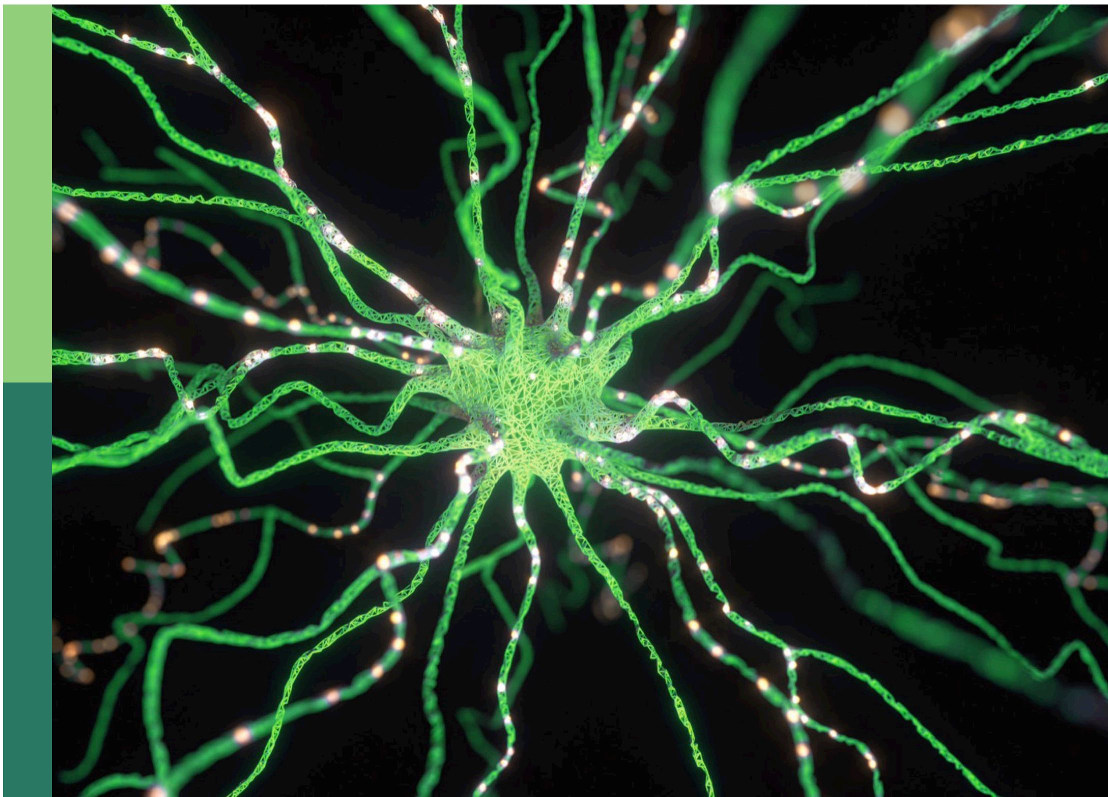
frontiers | Research Topics

Neurorobotics explores machine learning

Edited by
Fei Chen, Jose De Jesus Rubio and Mu-Yen Chen

Published in
Frontiers in Neurorobotics



ISSN 1664-8714
ISBN 978-2-83251-191-6
DOI 10.3389/978-2-83251-191-6

About Frontiers

Frontiers is more than just an open access publisher of scholarly articles: it is a pioneering approach to the world of academia, radically improving the way scholarly research is managed. The grand vision of Frontiers is a world where all people have an equal opportunity to seek, share and generate knowledge. Frontiers provides immediate and permanent online open access to all its publications, but this alone is not enough to realize our grand goals.

Frontiers journal series

The Frontiers journal series is a multi-tier and interdisciplinary set of open-access, online journals, promising a paradigm shift from the current review, selection and dissemination processes in academic publishing. All Frontiers journals are driven by researchers for researchers; therefore, they constitute a service to the scholarly community. At the same time, the *Frontiers journal series* operates on a revolutionary invention, the tiered publishing system, initially addressing specific communities of scholars, and gradually climbing up to broader public understanding, thus serving the interests of the lay society, too.

Dedication to quality

Each Frontiers article is a landmark of the highest quality, thanks to genuinely collaborative interactions between authors and review editors, who include some of the world's best academicians. Research must be certified by peers before entering a stream of knowledge that may eventually reach the public - and shape society; therefore, Frontiers only applies the most rigorous and unbiased reviews. Frontiers revolutionizes research publishing by freely delivering the most outstanding research, evaluated with no bias from both the academic and social point of view. By applying the most advanced information technologies, Frontiers is catapulting scholarly publishing into a new generation.

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the *Frontiers journals series*: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area.

Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers editorial office: frontiersin.org/about/contact

Neurorobotics explores machine learning

Topic editors

Fei Chen — Italian Institute of Technology (IIT), Italy

Jose De Jesus Rubio — Instituto Politécnico Nacional (IPN), Mexico

Mu-Yen Chen — National Cheng Kung University, Taiwan

Citation

Chen, F., De Jesus Rubio, J., Chen, M.-Y., eds. (2023). *Neurorobotics explores machine learning*. Lausanne: Frontiers Media SA. doi: 10.3389/978-2-83251-191-6

Table of contents

frontiers in Neurorobotics

ORIGINAL RESEARCH
published: 28 August 2020
doi: 10.3389/fnbot.2020.00060

A Brain-Inspired Model of Theory of Mind

Yi Zeng[1,2,3,4*†], *Yuxuan Zhao*[1†], *Tielin Zhang*[1], *Dongcheng Zhao*[1,4], *Feifei Zhao*[1] *and Enmeng Lu*[1]

[1] *Research Center for Brain-Inspired Intelligence, Institute of Automation, Chinese Academy of Sciences, Beijing, China,* [2] *Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Beijing, China,* [3] *National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China,* [4] *School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China*

Theory of mind (ToM) is the ability to attribute mental states to oneself and others, and to understand that others have beliefs that are different from one's own. Although functional neuroimaging techniques have been widely used to establish the neural correlates implicated in ToM, the specific mechanisms are still not clear. We make our efforts to integrate and adopt existing biological findings of ToM, bridging the gap through computational modeling, to build a brain-inspired computational model for ToM. We propose a Brain-inspired Model of Theory of Mind (Brain-ToM model), and the model is applied to a humanoid robot to challenge the false belief tasks, two classical tasks designed to understand the mechanisms of ToM from Cognitive Psychology. With this model, the robot can learn to understand object permanence and visual access from self-experience, then uses these learned experience to reason about other's belief. We computationally validated that the self-experience, maturation of correlate brain areas (e.g., calculation capability) and their connections (e.g., inhibitory control) are essential for ToM, and they have shown their influences on the performance of the participant robot in false-belief task. The theoretic modeling and experimental validations indicate that the model is biologically plausible, and computationally feasible as a foundation for robot theory of mind.

Keywords: theory of mind, false-belief task, brain inspired model, self-experience, connection maturation, inhibitory control

Edited by:
Jun Tani,
Okinawa Institute of Science and Technology Graduate University, Japan

Reviewed by:
Jing Zhao,
Yanshan University, China
Xiaoqian Mao,
Qingdao University of Science and Technology, China

***Correspondence:**
Yi Zeng
yi.zeng@ia.ac.cn

†*These authors have contributed equally to this work*

Received: *22 May 2020*
Accepted: *27 July 2020*
Published: *28 August 2020*

Citation:
Zeng Y, Zhao Y, Zhang T, Zhao D, Zhao F and Lu E (2020) A Brain-Inspired Model of Theory of Mind. Front. Neurorobot. 14:60. doi: 10.3389/fnbot.2020.00060

1. INTRODUCTION

Theory of Mind (ToM) is the ability to infer and understand other people's mental states to predict their behavior (Premack and Woodruff, 1978). It is a fundamental cognitive ability for the social brain. One of the most critical milestones in the ToM development is gaining the ability to attribute false belief: that is, to recognize that others can have beliefs about the world that are diverging (Wimmer and Perner, 1983). There is a wide variety of false-belief task (Huang and Liu, 2017; Scott and Baillargeon, 2017), but most of them can be divided into unexpected transfer task (Wimmer and Perner, 1983), unexpected contents task (Perner et al., 1987), and appearance-reality distinction (Flavell et al., 1983). Flavell et al. (1983) present a classical unexpected transfer task, Sally-Anne Test: Sally first placed a marble into her basket; then, she left the scene, and the marble was transferred by Anne and hidden in her box. Then Sally returned, and children were asked a belief question "Where will Sally look for her marble?" If the children pointed to the previous location of the marble, it meant that the children could understand that Sally held a false belief about the marble's location. Most 4-year-olds could point to the correct location, but most 3-year-olds failed—they predict that Sally will find her marble in the box.

As indicated in Asakura and Inui (2016), although ToM research has made progress on empirical findings and theoretical advances, relatively few efforts have been made from the biological plausible computational models' perspective, especially for false belief understanding. Based on findings of neural correlates and mechanisms of the false-belief task, we propose and build a Brain-inspired model of Theory of Mind (Brain-ToM model). And we challenge the false-belief task by incorporating the proposed model to humanoid robots. In this paper, we only focus on non-verbal unexpected transfer tasks as described below, including how to learn to understand object permanence and visual access from self-experience and use them to infer other's belief and predict their behavior. The object permanence is the ability to understand that objects continue existence even it cannot be perceived (Piaget and Cook, 1952).

From our point of view, self-experience in autobiographical memory and its utilization to infer other's belief or predict other's action is fundamental and crucial to the ToM. It is also mentioned as self-projection in Buckner and Carroll (2007) or using memories to understand others (Moreau et al., 2013). It enables real understanding of the self and others as well as their relationships, and utilize them to infer others' mental states based on personal experience from the self point of view. This perspective seems somewhat missing in existing research about the computational model.

In our opinion, an agent who can infer other's belief and predict their behavior should have the capability of self-other distinction as the premise. So in Zeng et al. (2016, 2017), we proposed a brain-inspired robot bodily self-model with the neural mechanisms of bodily self-perception based on extensions to primate mirror neuron system, and apply it to the humanoid robot for self-recognition. In this paper, based on the related findings for neural correlates and mechanisms of the ToM, we propose a Brain-ToM model to make the humanoid robot learn from self-experience. With the Brain-ToM model, the robot can pass the non-verbal unexpected transfer tasks adapted from Senju et al. (2011) and Southgate et al. (2007). The efforts may also provide a possible computational model and hints on how infant infers and understands other people's beliefs. Compared to the previous model, the characteristics of our model are with relatively more solid details from the biological brain. It explores the effect of self-experience as a core and is with considerations on the maturation of correlated brain areas (e.g., calculation capability) and their connections (e.g., inhibitory control). Besides, the model is naturally a brain-inspired spiking neural network model and is fundamentally based on brain plasticity principles.

The rest of this paper is organized as follows: Section 2 reviews the related work of computational models, the false belief tasks, and the brain regions in the ToM. In section 3, the architecture of the Brain-ToM model, the concrete neural network architecture, the Voltage-driven Plasticity centric Spiking Neural Networks (VPSNN), and the inhibitory control mechanism are introduced. The experimental settings, the experimental results and analyses are given in section 4. Some discussions and conclusions are drawn in sections 5, 6, respectively.

2. RELATED WORKS

In this section, we briefly review several related works, including the computational models, the false belief tasks, and the related brain regions of ToM.

2.1. Computational Models

Berthiaume et al. (2013) presented a constructivist connectionist model to simulate the false-belief task. The model encoded the location of an object, whether an agent has observed the object's movement, and the location where the agent came back to search. With the increased hidden units to improve computational power, the model would predict the correct search in two different false belief tasks—the approach task and the avoidance task. Their model was the first computational model to autonomously construct and transit between structures and to cover the two major false-belief task transitions. They suggested the view that the source of the transition is not developed in the understanding of beliefs, but changes in auxiliary skills such as: executive function, understanding and using representations, working memory, or language. Goodman et al. (2006) built two Bayesian models named CT model (copy theorist) and PT model (perspective theorist). Beliefs were only correlated to the location of the toy in the former model, and in the later model, the belief was not only correlated to the toy's location but also Sally's visual access, i.e., could Sally saw the toy moved or not. With the increase of resources and complexity in the PT model, the model could pass the false-belief task. Asakura and Inui (2016) designed a Bayesian framework that integrates theory-theory and simulation theory for false belief reasoning in the unexpected-contents task. This framework predicted other's belief by the self model and others model which were responsible for simulation-based and theory-based reasoning, respectively. In their opinion, the multiplicative effect of the ability to understand diverse beliefs and knowledge access could predict children's false belief ability. Their model provided good fits to a variety of ToM scale data for preschool children. Rabinowitz et al. (2018) designed a ToM neural network to learn how to model other agents by meta-learning. They constructed an observer who could collect agent's behavioral traces, and its goal was to predict the agent's future behavior. They applied the proposed ToMnet model in simple grid world environments, showing that the observer could model agents effectively and passed Sally Anne Test. And the observer needed not to be able to execute the behaviors itself. O'Laughlin and Thagard (2000) built a connectionist network

Abbreviations: ToM, theory of mind; STS, superior temporal sulcus; TPJ, temporo-parietal junction; IPL, inferior parietal lobule; pSTS, posterior superior temporal sulcus; PCC, posterior cingulate cortex; ACC, anterior cingulate cortex; mPFC, medial prefrontal cortex; vmPFC, ventral medial prefrontal cortex; dmPFC, dorsal medial prefrontal cortex; IFG, inferior frontal gyrus; vPMC, ventral premotor cortex; M1, primary motor cortex.

whose nodes represent the relevant event in the false-belief task, and passed the false-belief task by modifying the connection weight of excitatory links and inhibitory links. Milliez et al. (2014) presented a spatio-temporal reasoning system SPARK, which included a well-designed model of object position hypotheses and generated beliefs. They enabled the robot to passed the Sally-Anne test and performed well in dialog disambiguation. Patacchiola and Cangelosi (2020) proposed a developmental cognitive architecture for trust and ToM in humanoid robots. This architecture was inspired by psychological and biological observations. And it based on an actor-critic (AC) framework, an epigenetic robotic architecture (ERA), and a Bayesian network (BN). These modules represent the functions of the corresponding brain regions in ToM, and they could uncover the detailed mechanisms of trust-based learning in children and robots. Finally, they reproduced psychological experiments with the iCub humanoid robot, and the results are coherent with the real experimental data from children.

2.2. False Belief Tasks

There is a wide variety of false-belief tasks, here we introduce two non-verbal unexpected transfer tasks that will be adapted to verify the validity of the model.

Senju et al. (2011) investigated whether 18-month-olds infants would use their own past experience of visual access to attribute perception and consequent beliefs to other people. Infants are divided into two groups, one group wore opaque blindfolds, and another wore trick blindfolds which looked opaque but were actually transparent. The opaque blindfold and trick blindfold looked identical. The test stage is the same as Southgate's as described below. The puppet hid an object in the left box. After the actor wore the same blindfold, the puppet removed the object from the scene. The opaque blindfold group expected the actor to behave according to false belief, and the trick blindfolds did not. Their results show that 18-month-olds used self-experience with the blindfold to assess the actor's visual access and predict their behavior.

Southgate et al. (2007) used an anticipatory looking measure to test whether 2-year-olds infants have the ability of false belief understanding. In the familiarization trials, the puppet hid an object in the left or right box, then left the scene. The actor reached through the corresponding window after doors illuminated with the simultaneous chime. Note that "doors illuminated with simultaneous chime" indicated that the actor was going to reach the object. In one test trial, the puppet hid the object in the left box then move it to the right box. After the actor turned around, the puppet removed the object from the scene. In another test trial, the puppet hid the object in the left box, then the actor turned around. The puppet moved the object to the right box and hid it, then remove the object from the scene. For both test trails, the actor turned back and doors illuminated with simultaneous chime after the object was removed from the scene. Most infants could gaze toward the correct window. Their data demonstrated that 25-month-old infants had the ability of false belief understanding. The details of this experiment were illustrated in the figure of Southgate et al. (2007).

2.3. Brain Regions in Theory of Mind

Several brain regions, including the mPFC, bilateral TPJ, and precuneus, have been consistently found to be activated in various mentalizing tasks in healthy individuals (Green et al., 2015). Schurz et al. (2014) meta-analyzed 757 activation foci reported from 73 imaging studies of ToM that involved 1,241 participants, and their meta-analysis contained six different task groups—False belief vs. photo, Trait judgments, Strategic games, Social animations, Mind in the eyes, and rational actions. They found the mPFC and bilateral posterior TPJ were activated in all task groups. In false belief vs. photo stories task group, they found TPJp, IPL, precuneus, posterior cingulate gyrus, mPFC connectivity clusters 3 and 4, ventral parts of the mPFC, anterior cingulate gyrus, right anterior temporal lobe, and adjacent parts of the insula be activated. Molenberghs et al. (2016) conducted a series of activation likelihood estimation (ALE) meta-analyses on 144 datasets (involving 3,150 participants) to address the brain areas that implicated in specific types of ToM tasks. In terms of commonalities, consistent activation was identified in the medial prefrontal cortex and bilateral temporoparietal junction. Schurz and Perner (2015) reviewed nine current neurocognitive theories of how the ToM was implemented in the brain and evaluate them based on the results from a recent meta-analysis by Schurz et al. (2014). From theories about cognitive processes being associated with certain brain areas, they deduced predictions about which areas should be engaged by the different types of ToM tasks. These brain areas contain the mPFC, the pSTS, the TPJ, and the IPL.

3. METHODS

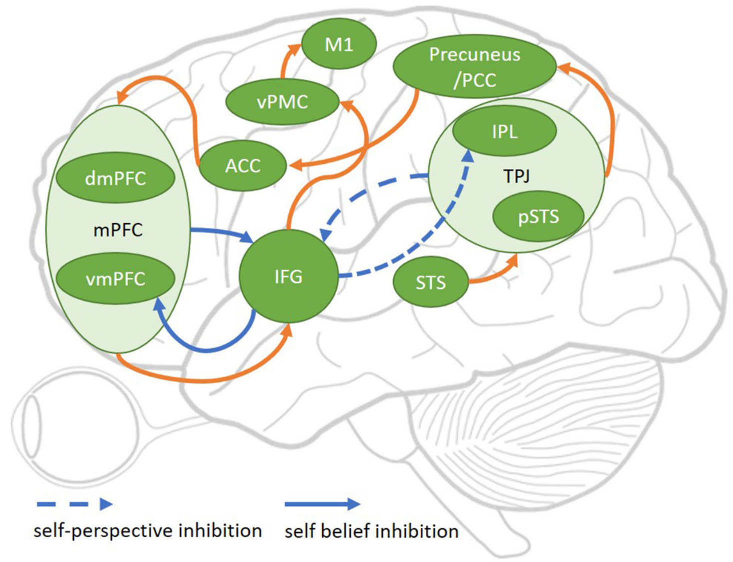
3.1. Architecture of the Brain-ToM Model

The architecture of the Brain-ToM model is shown in **Figure 1**.

The STS is sensitive to biological motion, and in our computational model, its function is to visually encode biological motion (Grossman and Blake, 2002).

The TPJ is considered as a crucial area in self-other distinction (Eddy, 2016; Bardi et al., 2017), controls representations relating to the self and other (Eddy, 2016), and involvement in self perspective-taking as well as other perspective-taking (Vogeley et al., 2001; van der Meer et al., 2011). There is no consensus on the anatomical definition of the extent and precise location of the TPJ (Igelstrom and Graziano, 2017), but in general, the TPJ contains two anatomically distinct regions including the IPL and pSTS (Abu-Akel and Shamay-Tsoory, 2011; Schurz et al., 2014; Igelstrom and Graziano, 2017). In our computational model, the TPJ is used to distinguish self and others, store self and other-relevant stimuli, and decide the output sequence of self and other-relevant stimuli.

The IPL is considered as a critical area in distinguishing the self from others and identifying the body ownership in our robot bodily self-model in Zeng et al. (2016, 2017), and some studies have indicated that it will be activated during lower-order self-perception (Schurz and Perner, 2015; Igelstrom and Graziano, 2017). So in our computational model, the IPL is used to store self-relevant stimuli. The pSTS (Frith and Frith, 1999; Schurz and Perner, 2015) is concerned with representing



Brain Area	Function in the computational model
STS	visually encodes biological motion
TPJ	distinguishes self and others store relevant stimuli
IPL	stores self-relevant stimuli
pSTS	stores other-relevant stimuli
Precuneus	infers visual access
PCC	sends information
ACC	belief reasoning
mPFC	stores beliefs
vmPFC	stores self-belief
dmPFC	stores other's belief
IFG	self-perspective inhibition self-belief inhibition
vPMC	encodes kinematics
M1	motion execution

FIGURE 1 | The Brain-ToM model (including major functional brain areas, pathways, and their interactions).

the actions of others and perspective taking (Frith and Frith, 2006). In our computational model, the pSTS is used to store other-relevant stimuli.

The precuneus is often activated during visuo-spatial imagery, episodic memory retrieval, self- processing operations (Cavanna and Trimble, 2006), and retrieving previous experiences (Molenberghs et al., 2016). And a main function of the precuneus in ToM is mental imagery to represent the perspective of another person (Cavanna and Trimble, 2006; Schurz et al., 2013, 2014) or modeling other people's views (Vogeley et al., 2004). In our computational model, the precuneus is the critical area for a machine to learn visual access from its own experience and uses it to infer other people's visual access. The PCC is the caudal part of the cingulate cortex, and the precuneus lies posterior and superior to the PCC (Leech and Sharp, 2014). In our computational model, the PCC receives the information from precuneus and sends it to ACC.

The anterior paracingulate cortex is often considered to be a part of the ACC and is used for representing mental states "decoupled" from reality (Gallagher and Frith, 2003). In our computational model, the ACC is the critical area in acquiring the ability of object permanence and then used it for belief reasoning.

The mPFC contains vmPFC and dmPFC. The vmPFC has typically been associated with self-referential processing, and the dmPFC has typically been associated with others-referential processing (Abu-Akel and Shamay-Tsoory, 2011; Denny et al., 2012; Jiang et al., 2016; Molenberghs et al., 2016). In our computational model, the mPFC is used to store the result of belief reasoning from ACC: the vmPFC stores the result of self-belief reasoning, and the dmPFC stores the other's belief reasoning.

The IFG is a critical area for the inhibition process: self-perspective inhibition and self-belief inhibition. The IFG inhibits self-perspective when self perspective and other-perspective are conflictive (Hartwright et al., 2012, 2015), and is suggested to inhibit self-belief to obtain correct task performance in the false-belief task (Mossad et al., 2016). Another function of IFG is encoding action goals and responding to goal-driven motions (Hamzei et al., 2016). The vPMC encodes kinematics based on motion goals from IFG, the encoded information is sent to M1. M1 encodes the strength and orientation of motion and controls the concrete motion execution (Georgopoulos et al., 1986).

As indicated in Green et al. (2015) and Jiang et al. (2016), the specific roles that brain areas have in the mentalization processes is not clear. Based on the neuroimaging studies as described above, we propose four pathways for robots learning from self-experience and uses it in the false-belief task, they are self-experience learning pathway, motivation understanding pathway, reasoning about one's own belief pathway and reasoning about other people's belief pathway.

The self-experience learning pathways is consist of object permanence learning pathway [Precuneus/PCC → ACC] and the visual access learning pathway [STS → TPJ(IPL) → Precuneus/PCC].

The test pathways are consist of motivation understanding pathway, reasoning about one's own belief pathway, and reasoning about other people's belief pathway.

The motivation understanding pathway is STS → pSTS → TPJ(IPL) → IFG.

The reasoning about one's own belief pathway contains belief reasoning pathway [STS → TPJ(IPL) → Precuneus/PCC → ACC → MPFC(vMPFC)] and the motor response pathway [MPFC(vMPFC) → IFG → vPMC → M1].

The reasoning about other people's belief pathway contains the true belief reasoning pathway and the false belief reasoning pathway.

The true belief reasoning pathway contains belief reasoning pathway [STS → TPJ(pSTS) → Precuneus/PCC → ACC → MPFC(dMPFC)] and the motor response pathway [MPFC(dMPFC) → IFG→ vPMC → M1].

The false belief reasoning pathway contains the belief reasoning pathway [STS → TPJ → IFG → TPJ(pSTS) → Precuneus/PCC → ACC → MPFC(dMPFC)] and the motor response pathway [MPFC → IFG → MPFC (dMPFC) → IFG → vPMC → M1].

In the reasoning about other people's belief pathway of false belief reasoning, the conflict between self-and-other perspective in TPJ will activate IFG, then IFG will inhibit self-relevant stimuli in IPL. So the others-relevant stimuli in pSTS will be the first output of TPJ, then it sends to Precuneus/PCC. In the motor response pathway of false belief reasoning, the conflict between self-belief and other's belief in mPFC will activate the IFG, then the IFG inhibits self-belief in vmPFC. So the other's belief in dmPFC will be the output of mPFC, then it will be sent to IFG for encoding action goals.

3.2. Concrete Neural Network Architecture

The concrete neural network architecture of the model is shown as **Figure 2**, and it uses the leaky integrate-and-fire model (LIF) neurons. This section describes (1) the input and output encoding information of different brain areas, (2) the Voltage-driven Plasticity-centric Spiking Neural Networks (VPSNN) used in Precuneus/PCC and ACC for visual access learning and object permanence learning respectively, and (3) the inhibitory control mechanism which was used to select correct output information of TPJ or mPFC.

3.2.1. STS

The STS encodes the processed results of visual perception and body information of the self and others at time t.

$$STS_t = [id_t, object_t, box_t, blindfold_t, turn_t]$$

We detect this information using traditional template matching methods and represent the result by neurons with an input synaptic current I of 1.0 or 0.0. The id_t uses two neurons to represent the identification of self or others. For the identification of id_t, we use the Fast R-CNN to recognize others at time t. More details could be found in our previous work (Zeng et al., 2017). The $object_t$ and box_t are both tuples consist of object or box identification information and its location information respectively. For the identification of the $object_t, box_t, blindfold_t$, we first collect their image templates, and then use the traditional template matching method to identify them at time t. The location of $object_t$ or box_t is calculated by the distance between the center of the black rectangles and the center of the object or box at time t. The $blindfold_t$ uses two neurons to represent the wearing state of the blindfold (wear or not wear) and uses another two neurons to represent whether there is a blindfold at time t. Here we define the state of self as wearing a blindfold if the blindfold covers most areas of its visual field, and define the state of others as wearing a blindfold if the blindfold covers the other's face. The $turn_t$ uses two neurons to represent the state of turning. Here the turning-around state of the robot itself is detected by the degree to which its head is twisted, and the turning-around state of the other robot is detected by whether its face or back is recognized.

3.2.2. TPJ

The input information of the TPJ is directly from STS, as

$$TPJ_{input} = STS_t$$

and the information is divided into self-relevant stimuli and others-relevant stimuli by the id_t, and then stored in IPL and pSTS, respectively. The information in IPL, pSTS, and the output of TPJ are encoded as.

$$IPL/pSTS/TPJ_{output} = [object_t, box_t, blindfold_t, turn_t]$$

3.2.3. Precuneus/PCC

The Precuneus/PCC is used for visual access learning. Here we use the VPSNN based on our previous work (Tielin et al., 2018) to train the robot to learn visual access.

The input information of Precuneus/PCC contains the current information from the output of TPJ and the previous information from working memory, and it could be represented as

$$Precuneus/PCC_{input_t} = TPJ_{output_t} + WM_t$$

where

$$WM_t = \zeta \times Precuneus/PCC_{input_{t-1}}$$

The ζ is the forgetting factor. In the training stage, the target output of Precuneus/PCC is the perceived location of the object (active the *unseen* signal if no object is detected). The output of Precuneus/PCC is encoded by the input synaptic current I of either 1.0 or 0.0 in the perception neurons,

$$Precuneus/PCC_{output_t} = [location_1, location_2, unseen]$$

There are 160 trials in the training process. Each training trial contains two images collected from the robot as shown in **Figure 4**. The first image is collected when putting the various objects in one location as shown in **Figure 4a**. The second image is collected when the robot is asked "Where is the [object label]?" in three scenes: (1) when the blindfold is interposed (**Figures 4b,c**), (2) when the robot has turned around, (3) when the object is moved to another location. For example, the first image is collected at time $t - 1$, and the second image is collected at time t. The input of the VPSNN is $Precuneus/PCC_{input_t}$, and the target output of the VPSNN is $Precuneus/PCC_{output_t}$. The $Precuneus/PCC_{input_t}$ receives two inputs: one is the raw TPJ_{output_t} and the other is the $Precuneus/PCC_{input_{t-1}}$ with a forgetting factor. In each training trial, there is no previous information from working memory when collecting the first image, so the $Precuneus/PCC_{input_{t-1}}$ is equal to the $TPJ_{output_{t-1}}$. The $Precuneus/PCC_{output_t}$ is the perceived location of the object at time t. The robot trains the self-experience of visual access to

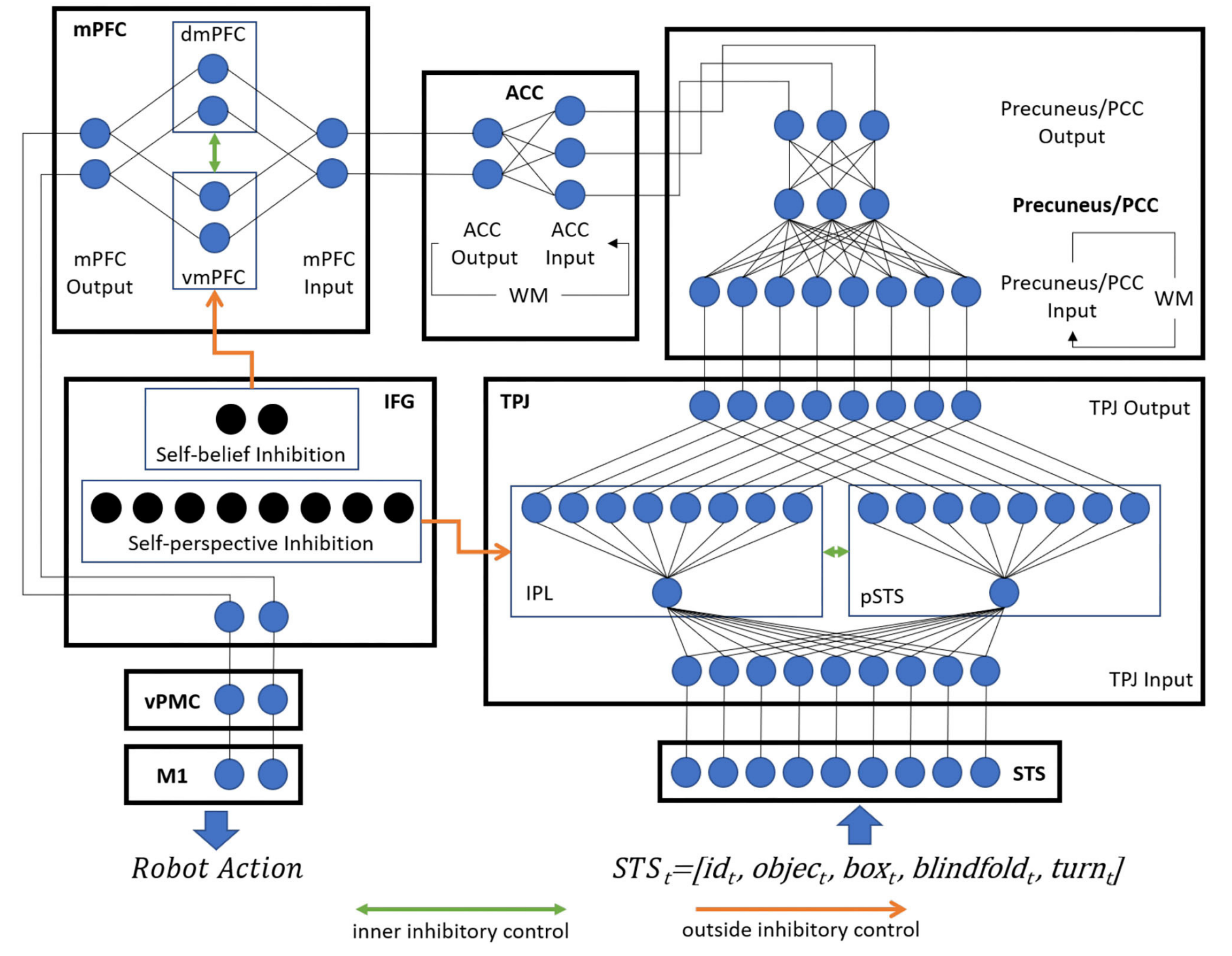


FIGURE 2 | The concrete neural network architecture of Brain-ToM model (WM denotes working memory).

the wearing of the blindfold or the turning around of the robot in the process of visual access learning, then uses it to infer itself and other robot's visual access in the Opaque-and-Transparent Blindfold Test and Turn Around Test.

3.2.4. ACC

The ACC is used for object permanence learning. Here we use the VPSNN to train the robot to learn object permanence. The input information of ACC contains the current information from the output of Precuneus/PCC and the previous information from working memory, and it could be represented as

$$ACC_{input_t} = Precuneus/PCC_{output_t} + WM_t$$

where

$$WM_t = \zeta \times ACC_{output_{t-1}}$$

The ζ is the forgetting factor. In the training stage, the target output of ACC is the location of the object. The output of ACC is encoded by the input synaptic current I of either 1.0 or 0.0 in the related neurons, i.e.,

$$ACC_{output_{t+1}} = [belief_{location_1}, belief_{location_2}]$$

There are 50 trials in the training process. Each training trial contains three images collected from the visual sensor, when (1) the various objects are put in one location (**Figure 5b**), (2) an object is hidden in the box (**Figure 5c**), and (3) the box is moved away (**Figure 5d**). For example, the first, second, third image is collected at time $t-1$, t, $t+1$, respectively. The input of the VPSNN is ACC_{input_t}, and the target output of the VPSNN is $ACC_{output_{t+1}}$. The ACC_{input_t} receives two inputs: one is the raw $Precuneus/PCC_{output_t}$ and the other is the $ACC_{output_{t-1}}$ with a forgetting factor. The $ACC_{output_{t-1}}$ is the perceived location of the object at time $t-1$. To train the ability of object permanence in the robot, we make the robot always perceive the location of the object at the end of each training trial. The $ACC_{output_{t+1}}$ is the perceived location of the object at time $t+1$. The robot trains itself a belief that objects are still where it has last located them, even they are out of its field of visual perception, and then uses it to infer itself and other robot's belief in the Opaque-and-Transparent Blindfold Test and Turn Around Test.

We train the visual access learning in Precuneus/PCC first, then the object permanence learning in ACC.

3.2.5. mPFC

The input of mPFC is identical to the output of ACC and distinguishes between the self-belief and other-belief by the source of the information: IPL or pSTS. The vmPFC and dmPFC both use two neurons to store the self-belief and other-belief about the location of the object.

$$\begin{aligned} &mPFC_{input_t} = ACC_{output_t} \\ &dmPFC_t = mPFC_{input_t}\ \textit{if the source is IPL} \\ &vmPFC_t = mPFC_{input_t}\ \textit{if the source is pSTS} \end{aligned}$$

The output of mPFC depends on the questions, which are set as follows: if the question is "Where is the ladybird according to the blue robot?" the other-belief stored in dmPFC tries to be the output of mPFC; if the question is "Where is the ladybird according to yourself?" the self-belief stored in vmPFC tries to be the output of mPFC.

$$mPFC_{output_t} = \begin{cases} dmPFC_t\ \textit{if the question is "Where is the} \\ \qquad\qquad \textit{ladybird according to the blue robot"} \\ vmPFC_t\ \textit{if the question is "Where is the} \\ \qquad\qquad \textit{ladybird according to yourself?"} \end{cases}$$

3.2.6. IFG

In the proposed model, IFG receives inputs from three sources: (1) the inhibit result neurons in TPJ that could stimulate IFG for self-perception inhibition, (2) the inhibit result neurons in mPFC that could stimulate IFG for self-belief inhibition, and (3) some other neurons in mPFC that could stimulate IFG to encode the action goal.

$$IFG_{input_t} = \begin{cases} I_{\textit{inhibit result neuron TPJ}} & \textit{self} - \textit{perception inhibition} \\ I_{\textit{inhibit result neuron mPFC}} & \textit{self} - \textit{belief inhibition} \\ mPFC & \textit{encode the action goal} \end{cases}$$

IFG uses the same number of neurons as IPL and as vmPFC to inhibit self-perception information and self-belief information, respectively, and uses another two neurons to encode the action goal, which is later sent to vPMC to control the robot's actions. The details of the inhibitory control mechanism of IFG could be found in section 3.4.

With the exception in the above mentioned evaluation, the synaptic plasticity only takes place in Precuneus/PCC and ACC in the process of training, while the weights of the other connections between various areas remain unchanged in the experiment.

3.3. VPSNN

For the mathematically modeling of brain regions such as Precuneus/PCC and ACC, here we select a standard VPSNN model (Tielin et al., 2018), which is a shallow feed-forward SNN and may well simulate input-output signals with the integration of supervised learning (with an additional teaching signal given directly to the output layer neurons) and unsupervised learning (tuned with biologically plasticity principles, e.g., STDP, and homeostatic membrane potential).

Two three-layer SNN architectures are designed for Precuneus/PCC (with 24 input neurons, three hidden neurons, and two output neurons) and ACC (with three input neurons, three hidden neurons, and two output neurons), respectively, as shown in **Figure 2**. The VPSNN includes four steps, namely: feed-forward information (including both membrane potential and spikes) propagation, unsupervised homeostatic state learning, supervised last layer learning, and passively updating synaptic weights based on STDP rules. In this paper, we take advantage of these four steps for the fast network tuning and update the methodologies of giving teaching signals from single SNN to two SNNs together for the better model integration.

3.3.1. The LIF Neuron Model

$$\tau_m \frac{dV}{dt} = -(V - V_L) - \frac{g_E}{g_L}(V - V_E) \tag{1}$$

$$\tau_E \frac{dg_E}{dt} = -g_E + \eta \sum_{j \in N_E} w_{j,i}\delta_t \tag{2}$$

The basic neuron model in VPSNN is the LIF model, which describes the dynamics of the membrane potential of V and synaptic-weight-related g_E, as shown in Equations (1) and (2). Once the pre-synaptic neurons fire, there is a non-linear increment of g_E, which will then propagate into V. The g_L is leaky conductance, V_L is leaky potential, τ_m and τ_E are conductance decay, η is the learning rate, and V_E is reversal potential.

3.3.2. The Feed Forward Propagation

The information propagation in the LIF neuron is slower compared with giving input directly into V. However, this is a specially designed procedure that will make the network-tuning focusing more on the homeostatic membrane potential adjustment and STDP learning. The information (especially the membrane potential) will be propagated from pre-synaptic neurons (e.g., V_j) into the post-synaptic neurons (e.g., V_i), and the whole feed-forward procedure V_i^{FF} is shown in Equation (3), in which the V_{th} is the firing threshold of the neurons.

$$\begin{cases} \tau_m \frac{dV_i}{dt} = -(V_i - V_L) - \frac{g_E}{g_L}(V_i - V_E) \\ \tau_E \frac{dg_E}{dt} = -g_E + \eta \sum_j^N w_{j,i} V_j \\ V_i = V_L, T_{ref} = T_0 \qquad\qquad if\ (V_i > V_{th}) \\ V_i^{FF} = V_i \end{cases} \tag{3}$$

3.3.3. Unsupervised Homeostatic Membrane Potential Learning

The basic homeostatic mechanism occurs in the input-output balance of the single neuron, described as the Equation (4).

$$\Delta E_i = V_i - \left(\sum_{j}^{N} w_{j,i} V_j - V_{th,i} \right) \tag{4}$$

The entire network homeostatic state can be represented as the addition of all of the neurons in each layer, i.e., $\Delta E = \sum_{i \in N} \Delta E_i$. Moreover, after the calculation (the detailed methodologies are shown in paper Tielin et al., 2018), the homeostatic membrane potential V_i^{Homeo} can be updated according to Equation (5). The η_i is the learning rate.

$$\Delta V_i^{Homeo} = -\eta_i \frac{V_i - \left(\sum_j^N w_{j,i} V_j - \sum_j^N V_{th,i} \right)}{-(V_i - V_L) - \frac{g_E}{g_L}(V_i - V_E)} \tag{5}$$

With the integration of Equations (3) and (5), the update rule of each neuron state V_i is shown in Equation (6). The t is the training time slot and T is the total training time in SNN learning.

$$\Delta V_i = \frac{t}{T} \Delta V_i^{FF} + \left(1 - \frac{t}{T} \right) \Delta V_i^{Homeo} \tag{6}$$

3.3.4. Supervised Last Layer Learning and STDP-Based Weights Consolidation

An additional teaching signal will be given to the network for guiding the proper network output. Here we add teaching signals into the last layer of SNN in the training procedure, as shown in Equation (7), in which V_T is teacher signal state, η^c is the learning rate.

$$dV_i = -\eta^c (V_i - V_T) \tag{7}$$

STDP rules (Bi and Poo, 2001; Dan and Poo, 2004; Bengio et al., 2015a,b) are further used for the knowledge consolidation from the membrane potential to synaptic weights, e.g., the synaptic weights could be passively updated by the changes of the pre- and post-synaptic neuron states. The function is as shown in Equation (8), in which the $V_i^{'}$ is the derivative value of V_i.

$$\Delta w_{j,i} \propto V_j V_i^{'} \tag{8}$$

3.4. Inhibitory Control Mechanism

The inhibitory control is used to select correct output information of TPJ or mPFC, and it can be divide into inner and outside inhibitory control. The inner inhibitory control cannot inhibit predominant information from self when the related information of self and others is conflictive, so we use the outside inhibitory control from IFG to inhibit the predominant information. Inhibitory control of one single neuron is shown in **Figure 3**. The IPL neurons (or vmPFC neurons) and pSTS neurons (or dmPFC neurons) receive electrovital currents of self-relevant stimuli and other-relevant stimuli, respectively. The input of inhibit neurons and temporary neurons depend on reasoning about other's belief (contains self-perspective inhibition and self-belief inhibition) or self-belief (contains other-perspective inhibition and other-belief inhibition), and the former is used to inhibit stimuli, the later is used to temporarily store uninhibited stimuli. In the process of self-perspective inhibition or self-belief inhibition, the input electrovital currents of inhibit neuron are equal to other-relevant stimuli, and the temporary neuron is equal to self-relevant stimuli (only in TPJ). So other-relevant stimuli will be the first output, and self-relevant stimuli will be the second output in TPJ, and other's belief will be exported in mPFC. In the process of other-perspective inhibition and other-belief inhibition, the inhibit electrovital currents are very big that it can completely inhibit other-relevant stimuli, and the temporary neuron is equal to other-relevant stimuli (only in TPJ). Therefore, in TPJ, self-related stimulus will be output first, other-related stimulus will be output second, and then self-belief will be output in mPFC.

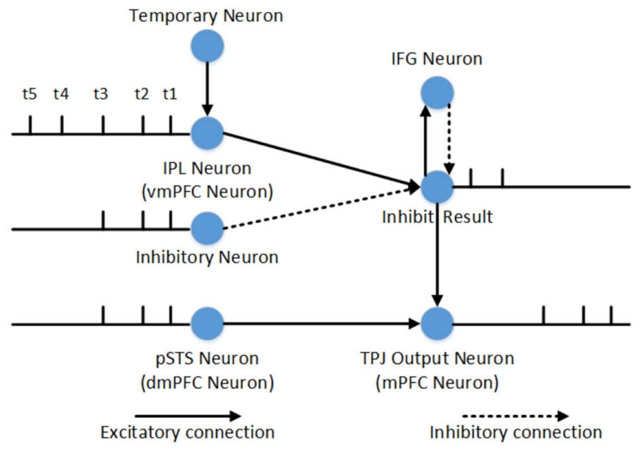


FIGURE 3 | Inhibitory control of one single neuron in reasoning about other's belief. At time t1, t2, and t3, the firing pattern of pSTS and IPL are identical, which means that the other-relevant stimuli and self-relevant stimuli are identical. The inhibitory neurons can inhibit self-relevant stimuli successfully. But at time t4 and t5, the other-relevant stimuli and self-relevant stimuli are in conflict with each other, and the inhibitory neurons cannot inhibit self-relevant stimuli. The inhibit result neurons will fire and stimulate IFG activation, while IFG activation will inhibit the firing of inhibit result neuron. The inhibit result neurons will combine with neurons in pSTS to generate other-relevant stimuli output firstly. Then the temporary neuron stimulates IPL, because the electrovital currents in pSTS and the inhibitory neuron is zero at this moment, self-relevant stimuli will be the second output of TPJ. Compared to the process of self-perspective inhibition, the only difference is that the inhibitory control in TPJ is used to decide the sequence of information output, and the inhibitory control in mPFC is used to decide which belief to export. Therefore, the process of self-belief inhibition does not require the involvement of temporary neuron.

4. EXPERIMENTS

In this section, we introduce the experimental settings and the result of our proposed model.

4.1. Experimental Settings

We deploy the computational model to humanoid robotics and use the Opaque-and-Transparent Blindfold Test and Turn

Around Test to validate the Brain-ToM model. Furthermore, we test the effect of self-experience, maturation of correlate brain areas (e.g., calculation capability) and their connections (e.g., inhibitory control) on the performance of participant robot in Opaque-and-Transparent Blindfold Test.

4.1.1. Opaque-and-Transparent Blindfold Test

The Opaque-and-Transparent Blindfold Test is adapted from Senju et al. (2011).

We enable all the robots to learn the visual access of blindfold from self experiences, as the infants' experience in Senju's experiment. In the proposed model, this process takes place in Precuneus (Vogeley et al., 2004; Cavanna and Trimble, 2006; Schurz et al., 2013, 2014). The robots are divided into two groups—the opaque blindfold group and the transparent blindfold group. The opaque blindfold and transparent blindfold look identical, at least the robot cannot distinguish them from appearance, but the transparent blindfold can make the robot who wears it see through, and the opaque blindfold cannot. **Figure 4** presents the visual inputs of the robot in the opaque blindfold group (with **Movie S1**) and the transparent blindfold group (with **Movie S2**). In this stage, the robots could not observe other robots wearing the blindfold, as the infants in Senju's experiment, so they have no opportunity to learn the property of the blindfold from the third-person point of view.

We suggest that the ability of learning object permanence is the prerequisite for the ToM. As indicated in Piaget and Cook (1952) and Bruce and Muhammad (2009), Piaget defined six developmental stages of object permanence. During the early stages (Stage I, Stage II, Stage III), children failed to find a hidden object. During Stage IV (8–12 months) children can retrieve an object when its concealment is observed. But they cannot find the object when it is continuously moving. During Stage V (12–18 months), the children can retrieve an object when it is hidden several times within his or her view. In summary, when an object was hidden in location A and then hidden in location B, the children would try to find the object in location A during Stage IV and would try to find it in location B during Stage V. With similar principles, here we enable all of the robots in the experiment acquire the ability of learning object permanence from their own self experiences, and in our model, ACC acts as a central role to realize this cognitive function (Gallagher and Frith, 2003). **Figure 5** shows the visual inputs of the robots in this process.

In the test stage, participant robots use the Brain-inspired Robot Bodily Self Model which we proposed in Zeng et al. (2016, 2017) to distinguish self and others. As in the experiment of Senju et al. (2011), the actor robot will try to find the hidden object in the box before the final test. By this way, the participant robot could understand that the actor has the same cognitive ability (e.g., visual ability) and the goal of the actor robot (**Movie S3**). In the final test, the opaque blindfold group and the transparent blindfold group are tested with the same process as shown in **Figure 6**. Then the participant robot be asked two questions: "Where is the ladybird according to the blue robot?" and "Where is the ladybird according to yourself?" We determine whether the robot can pass the task by detecting the direction of the finger which makes the results more intuitive.

4.1.2. Turn Around Test

The Turn Around Test is adapted from Southgate et al. (2007).

The robot learns the visual access of turning around from self-experience. The Turn Around Test is similar to the Opaque-and-Transparent Blindfold Test. The diversity of belief is caused by different blindfolds in the Opaque-and-Transparent Blindfold Test, and in Turn Around Test, it is caused by the behavior of turn around. The visual inputs of the participant robot are shown in **Figure 7**. As with the Opaque-and-Transparent Blindfold Test, the participant robot also be asked two questions: "Where is the ladybird according to the blue robot?" and "Where is the ladybird according to yourself?" And we determine whether the robot can pass the task by detecting the direction of the finger.

4.1.3. Maturation Test

The ability for the ToM comes with individual development process (Grosse Wiesmann et al., 2017). Grosse Wiesmann et al. (2017) discussed the influence of white matter structure on ToM by tract-based spatial statistics analysis and probabilistic tractography. They found that "the developmental breakthrough in false belief understanding is associated with age-related changes in local white matter structure in temporoparietal regions, the precuneus, and medial prefrontal cortex, and with increased dorsal white matter connectivity between temporoparietal and inferior frontal regions." And they thought "the emergence of mental state representation is related to the maturation of core belief processing regions and their connection to the prefrontal cortex" (Grosse Wiesmann et al., 2017). But their research focused on the 3- and 4-year-old children in the explicit false-belief tasks, and did not include younger infants who cannot pass the implicit false-belief task. They did not test whether this finding is also associated with an implicit task, because of the difficulties in performing MRI with toddlers.

Although the developmental neural basis for the implicit false-belief task is still not very clear, we hypothesize that the developmental process in implicit false belief understanding is relevant with explicit one, and will also be associated with the maturation of correlate brain areas and their connections. We aim to test this hypothesis by our computation model, and apply it to the Brain-ToM model that we developed for machine intelligence.

The maturation of correlate brain areas could be regarded as calculation capability in our model, and the calculation capability increases with the maturation of brain areas. The calculation capability in this model is proportional to the number of neurons in the hidden layer. We simulate immature Precuneus/PCC by reducing the number of the neurons in its own hidden layer, then verify the effect of this condition on the performance of the participant robot.

The maturation of the connection between brain areas is critical for information transmission and information integration, especially inhibitory connection and control. The inhibitory control is generally considered as a key mechanism in false-belief task (Leslie and Polizzi, 1998; Scott and Baillargeon, 2017), and we think that the maturation of connections between IFG and TPJ, IFG and vmPFC are the neural basis of self-perspective inhibition and self-belief inhibition, respectively. We

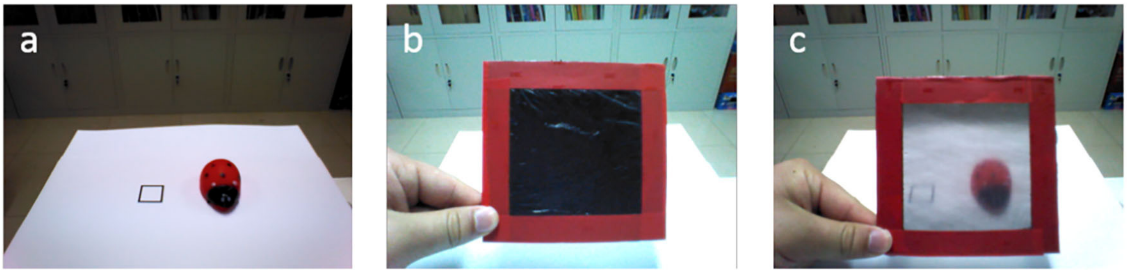


FIGURE 4 | Visual access learning of blindfold from self-experience. **(a)** An object is put on either of the black rectangles. **(b)** Visual inputs of the opaque blindfold group (with **Movie S1**). We interpose blindfold between the eyes of the robot and the object, then ask it "Where is the [object label]?" The robot will reply with the location of the object or with the fact that it did not see it. **(c)** Visual inputs of the transparent blindfold group (with **Movie S2**). The process is the same as the opaque blindfold group.

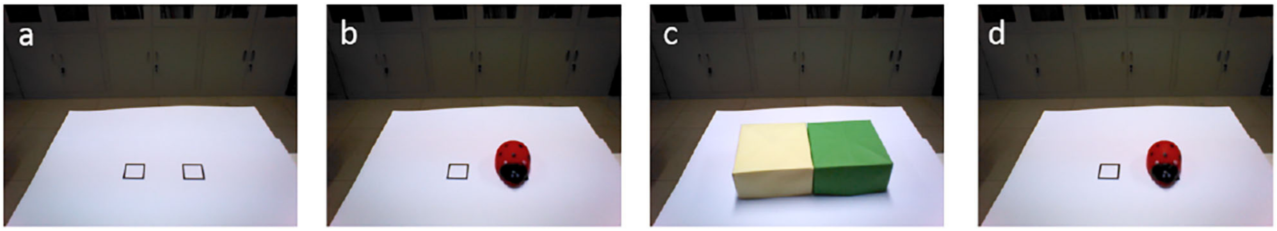


FIGURE 5 | Object permanence learning from self-experience. **(a)** The black rectangles are used to indicate the candidate positions of the object. **(b)** An object is put on either of the black rectangles, the robot can detect the location of the object—left side or right side. **(c)** The yellow box and the green box are used to hide the object. The robot cannot perceive the object in its visual field, hence cannot find it. It is similar to the early stages in Piaget's Stages of Object Permanence. **(d)** The boxes are removed, and the robot can perceive the object's location in its visual field.

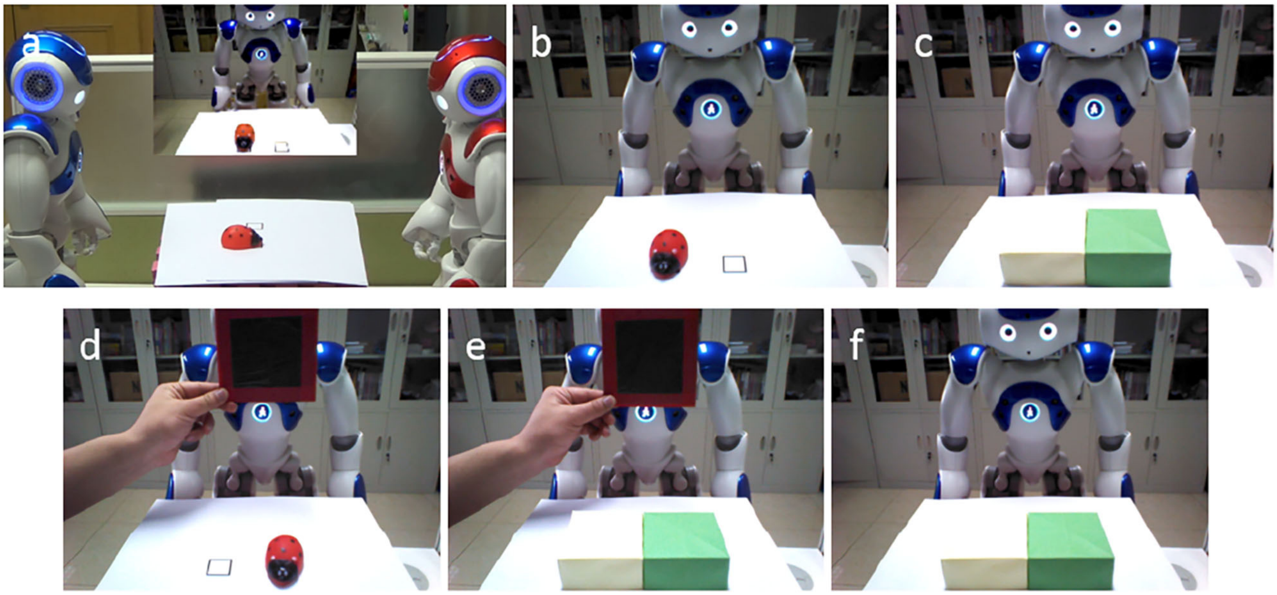


FIGURE 6 | Visual inputs of participant robot in the test stage. **(a)** The blue robot in the left is the actor, and the red robot in right is the participant who should infer the actor's belief. The middle screen in the **(a)**, and the remaining figures are the visual inputs of the participant robot. **(b)** An object (ladybird) is put on the left black rectangle. **(c)** The ladybird is hidden in the yellow box. **(d)** The blindfold is interposed between the actor (the blue robot) and the object (ladybird), and the object is moved to the right side. **(e)** The ladybird is hidden in the green box. **(f)** Finally, the blindfold is removed.

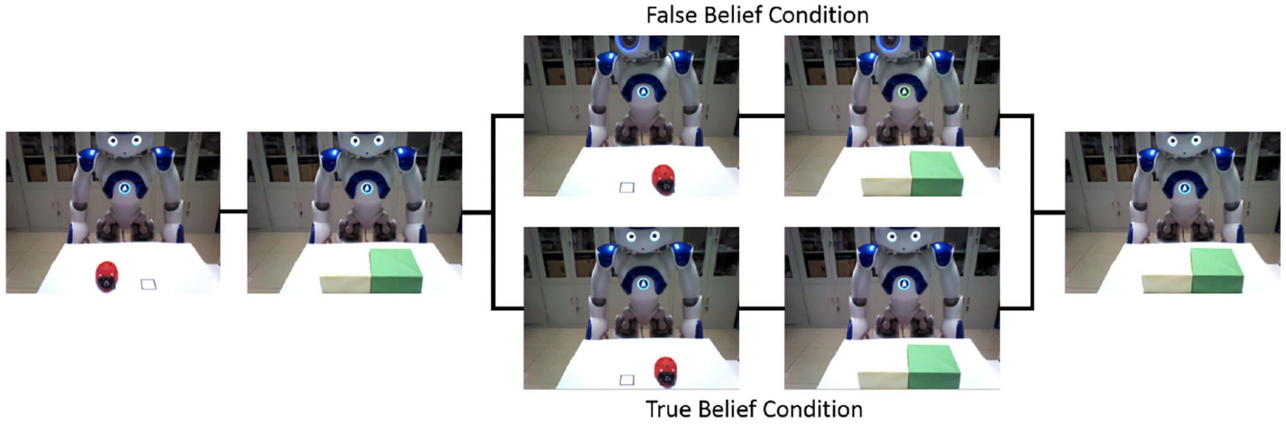


FIGURE 7 | Turn Around Test. An object (ladybird) is put on the left black rectangle and hidden it in the yellow box firstly. In the false belief condition, the object was moved to the other box when the actor robot turned around (**Movie S6**). And in true belief task, the actor robot did not turn around when the object was moved to the other box (**Movie S7**).

simulate immature connections between IFG and TPJ, IFG and vmPFC by set the synaptic weights as 0, then verify the effect of this condition on the performance of the participant robot.

4.2. Experimental Results and Analyses

In this section, we present the results of our model. Besides, we analyze the temporal and spatial activation of different brain areas during different tasks, the effect of self-experience, maturation of correlate brain areas (e.g., calculation capability) and their connections (e.g., inhibitory control) on the performance of participant robot in Opaque-and-Transparent Blindfold Test.

4.2.1. Opaque-and-Transparent Blindfold Test

The opaque blindfold group can be regarded as the false belief condition, as the actor robot's belief is inconsistent with the representations of reality. When asking the participant robot "Where is the ladybird according to the blue robot?" the participant robot will point to the yellow box on the left side. And when asking the participant robot "Where is the ladybird according to yourself?" the participant robot will point to the green box on the right side (**Movie S4**).

The transparent blindfold group can be regarded as the true belief condition, as the actor robot's belief is consistent with the representations of reality. When asking the participant robot the upper two questions, for both of them, the participant robot will point to the green box on the right side (**Movie S5**).

We repeat this experiment 20 times, the robot could pass the task every time, and we calculate the mean value and standard deviation of time consumption in different brain areas. To make the results more visible and clear, the data from STS, Precuneus/PCC, ACC, vPMC/M1 are excluded, because their time consumption is similar in different tasks. We select the time consumption in TPJ, IFG, and mPFC as examples. The time consumption of false belief reasoning (87.7 ms) is longer than true belief reasoning (69.8 ms). And the time consumption of reasoning about other's belief (78.8 ms) is longer than self-belief (15.2 ms). In the process of reasoning about self-belief, the time consumption of false belief condition (15.4 ms) and true belief condition (15 ms) are similar. In traditional true belief task, the time consumption will be shorter. The perception in TPJ is identical, hence the IFG will not be activated in this task and the time consumption is reduced.

Here we provide the time consumption in different tasks. Based on the belief about the object location, the task can be divided into reasoning about actor robot's false-belief task (other-incongruent condition) and true belief task (other-congruent condition), reasoning about participant robot's own belief task which contains self-incongruent condition (self-belief is divergent from other's) and self-congruent condition (self-belief is corresponding with other's). **Figure 8** shows the temporal and spatial activation of different brain areas during different tasks. This process only contains the perception conflict stage (as shown in **Figure 6d**) and motion response stage which have critical differences in different tasks. The time consumptions in STS and vPMC of different tasks are similar, about 448 and 500 ms, respectively. So we select 400-900 ms to show the process and the difference in different tasks. Reasoning about other's belief in the transparent group can be regarded as true belief task, but it must be noted that, the information from self-perspective and other-perspective are identical in tradition true belief task, while they conflict with each other in this task which is originally from Senju et al. (2011) based on human studies of 18-month-olds. For example, in the transparent blindfold task, the participant robot perceived from self-perspective that "I (participant robot) saw the object moved to the green box without blindfold" and the participant robot perceived from other-perspective that "the actor robot saw the object moved to the green box with blindfold," so the information is conflictive in this task. And in traditional true belief task, the information is identified as "I (participant robot) saw the object was moved to the green box" and "the actor saw the object was moved to the green box" without the difference caused by the transparent blindfold.

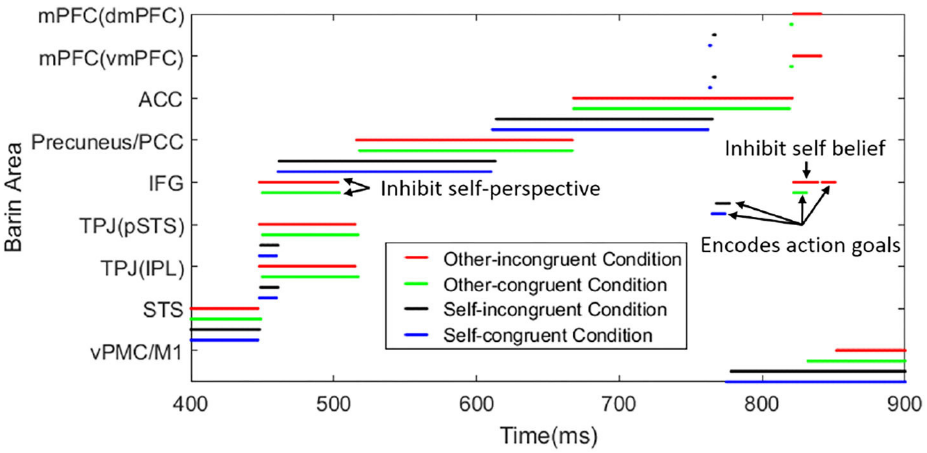


FIGURE 8 | The temporal and spatial activation of different brain areas during different tasks. The time consumption in STS (visually encoding biological motion), Precuneus/PCC (inferring visual access), ACC (belief reasoning), IFG (encoding action goals), and vPMC/M1 (encoding kinematics/motion execution) are similar in different tasks. In the process of TPJ activation (deciding the output sequence of self and other-relevant stimuli) in other-incongruent condition (red line) and other-congruent condition (green line), IFG is activated to inhibit self-perspective. In the process of mPFC activation (deciding belief for motor response) in other-incongruent condition, IFG is activated to inhibit self-belief. In the tasks of self-incongruent condition (black line) and self-congruent condition (blue line), the IFG is only activated in the process of encoding action goals. The value of total time consumption is self-congruent condition < self-incongruent condition < other-congruent condition < other-incongruent condition. To show the function of IFG easily, we make the arrow mark in the figure.

Our focus is the activation sequence of brain areas and reaction time in the various task, such as the other-incongruent condition spends more time than self-congruent condition rather than the numerical value of time consumption, these result is consistent with the functional neuroimaging studies in Mossad et al. (2016) and Dohnel et al. (2012).

In the process of inferring visual access which corresponds to **Figure 6d**, even though the visual inputs in both groups are identical, the output is different when the participant robot infer other's visual access with different self-experience. In other words, when inferring visual access of another person by self-experience, the opaque blindfold group will know the actor cannot see the moving object, and the transparent blindfold group will know the actor can. When inferring self visual access, the perception of visual inputs and the result of precuneus are identical in both groups.

In the process of inferring other's visual access, the IFG will not be activated when self-perspective and other-perspective is identical, as shown in **Figures 6b,c,f**. If the self-perspective and other-perspective are in conflict with each other in the process of reasoning about other's belief, the IFG will be activated to inhibit the information of self-perspective, as shown in **Figures 6d,e**.

In the process of motor response in reasoning about other's belief, the IFG will inhibit self-belief if the beliefs are conflictive. So the other's belief in dmPFC will be the output of mPFC. Then IFG receives input from mPFC and encodes action goals to control vPMC to action.

In addition, we also test the effect of blindfold position in this task. In the visual access of blindfold learning stage, we add a new phase: we put the blindfold on the desk or interpose it at others position to make that both the blindfold and object can be perceived by the participant robot, and we also ask the question "Where is the [object label]?" In the test stage, we put the blindfold on the desk or interpose it at other positions where the actor's visual inputs are not blocked. Both of the groups can infer the visual access of actor robot correctly, and conclude that the actor robot could see the object move to the right side. And self-belief is corresponding to other's belief in both groups. This additional test could prove that the actor robot does not use low-level features such as whether the blindfold exists when inferring other's mental state, and it also proves the effect of the bodily model in this task.

4.2.2. Turn Around Test

In the false belief condition, the actor robot's belief is inconsistent with the representations of reality. When asking the participant robot "Where is the ladybird according to the blue robot?" the participant robot will point to the yellow box on the left side. And when asking the participant robot "Where is the ladybird according to yourself?" the participant robot will point to the green box on the right side (**Movie S6**).

In the true belief condition, the actor robot's belief is consistent with the representations of reality. When asking the participant robot the upper two questions, for both of them, the participant robot will point to the green box on the right side (**Movie S7**).

We repeat this experiment 20 times, the robot could pass the task every time. The mechanism of Turn Around Test is similar to the Opaque-and-Transparent Blindfold Test, the only difference is that the self-perspective and other-perspective are identical in true belief task of Turn Around Test. So IFG was not activated in this stage. We select the time consumption in TPJ, IFG, and mPFC as examples. The time consumption of false belief reasoning (87.3 ms) is longer than true belief reasoning (16.1 ms). And the time consumption of reasoning about other's belief (51.7 ms) is longer than self-belief (15.3 ms). In the process of reasoning about self-belief, the time consumption of false belief condition (15.4 ms) and true belief condition (15.2 ms) are similar.

4.2.3. Maturation Test

The maturation of correlate brain areas will be regarded as calculation capability in our model, and the calculation capability increases with the maturation of brain areas. The calculation capability in this model is proportional to the number of neurons in the hidden layer. For example, If the neuron number involved in the calculation of the hidden layer in Precuneus/PCC is two or fewer, even though participant robot can learn visual access of blindfold, it failed in inferring other's visual access. As indicated in Myowa-Yamakoshi et al. (2011), 8-month-old infants and 12-month-old infants had experienced being blindfolded, when they saw a blindfolded actor did a successful goal-directed action which normally could not succeed with a blindfold, 12-month-old infants will look longer, but 8-month-old infants will not. We think that with the maturation of correlate brain areas as well as their connections, more neurons and synaptic connections will be included in the task processing.

The maturation of the connection between brain areas is critical for information transmission and information integration, especially inhibitory connection and control. The inhibitory control is generally considered as a key mechanism in false-belief task (Leslie and Polizzi, 1998; Scott and Baillargeon, 2017), and we think that the maturation of connections between IFG and TPJ, IFG and vmPFC are the neural basis of self-perspective inhibition and self-belief inhibition respectively. As shown in **Figure 9**, the inhibitory control uses inhibitory neurons and temporary neurons which store information temporarily for the selection of correct output information of TPJ or mPFC. In the process of reasoning about other's belief, the inhibitory neurons in TPJ and mPFC receive other-relevant information from STS or ACC, respectively. The information of inhibitory neurons in TPJ is identical with the information of other-relevant stimuli in pSTS, and the information of inhibitory neurons in mPFC is identical with the information of other's belief in dmPFC. The temporary neurons in TPJ receive self-relevant information from STS, and the temporary neurons' information is identical with the information of self-relevant stimuli in IPL. The self-perspective inhibition takes place in TPJ and the self-belief inhibition takes place in mPFC. Then we test the effect of these connections in the false-belief task, and observe that the different maturation of connections leads to different permanence in the task. In this figure, the inhibitory neurons (In) are used to inhibit information in IPL or vmPFC, and the inhibitory result (InR) is the result of their interaction. If the connections between IFG and TPJ, IFG and vmPFC are mature, the activated neurons of InR will activate IFG to inhibit the information in InR neurons, and then make the correct information as the output of TPJ and mPFC. These connections will not influence the process of reasoning about self-belief.

5. DISCUSSION

In this section, we will discuss the characteristics of the model, the reasons why robot experiments and cognitive experiments are not completely consistent, and the possible mechanisms of why toddlers fail in high inhibition tasks.

Compared to the previous models which we introduce in the related works, our model explores and is fundamentally based on the role of self-experience. In our model, robots learn to understand object permanence and visual access of blindfold or turn around from self-experience, then use it to infer other's belief and predict their actions. All of the participant robots learn the ability of understanding object permanence from the same experience. In the Opaque-and-Transparent Blindfold Test, they are divided into opaque blindfold group and transparent blindfold group. Even though the visual inputs of both groups in the test stage are identical, the different experience with an opaque blindfold or transparent blindfold leads to different performances. In Turn Around Test, the different behaviors of the actor robot in the test stage result in different performances. Compared with the recently published work from Patacchiola and Cangelosi (2020): (1) Our model is based on spiking neural networks, and just uses the STDP to successfully reproduce the complex cognitive function of ToM, hence more biological plausible. And their model is based on an actor-critic (AC) framework, an epigenetic robotic architecture (ERA) and a Bayesian network (BN). (2) Our model considered more brain regions that have been consistently found in many experimental paradigms of ToM, such as the TPJ that used for self-other distinction, the IFG that used for self-perspective inhibition and self-belief inhibition, etc. (3) Our model is used for the false belief task, which is one of the most classical and widely used experimental paradigms of ToM, and their model challenges a different task. The two studies have complementary contributions to the ToM models through bio-inspired mechanisms.

Through the integration of biological inspirations and computational modeling, we suggest that the self-experience, maturation of correlated brain areas (e.g., calculation capability) and connection between brain areas (e.g., inhibitory control) will have great influence on the participant's performance in the false-belief task.

As indicated in Scott and Baillargeon (2017), the false belief tasks contain spontaneous-response and elicited-response tasks that belong to the implicit task and explicit task, respectively. The difference in spontaneous-response and elicited-response tasks is that the former investigates the capacity of false belief understanding by spontaneous behavior such as anticipatory-looking, preferential-looking, etc with a non-verbal task, and the latter investigates this capacity by answering direct questions that predict agent's behavior who has a false belief with the verbal task. Children can pass the spontaneous-response task before 2 years old, but they can not pass the elicited-response tasks until about 4 years old. Our tasks on robots are not completely consisted with Senju et al. (2011) on 18-month-olds infants and Southgate et al. (2007) on 2-year-olds infants. Both of them used spontaneous-response to test the infants

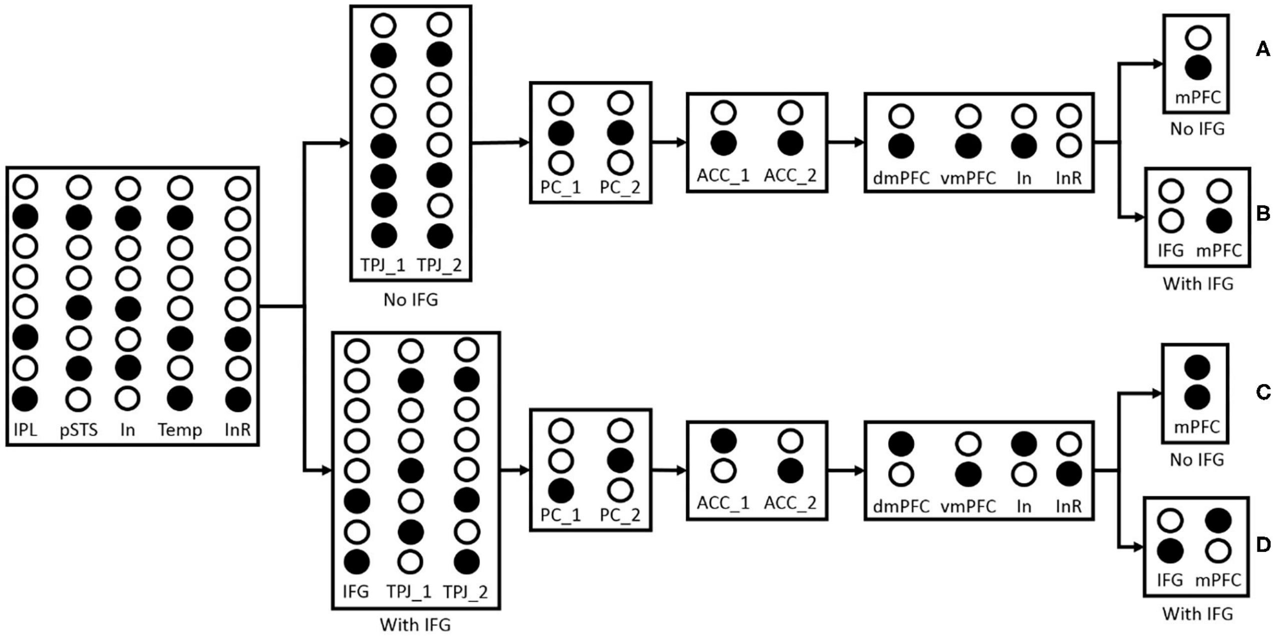


FIGURE 9 | The effect of IFG connection in the false-belief task. The solid circles indicate that the neurons have been activated. "No IFG" means the connection between IFG and TPJ or IFG and vmPFC is immature, and 'With IFG' means the connection between IFG and TPJ or IFG and vmPFC is mature. **(A)** Both of the connections between IFG and TPJ, IFG and vmPFC are immature. The first output of TPJ (TPJ_1) is different from IPL or pSTS because the self-perspective inhibition failed which caused by the immature connection between IFG and TPJ, and the second output (TPJ_2) is identical with IPL. The inferring visual access result of PC_1 and PC_2 in Precuneus/PCC are identical, and the belief reasoning result of ACC_1 and ACC_2 in ACC are identical. In the process of motor response, the InR neurons have not been activated because the belief in dmPFC and vmPFC are identical, so the output of mPFC is identical with dmPFC. The behavior of the participant robot to predict the actor robot's action is the same as the action caused by self-belief, so it failed in the false-belief task. **(B)** The connection between IFG and TPJ is immature, and the connection between IFG and vmPFC is mature. The participant robot also failed in this task because the result of inferring other's visual access is wrong which causes by the immature connection between IFG and TPJ. **(C)** The connection between IFG and TPJ is mature, and the connection between IFG and vmPFC is immature. The activated InR neurons activate IFG to inhibit self-perspective. The TPJ_1 and TPJ_2 could be output correctly. The participant robot could infer other's belief correctly, but it cannot inhibit the effect of self-belief without IFG. Both of the two candidate responses are activated when asked the question "Where is the ladybird according to the blue robot." The participant robot will tend to select action directed by self-belief and failed in the false-belief task. **(D)** Both of the connections between IFG and TPJ, IFG, and vmPFC are mature. With the connection to IFG, the participant robot will inhibit self-perspective in TPJ and inhibit self-belief in mPFC, then succeed in this task.

on whether they can pass the task. And in the test trial, they removed the object from the scene to make infants pass the task easier. In our task, we determine whether the robot can pass the task by detecting the direction of the finger which makes the results more intuitive. And in the test trial, we move the object to the other box. Setoh et al. (2016) found that 2.5-year-olds toddlers could succeeded in a traditional false-belief task with reduced processing demands. Toddlers could pass the elicited-intervention and low inhibition task (removing object from the scene) which is described by language and picture, but would fail in high inhibition task (moving the object to another box). They thought the reason why toddlers failed in high inhibition task is that toddlers cannot inhibit the response of the actual location of the object. We suppose that the core mechanism of belief reasoning is identical in both tasks, and the only difference should be in the process of motor response. In the motor response process of the elicited-intervention task, it may use the brain areas which control the hand movement. And in spontaneous-response task, it may use the brain areas which control the eye movement. The main reason why we use high inhibition task to replace low inhibition task is that the behavior of the robots in the true belief task is more intuitive to be understood (for the high inhibition task, the robot can point to the position of the object, while for the low inhibition task, the objects are removed outside the scene, and the robot cannot point to their positions).

And we suppose that the reason why toddlers failed in high inhibition task should be related to the lack of motor response ability rather than ToM (e.g., understanding that others have beliefs that are different from one's own). As shown in **Figure 9**, the connection between IFG and TPJ are matured but the connection between IFG and vmPFC are not, both of the two candidate responses are activated when the participant robot is asked "Where is the ladybird according to the blue robot." The participant robot will tend to select action directed by self-belief rather than randomly, as the result of behavior data shown in Setoh et al. (2016) and Samson et al. (2005). In the low inhibition task, the participant has no idea about the object's location and one of the two candidate motor responses are activated, so the participant robot can succeed in this task. And in Samson et al. (2005), the participant who has a lesion of the right inferior and middle frontal gyri performed well in low-inhibition false-belief task but failed in the high-inhibition task.

Here we don't attempt to compare our model with the traditional Theories of ToM as Theory Theory (Gopnik and Wellman, 1994), Simulation Theory (Gallese, 1998), etc., because what we focus on in this paper is to build a computationally feasible model which could uncover the detailed mechanisms of ToM and enhance our understanding of how the self-experience, maturation of correlated brain areas and connection between brain areas affect the participant's performance in the false-belief task.

6. CONCLUSION

The computational model for the robotic ToM is regarded as one of the Grand Challenges for Artificial Intelligence and Robotics (Yang et al., 2018). Here we proposed a Brain-ToM model based on existing biological findings of ToM, and this model shows its relevance to ToM of human from the mechanism and behavior perspectives.

In summary, we propose a Brain-ToM model to enable machines to acquire the ability of ToM through learning and inferring based on self-experience. We validate the model by deploying it on humanoid robots. Our model successfully enabled the robot to pass the false-belief task, which is a classical task designed to understand the nature and mechanisms of ToM from Cognitive Psychology. The model and its application on robots show that current understanding on the mechanisms of the ToM can be computationally unified into a consistent framework and enable the robots to be equipped with the initial cognitive ability of ToM.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/**Supplementary Material**, further inquiries can be directed to the corresponding author/s.

AUTHOR CONTRIBUTIONS

YZe conceived the initial idea. YZh and YZe designed the model, carried out, and analyzed the experiments. TZ, YZe, and DZ designed and implemented the SNN algorithm. FZ contributed to the inhibitory control mechanism. YZe, YZh, TZ, and EL wrote the manuscript.

FUNDING

This work was supported by the Strategic Priority Research Program of Chinese Academy of Sciences [Grant No. XDB32070100], the Beijing Municipality of Science and Technology [Grant No. Z181100001518006], the CETC Joint Fund [Grant No. 6141B08010103], the Major Research Program of Shandong Province [2018CXGC1503] and the Key Research Program of Frontier Sciences of Chinese Academy of Sciences [Grant No. ZDBS-LY-JSC013].

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2020.00060/full#supplementary-material

Movie S1 | Visual access learning of blindfold from self-experience (opaque blindfold group). This process enables the participant robot understand that other robots wearing a blindfold cannot see.

Movie S2 | Visual access learning of blindfold from self-experience (transparent blindfold group). This process enables the participant robot understand that other robots wearing a blindfold can see.

Movie S3 | Familiarization phase. This process enables the participant robot understand the goal of the actor robot.

Movie S4 | Opaque-and-transparent blindfold test (opaque blindfold group). The details in opaque blindfold group of blindfold test.

Movie S5 | Opaque-and-transparent blindfold test (transparent blindfold group). The details in transparent blindfold group of blindfold test.

Movie S6 | Turn around test (false-belief task). The details in false-belief task of turn around test.

Movie S7 | Turn around test (true belief task). The details in true belief task of turn around test.

REFERENCES

Abu-Akel, A., and Shamay-Tsoory, S. (2011). Neuroanatomical and neurochemical bases of theory of mind. *Neuropsychologia* 49, 2971–2984. doi: 10.1016/j.neuropsychologia.2011.07.012

Asakura, N., and Inui, T. (2016). A bayesian framework for false belief reasoning in children: A rational integration of theory-theory and simulation theory. *Front. Psychol*. 7:2019. doi: 10.3389/fpsyg.2016.02019

Bardi, L., Six, P., and Brass, M. (2017). Repetitive TMS of the temporo-parietal junction disrupts participant's expectations in a spontaneous theory of mind task. *Soc. Cogn. Affect. Neurosci*. 12, 1775–1782. doi: 10.1093/scan/nsx109

Bengio, Y., Lee, D.-H., Bornschein, J., Mesnard, T., and Lin, Z. (2015a). Towards biologically plausible deep learning. *arXiv [Preprint]. arXiv:1502.04156.*

Bengio, Y., Mesnard, T., Fischer, A., Zhang, S., and Wu, Y. (2015b). Stdp as presynaptic activity times rate of change of postsynaptic activity. *arXiv [Preprint]. arXiv:1509.05936.*

Berthiaume, V. G., Shultz, T. R., and Onishi, K. H. (2013). A constructivist connectionist model of transitions on false-belief tasks. *Cognition* 126, 441–458. doi: 10.1016/j.cognition.2012.11.005

Bi, G.-Q., and Poo, M.-M. (2001). Synaptic modification by correlated activity: Hebb's postulate revisited. *Annu. Rev. Neurosci*. 24, 139–166. doi: 10.1146/annurev.neuro.24.1.139

Bruce, S., and Muhammad, Z. (2009). The development of object permanence in children with intellectual disability, physical disability, autism, and blindness. *Int. J. Disabil. Dev. Educ*. 56, 229–246. doi: 10.1080/10349120903102213

Buckner, R. L., and Carroll, D. C. (2007). Self-projection and the brain. *Trends Cogn. Sci*. 11, 49–57. doi: 10.1016/j.tics.2006.11.004

Cavanna, A. E., and Trimble, M. R. (2006). The precuneus: a review of its functional anatomy and behavioural correlates. *Brain* 129(Pt 3), 564–583. doi: 10.1093/brain/awl004

Dan, Y., and Poo, M.-M. (2004). Spike timing-dependent plasticity of neural circuits. *Neuron* 44, 23–30. doi: 10.1016/j.neuron.2004.09.007

Denny, B. T., Kober, H., Wager, T. D., and Ochsner, K. N. (2012). A meta-analysis of functional neuroimaging studies of self- and other judgments reveals a spatial gradient for mentalizing in medial prefrontal cortex. *J. Cogn. Neurosci*. 24, 1742–1752. doi: 10.1162/jocn_a_00233

Dohnel, K., Schuwerk, T., Meinhardt, J., Sodian, B., Hajak, G., and Sommer, M. (2012). Functional activity of the right temporo-parietal junction and of the medial prefrontal cortex associated with true and false belief reasoning. *NeuroImage* 60, 1652–1661. doi: 10.1016/j.neuroimage.2012.01.073

Eddy, C. M. (2016). The junction between self and other? Temporo-parietal dysfunction in neuropsychiatry. *Neuropsychologia* 89, 465–477. doi: 10.1016/j.neuropsychologia.2016.07.030

Flavell, J. H., Flavell, E. R., and Green, F. L. (1983). Development of the appearance-reality distinction. *Cogn. Psychol*. 15, 95–120. doi: 10.1016/0010-0285(83)90005-1

Frith, C. D., and Frith, U. (1999). Interacting minds-a biological basis. *Science* 286, 1692–1695. doi: 10.1126/science.286.5445.1692

Frith, C. D., and Frith, U. (2006). The neural basis of mentalizing. *Neuron* 50, 531–534. doi: 10.1016/j.neuron.2006.05.001

Gallagher, H. L., and Frith, C. D. (2003). Functional imaging of "theory of mind". *Trends Cogn. Sci*. 7, 77–83. doi: 10.1016/S1364-6613(02)00025-6

Gallese, V. (1998). Mirror neurons and the simulation theory of mind-reading. *Trends Cogn. Sci*. 2, 493–501. doi: 10.1016/S1364-6613(98)01262-5

Georgopoulos, A., Schwartz, A., and Kettner, R. (1986). Neuronal population coding of movement direction. *Science* 233, 1416–1419. doi: 10.1126/science.3749885

Goodman, N. D., Baker, C. L., Bonawitz, E. B., Mansinghka, V. K., Gopnik, A., Wellman, H., et al. (2006). "Intuitive theories of mind: a rational approach to false belief," in *Proceedings of the Twenty-Eighth Annual Conference of the Cognitive Science Society* (Vancouver, BC), 1382–1387.

Gopnik, A., and Wellman, H. M. (1994). "The theory theory," in *Mapping the Mind: Domain Specificity in Cognition and Culture*, eds L. A. Hirschfeld and S. A. Gelman (Cambridge University Press) 257. doi: 10.1017/CBO9780511752902.011

Green, M. F., Horan, W. P., and Lee, J. (2015). Social cognition in schizophrenia. *Nat. Rev. Neurosci*. 16, 620–631. doi: 10.1038/nrn4005

Grosse Wiesmann, C., Schreiber, J., Singer, T., Steinbeis, N., and Friederici, A. D. (2017). White matter maturation is associated with the emergence of theory of mind in early childhood. *Nat. Commun*. 8:14692. doi: 10.1038/ncomms14692

Grossman, E. D., and Blake, R. (2002). Brain areas active during visual perception of biological motion. *Neuron* 35, 1167–1175. doi: 10.1016/S0896-6273(02)00897-8

Hamzei, F., Vry, M. S., Saur, D., Glauche, V., Hoeren, M., Mader, I., et al. (2016). The dual-loop model and the human mirror neuron system: an exploratory combined fMRI and DTI study of the inferior frontal gyrus. *Cereb. Cortex* 26, 2215–2224. doi: 10.1093/cercor/bhv066

Hartwright, C. E., Apperly, I. A., and Hansen, P. C. (2012). Multiple roles for executive control in belief-desire reasoning: distinct neural networks are recruited for self perspective inhibition and complexity of reasoning. *NeuroImage* 61, 921–930. doi: 10.1016/j.neuroimage.2012.03.012

Hartwright, C. E., Apperly, I. A., and Hansen, P. C. (2015). The special case of self-perspective inhibition in mental, but not non-mental, representation. *Neuropsychologia* 67, 183–192. doi: 10.1016/j.neuropsychologia.2014.12.015

Huang, Q., and Liu, X. (2017). Do infants have an understanding of false belief? *Adv. Psychol. Sci*. 25, 431–442. doi: 10.3724/SP.J.1042.2017.00431

Igelstrom, K. M., and Graziano, M. S. A. (2017). The inferior parietal lobule and temporoparietal junction: a network perspective. *Neuropsychologia* 105, 70–83. doi: 10.1016/j.neuropsychologia.2017.01.001

Jiang, Q., Wang, Q., Li, P., and Li, H. (2016). The neural correlates underlying belief reasoning for self and for others: evidence from ERPS. *Front. Psychol*. 7:1501. doi: 10.3389/fpsyg.2016.01501

Leech, R., and Sharp, D. J. (2014). The role of the posterior cingulate cortex in cognition and disease. *Brain* 137(Pt 1), 12–32. doi: 10.1093/brain/awt162

Leslie, A. M., and Polizzi, P. (1998). Inhibitory processing in the false belief task: two conjectures. *Dev. Sci*. 1, 247–253. doi: 10.1111/1467-7687.00038

Milliez, G., Warnier, M., Clodic, A., and Alami, R. (2014). "A framework for endowing an interactive robot with reasoning capabilities about perspective-taking and belief management," in *Ro-Man: the IEEE International Symposium on Robot and Human Interactive Communication* (Edinburgh), 1103–1109. doi: 10.1109/ROMAN.2014.6926399

Molenberghs, P., Johnson, H., Henry, J. D., and Mattingley, J. B. (2016). Understanding the minds of others: a neuroimaging meta-analysis. *Neurosci. Biobehav. Rev*. 65, 276–291. doi: 10.1016/j.neubiorev.2016.03.020

Moreau, N., Viallet, F., and Champagne-Lavau, M. (2013). Using memories to understand others: the role of episodic memory in theory of mind impairment in Alzheimer disease. *Ageing Res. Rev*. 12, 833–839. doi: 10.1016/j.arr.2013.06.005

Mossad, S. I., AuCoin-Power, M., Urbain, C., Smith, M. L., Pang, E. W., and Taylor, M. J. (2016). Thinking about the thoughts of others; temporal and spatial neural activation during false belief reasoning. *NeuroImage* 134, 320–327. doi: 10.1016/j.neuroimage.2016.03.053

Myowa-Yamakoshi, M., Kawakita, Y., Okanda, M., and Takeshita, H. (2011). Visual experience influences 12-month-old infants' perception of goal-directed actions of others. *Dev. Psychol*. 47, 1042–1049. doi: 10.1037/a0023765

O'Laughlin, C., and Thagard, P. (2000). Autism and coherence: a computational model. *Mind Lang*. 15, 375–392. doi: 10.1111/1468-0017.00140

Patacchiola, M., and Cangelosi, A. (2020). A developmental cognitive architecture for trust and theory of mind in humanoid robots. *IEEE Trans. Cybern*. 1–13. doi: 10.1109/TCYB.2020.3002892

Perner, J., Leekam, S. R., and Wimmer, H. (1987). Three-year-olds' difficulty with false belief: the case for a conceptual deficit. *Br. J. Dev. Psychol*. 5, 125–137. doi: 10.1111/j.2044-835X.1987.tb01048.x

Piaget, J., and Cook, M. (1952). *The Origins of Intelligence in Children, Vol. 8*. New York, NY: International Universities Press. doi: 10.1037/11494-000

Premack, D., and Woodruff, G. (1978). Does the chimpanzee have a theory of mind? *Behav. Brain Sci*. 1, 515–526. doi: 10.1017/S0140525X00076512

Rabinowitz, N. C., Perbet, F., Song, H. F., Zhang, C., Eslami, S., and Botvinick, M. (2018). Machine theory of mind. *arXiv [Preprint]. arXiv:1802.07740.*

Samson, D., Apperly, I. A., Kathirgamanathan, U., and Humphreys, G. W. (2005). Seeing it my way: a case of a selective deficit in inhibiting self-perspective. *Brain* 128(Pt 5), 1102–1111. doi: 10.1093/brain/awh464

Schurz, M., Aichhorn, M., Martin, A., and Perner, J. (2013). Common brain areas engaged in false belief reasoning and visual perspective taking: a meta-analysis of functional brain imaging studies. *Front. Hum. Neurosci*. 7:712. doi: 10.3389/fnhum.2013.00712

Schurz, M., and Perner, J. (2015). An evaluation of neurocognitive models of theory of mind. *Front. Psychol*. 6:1610. doi: 10.3389/fpsyg.2015.01610

Schurz, M., Radua, J., Aichhorn, M., Richlan, F., and Perner, J. (2014). Fractionating theory of mind: a meta-analysis of functional brain imaging studies. *Neurosci. Biobehav. Rev*. 42, 9–34. doi: 10.1016/j.neubiorev.2014.01.009

Scott, R. M., and Baillargeon, R. (2017). Early false-belief understanding. *Trends Cogn. Sci*. 21, 237–249. doi: 10.1016/j.tics.2017.01.012

Senju, A., Southgate, V., Snape, C., Leonard, M., and Csibra, G. (2011). Do 18-month-olds really attribute mental states to others? A critical test. *Psychol. Sci*. 22, 878–880. doi: 10.1177/0956797611411584

Setoh, P., Scott, R. M., and Baillargeon, R. (2016). Two-and-a-half-year-olds succeed at a traditional false-belief task with reduced processing demands. *Proc. Natl. Acad. Sci. U.S.A*. 113, 13360–13365. doi: 10.1073/pnas.1609203113

Southgate, V., Senju, A., and Csibra, G. (2007). Action anticipation through attribution of false belief by 2-year-olds. *Psychol. Sci*. 18, 587–592. doi: 10.1111/j.1467-9280.2007.01944.x

Tielin, Z., Yi, Z., Dongcheng, Z., and Mengting, S. (2018). "A plasticity-centric approach to train the non-differential spiking neural networks," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence* (New Orleans, LA), 1–8.

van der Meer, L., Groenewold, N. A., Nolen, W. A., Pijnenborg, M., and Aleman, A. (2011). Inhibit yourself and understand the other: neural basis of distinct processes underlying theory of mind. *NeuroImage* 56, 2364–2374. doi: 10.1016/j.neuroimage.2011.03.053

Vogeley, K., Bussfeld, P., Newen, A., Herrmann, S., Happe, F., Falkai, P., et al. (2001). Mind reading: neural mechanisms of theory of mind and self-perspective. *NeuroImage* 14(1 Pt 1), 170–181. doi: 10.1006/nimg.2001.0789

Vogeley, K., May, M., Ritzl, A., Falkai, P., Zilles, K., and Fink, G. R. (2004). Neural correlates of first-person perspective as one constituent of human self-consciousness. *J. Cogn. Neurosci*. 16, 817–827. doi: 10.1162/089892904970799

Wimmer, H., and Perner, J. (1983). Beliefs about beliefs: representation and constraining function of wrong beliefs in young children's understanding of deception. *Cognition* 13, 103–128. doi: 10.1016/0010-0277(83)90004-5

Yang, G.-Z., Bellingham, J., Dupont, P. E., Fischer, P., Floridi, L., Full, R., et al. (2018). The grand challenges of science robotics. *Sci. Robot*. 3, 1–14. doi: 10.1126/scirobotics.aar7650

Zeng, Y., Zhao, Y., and Bai, J. (2016). "Towards robot self-consciousness (i): Brain-inspired robot mirror neuron system model and its application in mirror self-recognition," in *International Conference on Brain Inspired Cognitive Systems* (Beijing), 11–21. doi: 10.1007/978-3-319-49685-6_2

Zeng, Y., Zhao, Y., Bai, J., and Xu, B. (2017). Toward robot self-consciousness (ii): Brain-inspired robot bodily self model for self-recognition. *Cogn. Comput*. 10, 307–320. doi: 10.1007/s12559-017-9505-1

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Neurorobotics

ORIGINAL RESEARCH
published: 20 October 2020
doi: 10.3389/fnbot.2020.591128

Perception-Action Coupling Target Tracking Control for a Snake Robot via Reinforcement Learning

Zhenshan Bing[1]*, *Christian Lemke*[2], *Fabric O. Morin*[1], *Zhuangyi Jiang*[1], *Long Cheng*[3]†, *Kai Huang*[3] *and Alois Knoll*[1]

[1] *Department of Informatics, Technical University of Munich, Munich, Germany,* [2] *Department of Informatics, Ludwig Maximilian University of Munich, Munich, Germany,* [3] *School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China*

Visual-guided locomotion for snake-like robots is a challenging task, since it involves not only the complex body undulation with many joints, but also a joint pipeline that connects the vision and the locomotion. Meanwhile, it is usually difficult to jointly coordinate these two separate sub-tasks as this requires time-consuming and trial-and-error tuning. In this paper, we introduce a novel approach for solving target tracking tasks for a snake-like robot as a whole using a model-free reinforcement learning (RL) algorithm. This RL-based controller directly maps the visual observations to the joint positions of the snake-like robot in an end-to-end fashion instead of dividing the process into a series of sub-tasks. With a novel customized reward function, our RL controller is trained in a dynamically changing track scenario. The controller is evaluated in four different tracking scenarios and the results show excellent adaptive locomotion ability to the unpredictable behavior of the target. Meanwhile, the results also prove that the RL-based controller outperforms the traditional model-based controller in terms of tracking accuracy.

Keywords: snake robot, target tracking, reinforcement learning, motion planning, visual perception

OPEN ACCESS

Edited by:
Fei Chen,
Italian Institute of Technology (IIT), Italy

Reviewed by:
Xingdong Li,
Northeast Forestry University, China
Xiao Huang,
Beijing Institute of Technology, China

****Correspondence:***
Zhenshan Bing
bing@in.tum.de

†Present address:
Long Cheng,
College of Computer Science and Artificial Intelligence,
Wenzhou University, Wenzhou, China

Received: *03 August 2020*
Accepted: *15 September 2020*
Published: *20 October 2020*

Citation:
Bing Z, Lemke C, Morin FO, Jiang Z, Cheng L, Huang K and Knoll A (2020) Perception-Action Coupling Target Tracking Control for a Snake Robot via Reinforcement Learning. Front. Neurorobot. 14:591128. doi: 10.3389/fnbot.2020.591128

1. INTRODUCTION

Inspired by real snakes, snake-like robots are designed as a class of hyper-redundant mechanisms in order to achieve the agility and adaptability of their biological counterparts. Their long and narrow bodies with many degrees of freedom (DOF) enable them to perform diverse tasks that could never be carried out by other kinds of mobile robots, such as search and rescue in disaster scenes (Evan, 2017), complex teleoperation in space (Walker and Hannan, 1999), and even minimally invasive cardiac surgery (Webster et al., 2006). However, this high level of flexibility also corresponds to a complex control task involving the internal regulation of body joints and external interaction with the ground, in which model-based methods usually fail to control the robots adaptively in a dynamically changing environment.

Vision-guided locomotion, as one of the essential skills for moving in changeable scenarios, is a must-have capability for snake-like robots, to ensure that they can be deployed in an unattended environment by human operators. With the help of the visual information, the snake-like robots can solve more complex and realistic tasks, such as target tracking and obstacle avoidance. Especially in field operations, such as disaster rescue tasks and surveillance tasks, the target tracking capability can greatly improve the performances of the snake-like robots. Yet this kind of locomotion control still remains a challenging task for snake-like robots, since it involves not only the locomotion but also the target information obtained by cameras. For controlling the locomotion, different types of methods have been studied including the sinusoid-based methods (Hirose, 1993), CPG-based methods (Bing et al., 2017), and the dynamics-based methods (Miller, 1988). However, none of these methods can be used directly to perform vision-based tracking tasks. Such tracking tasks require the robot to be agile and adapt to their target trajectories with unpredictable changes in velocity or direction of travel, which is extremely challenging for traditional model-based methods. Our proposal is completely different as it tackles the object tracking and camera control pipeline in an end-to-end manner, based on reinforcement learning.

Strategies based on Reinforcement Learning (RL) are promising solutions for performing target tracking for a snake-like robot. This is because a RL-trained controller can take the visual image directly as the input, while simultaneously fully exploring the locomotion capabilities compared with model-based methods. This is particularly suitable for snake-like robots with redundant degrees of freedom. Although RL-based methods have been adopted to control mobile vehicle platforms (Morbidi and Mariottini, 2013; Yun et al., 2018; Luo et al., 2019), the effectiveness of such methods for generating agile steering motions for snake-like robots has, nevertheless, not yet been studied extensively, especially when interacting with the environment. The reasons are 2-fold. First, the steering control for snake-like robots is complex, especially when it comes to the sudden change in velocity or direction of travel, as this requires the coordination of bodies with redundant degrees of freedom from one moving pattern to another in a short time. Second, when traditional methods are used on mobile platforms, target tracking is usually divided into tracking and control sub-tasks, which makes it difficult to tune the pipeline jointly, especially considering the aforementioned motion barrier for snake-like robots. To cope with this hard-to-predict tracking and movement complexity, the RL-based control strategies need to map the visual inputs to the joint space directly, in order to perform the corresponding motions, and must operate with adequately defined reward functions to train a policy successfully. Hence, the design of a target tracking controller for snake-like robots based on RL is challenging.

To design a target tracking controller for snake-like robots, this paper proposes a RL-based strategy. Our main contributions to the literature are summarized as follows. First, we offer a novel alternative to solving the target tracking task for a snake-like robot via reinforcement learning. The learned policy directly transforms the external and internal observations to a sophisticated motion pattern for performing perception-action coupling tasks. It is worth to note that this RL-based method can be applied in different types of snake-like robots instead of designed solely for the one used in this work. Second, we propose a customized reward function that takes contiguous distances into calculation. With this reward design, the learned locomotion successfully solves adaptive target tracking tasks, and, more surprisingly it also learns the ability to keep the target within its visual field, even though this behavior is not specifically rewarded. Third, we propose a tracking accuracy metric that takes both the distance and direction into consideration. Based on this metric, we demonstrate that the learned locomotion outperforms the model-based locomotion in terms of tracking accuracy.

2. RELATED WORK

As our work is related to the perception-driven locomotion of snake-like robots and perception-driven algorithms based on reinforcement learning, we briefly review the state-of-the-art research on both aspects in the following.

2.1. Vision-Based Snake-Like Locomotion

Trajectory or target tracking of snake-like robots is important and operators usually control their locomotion by indicating the expected direction of its head module (Kamegawa et al., 2004; Fukushima et al., 2012; Yamada et al., 2013; Tanaka and Tanaka, 2015). Under the velocity constraints, which prevent the body from slipping sideways, trajectory tracking locomotion control of snake-like robots has been investigated (Matsuno and Mogi, 2000; Prautsch et al., 2000; Transeth et al., 2007; Ishikawa et al., 2009; Tanaka and Matsuno, 2014). Liljeback proposed a straight line path-following controller of a planar snake-like robot under the Line-of-Sight (LOS) guidance law, but the robot's head could not track the desired trajectory (Liljeback et al., 2011). Matsuno derived a dynamic model to avoid the singular configuration of the robot body. Using this control law, their snake robot can accomplish trajectory tracking of the head module without converging to a singular configuration (Matsuno and Sato, 2005). However, their results were only tested on a sinusoid-like track and this dynamics-based method may not adapt to unknown scenarios with changing dynamics. Similar ideas can also be found in Tanaka et al. (2015) and Huang et al. (2017). Xiao performed autonomous locomotion in a known scene and the positions of the snake-like robot and the obstacles were acquired from external web cameras (Xiao et al., 2015). This method, in fact, is a trade-off idea since they could not use the built-in camera in the snake-like robot due to its undulating locomotion pattern. In fact, there are few research efforts about the onboard vision-based locomotion control of snake-like robots, since the undulation of the body cannot provide a stable base for vision sensors. Bing et al. (2019a) proposed an end-to-end target tracking for snake-like robots based on spiking neural network. However, the network controller only outputs the steering signals and the locomotion itself is further generated with model-based methods. The robot "IRS Souryu" equipped with a ToF camera and 3D range sensors performs real-time localization and mapping tasks in a rubble environment (Ohno et al., 2006). A semi-auto snake-like robot with a B/W camera and a structured light sensor was investigated to perform a localization task of a pole, navigate and then climb it (Ponte et al., 2014). A slithering gait that specially designed for snake-like robots to perform target tracking tasks is introduced in Bing et al. (2017). A detailed survey about perception-driven and obstacle-aided locomotion for snake robots can be found in Sanfilippo et al. (2017).

2.2. RL-Based Tracking

As a principle approach to temporal decision-making problems, RL-based approaches have been used for solving visual object tracking tasks that aim at finding the target position in contiguous frames and whereby steering the locomotion of an mobile agent. Gaskett designed a mobile robot that can perform visual servoing and wandering behaviors through a Q-learning method (Gaskett et al., 2000; Garcia-Aracil et al., 2011). The work clearly demonstrated that a direct mapping from image space to actuator command using RL is a promising method. Similar work is also given in Asada et al. (1996), Takahashi et al. (1999), Busquets et al. (2002), and Hafner and Riedmiller (2003). Miljkovic presented a novel intelligent visual servo controller for a robot manipulator using RL (Miljković et al., 2013). Based on their control scheme, the performance of the robot is improved and is able to adapt to the changing environment. In the recent ViZDoom platform for visual reinforcement learning, they provided two basic scenarios: a basic move-and-shoot bot and a maze-navigation bot. Yun developed a reinforcement learning scheme to utilize labeled video sequences for training their action-driven tracker (Yun et al., 2018). In Ding et al. (2018), a partial RL based tracking algorithm was proposed to achieve adaptive control of a wheeled mobile robotic system working in highly complex and unpredictable environment. The controller required less calculation time than other optimization technologies and exhibited higher accuracy at the same time. As far as we are aware, to date, there have been no studies that employ reinforcement algorithm to control snake-like robots for performing vision-based locomotion, except for one of our previous research that used RL to design energy-efficient gaits (Bing et al., 2019b).

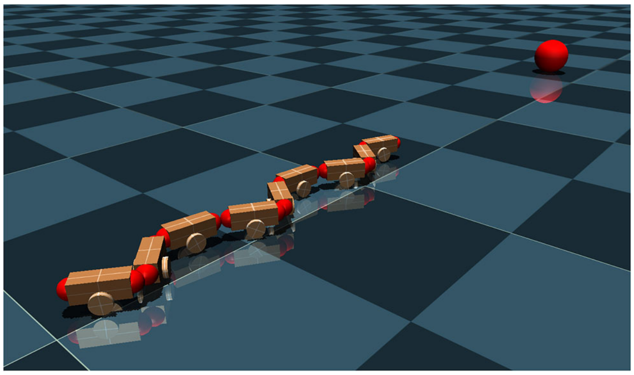


FIGURE 1 | Target tracking scene for a snake-like robot.

3. MODELS AND TASKS

This section first introduces the models of the snake-like robot. Then, the target tracking task is presented, together with the target tracking metric for evaluating the performances of different algorithms.

3.1. Models

The target tracking scene is modeled and simulated in MuJoCo (Todorov et al., 2012), in which a red ball is used as the target and a snake-like robot is the tracker (see **Figure 1**). The easily detectable sphere has a radius of 0.2 *m* and is placed in front of the snake at the distance of 4 *m*. The robot is inspired by the *ACM* snake-like robot (Hirose, 1993), which uses eight joints and nine identical modules. A head camera is used as the visual system for the snake-like robot, which is positioned in the center

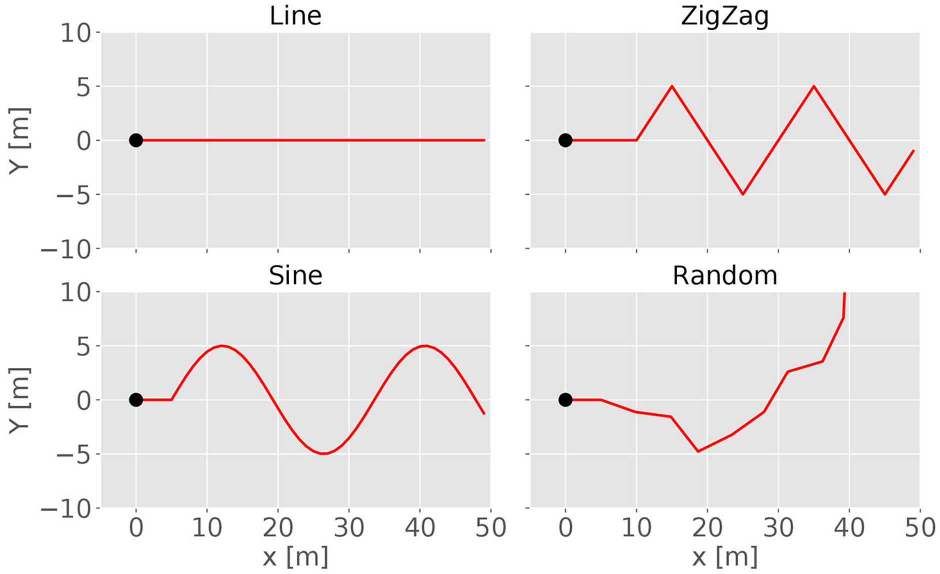


FIGURE 2 | Each of these four diagrams visualizes one of the testing tracks. The tracks define the trajectory of the target. The black dots indicate the start positions of the track. Only a limited length is displayed because the tracks are continuing. The first three figures show the *line*, *zigzag*, and *sine* track, respectively. The last sub-figure gives an example of a *random* track.

of the first head module. Due to the camera position and view volume, the ground area in front of the robot is clipped during rendering. The purpose of this camera is to obtain information of the moving target. More details about the model of the robot can be found in Bing et al. (2019b).

3.2. Tasks Description

In our target tracking task, the snake-like robot has to follow a target while maintaining a certain distance from it. The target moves with a specified velocity $v_t = 0.3\ m/s$ on the trajectories of random tracks with constrained conditions. The random-tracks consists of short straight forward sections linked by abrupt random direction changes with angles between $-60°$ and $60°$. A random seed is used to generate arbitrarily random tracks during the training process.

For evaluating the performances of the controller, we also design four predefined tracks for testing as shown in **Figure 2**. The *line* track is used to test a simple forward locomotion and is therefore the easiest. For testing the steering behavior, the *wave* track offers a continuous curve where the robot has to alternately change its steering direction. A modified sinus wave defines the *wave* track. The *zigzag* and *random* track scenarios test the robot's capability to handle abrupt directional changes. The *zigzag* track is defined by alternating abrupt left and right turns of 60°. The *random* track consists of short straight forward sections linked by abrupt random direction changes with angles between $-60°$ and $60°$. Heess et al. (2017) described that starting with easier tasks for training supports a faster learning process in RL. Thus, all the tracks have a short straight segment at the beginning, which enhances the learning process because it is easier to move straight forward at the beginning of a training process.

3.3. Tracking Metrics

As illustrated in **Figure 3**, the location of the target at time t can be represented as $\mathbf{X}_T(t) = (x_T(t), y_T(t))$ in the global frame of reference $\mathcal{R}_\mathcal{G}$. Similarly, let the position of the snake-like robot at time t be denoted by $\mathbf{X}_F(t) = (x_F(t), y_F(t))$. For simplicity, we discretize the time with steps of $\delta_t = 0.01\ ms$ and use the notation n to refer to the n^{th} time step. Let $d(n) = ||\mathbf{X}_T(n) - \mathbf{X}_F(n)||_2$ be the distance between the target and the robot at time-slot n, where $||\cdot||_2$ denotes the L_2 norm. In addition, $\theta_T = \arctan\frac{y_T(t)}{x_T(t)}$ represents the global angle of the target and $\theta_F = \arctan\frac{y_F(t)}{x_F(t)}$ is the angle of the head module in $\mathcal{R}_\mathcal{G}$. Then, the absolute relative angle between the head and the target ϕ_t can be calculated as

$$\phi_t = |\theta_T - \theta_F| \tag{1}$$

An expected tracking is to minimize the distance difference from $d(n)$ to desired distance d_t and aims at the target in the middle of the field of view (FoV) to greatest extent. Thus, the Averaged Tracking Error (*ATE*) is defined as

$$ATE = \sum_{i=0}^{N}\left(\frac{|d(n) - d_t|}{d_t} \times \frac{\phi_t(n)}{\phi_{t_{max}}}\right) \times \frac{1}{N}, \tag{2}$$

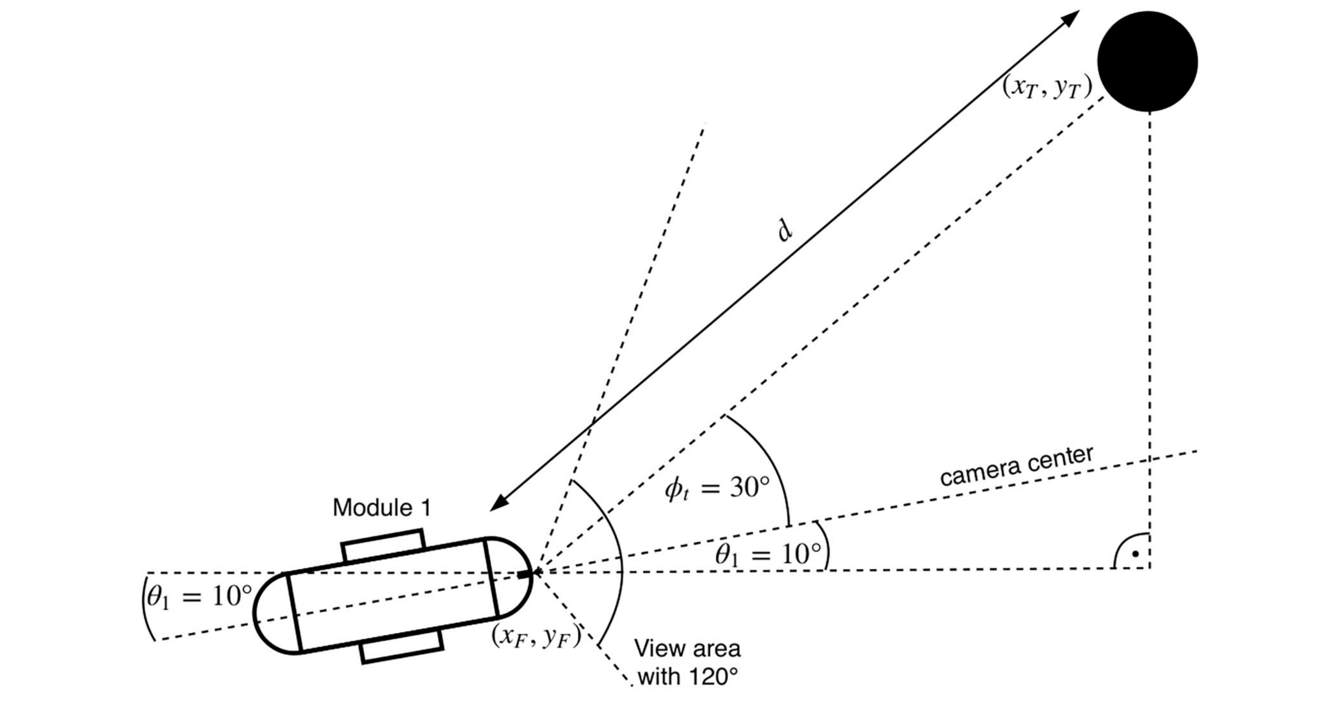


FIGURE 3 | An overview of the used angles between the camera and the target. It shows the first module and the target with their corresponding relative angles from a top view perspective. Here, the head to target angle ϕ_t has a value of 30°. The global angle of the head θ_F is 10°. The target position is at (x_T, y_T) and has a Euclidean distance d to the camera position (x_F, y_F).

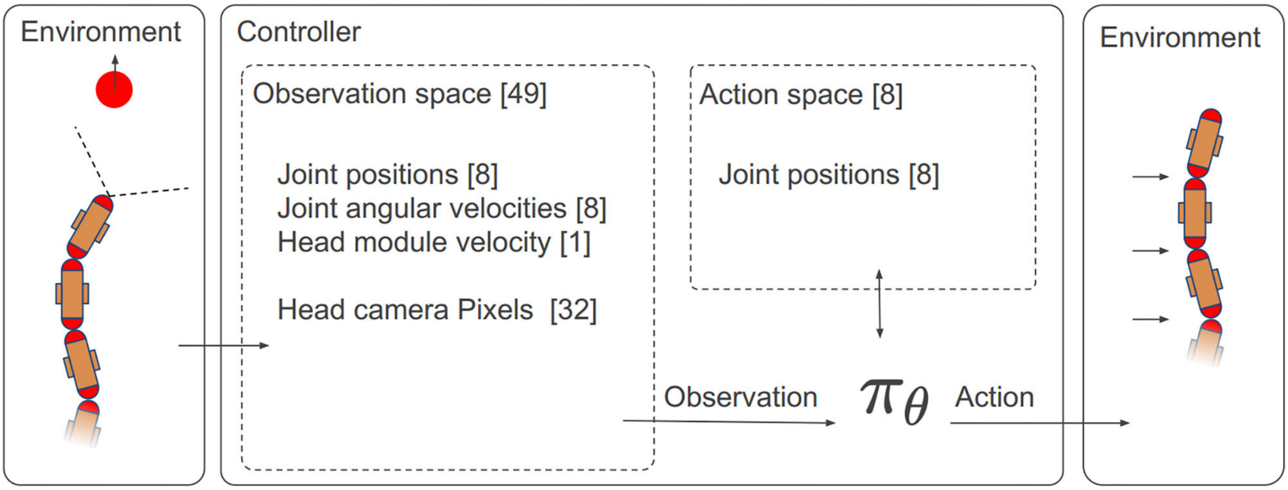


FIGURE 4 | The overall architecture of the RL controller with the details of the observation space and the action space.

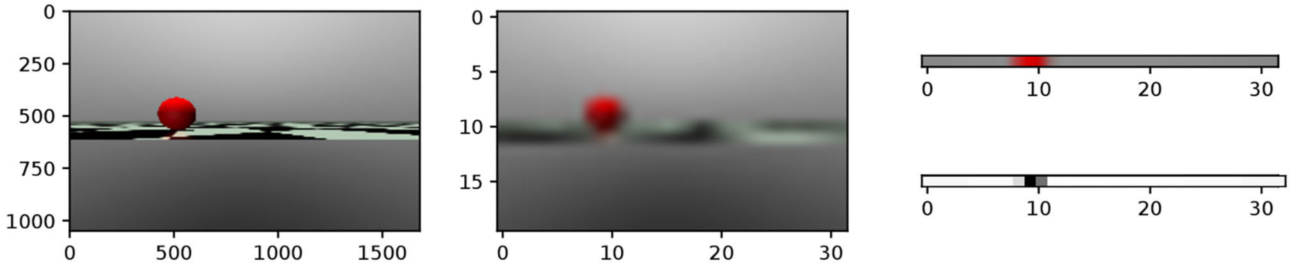


FIGURE 5 | This figure shows the visual information of the target at the distance of 4.0 m with 30° on the left. The left figure is the original image with 1, 685 × 1, 050 from the sensor. The middle figure is rendered with 32 × 20 pixels. The right array of pixels from the 10th row is used to estimate the relative position and distance of the target.

TABLE 1 | The observation space $\mathbf{o}_i^t$ of the controller.

Symbols	Descriptions
α_{1-8}	Relative joint angular positions
$\dot{\alpha}_{1-8}$	Relative joint angular velocity
v_1	Absolute head module linear velocity
$p_{10,1-32}$	Pixel 1–32 of the 10th row of the camera image

where N is the amount of the time steps. $|d(n) - d_t|$ calculates the absolute distance error in the normal direction. $\frac{1}{d_t}$ is used to calculate the error ratio against the desired target. $\frac{\phi_t(n)}{\phi_{t_{max}}}$ indicates the target's deviation in the FoV, where $\phi_{t_{max}} = 60^\circ$.

It should be noted that the tracking metric *ATE* is not used the reward function for our RL controller, since the reward signal only depicts a desired behavior that keeps a distance with the target while the tracking metric specifically measures the tracking accuracy. For RL tasks, it is more realistic to simply use an intuitive reward instead of regularizing it with a specific metric.

4. BASELINE EXAMPLE

This section briefly explains the start-of-the-art and widely used method for controlling the locomotion of snake-like robots, which is the model-based *gait equation* controller.

The *gait equation* method is a kinematic locomotion controller that describes the gaits using a sinusoid-like wave. This method was first proposed as the *serpentine curve* (Hirose, 1993) by Hirose who gained inspiration from real snakes. In this work, an undulation gait equation developed in Tesch et al. (2009) is used for the purpose of comparison. The *gait equation* controller is modeled as

$$\alpha(m, t) = (\frac{m}{M}x + y) \times A \times \sin(\omega t + \lambda m) + C\,. \qquad (3)$$

$\alpha(m, t)$ presents the joint angle value at time t, where m is the joint index and M is the joint amount. λ and ω are the spatial and temporal frequency of the movement, respectively. The spatial frequency represents the cycle numbers of the wave and the temporal frequency represents the traveling speed of the wave. A is the serpentine amplitude and x and y are the constants for shaping the body curve. C is the amplitude bias for steering the direction of a slithering locomotion.

The target tracking locomotion for the *gait equation* controller is divided into two sub-tasks, namely, lateral localization and speed control. Similar ideas can be also found in Bing et al. (2017) and Bing et al. (2019a). In the FoV of the robot, the target will be identified as a group of red pixels. For the lateral localization control, the moment of the red pixels is calculated and then used as the control target for a proportional integral (PI) controller, since it indicates the relative position of the target in the FoV of the robot. For the speed control, the number of the red pixels are counted to represent the distance from the robot to the target. In order to have a more accurate estimation, the visual image is rendered with a higher resolution 128 × 80 × 3. This is because a higher resolution can generate more amounts of red pixels for the same target and then result in a more accurate control performance. Due to the page limit, the implementation details of the *gait equation* controller will not be further explained. To ensure a fair comparison, we make many attempts to find the best control parameters to optimize the performance. But only the best tracking results are selected as the baseline example for further usage.

5. PROPOSED RL-BASED CONTROLLER

This section presents the details of the proposed RL-based controller, including the construction of the network and the training configuration and results.

5.1. Reinforcement Learning Setup

The most important components of a RL controller are the observation space, the action space, and the reward function. The overall architecture is shown in **Figure 4**.

5.1.1. Observation Space

The snake-like robot solely use a RGB vision sensor to track the target. Due to the undulation of the locomotion, the rendered image from the robot keeps shifting in the horizontal direction. In order to reduce unnecessary dimensions and enhance the computing efficiency, the following steps describe the image processing pipeline:

1. The image is directly rendered with 32 × 20 × 3 pixels. The middle figure in **Figure 5** shows an example of the rendered RBG image with 32 × 20 × 3 resolution.
2. The 10th row is extracted from the rendered image, since this line contains the pixels at which the target is located, as shown at the top right of **Figure 5**.
3. The color space is then transformed from RGB to gray with values in the range of $[0, 1]$ based on the intensity of the red pixel, as shown at the bottom right of **Figure 5**.

The resulting 32 pixels $p_{10,1-32}$ contain information about the target's relative position. Furthermore, its distance can also be estimated by the gray value intensities of the target's corresponding pixels. This value increases when the target is closer and decreases when the target is further away. These changes are caused by the illumination of the bright background. The further away the target is, the smaller it is rendered, resulting in a lower pixel count.

The controller also has to control the locomotion of the snake-like robot. To be able to propel forward, it performs a lateral

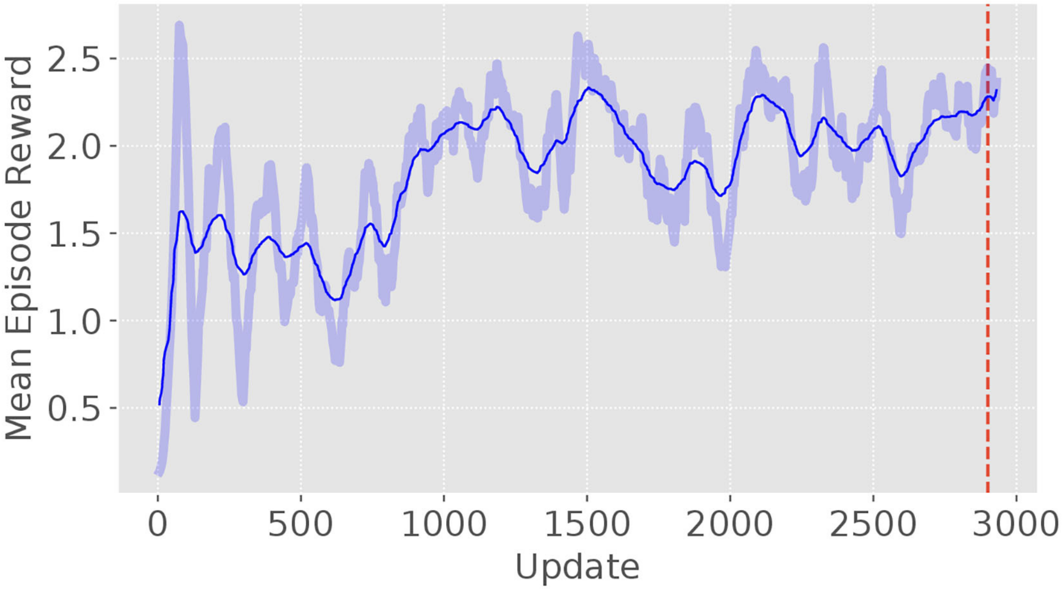


FIGURE 6 | The learning curve of the autonomous target tracking model. It is trained with 3 million time steps with 1,000 time steps per episode and 1,024 time steps per update.

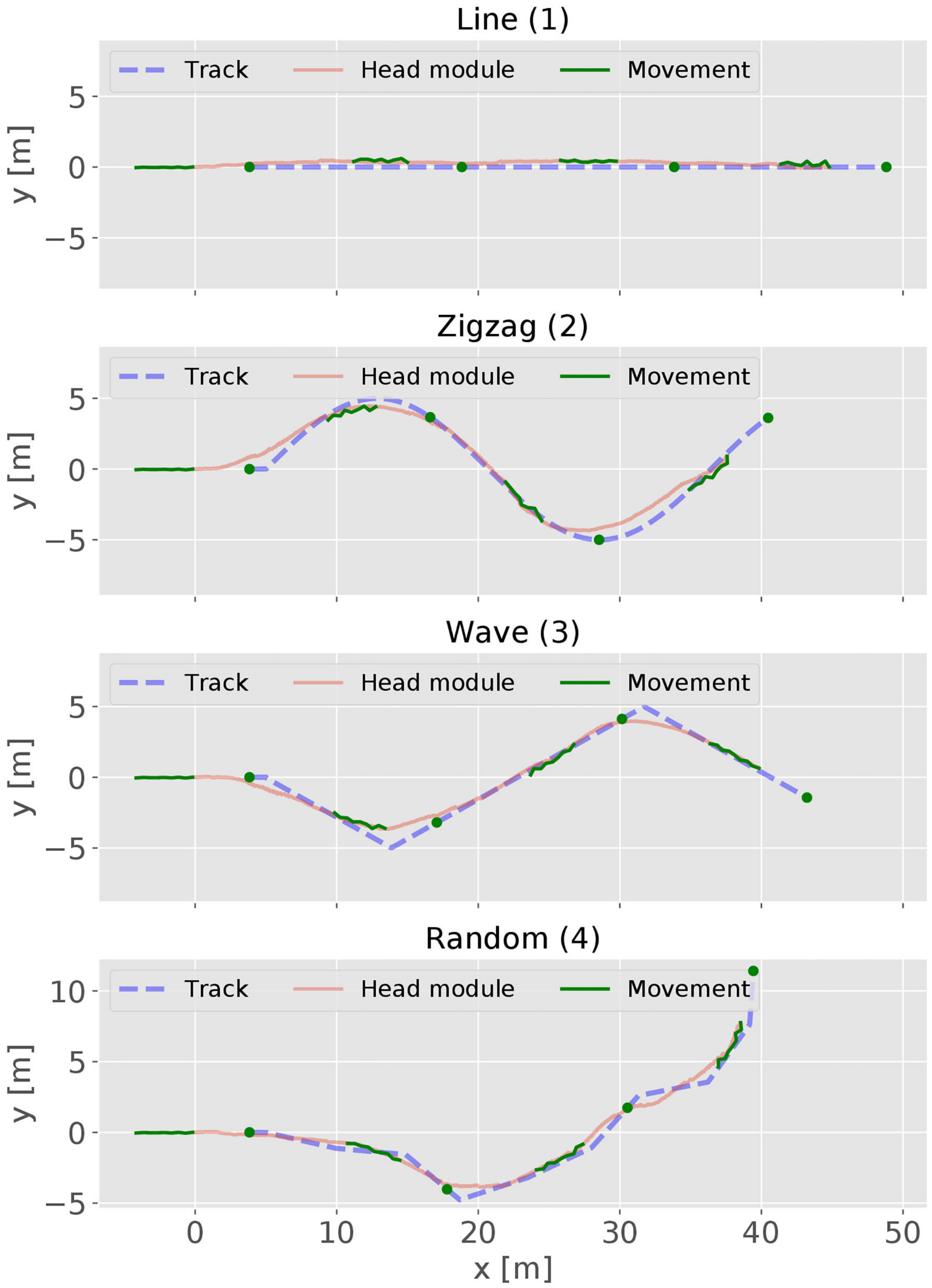


FIGURE 7 | The trajectories of the snake-like robot and the testing tracks. In addition, the body curve of the snake-like robot and the target position are added and captured every 1,000 time step.

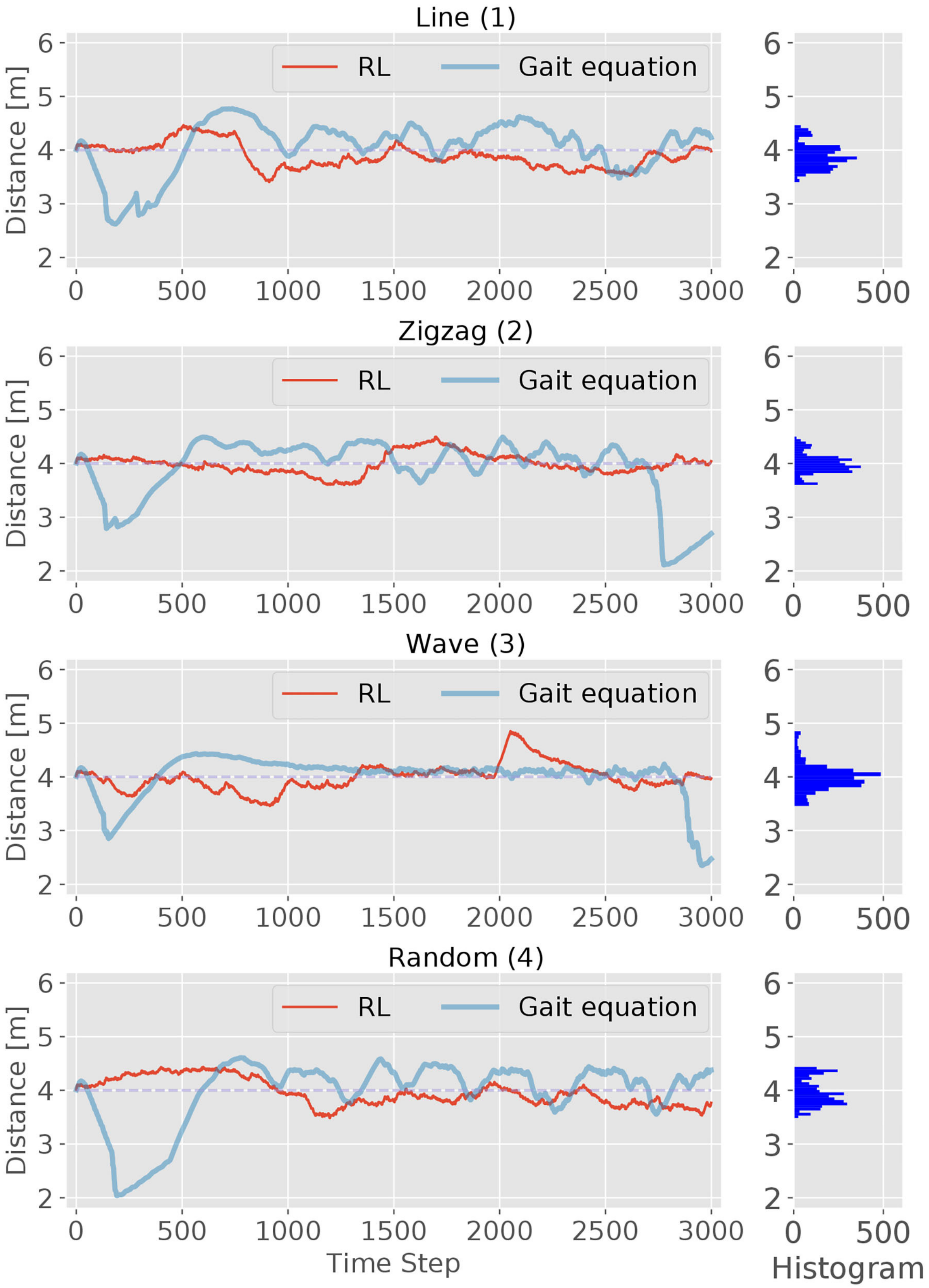


FIGURE 8 | The performance of the RL and *gait equation* controller in maintaining a certain distance from the target is shown in this figure. Each diagram represents one episode run on a track with 3,000 time steps. The target distance is set to 4.0 m and the range limits are set at 2.0 and 6.0 m. The histogram on the right of each diagram shows the distance distribution of the RL controller. Overall the distance varies in an adequate range around the target distance.

undulation motion pattern with its joints. At each time step, it receives the current joint angular positions α_{1-8} and the angular joint velocities $\dot{\alpha}_{1-8}$ to learn the locomotion and represent the proprioceptive awareness of the robot. In addition to these parameters, the controller also receives the global head module velocity along the moving direction v_1. It has been observed that this helps it to estimate its global velocity. This is required to control the velocity of the robot. In summary, an overall 49-DOF observation space is used in this work, as shown in **Table 1**.

5.1.2. Action Space

The action space $\mathbf{a}_i^t$ of the RL controller corresponds to the eight joint positions of the robot, which linearly map finite continuous values in the range of $[-1.5, 1.5]$ to $[-90^\circ,\ 90^\circ]$.

5.1.3. Reward Function

In the target tracking task, the snake-like robot follows a moving target while maintaining a specified distance d_t from it. Meanwhile, the robot should also try to maintain the target in the center of its FoV. Therefore, the distance-keeping and lateral localization in the FoV are the two criteria to find a successful behavior.

Thus, a distance-keeping reward is designed. Let $d_r = 2\ m$ define the distance radius from $t_d = 4\ m$ to its maximum distance $d_{max} = 6\ m$ and minimum distance $d_{min} = 2\ m$. In this approach, the reward represents the distance change between the head module and the target. The less distance changed toward the target distance d_t, the higher the reward. Similarly, the lower reward is received by increasing or decreasing the distance from the target distance d_t. The reward function is defined as follows:

$$\begin{aligned} r_d &= (1 - \frac{|d_t - d_{\text{after}}|}{d_r}) - (1 - \frac{|d_t - d_{\text{before}}|}{d_r}) \\ &= \frac{|d_t - d_{\text{before}}| - |d_t - d_{\text{after}}|}{d_r} \end{aligned}$$

Here, d_{before} defines the distance before the action, whereby the distance after the action is defined as d_{after}. Note that the distance change can also be denoted as velocity, since the measurements are time dependent. The term $|d_t - d|$ calculates the absolute distance difference between d_t and the current head position. The resulting normalization $\frac{|d_t - d|}{d_r}$ is 0, if $d = d_t$ and 1 if $d = d_t \pm d_r$. This effect is inverted by $1 - \frac{|d_t - d|}{d_r}$. As a result, the maximum reward of 1 is achieved if $d = d_t$.

In the task of autonomous target tracking, it is important to maintain vision of the target in order to react to its movement changes. It is worth noting, different from the evaluation metric defined in (2), that the reward function does not explicitly reward that behavior. The agent has to learn independently that it must observe the target's position in order to follow it.

5.2. Training

In order to map the input (observation $\mathbf{o}_i^t$) and the output (action $\mathbf{a}_i^t$), a fully connected 2-hidden-layer neural network is constructed as an approximator to the policy π_θ. The input layer and the output layer share the same dimensions as the observation space $\mathbf{o}_i^t$ and action space $\mathbf{a}_i^t$. The proximal policy optimization (PPO) algorithm adapted from is used to train the network, since PPO performs better on continuous action space tasks while being much simpler to implement and tune (Schulman et al., 2017). We train our policy network on a computer with an i7-9750H CPU and a Nvidia RTX 2070 GPU.

The model is trained by using the random track with a changing random seed for every episode. Therefore, a variety of tracks are generated and the model will not overfit to a specific track. This is necessary because it has been observed that the model tended to overfit while training on unvarying trajectories. As result for overfitting, the controller was not be able to adapt to other trajectory patterns. Based on the learning curve, a total of 3 million time steps (about 3,000 updates) were used for training (see **Figure 6**). The training process will terminate itself either when the target is out of view or reaching the end of the total time-steps. The mean reward gradually increases and levels up at around 2.4 with some fluctuation. This is because performance of the controller varies from the randomly changing track for each episode. The model at update 2,900 was selected for the further usage.

6. RESULTS AND DISCUSSIONS

This section will first describe the performance of the gaits generated by the RL controller in testing tracks. Then, we compare our gaits to the gaits generated from traditional model-based method in terms of tracking accuracy. Finally, we will give the limitation of the simulated results.

6.1. Results

The performance of the RL controller was tested on four different tracks (see **Figure 2**). For evaluation, the episode length is set to $3,000$ time steps. The trajectories of the head module of robot during the evaluation are shown using red solid lines in **Figure 7**, together with its corresponding track pattern (blue dash lines). In addition, the body curves are plotted every $1,000$ time step with the target position at that time using green lines and dots. For all four tracks, the RL controller was able to successfully follow the target. By comparing the trace of the snake-like robot and the track, a variation is observed in which the trajectories are not matched to each other. In some sections, they go in parallel or cross each other. This indicates that the snake-like robot is not heading directly for the target's position. In some cases, the head module's trace takes a shortcut in the curves of the target's track. However, the trace of the head module is maintaining a visible minimum distance. We can thus conclude that the controller performs a

TABLE 2 | The statistics for the head to target distances (unit: meters).

Track	Mean	Std	Min	Max
Line	3.99	0.18	3.60	4.49
Random	3.96	0.24	3.48	4.43
Wave	3.99	0.24	3.46	4.85
Zigzag	3.88	0.22	3.40	4.46

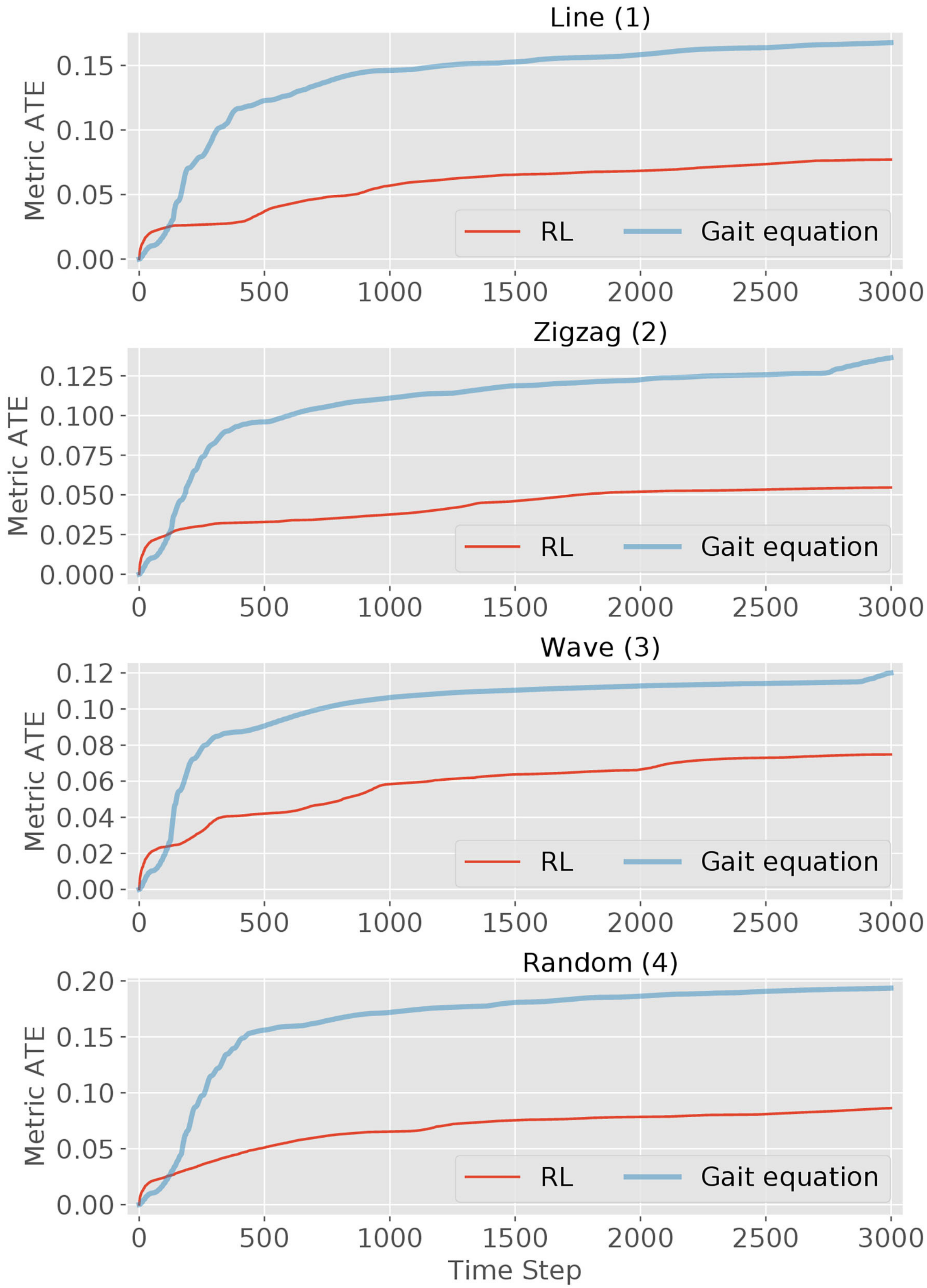


FIGURE 9 | The tracking metric curves of the RL controller and *gait equation* controller over the elapsed time (3,000 timestep in total) in four scenes. The tracking metric is defined in (2).

successful path-following behavior. Besides, the controller had to maintain a certain distance to the target. The red lines and the histogram in **Figure 8** show the distance distribution of runs on all tracks. In all runs, the distance varies around the target of 4.0 m which is measured at the center of the head. The oscillation movement of the head causes constant minimal distance changes. **Table 2** shows the statistics of the runs. As result, the controller was able to maintain the distance from the target with an adequate variance.

6.2. Comparisons

In order to evaluate the performances of the RL controller against the model-based method, the *gait equation* controller (section 4) was also executed on those four testing tracks.

We first compare the performance of the tracking accuracy. **Figure 8** shows the traces of the distance between the target and the robot for both controllers. In general, we can observe that the RL controller has a better tracking accuracy than the *gait equation* controller. For the RL controller, it exhibits better performances at the beginning process on all different tracks and keeps the distance very close to the desired value. Then with the accumulated error, all these four figures reach a relatively large error at some point, but then correct its direction to the right course. For the RL controller, the lag of tracking is much more smaller. After the starting of the movement, RL-based controller also exhibits better tracking accuracy. For the *gait equation* controller, it deviates most at the beginning of the movement for all four tracks. This is because the controller needs to response to the changing visual information and the effect will only show after the error has been accumulated for a period of time.

The second performance indicator is the tracking metric *ATE* defined in (2). The *ATE* is plotted over time steps in **Figure 9**. It can be observed that for both controllers, the metric curve gets higher due to the accumulated error with time passing by. But for all four scenes, the RL controller outperforms the *gait equation* controller: by around 50% in the simple *line* and *wave* scene and by 70% in the other difficult scenes. In conclusion, the RL controller outperform the *gait equation* controller both in terms of distance tracking accuracy and the averaged tracking error.

Since our training will terminate itself once the robot lose sight of the target, there will be no target recovery behavior obtained during training, such as retaining. In fact, the target recovery behavior is associated with some memory-like function that can predict the motion tendency of a moving object.

6.3. Limitations

It is worth noting that, our RL-based controller is demonstrated by simulations now and has not been support with results of physical snake-like robot yet. In order to ensure the validity of the simulated results, we first try to close the simulation-to-reality gap by setting simulation parameters with real-world properties (e.g., dimension, density, friction, etc.). Second, all the methods implemented in simulation can also be produced in a real-world setup.

For prototype experiments, the main challenge is how to train the RL controller in a real-world setup, which usually requires millions of episodes. Different from a robotic arm that can be set to its initial condition easily, there is no good way to reset the training scene for mobile robots in real world. Some algorithms (Fu et al., 2017; Hwangbo et al., 2019) may directly transfer the learned policy from simulation and implement it in real-world scenario. But this is out of the scope of this paper.

7. CONCLUSION

Performing target-tracking tasks for snake-like robots is a challenging task, since it not only involves designing agile locomotion patterns for the robot, but also overcoming difficulties to obtain stable visual information due to the inherent undulatory motions. In this paper, we try to solve this complex perception-to-action control task by using reinforcement learning, which directly maps the vision space to the joint space and reduces the computational complexity of dealing with object tracking and robot motion control in separate components. In our test scenarios, the learned gait shows much better tracking performances than the model-based method. Our work contributes to designing sophisticated and efficient moving patterns for perception-driven tasks with a snake-like robot.

Our future work will aim at performing tracking tasks with more complex visual information. For instance, the perception of the visual information can be replaced with more sophisticated technologies. To improve the adaptability of our RL controller, we will further investigate locomotion skills for more challenging scenarios, such as in a obstacle surrounding environment and the capability to recover tracking when the target runs out of the visual field of the robot.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author/s.

AUTHOR CONTRIBUTIONS

ZB, CL, and AK brought up the core concept. ZB and CL conducted the experiments and analyzed the results. ZB, CL, ZJ, LC, and KH wrote the paper. All authors contributed to the article and approved the submitted version.

FUNDING

This research has received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 785907 (Human Brain Project SGA2) and the Specific Grant Agreement No. 945539 (Human Brain Project SGA3). Meanwhile, it was also supported by the German Research Foundation (DFG) and the Technical University of Munich (TUM) in the framework of the Open Access Publishing Program.

REFERENCES

Asada, M., Noda, S., Tawaratsumida, S., and Hosoda, K. (1996). Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Mach. Learn*. 23, 279–303.

Bing, Z., Cheng, L., Chen, G., Röhrbein, F., Huang, K., and Knoll, A. (2017). Towards autonomous locomotion: CPG-based control of smooth 3D slithering gait transition of a snake-like robot. *Bioinspir. Biomim*. 12:035001. doi: 10.1088/1748-3190/aa644c

Bing, Z., Jiang, Z., Cheng, L., Cai, C., Huang, K., and Knoll, A. (2019a). "End to end learning of a multi-layered SNN based on R-STDP for a target tracking snake-like robot," in *2019 International Conference on Robotics and Automation (ICRA)*, 9645–9651. doi: 10.1109/ICRA.2019.8793774

Bing, Z., Lemke, C., Jiang, Z., Huang, K., and Knoll, A. (2019b). "Energy-efficient slithering gait exploration for a snake-like robot based on reinforcement learning," in *The 28th International Joint Conference on Artificial Intelligence (IJCAI-19)* (Macao: IJCAI). doi: 10.24963/ijcai.2019/785

Busquets, D., de Mántaras, R. L., De, R. L. O., Antaras, M., Sierra, C., and Dietterich, T. G. (2002). "Reinforcement learning for landmark-based robot navigation," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (Bologna), 841–843. doi: 10.1145/544862.544938

Ding, L., Li, S., Gao, H., Chen, C., and Deng, Z. (2018). Adaptive partial reinforcement learning neural network-based tracking control for wheeled mobile robotic systems. *IEEE Trans. Syst. Man Cybern. Syst*. 50, 2512–2523. doi: 10.1109/TSMC.2018.2819191

Evan, A. (2017). *What CMU's Snake Robot Team Learned While Searching for Mexican Earthquake Survivors*. Available online at: https://spectrum.ieee.org/automaton/robotics/industrial-robots/cmu-snake-robot-mexico-earthquake

Fu, J., Luo, K., and Levine, S. (2017). Learning robust rewards with adversarial inverse reinforcement learning. *arXiv* 1710.11248.

Fukushima, H., Satomura, S., Kawai, T., Tanaka, M., Kamegawa, T., and Matsuno, F. (2012). Modeling and control of a snake-like robot using the screw-drive mechanism. *IEEE Trans. Robot*. 28, 541–554. doi: 10.1109/TRO.2012.2183050

Garcia-Aracil, N., Perez-Vidal, C., Sabater, J. M., Morales, R., and Badesa, F. J. (2011). Robust and cooperative image-based visual servoing system using a redundant architecture. *Sensors* 11, 11885–11900. doi: 10.3390/s111211885

Gaskett, C., Fletcher, L., and Zelinsky, A. (2000). "Reinforcement learning for a vision based mobile robot," in *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)* (*Cat. No.00CH37113)*, Vol. 1 (Takamatsu), 403–409. doi: 10.1109/IROS.2000.894638

Hafner, R., and Riedmiller, M. (2003). "Reinforcement learning on an omnidirectional mobile robot," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)* (*Cat. No.03CH37453)*, Vol. 1 (Las Vegas, NV), 418–423.

Heess, N., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., et al. (2017). Emergence of locomotion behaviours in rich environments. *arXiv* 1707.02286.

Hirose, S. (1993). *Biologically Inspired Robots: Snake-Like Locomotors and Manipulators*, Vol. 1093. Oxford: Oxford University Press.

Huang, K., Shao, K., Zhen, S., and Sun, H. (2017). A novel approach for modeling and tracking control of a passive-wheel snake robot. *Adv. Mech. Eng*. 9:1687814017693944. doi: 10.1177/1687814017693944

Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., et al. (2019). Learning agile and dynamic motor skills for legged robots. *Sci. Robot*. 4:eaau5872. doi: 10.1126/scirobotics.aau5872

Ishikawa, M., Morin, P., and Samson, C. (2009). "Tracking control of the trident snake robot with the transverse function approach," in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC) Held Jointly With 2009 28th Chinese Control Conference* (Shanghai), 4137–4143. doi: 10.1109/CDC.2009.5399958

Kamegawa, T., Yarnasaki, T., Igarashi, H., and Matsuno, F. (2004). "Development of the snake-like rescue robot "kohga"," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04*, Vol. 5 (New Orleans, LA), 5081–5086. doi: 10.1109/ROBOT.2004.1302523

Liljeback, P., Pettersen, K. Y., Stavdahl, A, and Gravdahl, J. T. (2011). Controllability and stability analysis of planar snake robot locomotion. *IEEE Trans. Autom. Control* 56, 1365–1380. doi: 10.1109/TAC.2010.2088830

Luo, W., Sun, P., Zhong, F., Liu, W., Zhang, T., and Wang, Y. (2019). End-to-end active object tracking and its real-world deployment via reinforcement learning. *IEEE Trans. Pattern Anal. Mach. Intell*. 1, 1317–1332. doi: 10.1109/TPAMI.2019.2899570

Matsuno, F., and Mogi, K. (2000). "Redundancy controllable system and control of snake robots based on kinematic model," in *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No.00CH37187)*, Vol. 5 (Sydney, NSW), 4791–4796. doi: 10.1109/CDC.2001.914686

Matsuno, F., and Sato, H. (2005). "Trajectory tracking control of snake robots based on dynamic model," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (Barcelona), 3029–3034. doi: 10.1109/ROBOT.2005.1570575

Miljkovicć, Z., Mitić, M., Lazarević, M., and Babić, B. (2013). Neural network reinforcement learning for visual control of robot manipulators. *Expert Syst. Appl*. 40, 1721–1736. doi: 10.1016/j.eswa.2012.09.010

Miller, G. S. (1988). The motion dynamics of snakes and worms. *ACM Siggraph Comput. Graph*. 22, 169–173. doi: 10.1145/378456.378508

Morbidi, F., and Mariottini, G. L. (2013). Active target tracking and cooperative localization for teams of aerial vehicles. *IEEE Trans. Control Syst. Technol*. 21, 1694–1707. doi: 10.1109/TCST.2012.2221092

Ohno, K., Nomura, T., and Tadokoro, S. (2006). "Real-time robot trajectory estimation and 3d map construction using 3D camera," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Beijing), 5279–5285. doi: 10.1109/IROS.2006.282027

Ponte, H., Queenan, M., Gong, C., Mertz, C., Travers, M., Enner, F., et al. (2014). "Visual sensing for developing autonomous behavior in snake robots," in *2014 IEEE International Conference on Robotics and Automation (ICRA)* (Hong Kong), 2779–2784. doi: 10.1109/ICRA.2014.6907257

Prautsch, P., Mita, T., and Iwasaki, T. (2000). Analysis and control of a gait of snake robot. *IEEJ Trans. Ind. Appl*. 120, 372–381. doi: 10.1541/ieejias.120.372

Sanfilippo, F., Azpiazu, J., Marafioti, G., Transeth, A. A., Stavdahl, A., and Liljebäck, P. (2017). Perception-driven obstacle-aided locomotion for snake robots: the state of the art, challenges and possibilities. *Appl. Sci*. 7:336. doi: 10.3390/app7040336

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv* 1707.06347.

Takahashi, Y., Takeda, M., and Asada, M. (1999). "Continuous valued q-learning for vision-guided behavior acquisition," in *Proceedings. 1999 IEEE/SICE/RSJ. International Conference on Multisensor Fusion and Integration for Intelligent Systems. MFI'99 (Cat. No.99TH8480)* (Taipei), 255–260. doi: 10.1109/MFI.1999.815999

Tanaka, M., and Matsuno, F. (2014). Control of snake robots with switching constraints: trajectory tracking with moving obstacle. *Adv. Robot*. 28, 415–429. doi: 10.1080/01691864.2013.867285

Tanaka, M., and Tanaka, K. (2015). Control of a snake robot for ascending and descending steps. *IEEE Trans. Robot*. 31, 511–520. doi: 10.1109/TRO.2015.2400655

Tanaka, M., Tanaka, K., and Matsuno, F. (2015). Approximate path-tracking control of snake robot joints with switching constraints. *IEEE/ASME Trans. Mechatron*. 20, 1633–1641. doi: 10.1109/TMECH.2014.2367657

Tesch, M., Lipkin, K., Brown, I., Hatton, R., Peck, A., Rembisz, J., et al. (2009). Parameterized and scripted gaits for modular snake robots. *Adv. Robot*. 23, 1131–1158. doi: 10.1163/156855309X452566

Todorov, E., Erez, T., and Tassa, Y. (2012). "Mujoco: a physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033. doi: 10.1109/IROS.2012.6386109

Transeth, A. A., van de Wouw, N., Pavlov, A., Hespanha, J. P., and Pettersen, K. Y. (2007). "Tracking control for snake robot joints," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems* (San Diego, CA), 3539–3546. doi: 10.1109/IROS.2007.4399174

Walker, I. D., and Hannan, M. W. (1999). "A novel 'elephant's trunk' robot," in *1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Cat. No. 99TH8399)* (Atlanta, GA: IEEE), 410–415. doi: 10.1109/AIM.1999.803204

Webster, R. J., Okamura, A. M., and Cowan, N. J. (2006). "Toward active cannulas: miniature snake-like surgical robots," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Beijing), 2857–2863. doi: 10.1109/IROS.2006.282073

Xiao, X., Cappo, E., Zhen, W., Dai, J., Sun, K., Gong, C., et al. (2015). "Locomotive reduction for snake robots," in *2015 IEEE International Conference on Robotics and Automation (ICRA)* (Seattle), 3735–3740. doi: 10.1109/ICRA.2015.7139718

Yamada, H., Takaoka, S., and Hirose, S. (2013). A snake-like robot for real-world inspection applications (the design and control of a practical active cord mechanism). *Adv. Robot*. 27, 47–60. doi: 10.1080/01691864.2013.752318

Yun, S., Choi, J., Yoo, Y., Yun, K., and Choi, J. Y. (2018). Action-driven visual object tracking with deep reinforcement learning. *IEEE Trans. Neural Netw. Learn. Syst*. 29, 2239–2252. doi: 10.1109/TNNLS.2018.2801826

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

ORIGINAL RESEARCH
published: 21 May 2021
doi: 10.3389/fnbot.2021.664135

A New Hydrogen Sensor Fault Diagnosis Method Based on Transfer Learning With LeNet-5

*Yongyi Sun[1,2], Shuxia Liu[2], Tingting Zhao[3], Zhihui Zou[3], Bin Shen[3], Ying Yu[2], Shuang Zhang[2] and Hongquan Zhang[1,3]**

[1] Key Laboratory of Electronics Engineering, College of Heilongjiang Province, Heilongjiang University, Harbin, China, [2] Department of Information, Liaoning Police Academy, Dalian, China, [3] School of Automation, Harbin Engineering University, Harbin, China

OPEN ACCESS

Edited by:
Jose De Jesus Rubio,
Instituto Politécnico Nacional (IPN), Mexico

Reviewed by:
Wellington Pinheiro dos Santos,
Federal University of Pernambuco, Brazil
Faisal Budiman,
Telkom University, Indonesia
Peng Zhang,
Huazhong University of Science and Technology, China
Qili Zhao,
Nankai University, China
Zhan Yang,
Soochow University, China
Ming Zhu Sun,
Nankai University, China
Qi Li,
Changchun University of Science and Technology, China

***Correspondence:**
Hongquan Zhang
1277861501@hrbeu.edu.cn

Received: *09 February 2021*
Accepted: *16 April 2021*
Published: *21 May 2021*

Citation:
Sun Y, Liu S, Zhao T, Zou Z, Shen B, Yu Y, Zhang S and Zhang H (2021) A New Hydrogen Sensor Fault Diagnosis Method Based on Transfer Learning With LeNet-5. Front. Neurorobot. 15:664135. doi: 10.3389/fnbot.2021.664135

The fault safety monitoring of hydrogen sensors is very important for their practical application. The precondition of traditional machine learning methods for sensor fault diagnosis is that enough fault data with the same distribution and feature space under the same working environment must exist. Widely used fault diagnosis methods are not suitable for real working environments because they are easily complicated by environmental conditions such as temperature, humidity, shock, and vibration. Under the influence of such complex conditions, the acquisition of sensor fault data is limited. In order to improve fault diagnosis accuracy under complex environmental conditions, a novel method of transfer learning (TL) with LeNet-5 is proposed in this paper. Firstly, LeNet-5 is applied to learn the features of the data-rich datasets of gas sensor faults in a normal environment and to adjust the parameters accordingly. The parameters of the LeNet-5 are transferred from the task in the normal environment to a task in a complex environment by using the TL method. Then, the migrated LeNet-5 is used for the fault diagnosis of gas sensors with a small amount of fault data in a complex environment. Finally, a prototype hydrogen sensor array is designed and implemented for experimental verification. The gas sensor fault diagnosis accuracy of the traditional LeNet-5 was 88.48 ± 1.04%, while the fault diagnosis accuracy of TL with LeNet-5 was 92.49 ± 1.28%. The experimental results show that the method adopted presents an excellent solution for the fault diagnosis of a hydrogen sensor using a small quantity of fault data obtained under complex environmental conditions.

Keywords: hydrogen sensor, fault diagnosis, transfer learning, LeNet-5, machine learning

INTRODUCTION

With the gradual depletion of traditional energy sources and the development of clean fuel, clean fuel plays a prominent role throughout many fields (Tsujita et al., 2005; Brown et al., 2015). As hydrogen is a clean fuel, its application range is therefore rapidly expanding (Chalk and Miller, 2006). It is widely used in meteorological science, aerospace technology, the metallurgical industry, the electronics industry, national defense, the chemical industry, and so on, and its consumption is also increasing rapidly (Poirier and Sapundzhiev, 1997; Winter, 2005; Staffell et al., 2019).

Hydrogen is a colorless, odorless, flammable, and explosive gas. It is necessary to monitor hydrogen concentrations using hydrogen sensors because it is considered a dangerous chemical (Song et al., 2019).

Semiconductor gas sensors have been widely used in hydrogen detection based on SnO_2-sensitive materials (Fedorenko et al., 2017; Zhang Q. et al., 2018). However, they can be hindered by sensor aging, environmental temperature and humidity, sensitive material falling off the sensor, gas adsorption poisoning, and other factors that could affect the reliability of the sensors. Hydrogen sensors are prone to failure in its hydrogen safety detection function, which may lead to combustion and explosion. Therefore, hydrogen sensors' fault diagnosis is of great importance. Ingimundarson et al. proposed model-based detection of hydrogen leaks in 2008 (Ingimundarson et al., 2008). Ma et al. proposed hydrogen sensor for fault detection of power transformer in 2012 (Ma et al., 2012). Song et al. proposed a fault diagnosis and reconfiguration strategy for hydrogen sensor array in 2019 (Song et al., 2019). Sun et al. proposed a new convolutional neural network method for hydrogen sensor fault diagnosis in 2020 (Sun et al., 2020).

Recently, traditional machine learning (ML) methods have been widely used for fault diagnoses, such as the extreme learning machine (ELM) (Song et al., 2019), empirical mode decomposition (Chen Y. S. et al., 2016), support vector machines (SVM) (Hu et al., 2005), KNN (Yang et al., 2016b), non-negative matrix factorization (Yang et al., 2016a), gray forecasting (Chen Y. et al., 2016), learning vector quantization (LVQ) (Bassiuny et al., 2007), random forest (RF) (Mohapatra et al., 2020), and kernel principal component analysis (KPCA) (Navi et al., 2018). These methods can effectively extract fault features to a certain extent, but there are some limitations. ML methods are unable to generate discriminative features of fault signal data, there methods always combined with the signal features extraction process, the features should be predesigned. However, predesigning handcrafted features is difficult. The features extraction process of fault signal is an exhausted work and impacts the fault diagnosis result. These methods require experts to have a rich mathematical knowledge about manual feature extraction, and the degree of automation of feature extraction is greatly limited. The method selected by the experts directly affects the diagnosis results.

As a branch of ML, deep learning (DL) has powerful functionality and flexibility. DL does not need to rely on expert experience and manual feature extraction (Zhang W. et al., 2018), so many scholars have gradually introduced DL methods, such as the deep belief network (Shao et al., 2018; Wang et al., 2020), sparse autoencoders, and convolution neural networks (CNNs) (Wen et al., 2018; Wu and Zhao, 2018; Gou et al., 2020; Sun et al., 2020) into fault diagnosis processes. These methods can improve the accuracy of fault diagnosis, but there are some limitations. These methods require certain preconditions: sufficient sample data are required, and the training and test samples need to have the same data distribution. When the distributions of sample data are different, the performances of the above methods would drop. They does not consider the use of fault data under different environments for fault diagnosis.

The concept of transfer Learning (TL) was first proposed in 1995 at a conference on neural information processing systems (Thrun, 1995). TL is adopted to improve the performance of the current task by using data, models, or tasks from the source task that are different from (but similar to) the target task (Pan and Yang, 2009; Chen et al., 2019). When the data attributes and feature spaces in the source domain and the target domain are similar but not identical, previous learning experience is used to solve the crossing domain problem (Pan and Yang, 2009; Wen et al., 2017a). There are many scenarios of TL, such as multi-task learning (Caruan, 1997) and domain adaptation (Saenko et al., 2010). Model-based TL can use the pre training knowledge acquired in the source domain to transform and summarize the deep model (Donahue et al., 2014). As a new ML method, many scholars have started to introduce the TL method into the process of fault diagnosis under variable conditions (Wen et al., 2017b; Wu et al., 2020). However, this method is rarely used in gas sensor fault diagnosis.

In this paper, a gas sensor fault diagnosis method based on TL with LeNet-5 in a complex environment is proposed. A large set of gas sensor fault signal data under normal environmental conditions is adopted to train the traditional LeNet-5 model until an excellent performance is observed. However, it is difficult to obtain an amount of fault signal data due to the limitation of experimental conditions under complex environment, so the fault signal data is limited. The traditional model and parameters of the LeNet-5 can transfer to a new target task with a small amount of fault data using the TL method. The TL with LeNet-5 method could make use of gas sensor fault data from different environments, resulting in a better performance in complex environments. The benefits of this novel method improve the accuracy of fault diagnosis in complex environments by utilizing gas sensor fault signal data from normal and complex environments when only a small quantity of target fault data exists.

The remainder of this article is organized as follows. The second section introduces the theoretical fundamentals. In the third section, a novel model based on TL with LeNet-5 for hydrogen sensor fault diagnosis is introduced. The fourth section verifies the performance of the proposed method. The fifth section contains the conclusions and future researches.

THEORETICAL FUNDAMENTALS

CNNs and LeNet-5

CNNs are widely used in image processing. They consist of a convolutional layer, pooling layer, and full connection (FC) layer. The convolutional layer can extract features via a convolutional operation on the previous layers of different convolutional kernels. More features can be extracted by using multiple convolutional kernels. The pooling layer can sample the features extracted from the convolutional layer. The sampling method can be divided into two parts: maximum sampling and mean sampling. In this paper, the maximum sampling method is adopted. Each node of the FC layer is connected with all nodes of the previous layer, which are used to integrate the features extracted from the front edge (Wu and Zhao, 2018). The mathematical model of the CNN follows Equation (1).

$$x_j^l = f(\sum\nolimits_{i\in M_j} x_j^{l-1} \times k_{ij}^l + b_j^l) \tag{1}$$

where M_j denotes the input characteristic graph, k denotes the convolution kernel, b denotes the network bias, x_j^l denotes the l layer output, and x_j^{l-1} denotes the l layer input. The calculation method for subsampling layer neurons follows Equation (2):

$$x_j^l = f(\beta_j^l down(x_i^{l-1}) + b_j^l) \tag{2}$$

where $down(x_i^{l-1})$ denotes the subsampling function and β denotes the network multiplicative bias. The CNN's output layer model follows Equation (3):

$$O = (b_o + w_o f_v) \tag{3}$$

where f_v denotes the eigenvector, b_o, w_o denotes the deviation vector and the weight matrix.

There are many CNN models; for example, GoogLeNet (Szegedy et al., 2015), AlexNet (Krizhevsky et al., 2017), and LeNet-5 (LeCun, 2015). As a classic CNN, LeNet-5 is widely used for handwritten digital character recognition (Tivive and Bouzerdoum, 2005) and fault diagnosis (Wen et al., 2018; Sun et al., 2020). LeNet-5 is a CNN with a gradient-based learning structure, and its input layer is an image with a size of 32× 32 pixels. The typical LeNet-5 structure consists of two alternating convolutional layers, two pooling layers, and the two-layer FC artificial neural network. Compared with Alenet, GoogLenet, VGG16, and other CNN algorithms, LeNet-5 method has simple structure and high accuracy (Wen et al., 2018; Lu et al., 2019), and has achieved good results in hydrogen sensor fault diagnosis (Sun et al., 2020). Therefore, this study adpots LeNet-5 as classifier.

Transfer Learning

TL is committed to transferring information of knowledge obtained on sufficient labeled data of a source domain to a small amount of data of a target domain. From the data volume, the source domain data are easy to obtain, the data samples are sufficient, the target domain data are difficult to obtain, and the data samples are very limited. When the content of previous learning and the content of new problems are similar, and the potential data share some common characteristics, the migration effect is significant. For example, it is easier for a person to learn to ride a motorcycle after learning to ride a bicycle. The domain and task follow Equations (4) and (5)

$$D = \{X, P(X)\} \tag{4}$$

$$T = \{Y, P(Y/X)\} \tag{5}$$

where D denotes the domain and T denotes the task. X, Y are the feature space and category space, respectively, and $P(X)$, $P(Y/X)$ are the marginal probability distribution and the conditional probability density, respectively. TL based on parameters migration is adopted in this paper; that is, some parameters are shared between the target domain model and the source domain model. Its purpose is to mine the prior distributions or parameters shared between the source domain and target domain.

PROPOSED MODEL FOR FAULT DIAGNOSIS OF HYDROGEN SENSORS BASED ON TL WITH LENET-5

In this section, a novel model of TL with LeNet-5 is proposed for the fault diagnosis of hydrogen sensors. Firstly, a method for preprocessing the raw signal of hydrogen sensors is adopted. Secondly, TL with the LeNet-5 method is proposed for gas sensor fault diagnosis.

Hydrogen Sensor Fault Signal Pre-treatment

In this paper, the data preprocessing method we adopted changes the raw fault signal into a two-dimensional gray image conversion (Sun et al., 2020). We normalized the fault data. This method could not only realize end-to-end data conversion, but also eliminate the influence of expert experiences as much as possible without any predefined parameters. Supposing that the sensor fault signal is $L(n)$, it follows Equation (6)

$$L(n), n = 1, 2, \cdot\cdot\cdot\cdot, N \times M \tag{6}$$

and the resolution of the two-dimensional gray image is $N \times M$ pixels, where N is the width and M is the height. To ensure the symmetry of $L[i]$, the MOL as the matrix of $L[i]$ follows Equation (7)

$$MOL = \begin{bmatrix} L(1) & \cdots & L(N) \\ \vdots & \ddots & \vdots \\ L((M-1)\,N+1) & \cdots & L(NM) \end{bmatrix} \tag{7}$$

The process of sensor fault signal pretreatment is shown in **Figure 1**.

The Proposed Method of the TL With LeNet-5

Many CNN models have been proposed in recent years. This paper adopts the classic LeNet-5 model, which has been applied in many fields. The proposed LeNet-5 method consists of two parts: feature extraction and fault classification. It requires an image size of 32× 32 pixels as the input; however, in order to improve the results of the gas sensor fault classification, we changed the size of the LeNet-5 input image. The revised width N of the gray image is 50 pixels, and the height M is 40 pixels. These adjustments depend on the volume of the raw fault signal and the architecture of feature extraction. The LeNet-5 consists of two convolution layers, two pooling layers, and two FC layers with two strategies: dropout and batch normalization (BN). The LeNet-5 structure proposed in this paper is shown in **Figure 2**.

This paper proposes TL with LeNet-5 method for gas sensor fault diagnosis in a complex environment, which involves two domains: the source domain and task domain. The source

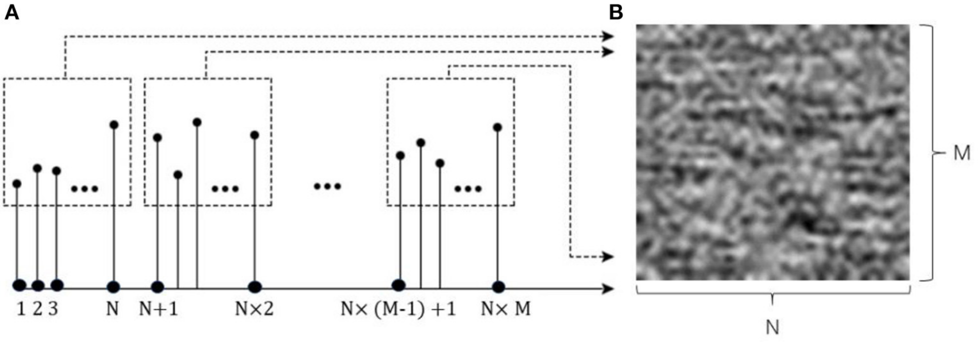


FIGURE 1 | The process of gas sensor fault signal pretreatment. **(A)** One dimensional time domain gas sensor fault signal, **(B)** Two dimensional gray image.

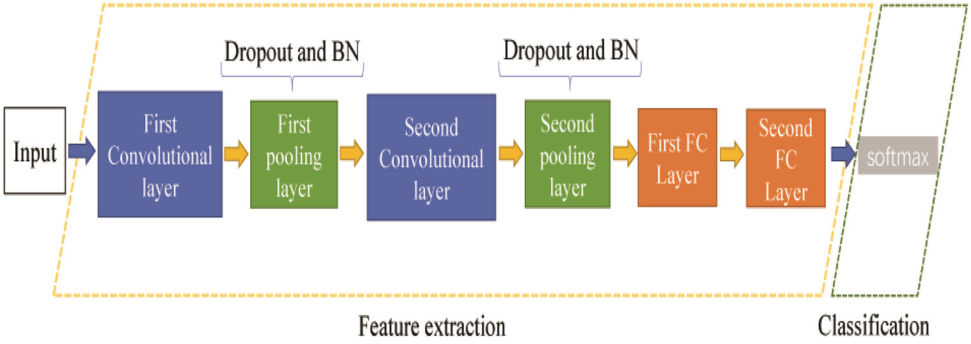


FIGURE 2 | The LeNet-5 structure proposed in this paper.

domain contains S kinds of gas sensor signal modes with sufficient fault data under a normal environment. The target domain contains T kinds of gas sensor signal modes with a small amount of fault data under a complex environment. The process of fault diagnosis is presented in six steps.

(1) S kinds of signal mode data in the source domain are preprocessed and converted into two-dimensional gray images.
(2) The images of source domain are input into the LeNet-5 model for training.
(3) The trained LeNet-5 model and parameters are transferred to the target domain.
(4) T kinds of signal mode data in the task domain are preprocessed and converted into two-dimensional gray images.
(5) The images of task domain are placed into the TL with LeNet-5 model for training, and the model parameters are fine tuned.
(6) The test sample data are adopted to test the trained model in order to verify the accuracy of the proposed method. The detailed process of the TL with LeNet-5-based gas sensor fault diagnosis model described in this paper is shown in **Figure 3**.

EXPERIMENT AND VALIDATION OF THE PROPOSED METHOD

Experimental Setup

Based on the environmental adaptability, reliability, and performance tests, together with the related literature, several typical fault signal modes of the SnO_2 sensors are summarized in this study. These include heating wire disconnection (HWD), aging of the heating wire (AHW), aging of the sensitive body (ASB), exfoliation of the sensitive body (ESB), and false welding of the sensitive body (FWSB) (Sun et al., 2020). In order to obtain the data from five modes of fault signals under normal and complex environments, we collected fault data through a self-made special gas sensor test system. The test system is composed of a constant temperature and humidity box, a shaking table, an electric fan, a standard hydrogen concentration bottle, a standard air bottle, a gas molecular flow meter, a gas mixer, a bidirectional regulated power supply, a data collector, a computer, a temperature sensor, a humidity sensor, a sensor chamber, and six SnO_2 sensor arrays.

The constant temperature and humidity box provided the test environment for temperature and humidity changes, the shaking table provided the test environment for vibration changes, and the electric fan provided the test environment

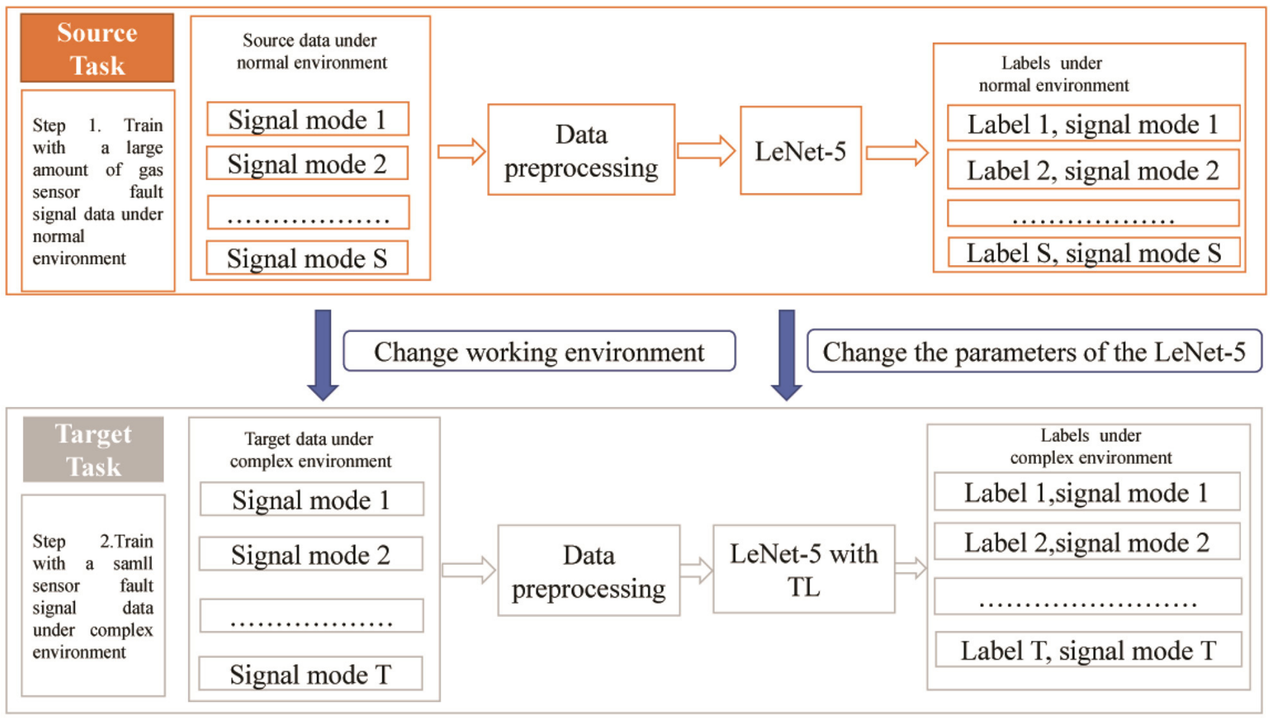


FIGURE 3 | A detailed process of transfer learning with the LeNet-5-based gas sensor fault diagnosis model.

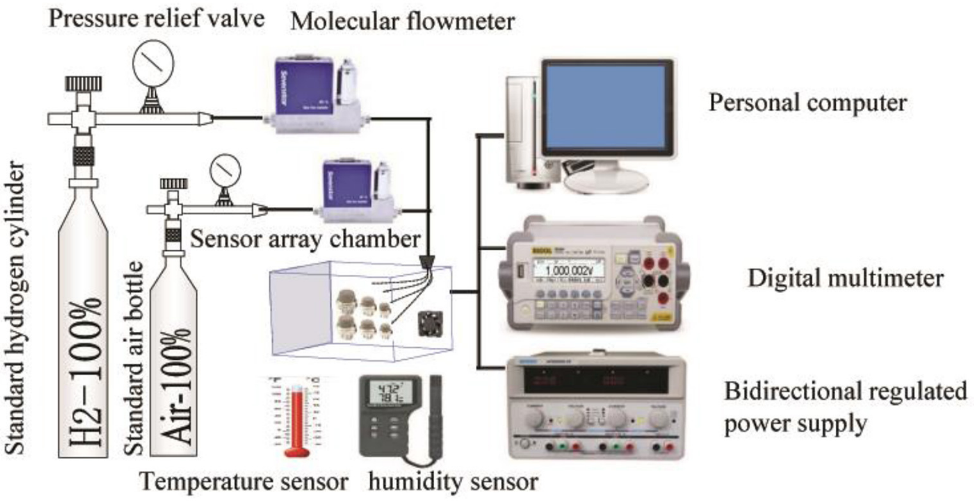


FIGURE 4 | A system diagram of hydrogen sensor arrays.

for wind changes. The hydrogen sensor array system diagram is shown in **Figure 4**. A sensor array and gas chamber were placed into the constant temperature and humidity box and vibration table, respectively, to simulate temperature, humidity, and vibration variations in the environment. The fan was placed in the gas chamber to simulate wind changes in the test environment.

The structure of the SnO_2 sensor model (MQ-8) is shown in **Figure 5**. It is composed of a four-leg plastic base, nickel-plated copper column, stainless steel explosion-proof net, metal buckle ring, nickel-chromium heating wire, gas-sensitive body, gold lead, and porcelain tube carrier. The gold lead and the porcelain tube carrier were connected via gold slurry sintering welding, the nickel-chromium heating wire and the gold lead

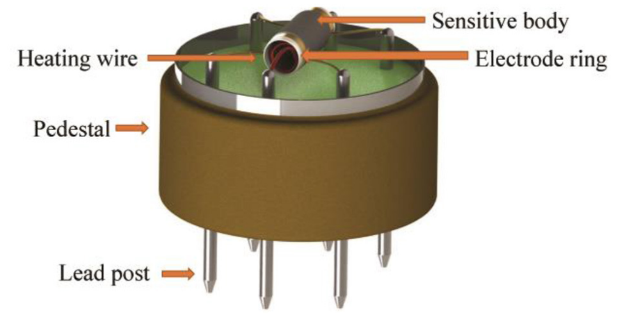


FIGURE 5 | The MQ-8 sensor structure diagram.

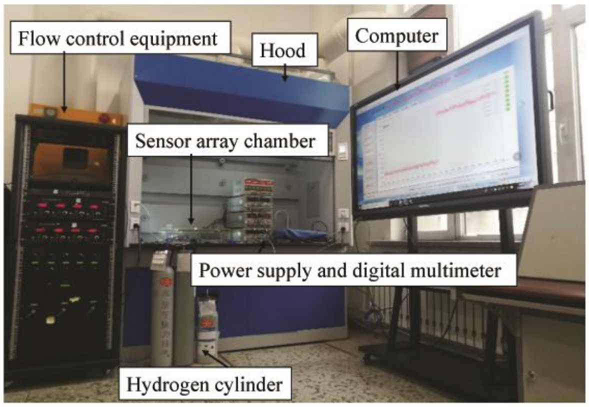


FIGURE 6 | The experimental setup of MQ-8 gas sensor array.

were connected via tin welding with the nickel-copper column, and the nickel-chromium heating wire and the gas-sensitive body were the key components of the hydrogen sensors. The nickel-chromium heating wire can provide a high-temperature working environment for the sensors. The function of the sensitive body was to detect the concentration of hydrogen and convert the value of the concentration into the resistance change. The function of the gold lead wire was to pass the information of the resistance change to the outside of the sensors through the nickel-copper column. The functional components of the gas sensor mentioned above are the main factors leading to the failure of the MQ-8 sensor. These variables keep constant during the experiment. The process of data acquisition is listed as follows: in the sensor array, each sensor has a separate power supply and a separate voltage divider. When the signals of the six sensors are collected, they are input to the single-chip microcomputer, integrated into the data acquisition card, and finally uploaded to the upper computer. The experimental device (the MQ-8 sensor array) is shown in **Figure 6**. The DL program was run on a 3.0 GHz Intel CPU and 8 GB RAM with Python 3.7.4 and TensorFlow 1.15.0 running Windows 10.

The general static calibration method of gas sensor is used to put the sensor into a standard gas box, and inject pure measured gas on the basis of the known space structure volume of the gas tank. After conversion, the standard gas concentration can be obtained. The standard gas concentration is placed on the sensor, and the sensor has output, then the standard gas concentration can be established the corresponding relationship with output can achieve the purpose of sensor calibration.

In this study, the normal environmental conditions are defined as a standard atmospheric pressure, temperature range between 17 to 23°C, humidity concentration range from 30 to 60%. The complex environmental conditions are defined as high humidity concentration range from 90 to 95%, low humidity concentration range from 10 to 20%, low temperature range from −10 to −30°C, high temperature range from 40 to 60°C. The wind is five meters per second, and it vibrates. The concentration range of hydrogen is 0.1–1%.

The data from six signal modes (i.e., without fault and the five fault types) were obtained through the instrument and equipment we set up. We obtained the six signal modes of the MQ-8 sensor under a normal environment, as shown in **Figure 7**. The fault signal data were stable, so we used Matlab to simulate the six signal modes under a normal environment and increased the fault signal data number of the six signal modes. The sample data includes real samples and Matlab simulation samples under normal environment. The simulation data under different fault modes were obtained by the following ways: (1) The HWD fault was a linear signal with a larger slope which was superimposed on the normal output signal from a certain moment. The signal was stable at a certain value and at a certain moment. (2) The AHW fault was a linear signal with small slope superimposed at a certain moment of the normal output signal. (3) The aging cycle of ASB fault was long, so the aging process is accelerated in the simulation for the convenience of research, a linear signal with very small slope was superimposed on the normal signal from a certain time. (4) The ESB fault was to add a constant deviation data on the normal output signal from a certain time. (5) The output signal of the FWSB fault was 0 at a random time, and the output was normal at a certain time. On this basis, the white noise signal was superimposed.

Gas sensors often encounter complex environments in practice. In order to observe gas sensor fault signals in complex environments, we changed the temperature and humidity of a constant humidity incubator to increase the noise interference. The vibration noise interference could be increased by changing the vibration spectrum of the shaking table; the wind speed noise interference could be increased by the wind force generated by the electric fan in the gas chamber. In the end, the fault signals were obtained under complex environmental conditions. Therefore, the experiment under complex environment constructs a transformation environment, but carries on the measurement under the stable situation. Due to the complexity of the test conditions, we only get twenty-nine fault data for each fault type.

The six signal modes of the MQ-8 sensor under a complex environment are shown in **Figure 7**. As shown, the collected gas sensor fault signal changed greatly and was complex, so it was difficult to obtain these data through the Matlab simulation.

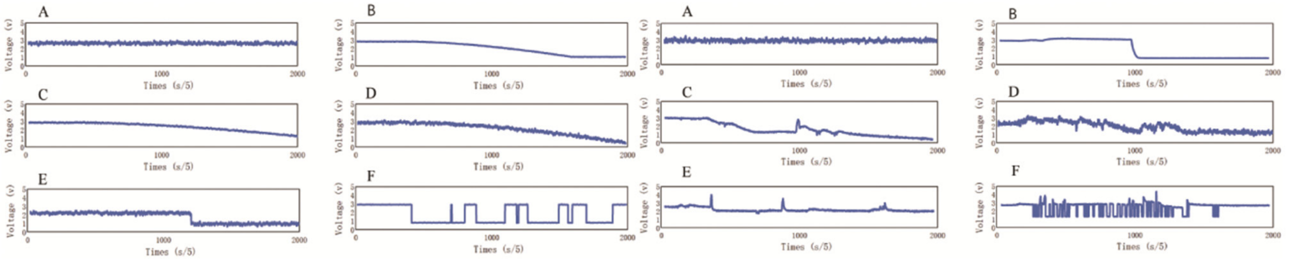


FIGURE 7 | Six signal modes of MQ-8 gas sensor under a normal environment and complex environment. **(A)** Normal signal; **(B)** The signal including heating wire disconnection (HWD) fault; **(C)** The signal including aging of the heating wire (AHW) fault; **(D)** The signal including aging of the sensitive body (ASB) fault; **(E)** The signal including exfoliation of the sensitive body (ESB) fault; **(F)** The signal including false welding of the sensitive body (FWSB) fault.

TABLE 1 | Signal mode types and samples of every type under normal and complex environments using transfer learning with LeNet-5.

	Source task	Target task
Working environments	Normal environment	Complex environment
Signal mode types	6	6
Samples of every type	100	29

Through the experiments, the MQ-8 gas sensor signal mode types and samples of every type (i.e., the five fault types and without fault) under normal and complex environments were obtained to verify the effectiveness of TL with LeNet-5, as shown in **Table 1**.

Validation of the TL With LeNet-5 Method

To validate the advantages of the proposed model in the fault diagnosis of a hydrogen sensor, tests were performed. The results of TL with LeNet-5 training and inference are presented in this section.

TL With LeNet-5 Training

There are data-rich sensor fault training samples under a normal environment in the source data compared with the target data under a complex environment. The LeNet-5 was trained and transferred from a normal environment to a complex environment. 100 samples of fault signal modes for each type in the source task were used to train the traditional LeNet-5. In the target task, only 20 samples of signal modes for each type were obtained to train the transferred LeNet-5. Nine samples of signal modes for each type in the target task were obtained for the test. The details of the labels and samples under normal and complex environmental conditions are shown in **Table 2**.

TABLE 2 | Labels and samples under normal and complex environmental conditions.

Label	Signals modes description	Normal environment conditions	Complex environmental conditions	
		Number of training samples	Number of training samples	Number of test samples
1	Normal signal	100	20	9
2	HWD signal	100	20	9
3	AHW signal	100	20	9
4	ASB signal	100	20	9
5	ESB signal	100	20	9
6	FWSB signal	100	20	9

The gas sensor signal data of six modes under normal environmental conditions were the source data of the transfer learning with LeNet-5 method. The data under complex environmental conditions were the target data of the transfer learning with LeNet-5 method. They were also the training data for other methods without transfer in the experiment.

As shown in **Figures 8**, **9**, the six sensor signal modes under normal and complex environments were converted into two-dimensional gray images, and the size of each image was 50× 40 pixels.

The two-dimensional gray images under a normal environment were input into the traditional LeNet-5 for training, and the number of experimental samples was 100 sets. The traditional LeNet-5 was trained for 500 iterations. As can be seen from **Figure 10**, the training accuracy reached 100%, and the training loss was close to 0 after about 50 iterations.

In order to verify the effectiveness of TL with LeNet-5 method, two methods were used to train the gray images. As shown in **Figure 11**, firstly, the traditional LeNet-5 model and parameters, which were trained in the source task, were transferred to the target task. The two-dimensional gray images under a complex environment were used as the target domain data for retraining. The TL with LeNet-5 method was trained for 500 iterations. The training accuracy reached 100%, and the training loss was close to 0 after about 150 iterations. Secondly, the two-dimensional gray images under a complex environment were input into the traditional LeNet-5 directly. The traditional LeNet-5 was trained for 500 iterations. The training accuracy reached 100%, and the training loss was close to 0 after about 200 iterations. The two methods both used 20 sets of experimental samples.

TL With LeNet-5 Inference

To obtain better results, the TL with LeNet-5 used cross-validation method. The experiments repeated 30 times. The

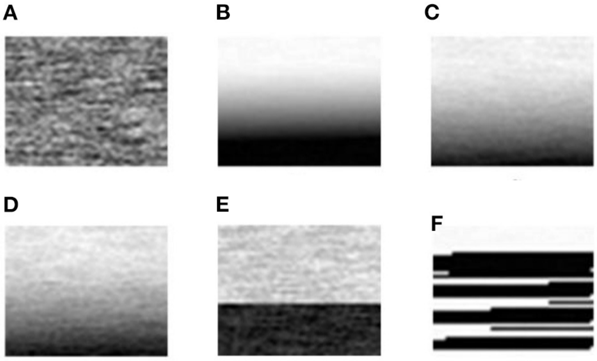


FIGURE 8 | Converted two-dimensional gray images of the gas sensor signals of six modes under a normal environment. **(A)** Normal signal; **(B)** The signal including heating wire disconnection (HWD) fault; **(C)** The signal including aging of the heating wire (AHW) fault; **(D)** The signal including aging of the sensitive body (ASB) fault; **(E)** The signal including exfoliation of the sensitive body (ESB) fault; **(F)** The signal including false welding of the sensitive body (FWSB) fault.

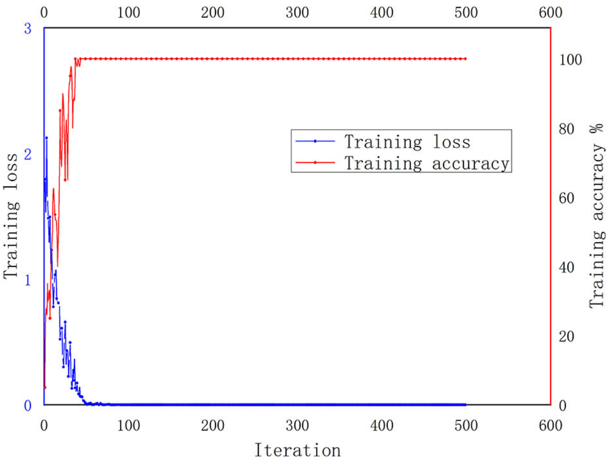


FIGURE 10 | The LeNet-5 method's training accuracy and loss from the source task under a normal environment.

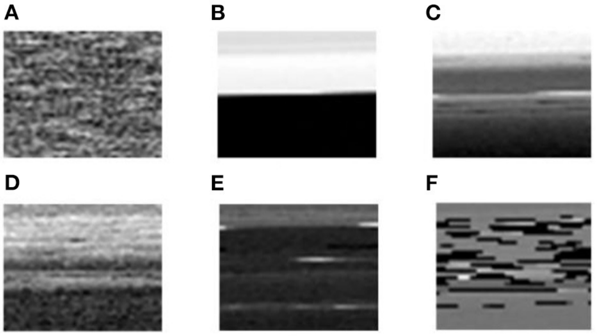


FIGURE 9 | Converted two-dimensional gray images of the gas sensor signals of six modes under a complex environment. **(A)** Normal signal; **(B)** The signal including heating wire disconnection (HWD) fault; **(C)** The signal including aging of the heating wire (AHW) fault; **(D)** The signal including aging of the sensitive body (ASB) fault; **(E)** The signal including exfoliation of the sensitive body (ESB) fault; **(F)** The signal including false welding of the sensitive body (FWSB) fault.

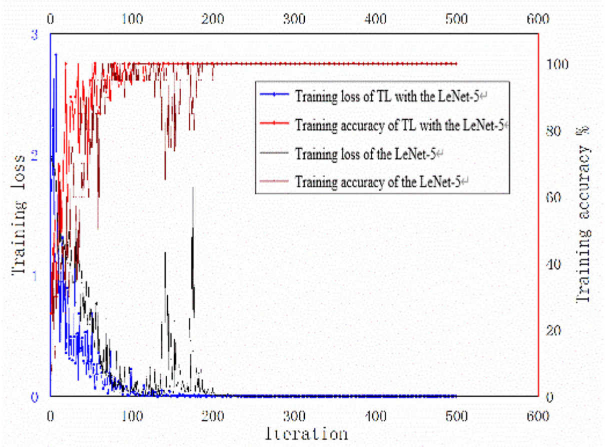


FIGURE 11 | Training accuracy and loss of transfer learning with the LeNet-5 method and the traditional LeNet-5 method under a complex environment for the target task.

diagnosis results of TL with LeNet-5 compared with the results of the traditional LeNet-5 without transfer, compared under a complex environment (in terms of accuracy). The total fault diagnosis accuracy of the traditional LeNet-5 was 88.48 ± 1.04%, while the total fault diagnosis accuracy of TL with LeNet-5 was 92.49 ± 1.28%. All the results of the fault diagnosis accuracy for different signal modes are shown in **Table 3**. The boxplot of total fault diagnosis accuracy is shown in **Figure 12**.

Discussion

In this study, the experimental conditions are limited, and only twenty-nine fault data are available for each fault under complex environment. The accuracy of fault diagnosis can be improved by using TL with LeNet-5 method. As shown in **Figure 11**, the convergence of the accuracy and the loss of the TL with LeNet-5 training were faster compared with the traditional LeNet-5 method without transfer. As can be seen in **Table 3**, transferring to different target task results in different performance. That is to say, the similarities of source task and target task could affect the performance of transfer learning.

Two other methods (Zhang P. et al., 2018) were added to comprehensively evaluate the performance: (1) using only the samples from normal environment to train the LeNet-5 model, and the same testing data as in the TL method were tested. (2) Using both the samples from the normal environment and the complex environment to directly train the LeNet-5 model (Without TL), and the same testing data as in the TL method were tested. The diagnostic results were 87.05% and 90.75%, respectively.

We also compared the TL with LeNet-5 method to traditional ML methods, such as LVQ (Bassiuny et al., 2007), ELM (the kernels is 116) (Song et al., 2019), SVM (the gamma

TABLE 3 | Fault diagnosis accuracy of the different methods.

Signals modes description	From normal environment to complex environment		
	Without transfer (%)*	Transfer learning (%)*	Improvements (%)*
Normal signal	100.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.00
HWD signal	88.53 ± 2.03	98.89 ± 3.39	10.36 ± 4.05
AHW signal	87.42 ± 4.82	88.90 ± 2.92	1.48 ± 3.84
ASB signal	88.16 ± 2.82	89.64 ± 2.82	1.48 ± 4.82
ESB signal	77.43 ± 2.03	78.54 ± 2.82	1.11 ± 3.39
FWSB signal	89.27 ± 2.03	99.26 ± 2.82	9.99 ± 3.39
Total	88.48 ± 1.04	92.49 ± 1.28	4.01 ± 1.61

*$\overline{X}$ ± *SD.*

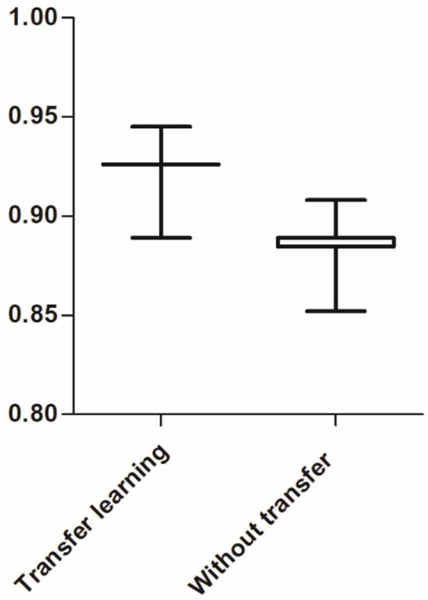


FIGURE 12 | Boxplot of fault diagnosis accuracy of the different methods.

value of polynomial kernel is 2) (Hu et al., 2005), KNN (Yang et al., 2016b), and RF (Mohapatra et al., 2020). All the experiments repeated 30 times, respectively, and all the results are presented in **Table 4**. The novel method had a higher accuracy than the traditional ML methods in a complex environment.

CONCLUSIONS AND FUTURE RESEARCHES

In this paper, a novel TL with LeNet-5 method was proposed for gas sensor fault diagnosis. The novel method has been validated by our self-made experimental system dataset. Traditional LeNet-5 without TL and other traditional ML methods were adopted for comparison.

TABLE 4 | Diagnosis accuracy based on seven different methods under a complex environment.

Methods	Accuracy (%)
LVQ	77.48 ± 1.12
ELM	79.50 ± 0.48
SVM	87.10 ± 0.92
KNN	85.19 ± 0.50
RF	88.01 ± 0.58
LeNet-5	88.48 ± 1.04
TL with LeNet-5	92.49 ± 1.28

In practice, there are usually abundant fault signal data under normal environmental conditions and limited fault signal data under complex environmental conditions. Furthermore, fault signal data in normal and complex environments might have different distributions. LeNet-5 improves the fault diagnosis accuracy of gas sensors in the same environment where the training data are abundant; however, it is not suitable for fault diagnosis in complex environments with limited training data. The experimental results show that the TL with LeNet-5 method could improve the accuracy of the fault diagnosis compared with the LeNet-5 without TL method and other traditional ML methods, which cannot take advantage of fault signal data in different distributions. The proposed method can provide a good fault diagnosis scheme for hydrogen sensors when only a small amount of fault data existing under complex environment.

The limitations of the proposed method is that, the common hydrogen sensor signal modes are needed to be represented in the dictionary list type. Otherwise the signal modes which have not been learned would be misclassified to be the known ones. Based on the limitation, the method can be modified to find an unknown signal mode in our future research work.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

YS designed the research and wrote the first draft of the manuscript. HZ helped to organize the manuscript. TZ, ZZ, and BS processed the data. HZ, SL, YY, and SZ revised the final version. The work presented in this study was carried out by all authors in collaboration with one another. All authors have read and approved the final manuscript.

FUNDING

This research was supported in part by the National Natural Science Foundation of China under Grant 62071138, in part by the Key Project of Education Department of Liaoning Province of China under Grant JYT2020LZ01, in part by the Key R&D plan of Liaoning Province of China under Grant 2019JH2/10300052, in part by the soft science research project of the Ministry of public security under Grant 2020LLYJLNST050, in part by the innovation team support plan of Liaoning Police College of China, in part by the teaching research project of Liaoning Police College of China under Grant JXYJ202007, in part by the Key R&D Projects of department of science and technology of Liaoning Province of China under Grant 2020JH2/10300109 and in part by the science and technology projects of Guizhou Province of China under Grant [2020] 2Y027.

REFERENCES

Bassiuny, A., Li, X., and Du, R. (2007). Fault diagnosis of stamping process based on empirical mode decomposition and learning vector quantization. *Int. J. Mach. Tools Manuf.* 47, 2298–2306. doi: 10.1016/j.ijmachtools.2007.06.006

Brown, D., Lewis, C., and Weinberger, B. I. (2015). Human exposure to unconventional natural gas development: a public health demonstration of periodic high exposure to chemical mixtures in ambient air. *J. Environ. Sci. Health Part A* 50, 460–472. doi: 10.1080/10934529.2015.992663

Caruan, R. (1997). Multitask learning. *Mach. Learn.* 28, 41–75. doi: 10.1023/A:1007379606734

Chalk, S., and Miller, J. F. (2006). Key challenges and recent progress in batteries, fuel cells, and hydrogen storage for clean energy systems. *J. Power Sources* 159, 73–80. doi: 10.1016/j.jpowsour.2006.04.058

Chen, X., Wang, S., Fu, B., Long, M., and Wang, J. (2019). "Catastrophic forgetting meets negative transfer: batch spectral shrinkage for safe transfer learning," in *Advances in Neural Information Processing Systems 32*, eds H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Red Hook, NY: Curran Associates, Inc.), 1906–1916.

Chen, Y., Yang, J., Xu, Y., Jiang, S., and Liu, X. (2016). Status self-validation of sensor arrays using gray forecasting model and bootstrap method. *IEEE Trans. Instrum. Meas.* 65, 1626–1640. doi: 10.1109/TIM.2016.2540942

Chen, Y. S., Xu, Y. H., Yang, J. L., Shi, Z., Jiang, S. D., and Wang, Q. (2016). Fault detection, isolation, and diagnosis of status self-validating gas sensor arrays. *Rev. Sci. Instrum.* 87:045001. doi: 10.1063/1.4944976

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., and Zhang, N. (2014). "A deep convolutional activation feature for generic visual recognition," in *International Conference on Machine Learning (ICML)* (Beijing), 647–655.

Fedorenko, G., Oleksenko, L., Maksymovych, N., Skolyar, G., and Ripko, O. (2017). Semiconductor gas sensors based on pd/so2 nanomaterials for methane detection in air. *Nanoscale Res. Lett.* 12:329. doi: 10.1186/s11671-017-2102-0

Gou, L., Li, H., Zheng, H., Li, H., and Pei, X. (2020). Aeroengine control system sensor fault diagnosis based on cwt and cnn. *Math. Probl. Eng.* 2020:5357146. doi: 10.1155/2020/5357146

Hu, Z. H., Cai, Y. Z., Li, Y. G., and Xu, X. M. (2005). Data fusion for fault diagnosis using multi-class support vector machines. *J. Zheijang Univ. Sci. A* 6, 1030–1039. doi: 10.1631/jzus.2005.A1030

Ingimundarson, A., Stefanopoulou, A. G., and McKay, D. A. (2008). Model-based detection of hydrogen leaks in a fuel cell stack. *IEEE Trans. Control Syst. Technol.* 16, 1004–1012. doi: 10.1109/TCST.2007.916311

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Commun. ACM* 60, 84–90. doi: 10.1145/3065386

LeCun, Y. (2015). *LeNet-5, Convolutional Neural Networks*. Available online at: https://yann.lecun.com/exdb/lenet.

Lu, S., Qian, G., He, Q., Liu, F., Liu, Y. B., and Wang, Q. J. (2019). *In situ* motor fault diagnosis using enhanced convolutional neural network in an embedded system. *IEEE Sens. J.* 20, 8287-8296. doi: 10.1109/JSEN.2019.2911299

Ma, G., Li, C., Luo, Y., Mu, R., and Wang, L. (2012). High sensitive and reliable fiber Bragg grating hydrogen sensor for fault detection of power transformer. *Sens. Actuators B Chem.* 169, 195–198. doi: 10.1016/j.snb.2012.04.066

Mohapatra, D., Subudhi, B., and Daniel, R. (2020). Real-time sensor fault detection in tokamak using different machine learning algorithms. *Fusion Eng. Des.* 151:111401. doi: 10.1016/j.fusengdes.2019.111401

Navi, M., Meskin, N., and Davoodi, M. (2018). Sensor fault detection and isolation of an industrial gas turbine using partial adaptive kpca. *J. Process Control* 64, 37–48. doi: 10.1016/j.jprocont.2018.02.002

Pan, J., and Yang, Q. (2009). A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* 22, 1345–1359. doi: 10.1109/TKDE.2009.191

Poirier, M., and Sapundzhiev, C. (1997). Catalytic decomposition of natural gas to hydrogen for fuel cell applications. *Int. J. Hydrogen Energy* 22, 429–433. doi: 10.1016/S0360-3199(96)00101-2

Saenko, K., Kuli, B., Fritz, M., and Darrell, T. (2010). "Adapting visual category models to new domains," in *European Conference on Computer Vision (ECCV)* (Heraklion), 213–226.

Shao, H., Jiang, H., Zhang, H., and Liang, T. C. (2018). Electric locomotive bearing fault diagnosis using a novel convolutional deep belief network. *IEEE Trans. Ind. Electron.* 65, 2727–2736. doi: 10.1109/TIE.2017.2745473

Song, K., Xu, P., Chen, Y., Zhang, T., Wei, G., and Wang, Q. (2019). A fault diagnosis and reconfiguration strategy for self-validating hydrogen sensor array based on mwpca and elm. *IEEE Access* 7,115075–115092. doi: 10.1109/ACCESS.2019.2936128

Staffell, I., Scamman, D., Abad, A. V., and Balcombe, P. (2019). The role of hydrogen and fuel cells in the global energy system. *Energy Environ. Sci.* 12, 463–491. doi: 10.1039/C8EE01157E

Sun, Y., Zhang, H., Zhao, T., Zou, Z., Shen, B., and Yang, L. (2020). A new convolutional neural network with random forest method for hydrogen sensor fault diagnosis. *IEEE Access* 8, 85421–85430. doi: 10.1109/ACCESS.2020.2992231

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2015). "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Boston, MA), 1–9.

Thrun, S. (1995). "Extracting rules from artificial neural networks with distributed representations," in *Advances in Neural Information Processing Systems* (Denver, CO), 505–512.

Tivive, F. H. C., and Bouzerdoum, A. (2005). "An eye feature detector based on convolutional neural network," in https://ieeexplore.ieee.org/xpl/conhome/10550/proceeding *Proceedings* of the Eighth International Symposium on Signal Processing and Its Applications (Sydney, NSW).

Tsujita, W., Yoshino, A., Ishida, H., and Moriizumi, T. (2005). Gas sensor network for air-pollution monitoring. *Sens. Actuators B Chem.* 110, 304–311. doi: 10.1016/j.snb.2005.02.008

Wang, Y., Pan, Z., Yuan, X., Yang, C., and Gui, W. (2020). A novel deep learning-based fault diagnosis approach for chemical process with extended deep belief network. *ISA Trans.* 96, 457–467. doi: 10.1016/j.isatra.2019.07.001

Wen, L., Gao, L., and Li, X. (2017a). A new deep transfer learning based on sparse auto-encoder for fault diagnosis. *IEEE Trans. Syst. Man Cybern. Syst.* 49, 136–144. doi: 10.1109/TSMC.2017.2754287

Wen, L., Li, X., and Gao, L. (2017b). A transfer convolutional neural network for fault diagnosis based on resnet-50. *Neural Comput. Appl.* 32, 6111–6124. doi: 10.1007/s00521-019-04097-w

Wen, L., Li, X., Gao, L., and Zhang, Y. (2018). A new convolutional neural network-based data-driven fault diagnosis method. *IEEE Trans. Ind. Electron.* 65, 5990–5998. doi: 10.1109/TIE.2017.2774777

Winter, C. J. (2005). Into the hydrogen energy economy-milestones. *Int. J. Hydrogen Energy* 30, 681–685. doi: 10.1016/j.ijhydene.2004.12.011

Wu, H., and Zhao, J. (2018). Deep convolutional neural network model based chemical process fault diagnosis. *Comput. Chem. Eng.* 115, 185–197. doi: 10.1016/j.compchemeng.2018.04.009

Wu, Z., Jiang, H., Zhao, K., and Li, X. (2020). An adaptive deep transfer learning method for bearing fault diagnosis. *Measurement* 151:107227. doi: 10.1016/j.measurement.2019.107227

Yang, J., Chen, Y., Zhang, L., and Sun, Z. (2016a). Fault detection, isolation, and diagnosis of self-validating multifunctional sensors. *Rev. Sci. Instrum.* 87:065004. doi: 10.1063/1.4954184

Yang, J., Sun, Z., and Chen, Y. (2016b). Fault detection using the clustering-kNN rule for gas sensor arrays. *Sensors* 16:2069. doi: 10.3390/s16122069

Zhang, P., Ma, X., Chen, L., Zhou, J., and Wang, C. (2018). Decoder calibration with ultra small current sample set for intracortical brainmachine interface. *J. Neural Eng.* 15:026019. doi: 10.1088/1741-2552/aaa8a4

Zhang, Q., Zhou, Q., Lu, Z., Wei, Z., Xu, L., and Gui, Y. (2018). Recent advances of SnO_2-based sensors for detecting fault characteristic gases extracted from power transformer oil. *Front. Chem.* 6:364. doi: 10.3389/fchem.2018.00364

Zhang, W., Li, C., Peng, G., Chen, Y., and Zhang, Z. (2018). A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load. *Mech. Syst. Signal Process.* 100, 439–453. doi: 10.1016/j.ymssp.2017.06.022

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

BRIEF RESEARCH REPORT
published: 28 April 2022
doi: 10.3389/fnbot.2022.857595

Return Strategy and Machine Learning Optimization of Tennis Sports Robot for Human Motion Recognition

Yuxuan Wang[1,2], *Xiaoming Yang*[3,4], *Lili Wang*[4], *Zheng Hong*[5] *and Wenjun Zou*[1]*

[1] Sports Institute, Nanchang JiaoTong Institute, Nanchang, China, [2] Graduate School, University of Perpetual Help System Dalta, Las Piñas, Philippines, [3] Faculty of Educational Studies, Universiti Putra Malaysia, Kuala Lumpur, Malaysia, [4] College of Physical Education, East China University of Technology, Nanchang, China, [5] School of Software, Nanchang University, Nanchang, China

At present, there are many kinds of intelligent training equipment in tennis sports, but they all need human control. If a single tennis player uses the robot to return the ball, it will save some human resources. This study aims to improve the recognition rate of tennis sports robots in the return action and the return strategy. The human-oriented motion recognition of the tennis sports robot is taken as the starting point to recognize and analyze the return action of the tennis sports robot. The OpenPose traversal dataset is used to recognize and extract human motion features of tennis sports robots under different classifications. According to the return characteristics of the tennis sports robot, the method of tennis return strategy based on the support vector machine (SVM) is established, and the SVM algorithm in machine learning is optimized. Finally, the return strategy of tennis sports robots under eight return actions is analyzed and studied. The results reveal that the tennis sports robot based on the SVM-Optimization (SVM-O) algorithm has the highest return recognition rate, and the average return recognition rate is 88.61%. The error rates of the backswing, forward swing, and volatilization are high in the return strategy of tennis sports robots. The preparation action, backswing, and volatilization can achieve more objective results in the analysis of the return strategy, which is more than 90%. With the increase of iteration times, the effect of the model simulation experiment based on SVM-O is the best. It suggests that the algorithm proposed has a reliable accuracy of the return strategy of tennis sports robots, which meets the research requirements. Human motion recognition is integrated with the return motion of tennis sports robots. The application of the SVM-O algorithm to the return action recognition of tennis sports robots has good practicability in the return action recognition of tennis sports robot and solves the problem that the optimization algorithm cannot be applied to the real-time requirements. It has important research significance for the application of an optimized SVM algorithm in sports action recognition.

Keywords: OpenPose traversal dataset, human motion characteristics, machine learning, support vector machine algorithm, human motion recognition

OPEN ACCESS

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
Yaodong Gu,
Ningbo University, China
Mustafa Sögüt,
Middle East Technical University, Turkey
Jose Maria Gimenez-Egido,
University of Murcia, Spain
Ersan Arslan,
Tokat Gaziosmanpasa University, Turkey
Ruey-Shun Chen,
National Chiao Tung University, Taiwan

****Correspondence:***
Wenjun Zou
06005@ncjti.edu.cn

Received: *18 January 2022*
Accepted: *18 March 2022*
Published: *28 April 2022*

Citation:
Wang Y, Yang X, Wang L, Hong Z and Zou W (2022) Return Strategy and Machine Learning Optimization of Tennis Sports Robot for Human Motion Recognition.
Front. Neurorobot. 16:857595.
doi: 10.3389/fnbot.2022.857595

RESEARCH PURPOSE AND CURRENT SITUATION

With the progress of computer technology and its wide application in life practice, behavior detection and action recognition projects based on various algorithms are being applied and studied in related fields (Aslan and Durdu, 2020). Behavior monitoring includes bee colony behavior and human behavior monitoring through graphics, temperature, humidity, sound, and other information (Jalal and Kamal, 2019). Introducing human motion recognition and motion intention prediction into the machine can bring the users a better sense of experience. People and robots coexist or work together. On the premise of ensuring that robots do not harm humans, it is necessary to explore efficient human–machine cooperation schemes to complement the advantages of humans and robots (Kahanowich, 2021). Generally, human motion is obtained in the form of a video or image and then recognized. With the development of science and technology and the electronic industry, wearable sensor devices are being used to recognize human actions (Hou et al., 2019; Jalal and Quaid, 2019; Zhao, 2020). With the growth of wireless technology and the expansion of coverage, wireless fidelity (WiFi) signal is being used to recognize human motion, and good results have been achieved, which is also the latest research trend (Zhao et al., 2019; Zhu et al., 2021). Generally, data are first collected, then denoised, and processed in human motion recognition. Then, feature quantity is extracted, trained, and classified to realize the final recognition. Almost all research teams achieve human motion recognition according to this general process. In these five parts, data denoising and feature extraction are the two key links (Gurbuz, 2019; Wang and Zhang, 2019; Xiong et al., 2020). Researchers are deeply improving and developing these two links to improve recognition accuracy.

The training law of tennis strategy and tactics is a difficult work faced by coaches and players. In the evaluation study of tennis serve and return practice, Krause et al. (2019) found that every hitting technique and step running position of the opponent could determine the return quality. Athletes also need to combine their own playing characteristics and styles in peacetime training to figure out every technical detail of the opponent to benefit from it (Krause et al., 2019). People hope to find a method that can be applied to all players to make them adopt strategies and tactics in a broader range and find weaknesses through analysis. Human motion recognition is a new human–computer interaction mode. It extracts and classifies human action features through computer vision technology, identifies human actions, obtains action information, and makes the machine "read" human body language (Oudah and Al-Naji, 2020; Pourdarbani et al., 2020; Yang et al., 2021). At present, significant progress has been made in the research and application of machine learning represented by deep learning (DL), which has effectively promoted the development of artificial intelligence. DL is a method of machine learning. In 2010, a large image dataset named ImageNet appeared. DL requires a lot of computational power, so some researchers combine the central processing unit (CPU) to train the DL model, which has been integrated into the current research method. Multiple researchers do various experiments through the standard datasets of machine learning and promote the research process by comparing the different methods. Giles and Kovalchik (2020) studied the direction change in the tracking data of professional tennis players and found that the support vector machine (SVM) algorithm tested the non-linear kernel method. The nearest neighbor method was used to test the simple neural network. It was usually recommended to experiment with the data of some shapes (Giles and Kovalchik, 2020; Kunze et al., 2020). ImageNet is one of the most influential datasets, which effectively promotes the development of the DL model (Alexopoulos and Nikolakis, 2020). Moreover, according to the tennis sports robots' return action selection and return strategy characteristics, there are many other datasets besides the ImageNet dataset. Based on that, many researchers also put forward new questions. These datasets also promote the development of relevant research (Liu et al., 2019).

For the problem of reduced human resources in tennis training, the robot can return the ball according to human actions. With the human motion recognition in the return strategy of tennis sports robots as the main starting point, this study extracts the human motion features through the tennis sports information under different classifications and the OpenPose traversal dataset. The return strategy of tennis sports robots is studied through machine learning. Besides, this study also focuses on the SVM algorithm optimization in machine learning. The return strategy of the tennis sports robot is optimized by the SVM-optimization (SVM-O) algorithm. This study provides a reference basis for robot return strategy in tennis sports and has crucial research significance.

Section Research Purpose and Current Situation introduces the research background of the machine learning algorithm of tennis robots for human motion recognition, which paves the way for the development of the SVM algorithm. Section Literature Review discusses the worldwide research on ball return strategy and human motion recognition. Section Research on Machine Learning and Related Sports Robots introduces machine learning and robots on the playground. Section Design of the SVM-O Model introduces the SVM algorithm and the optimization process of SVM. Section Research Model and Framework mainly aims at the ball return action recognition and ball return strategy design of the tennis robot. Section Results and Discussion analyzes the regression error rate, regression accuracy rate, regression recall rate, regression score rate, and regression recognition rate of eight actions in the tennis robot regression strategy and compares the regression accuracy rate of tennis robot under different model training. Finally, the whole research is summarized and analyzed, and the research limitations are put forward.

LITERATURE REVIEW

According to the research on the return strategy, Sharma and Kumar (2021) designed the badminton robot through the learning method of the database and obtained the movement track of badminton through the binocular camera. However, due to the fixed base and position of the robot, the receiving distance was short when returning the ball. Yunardi et al. (2021) designed an omnidirectional mobile tennis robot. Through the design of a robot manipulator, the robot could return the ball at different angles and forces and could move in any direction and return the ball at different angles on the playground. Based on the fully connected neural network, the return strategy of the badminton robot was studied. The neural network was optimized through the activation function and residual connection of the neurons. The training was designed according to the return point of badminton as the input value. However, only the coordinates of the return point of the badminton in returning the ball were researched. The return type and the return action of the opponent were not taken into account, so it was difficult to return the ball for the action of the robot facing the human body (Gao et al., 2020). In the data-processing stage, three-dimensional (3D) is much more complex than two-dimensional (2D). The 2D human pose recognition is more mature than 3D in terms of data and models. The 2D models also have a lot of outdoor and natural data, but almost all 3D data have indoor data. Because of the complexity of the 3D annotation and recognition, massive sensors and cameras are needed to collect data. Before 2015, the regression method was widely used to confirm the coordinates of human-joint points. Its innovation was to extract them from the 3 and 7 layers of convolutional neural network (CNN) and then conduct the convolution operation, which is called the spatial fusion model. Faisal et al. (2019) used the spatial fusion model to extract the internal relationship between joint points, but this method was difficult to expand in the model. The critical path method (CPM) proposed in 2016 has strong robustness, and many subsequent methods are improvements based on this method.

The contribution of the CPM is to use sequential convolution architecture to express spatial and texture information. The network is divided into several stages, and each stage has a part in supervision and training. The previous stage uses the original image as the input. The latter uses the feature image of the previous stage as the input, mainly to integrate spatial information, texture information, and center constraints. In addition, Li et al. (2020) simultaneously used multiple scales to process the input characteristics and responses for the same convolution architecture, ensuring accuracy and considering the distance relationship between the various components. Fine-tuning was conducted based on CNN. Its innovation lies in the use of a geometric transformation kernel in the convolution layer, which can model the dependence between the joint points. Moreover, a bidirectional tree model was proposed, so that the feature channel of each joint could receive the information of the other joints, which was called information transmission. This tree structure can also estimate the attitude of multiple people. However, the accuracy of this multi-person attitude estimation is not high, and the method based on a single person is better. OpenPose is a framework for real-time estimation of the human body, face, and hand morphology proposed by the cognitive computing laboratory of Carnegie Mellon University (Jaruenpunyasak et al., 2022). It provides 2D and 3D multi-person keypoint detection and a calibration toolbox for estimating specific area parameters. It can accept many kinds of input, including images, videos, and webcams. Similarly, the output of OpenPose is also diverse. The input and output parameters can also be adjusted according to different needs. At present, the sensor technology of human motion recognition takes the human motion posture database as the action classifier, obtains the image of human motion through the sensor, subdivides it according to the motion angle and speed, and finally classifies the information with the classifier.

This thesis uses the OpenPose traversal dataset to extract the information features of human motion recognition nodes and extracts human motion features through tennis motion information under different classifications and the OpenPose traversal dataset. The return strategy of the tennis robot is studied by machine learning and optimized by the SVM-O algorithm.

RESEARCH ON MACHINE LEARNING AND RELATED SPORTS ROBOTS

Algorithm Design of Machine Learning

Machine learning is the general name of a class of algorithms. These algorithms intend to mine the hidden laws from massive historical data and use them for prediction or classification. More specifically, machine learning can be seen as looking for a function, whose input is the sample data and the output is the expected result. However, this function is too complex to be expressed formally (Mohabatkar and Ebrahimi, 2021). It should be noted that the goal of machine learning is to make the learned functions well applicable to the new sample data, not just perform well on the training samples (Ghorbanzadeh et al., 2019). The ability of the learned function to apply to new samples is called generalization ability. **Figure 1** shows the specific algorithm steps of machine learning.

The machine learning steps in **Figure 1** are divided into the following three steps. First, an appropriate model is selected, which usually depends on the actual problem. Suitable models need to be selected for different problems and tasks. The model is a set of functions. Next, the quality of a function is judged, which needs to determine a measurement standard, that is, the loss function. The determination of loss function also depends on specific problems. For example, Euclidean distance is generally used in regression problems, and the cross-entropy cost function is generally used in classification problems. Finally, the "best" function is found. The commonly used methods include gradient descent algorithm, ordinary least squares, and other tricks. The "best" function needs to be tested on a new sample after it is learned. It is a good function only if it performs well on the new sample. Machine learning is a huge family system, involving multiple algorithms, tasks, and learning theories. **Figure 2** is the learning roadmap of machine learning.

In **Figure 2**, different colors represent different learning theories—orange represents tasks and green represents methods. According to the task types, machine learning models can be divided into regression, classification, and structure. The regression model is also called the prediction model. The

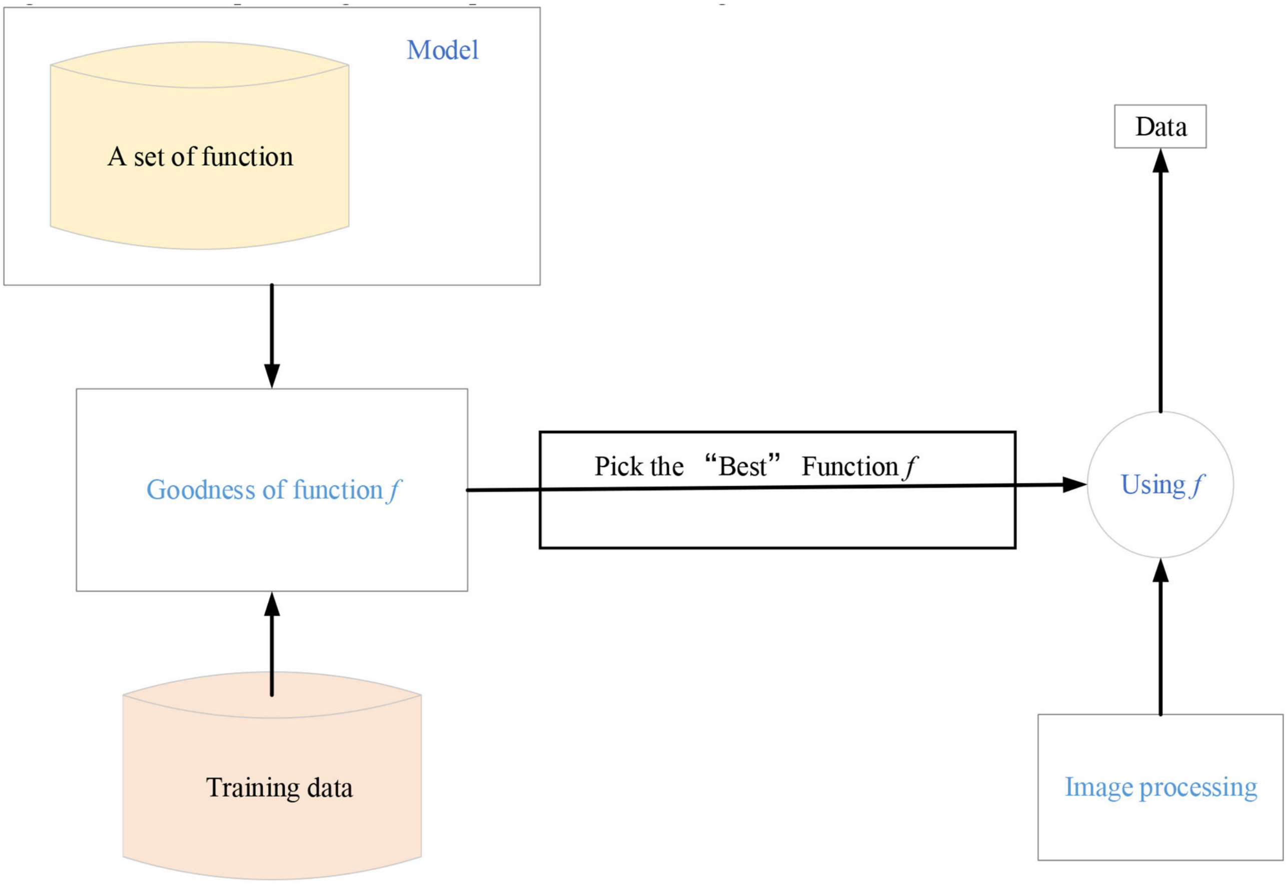


FIGURE 1 | Machine learning steps.

output is a numerical value that cannot be enumerated. The classification models are divided into binary classification and multi-classification. The common binary classification problems are spam filtering, and the common multi-classification problems are automatic document classification. The output of the structured learning model is no longer a value with the fixed length, but the text description of the figure. From the perspective of the method, it can be divided into linear and non-linear models. The linear model is relatively simple, but its role cannot be ignored. It is the basis of the non-linear model. Many non-linear models are transformed from the linear model. The non-linear models can be divided into traditional machine learning models, such as SVM, K-nearest Neighbor (KNN), decision tree, and DL model. According to the learning theory, the machine learning model can be divided into supervised, semi-supervised, unsupervised, transfer, and reinforcement learning. When the training sample is labeled, it is supervised learning. When part of the training sample is labeled and part of the training sample is not labeled, it is semi-supervised learning. When all training samples are unlabeled, it is unsupervised learning. Transfer learning is to transfer the trained model parameters to the new model to help in the new model training. Reinforcement learning is to learn the optimal policy, which enables an agent to act according to the current state in a specific environment to obtain the maximum reward. The most significant difference between reinforcement learning and supervised learning is that each decision in reinforcement learning does not define right or wrong, but aims to get the most cumulative rewards.

Robots on the Playground

Robot technology became the hottest topic in the investment circle together with unmanned aerial vehicle technology as early as 5 years ago. However, this topic was not pushed to the sports world until the "human–machine war" between Google AlphaGo and South Korean chess player Li Shishi.

The Japanese Volleyball Association uses robots to practice with athletes to improve the level of the athletes. This "defense robot" has three pairs of mechanical arms, which can move from one end to the other in front of the net to imitate the action of defense when smashing the attacker. The defensive robot is studied by the Japanese Volleyball Association and the University of Tsukuba. The coach can manipulate the robot arm back and forth according to the training needs. If the coach thinks the team can perform better in a past game, the robot can reproduce the game's situation at that time. The robot can imitate the possible actions of the opponent on the field in the future according to the

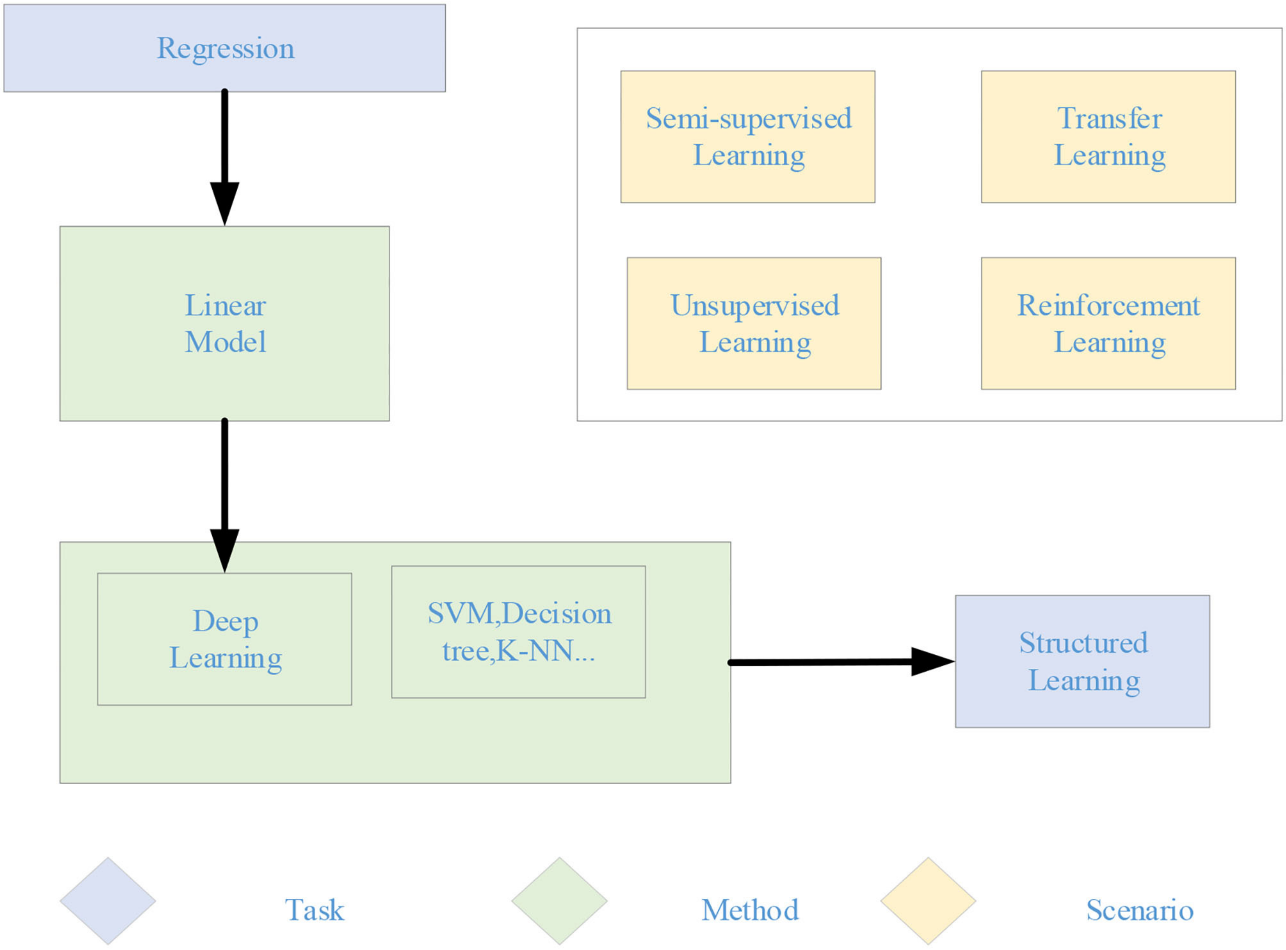


FIGURE 2 | Roadmap for machine learning.

strategic style of the opponent in the future. After the training starts, the coach only needs to press a button, and the robot's mechanical arm will quickly reach the designated position. The moving speed of this robot can reach 3.7 m/s, which is faster than that of the ordinary athletes running on the field.

Robomintoner, developed by Chengdu Electric Technology Chuangpin Robot Technology Co., Ltd., has reached the level of ordinary badminton lovers. Its technical core is the positioning and navigation of the whole field, the visual tracking and recognition of high-speed moving objects, and the control of the motion system. The robot mainly recognizes the trajectory of the badminton through binocular vision and predicts the landing point. It will tell the motion system the landing point of the badminton through Bluetooth communication, and the robot will move to the position where the badminton will land in advance.

The track and field training robots have a speed of 44.6 km/h, which is comparable to the world record of Usain Bolt in 2009. The main function is to see the difference between the user's current performance and the previous performance in real-time by simulating the runner's previous best performance during training to help the runner improve the performance. In terms of configuration, this small remote-controlled machine car has nine infrared sensors, drives small wheels through an Arduino board, and is equipped with a light-emitting diode (LED) light and a GoPro camera on the body.

The table tennis serving robot users can control the five built-in motors of the equipment in the supporting application of Trainerbot and set the serving into the up rotation, down rotation, side rotation, and random modes. The Trainerbot table tennis robot is 32 cm high and weighs only 1.2 kg. It can accommodate up to 30 table tennis balls. It can serve one ball every 0.5 s at the fastest and every 3 s at the slowest. Moreover, the table tennis coach robot can be used to recognize the ball's movement. The camera can track the position of table tennis 80 times per s, including ball speed, rotation speed, rotation direction, and other data. The sensor can analyze the speed of the ball 1,000 times per s and predict where the ball will land. Forpheus can calculate the angle and point at which the racket should hit back, with an error of <5 cm. Of course, the goal of Forpheus is to help mankind improve the level of table tennis. Therefore, in addition to tracking the table tennis ball, the landing

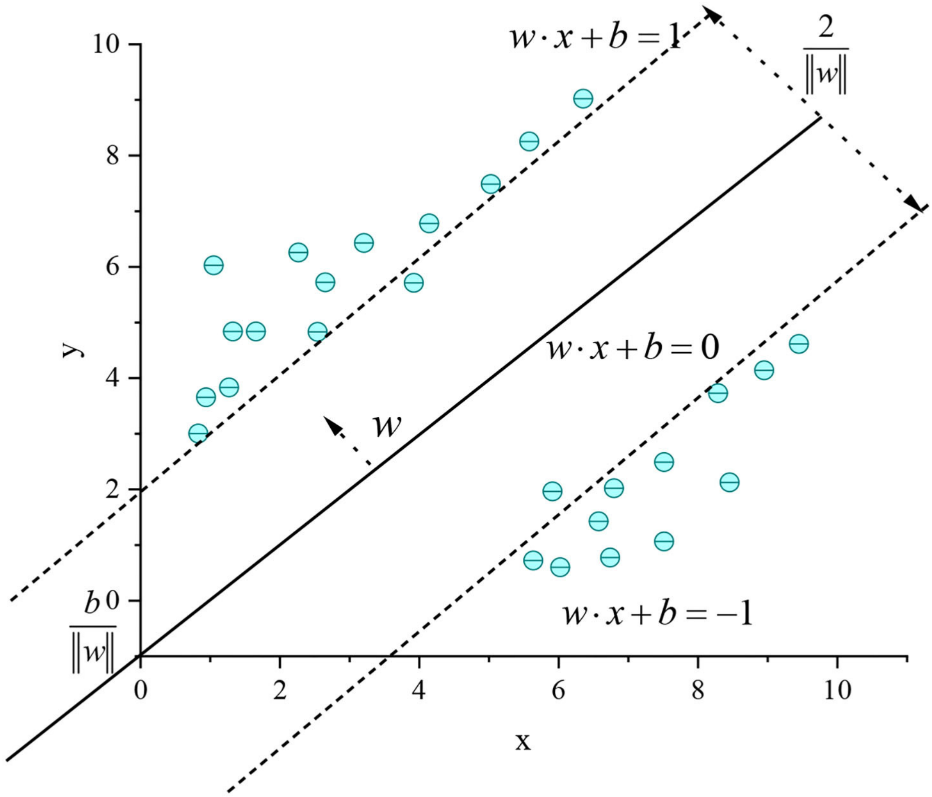


FIGURE 3 | SVM regression model.

point of the table tennis ball will be displayed by projection to help the athletes adjust their movements.

The tennis collection robot itself is equipped with advanced sensors. It detects, locates, and collects tennis balls by using the onboard computer to analyze the collected information and a wide-angle camera. In addition to the configured sensors, it also has a "Tennibot site" installed on the tennis net post. The site is a camera used to track the robot's position, which is connected with the robot through wireless communication. With the site to detect the robot's position, the robot's supporting iPhone Operating System (iOS) and Android application (APP) can be used to control which area of the playground it drives, set specific routes, or manual control. It can even be operated on an Apple watch, which users can choose. It is to avoid the robot appearing at the foot and affecting the playing when the user plays. The speed of collecting the tennis balls is 1.4 miles/h, about 38 m/min. A basket can collect 80 tennis balls, with a battery life of 4–5 h and only 90 min of charging each time, which can meet the needs of long-term practice. After use, the robot can be lifted and transported away through the handle like a suitcase.

DESIGN OF THE SVM-O MODEL

Support vector machine is a binary classification model, which is initially used in the case of linear separability. According to the diversity of the human actions studied, more action samples can be classified reasonably and as quickly as possible. This thesis optimizes the SVM sample classification process.

Given the training samples, the basic idea of the SVM algorithm classification learning is to find a partition hyperplane in the sample space based on the training samples to separate the samples of different categories. However, many partition hyperplanes can separate the training samples, and the most reliable one needs to be found (Shi, 2020). Intuitively, the one in the middle of the training sample should be found, because the partition hyperplane has the best tolerance to the local disturbance of the training sample. It means that the classification result of the partition hyperplane is the most robust and has the strongest generalization ability to the unseen examples. **Figure 3** shows the SVM regression model.

In **Figure 3**, the hyperplane in the sample space can be divided by a linear equation $w \cdot x + b = 0$, where w is the normal vector of the hyperplane partition, which determines the direction of the hyperplane; b is the displacement term, which determines the distance between the hyperplane and the origin. The partition hyperplane can be determined by w and b. The interval between two heterogeneous support vectors and the hyperplane is $\gamma = \frac{2}{\|w\|}$. In tennis sports, the robot return sample can be mapped from the original to a high-dimensional feature space, so that the sample can be linearly separable in this feature space. If the original space has a finite number of attributes, there must be a high-dimensional feature space to divide the return samples.

$\phi(x)$ is set as the feature vector after x is mapped to the high dimension. Then, the model corresponding to $\phi(x)$ dividing the hyperplane in the feature space can be expressed as:

$$f(x) = w^T\phi(x) + b \tag{1}$$

w and b are model parameters. There is

$$\min_{(w,b)}\frac{1}{2}\|w\|^2, st, y_i(w^T\phi(x_i) + b) \geq 1, i = 1, 2, ..., m \tag{2}$$

Equation (3) is a dual problem.

$$\max_\alpha\left(\sum_{i=1}^{m} a_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \phi(x_i)^T\phi(x_j)\right)$$
$$s.t. \sum_{i=1}^{m}\alpha_i y_i = 0, \alpha_i \geq 0, i = 1, 2, ..., m \tag{3}$$

$\phi(x_i)^T\phi(x_j)$ is the inner product after the training samples x_i and x_j are mapped to the feature space. The dimension of the feature space may be very high or even infinite. Therefore, the direct calculation is difficult, and a kernel function can be envisaged:

$$k(x_i, x_j) \leq \phi(x_i), \phi(x_j) \geq \phi(x_i)^T\phi(x_j) \tag{4}$$

Equation (4) is the inner product of x_i and x_j in feature space. It is equal to their calculation by function $k(\cdot,\cdot)$ in the original sample space. Equation (4) is solved to obtain the following:

$$f(x) = w^T\phi(x) + b = \sum_{i=1}^{m}\alpha_i y_i \phi(x_i)^T\phi(x) + b$$
$$= \sum_{i=1}^{m}\alpha_i y_i k(x, x_j) + b \tag{5}$$

Common kernel function: Equation (6) is the Gaussian kernel function; σ is the bandwidth of the Gaussian kernel.

$$k(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|}{2\sigma^2}), \sigma > 0 \tag{6}$$

Equation (7) is the Laplace kernel function:

$$k(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|}{\sigma}), \sigma > 0 \tag{7}$$

Equation (8) is the Sigmoid kernel function; tanh is a hyperbolic tangent function.

$$k(x_i, x_j) = \tanh(\beta x_i^T x_j + \theta), \beta > 0, \theta < 0 \tag{8}$$

All of the above samples must be divided correctly. The samples that do not meet the constraints can be optimized

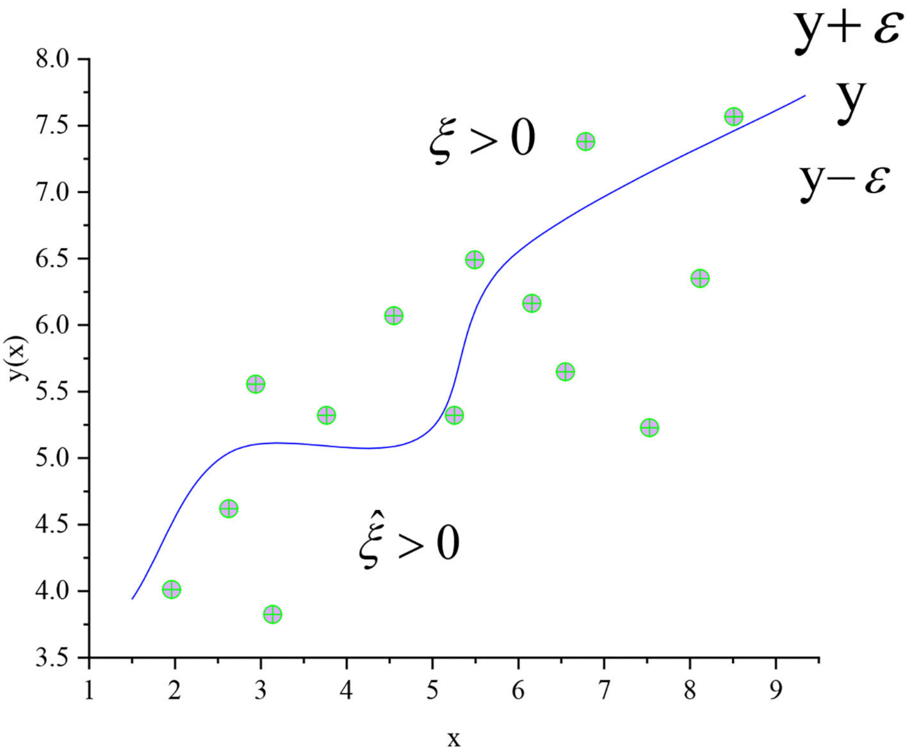


FIGURE 4 | Regression curve of SVM-O.

while maximizing the interval to reduce the error. The most commonly used error function is the least square sum error function. **Figure 4** shows the optimization effect of the SVM regression curve.

In **Figure 4**, the regression curve of the SVM in **Figure 3** is optimized. C is the regularization coefficient and the relaxation variable ξ is introduced. In the regression problem, each data point x_n needs two relaxation variables $\xi_n \geq 0$ and $\hat{\xi}_n \geq 0$. $\xi_n \geq 0$ corresponds to $t_n > y(x_n) + \varepsilon$ (above the curve in **Figure 4**) and corresponds to $t_n < y(x_n) + \varepsilon$ (below the curve in **Figure 4**). Equation (9) is a function that minimizes the regularization error. Equations (10) and (11) are the corresponding conditions.

$$C\sum_{n=1}^{N} E_\varepsilon(y(x) - t_n) + \frac{1}{2}\|w\|^2 \tag{9}$$

$$\begin{aligned} t_n &\leq y(x_n) + \varepsilon + \hat{\xi}_n \\ t_n &\geq y(x_n) - \varepsilon - \hat{\xi}_n \end{aligned} \tag{10}$$

The updated error function is as follows:

$$C\sum_{n=1}^{N} (\xi_n + \hat{\xi}_n) + \frac{1}{2}\|w\|^2 \tag{11}$$

The Lagrange multiplier method is introduced to optimize the equation with constraints and minimize the error function:

$$\begin{aligned} L = C\textstyle\sum_{n=1}^{N} (\xi_n + \hat{\xi}_n) + \frac{1}{2}\|w\|^2 - \sum_{n=1} N(\mu_n\xi_n + \hat{\mu}_n\hat{\xi}_n) \\ - \textstyle\sum_{n=1}^{N} a_n(\varepsilon + \xi_n + y_n - t_n) - \sum_{n=1}^{N} \hat{a}_n(\varepsilon + \hat{\xi}_n - y_n + t_n) \end{aligned} \tag{12}$$

By finding the derivative of the Lagrange function for w, b, ξ_n and $\hat{\xi}_n$ as 0, it can be obtained that

$$\begin{aligned} &\tfrac{\delta L}{\delta w} = 0 \Rightarrow \textstyle\sum_{n=1}^{N} (a_n - \tilde{a}_n)\phi(x_n) \\ &\tfrac{\delta L}{\delta b} = 0 \Rightarrow \textstyle\sum_{n=1}^{N} (a_n - \tilde{a}_n) = 0 \\ &\tfrac{\delta L}{\delta \xi_n} = 0 \Rightarrow a_n + \mu_n = C, \tfrac{\delta L}{\delta \hat{\xi}_n} = 0 \Rightarrow \tilde{a}_n + \tilde{\mu}_n = C \end{aligned} \tag{13}$$

The new input variable can be obtained by using the equation below:

$$\begin{aligned} y(x) &= \sum_{n=1}^{N} (a_n - \tilde{a}_n)\phi(x_n)\phi(x) + b \\ &= \sum_{n=1}^{N} (a_n - \tilde{a}_n)k(x_n, x) + b \end{aligned} \tag{14}$$

The value of b in Equation (1) can be obtained from Equation (15):

$$b = t_n - \varepsilon - w^T\phi(x_n) = t_n - \varepsilon - \sum_{m=1}^{N} (a_m - \tilde{a}_m)k(x_n, x_m) \tag{15}$$

TABLE 1 | Analysis of tactical action characteristics of return ball of tennis sports robot.

Return strategy	Characteristic	Optimization objectives
Continuous return	Continuity	Change different ball speeds, forces and directions
Return with more strengths	Depth continuity, velocity	Return quickly
Moving return	Continuous power	Improve return accuracy and continuity
Return with endurance and defense	High endurance	Slash and straight ball
Changeable return	Change the speed, direction and kinetic energy of the ball	Optimize the rotation and direction of the shot
Attack front court	Aggressive	Return forehand, backhand

Another method for SVM regression does not fix the width of the insensitive area ε but fixes the proportion v of the data points outside the pipeline. The maximization equation can prove that at most vN data points fall outside the insensitive pipeline, and at least vN data points are support vectors on or outside the pipeline. The value of vC is generally determined by cross-validation.

$$\begin{aligned} &\tilde{L}(a, \tilde{a}_n) = -\tfrac{1}{2}\textstyle\sum_{n=1}^{N}\sum_{m=1}^{N} (a_n - \tilde{a}_n)(a_m, -\tilde{a}_m)k(x_n, x_m) + \sum_{n=1}^{N} (a_n - \tilde{a}_n)t_n \\ &0 \leq a_n \leq \tfrac{C}{N}, 0 \leq \tilde{a}_n \leq \tfrac{C}{N}, \textstyle\sum_{n=1}^{N} (a_n - \tilde{a}_n) = 0, \sum_{n=1}^{N} (a_n + \tilde{a}_n) \leq vC \end{aligned} \tag{16}$$

RESEARCH MODEL AND FRAMEWORK

Recognition of Return Motion of Tennis Sports Robot

The robot return in tennis sports cannot be realized only with a heavy swing. It needs to study the combination of human action routines and constantly return the ball to the other party. For example, in diagonal backhand, the robot needs to catch the opportunity ball that the opponent's oblique return angle is not good, the ball speed is not fast, and the ball falls in the center of the court or in a comfortable position in front of the body. The robot's backhand position can be changed to the forehand to play the diagonal ball and return the ball effectively. These return methods provide the structure of tennis return tactics and can be used to teach tennis sports robot return. **Table 1** is the analysis of the return tactics of tennis sports robot.

According to **Table 1** and the high-performance camera processing of the tennis robot, the robot responds to the falling point of the opponent's returning the ball. The tennis movement is set as the return movement of 8 tennis sports robots, which are preparation action, swing, preparation position, backswing, forward swing, hitting, follow-up swing, and return to preparation position (Haryanto,

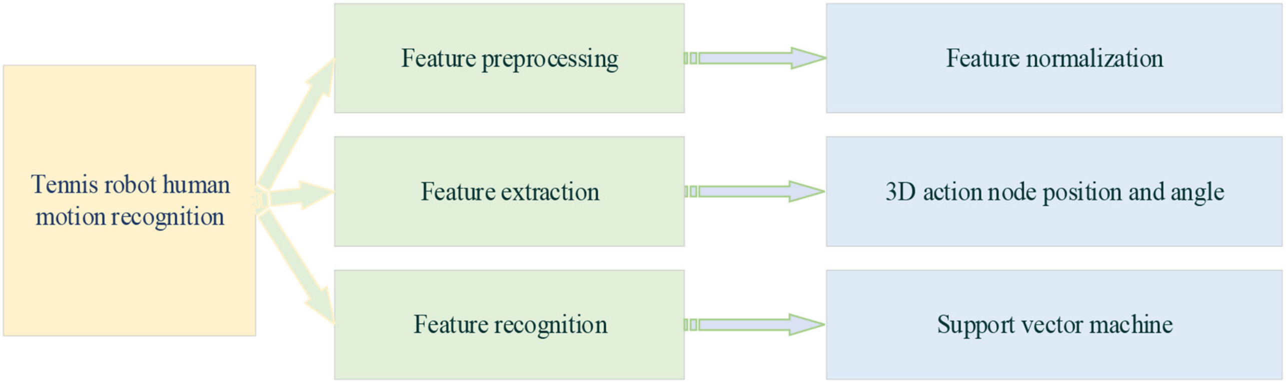


FIGURE 5 | Motion recognition of tennis sports robots.

2020). In tennis robots' data-processing stage of return motion recognition, 3D human motion recognition is much more complex than 2D. Two-dimensional human motion recognition is more mature than 3D in terms of data and models. 2D models also have many outdoor and natural datasets. However, almost all 3D human motion recognition datasets are indoor. Because of the complexity of 3D annotation and recognition, many sensors and cameras are needed to collect data. **Figure 5** is a motion recognition system of a tennis sports robot.

Figure 5 is the dataset feature extraction of the return strategy of tennis sports robots for human motion recognition. First, human motion recognition is carried out, followed by the extraction of motion features. The collected serving action images of the human body in tennis sports are put into the OpenPose skeleton extraction network to extract the keypoint coordinate data of human action. Then, the human action information under different classifications is extracted as action features and saved as the corresponding text (TXT) documents (Neff et al., 2019). Then, the features are integrated. The extracted feature information is integrated with the corresponding images in a TXT file, and the useless and redundant datasets are removed simultaneously. Finally, the TXT information is integrated as input and output tag comma-separated value (CSV) files, respectively. Among them, the input features include the key points of human service and the line features connected by different bone points and the surface features formed by the combination of different lines. These features will be extracted and learned through different classification algorithms.

In a convolutional neural network, the human motion figures are put into the network in the matrix form. The x, y, and z axes are taken as red, green, blue (RGB)—three channels of the figures. Then, the data are arranged in a row to get a vector with n columns. However, for convenience, n is decomposed. Every 100 groups of data form a return action figure of a tennis sports robot, that is, 1*100*3 represents a return action figure of a tennis sports robot. Of course, if the input is modified, the label file should also be modified accordingly. Before modification, each piece of data will have a label. However, after modification, every 100 pieces of data only need to output one label. Shell is adopted to process label files. Finally, the eight return motions in the label file are replaced with numbers from 1 to 8. The human bone information read from the input CSV is taken as the input and the CSV file of the Y label as the output. The training and verification sets are divided according to the proportion of 0.3. Then, they are converted into a NumPy matrix to participate in the operation. The decision tree, random forest, neural network, and SVM are selected as the machine learning model classifiers to test the model effect. The evaluation of the model is mainly based on the confusion matrix, accuracy, recall, and f_1 score. **Table 2** shows the recognition and evaluation results of the return motion of the tennis sports robot under different classifiers.

TABLE 2 | Recognition and evaluation results of the return motion of the tennis sports robot under different classifiers.

Evaluation parameters	SVM classifier	Decision tree classifier	Random forest classifier	Multi-layer perceptron (MLP) classifier
Accuracy	0.99	0.73	0.99	0.99
Recall	0.99	0.72	0.99	0.98
f_1 score	0.99	0.72	0.99	0.98
Model call time	0.00015	0.00001	0.00002	0.00001

Table 2 shows that the SVM classifier is superior to the decision tree classifier, random classifier, and MLP classifier in action recognition of the tennis sports robot in the accuracy, recall rate, f_1 score, and model call time. It has high accuracy and meets the tennis sports robot's return action recognition requirements.

Design of Return Strategy Algorithm for Tennis Sports Robot

The landing areas of the tennis serve are two diagonally opposite areas. Before serving, it is essential to stand in the area behind the end line and between the midpoint and the hypothetical

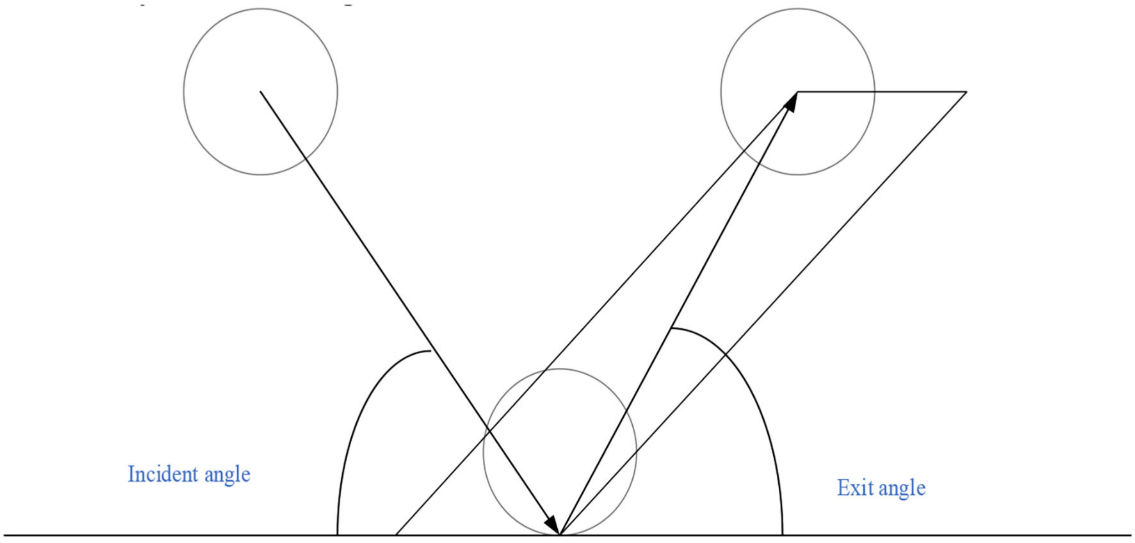


FIGURE 6 | Return state of the tennis sports robot.

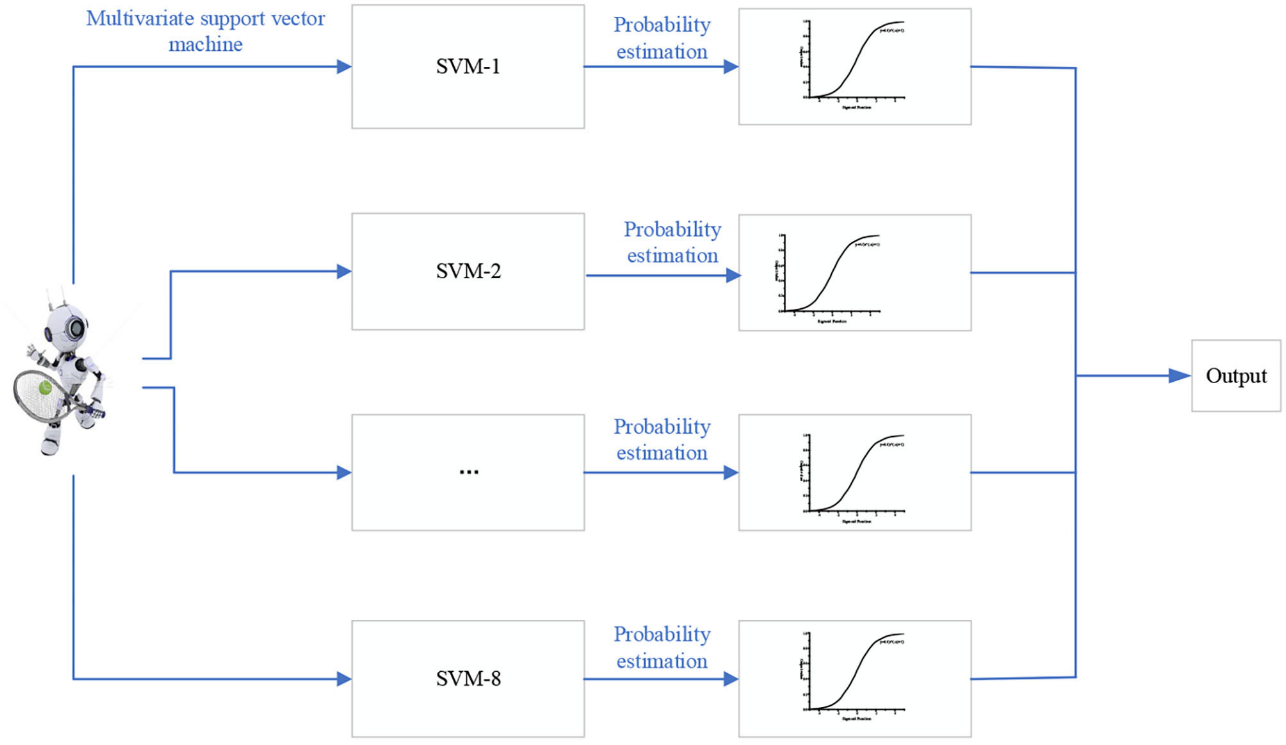


FIGURE 7 | Return strategy model of the tennis sports robot based on SVM.

extension line of the sideline, throw the tennis ball into the air, hit the ball with the racket before the ball touches the ground, and complete the sending of the ball when the racket contacts the ball. When facing a tennis player serving, the tennis sports robot cannot change the position of the original station by walking or running. It is best to stand in the specified position and not touch other areas. The player serves from the end line of the right area at the beginning of each game and then changes to the left area to serve after they score or lose points. In serving, the ball should cross the net and fall on the opposite service area in the opposite

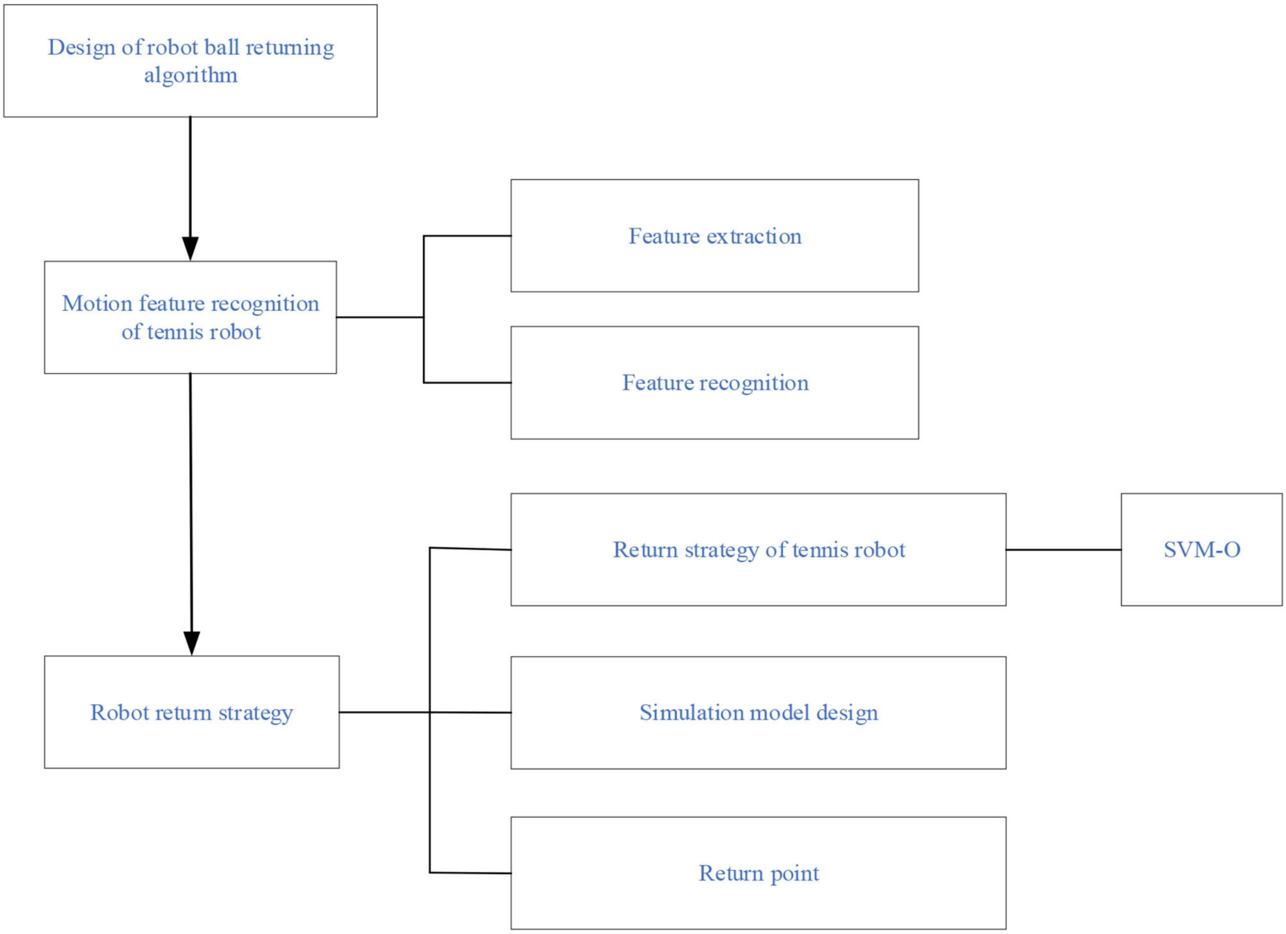

Design of robot ball returning algorithm
Motion feature recognition of tennis robot
Feature extraction
Feature recognition
Return strategy of tennis robot
SVM-O
Robot return strategy
Simulation model design
Return point

FIGURE 8 | Implementation framework of the return strategy of the tennis sports robot.

corner, or it can fall on the surrounding line. However, another service is required if the ball touches the net and falls into the opponent's service area during the service, and the tennis sports robot is not ready to return. **Figure 6** shows the return state of the tennis sports robot.

In **Figure 6**, after the ball rebounds, the tennis sports robot starts to stare at the ball, which is the same as the ball pressure with a deep landing point. The reason is that the backswing is too late. If the human movement causes the ball speed to reach 260 km/h, the tennis compression is more obvious. The tennis ball forms the incident angle and exit angle with the ground in returning the ball. In a tennis sports robot, the average serving speed of the human body is 150–250 km/h. According to different return points, the opponent serves within a certain speed range no matter how fast the service is. Through the ball return and landing points in different speed ranges, the robot can aim at the other party's ball return and landing point area to return the ball. The tennis sports robot can use SVM to get the optimal return strategy. **Figure 7** is the return strategy model of a tennis sports robot based on SVM.

Figure 7 is a multivariable SVM learning based on eight ball return actions of a tennis sports robot. However, the maximum interval characterizes the boundary between the current classifier and the dataset because the model is linearly separable from the 2D traditional model. The data volume is modeled and classified based on the feature extraction of human motion recognition images. With the two classifiers in **Figures 3**, **4** as examples, the maximum interval of blue lines in the curve is greater than that of the black lines. Therefore, the blue line is selected as the classifier. Machine learning is carried out for the return motion of the tennis sports robot, and the probability value is the output through the probability estimation of accuracy. **Figure 8** is the implementation framework of the return strategy of the tennis sports robot.

In **Figure 8**, the algorithm design of the tennis sports robot's return strategy is carried out using machine learning and the SVM-O algorithm. Feature extraction is carried out through the motion characteristics of human motion recognition, and the SVM algorithm is designed and studied for the return ball strategy of the tennis sports robots. Finally, the

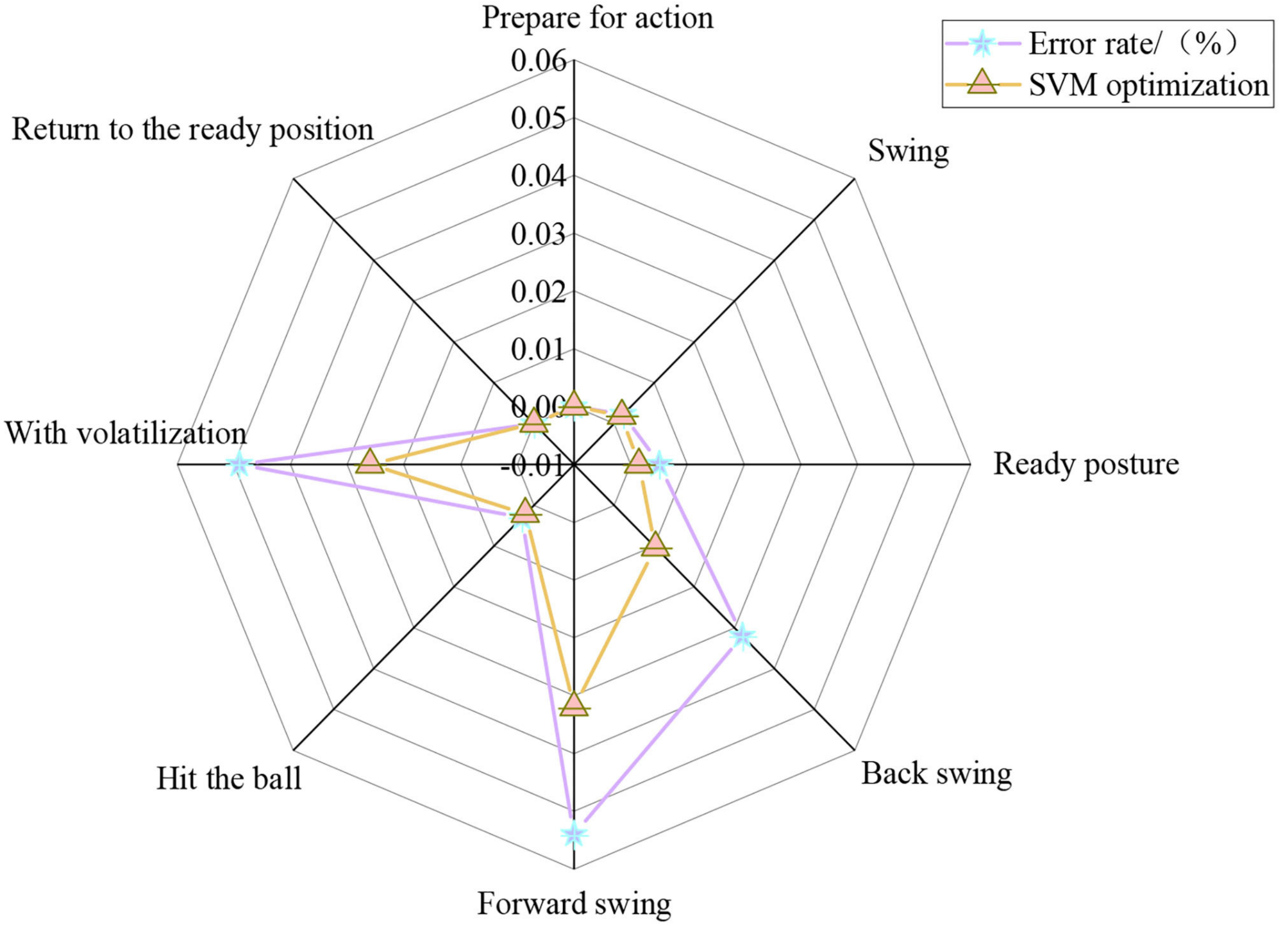


FIGURE 9 | Error rate analysis of 8 kinds of return movements of tennis sports robot based on SVM-O.

accuracy of the return strategy of the tennis sports robot is simulated.

RESULTS AND DISCUSSION

Return Analysis of the Tennis Sports Robot

To optimize machine learning, the return error rate, return accuracy, return recall rate, return score rate, and return recognition rate of eight actions in the return strategy of a tennis sports robot are analyzed through the optimization analysis of the SVM algorithm. **Figure 9** displays the analysis results of the error rate of different return actions of tennis sports robots based on SVM-O.

In **Figure 9**, the average error rates of preparation, swing, ready posture, backswing, forward swing, hitting, volatilization, and returning to the ready position are 0, 0.21, 0.32, 2.12, 4.3, 0.26, 3.75, and 0%, respectively. The error rate of the backswing, forward swing, and volatilization in the return strategy of the tennis sports robot are high. The return accuracy, return recall, and return score of eight return actions are analyzed in varying degrees to further analyze the return strategy of the tennis sports robots. **Figure 10** shows the analysis of different return strategies of the tennis sports robots.

Figure 10 shows that in the ball return strategy of the tennis sports robots after optimization, the action with the highest accuracy rate of machine learning is swing, 94%. The action with the highest recall rate is the preparation action, 98.21%, and the action with the highest-scoring rate is volatilization, 96.79%. Regarding the return accuracy, return recall, and return score, the average values of preparation, swing, ready posture, backswing, forward swing, hitting, volatilization, and returning to the ready position are 93.38, 89, 86.93, 90.10, 86.28, 85.84, 90, and 86.73%. It reveals that in the return strategy of the tennis sports robots, preparation, backswing, and volatilization can achieve more objective results in the analysis of the return strategy. CNN, SVM-Linear Discriminant Analysis (SVM-LDA) algorithm, and SVM—common spatial pattern (SVM-CSP) algorithm are compared and analyzed to study the superiority of the SVM-O algorithm in the ball return recognition rate of the tennis sports robot. **Figure 11** shows the return recognition rate of the tennis sports robots under different models.

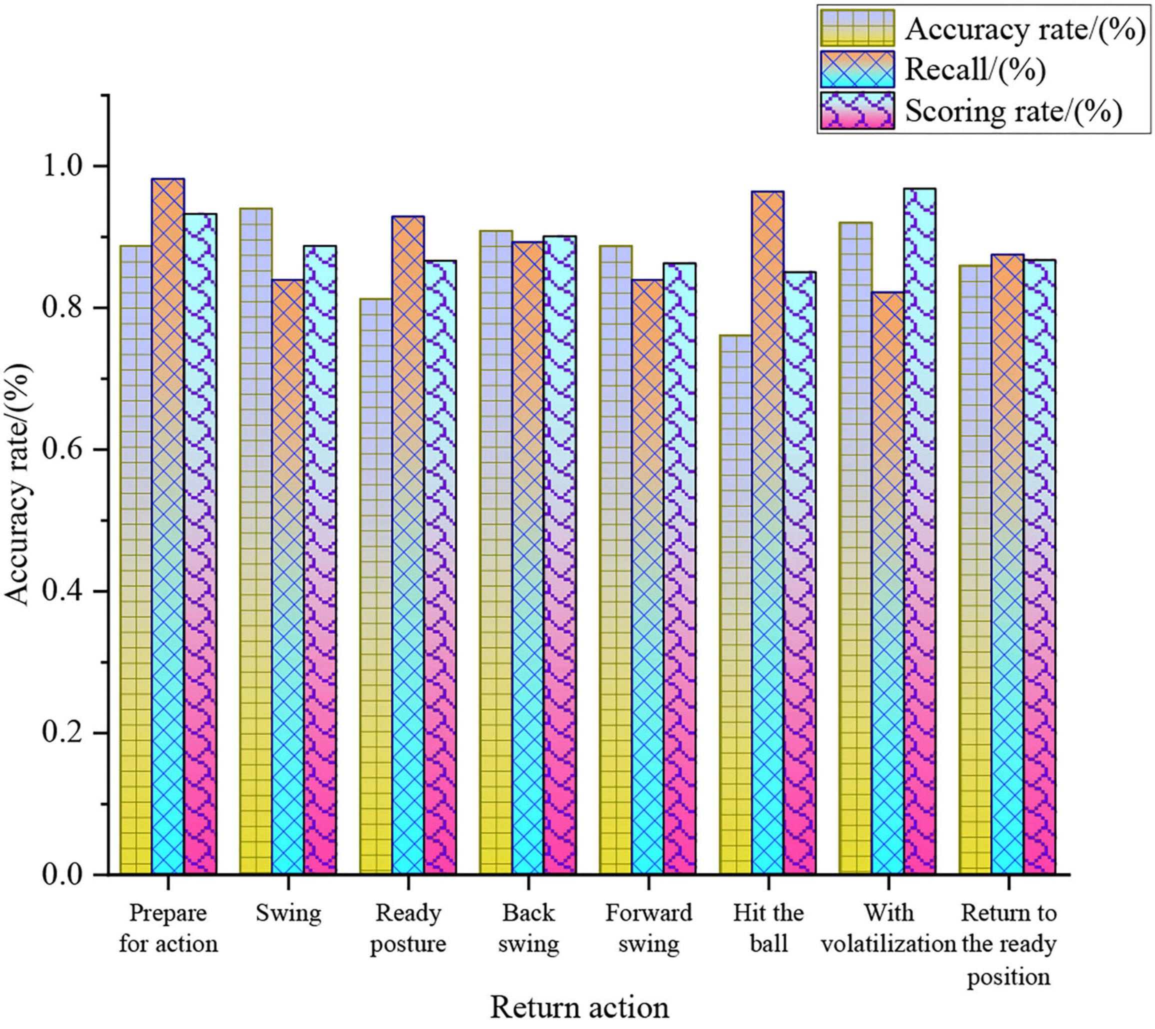


FIGURE 10 | Analysis of different return strategies of the tennis sports robot.

Figure 11 shows that the tennis sports robot based on the SVM-O algorithm has the highest return recognition rate, with an average return recognition rate of 88.61%. Then, the average return recognition rates of the tennis sports robot based on the SVM-LDA algorithm, SVM-CSP algorithm, and CNN algorithm are 85.49, 85.49, and 83.82%.

Simulation Effect of the Return Accuracy of the Tennis Sports Robot Under Different Model Training

The maximum number of iterations is set to 1 million times in the whole robot return strategy to refer to the final accuracy of the tennis sports robot return. In the simulation process, different models are simulated with different iteration times to simulate the return accuracy of the tennis robot. **Figure 12** is the simulation effect of the return accuracy of the tennis sports robot.

Figure 12 shows that when the base number of simulation times is small, there is little difference between the four models for the return accuracy of the tennis robot. With the increase of the iteration times, the effect of the model simulation experiment based on SVM-O is the best, followed by SVM-LDA, SVM-CSP, and finally CNN.

Discussion

First, the OpenPose traversal dataset was used to extract the information features of the human motion recognition nodes. Then, 8 kinds of tennis robots' return movements were integrated with the human motion characteristics, and the 8 kinds of tennis robots were used to analyze the return movements. By selecting these 8 basic return movements, the complexity of the human motion recognition process in tennis sports robots is reduced. The results show that when the number of return

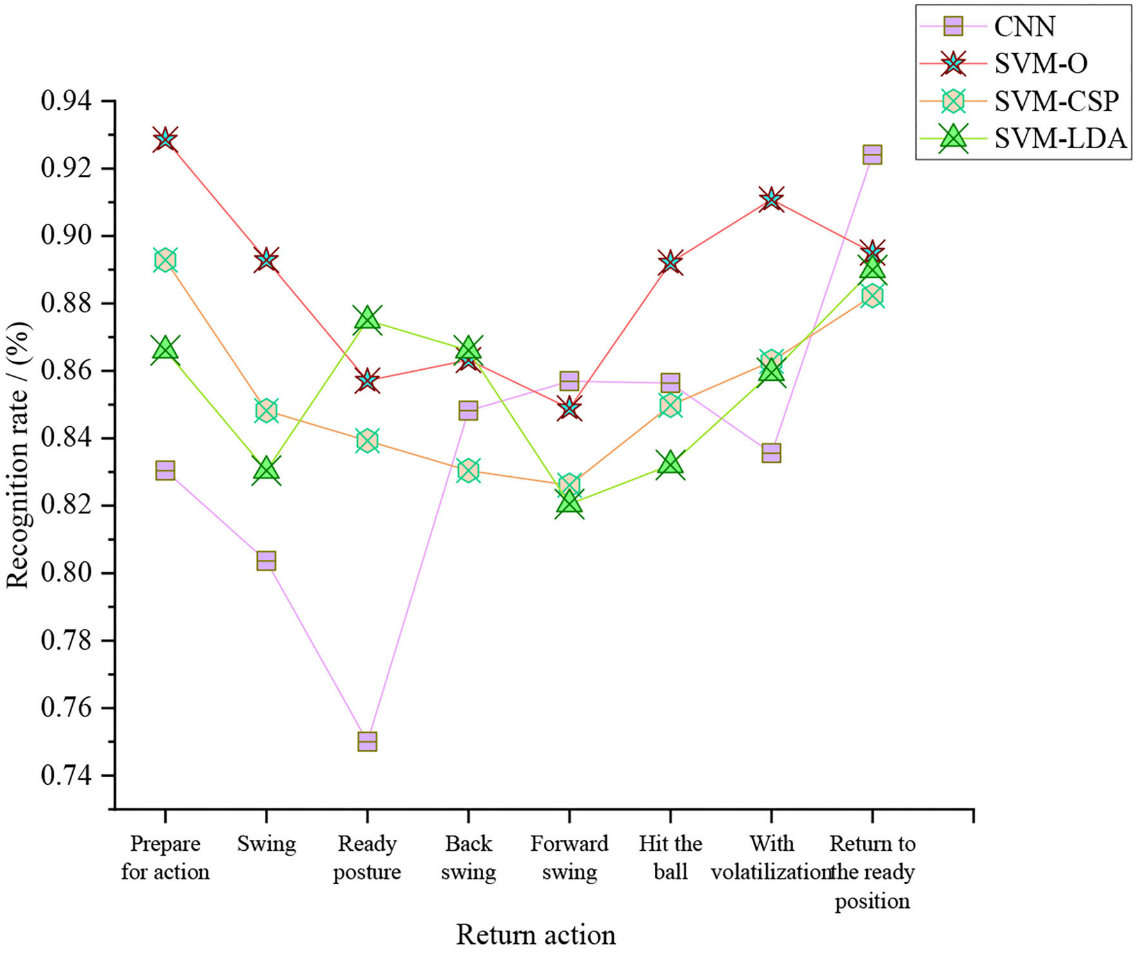


FIGURE 11 | Return recognition rate of the tennis sports robot under different models.

actions of the robot is set to 1 million, the recognition accuracy of the tennis sports robot to the CNN model is too small, and it is not easy to distinguish the return actions of the tennis sports robot. The smaller the number of iterations, the greater the impact on the recognition accuracy. The automatic ball-picking robot operates on the image data according to the previous tennis movement. The eagle eye system is composed of multiple high-speed cameras. Through the application of image recognition, image fusion, and 3D reconstruction technology, the configuration of the tennis robot is too high, and the return action of the tennis robot is not fully realized. The exploration is done to analyze the return action of the tennis sports robots based on the hitting strength and predicted landing point of the tennis sports robot. It is not easy to distinguish the characteristics of human motion by recognizing and analyzing human motion through the image obtained by the camera. However, the recognition of these feature points is significant. The image acquisition of the robot camera improves the efficiency of computational image recognition. However, when these human action feature points greatly impact the recognition results, this method is not desirable. Therefore, the return strategy of tennis robots is studied and analyzed by combining human motion recognition and robot return action.

CONCLUSION

Research and analysis show that there is almost no error rate in the preparation and return to the ready position of return strategy of the tennis robot. The error rate of the backswing, forward swing, and volatilization is higher than that of the other movements. Through the optimized SVM, the accuracy of the swing motion is as high as 94.00%. The action with the highest recall rate is preparation, 98.21%. The action with the highest score is swing, 96.79%. Regarding the return accuracy, return recall, and return score, the average values of preparation action, return action, and volatilization action are higher than the other return actions, which are 93.38, 90.10, and 90%, respectively. The research on the return ball recognition rate of the tennis robots under different models shows that the tennis robots' return ball recognition rate based on the SVM-O algorithm is higher than that of the SVM-LDA algorithm, SVM-CSP algorithm, and CNN algorithm. Its average return

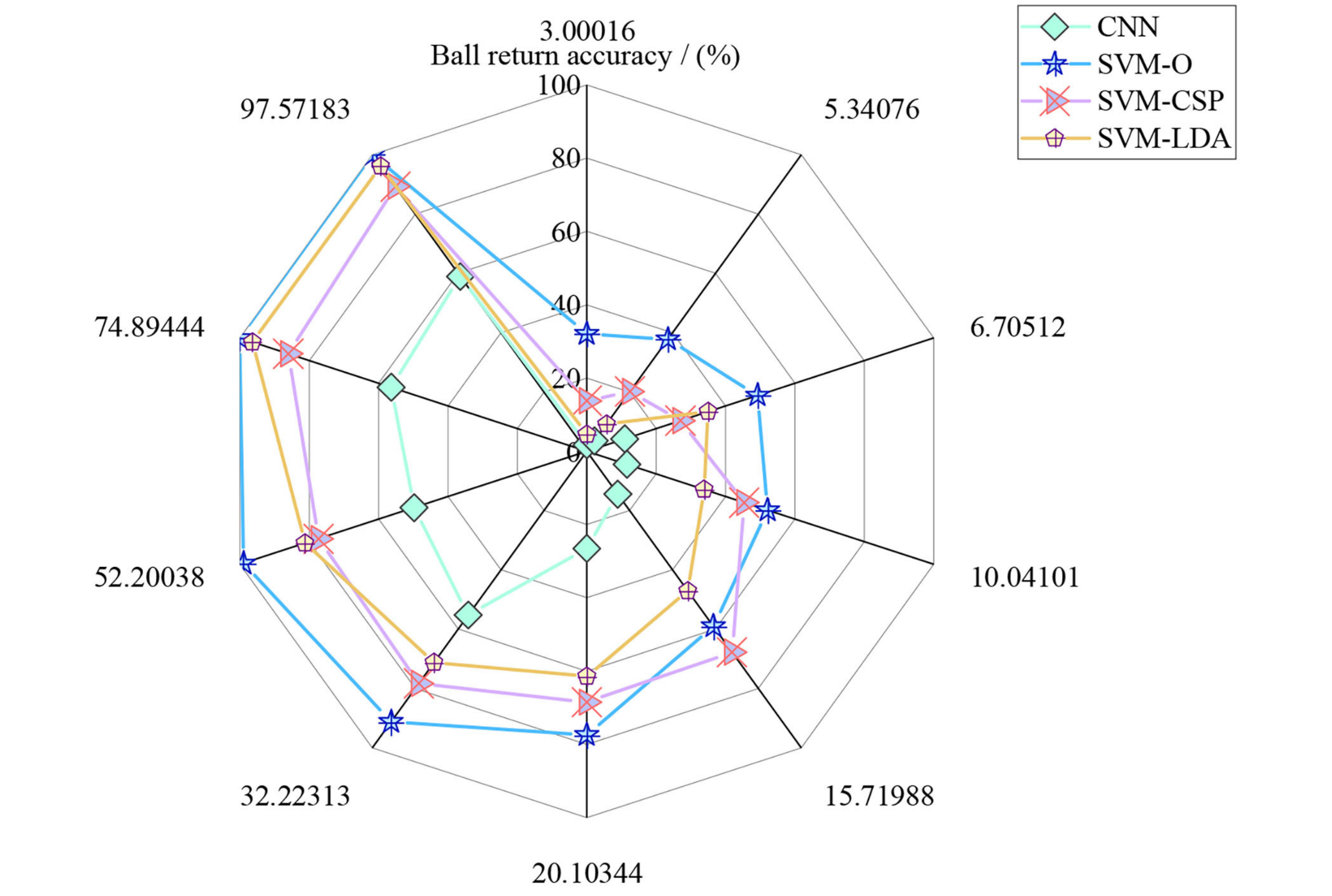


FIGURE 12 | Simulation effect of the return accuracy of the tennis sports robot.

ball recognition rate is 88.61%. To sum up, the simulation effect of the tennis robot's return strategy model based on SVM-O is the best, improving the diversity of the tennis robot's return strategies.

The required operations such as picking up, putting, and collecting balls in the tennis court are completed through the previous research on the design of the robot arm of the automatic ball-picking robot in tennis sports. The tennis robot's return strategy implementation model based on the SVM-O is proposed. It can integrate human action recognition with the return action of the tennis sports robot. Applying the SVM-O algorithm to the tennis sports robot's return action recognition solves the problem that the optimization algorithm cannot be applied to real-time requirements. It has important research significance for the application of an optimized SVM algorithm in sports action recognition. Tennis robots can combine different return strategies according to the recognition rate and accuracy of the return action. However, according to the work of the tennis robot, its internal structure will delay the operation of the system. Due to the complexity of human motion recognition, the robot cannot quickly launch the ball return according to the ball return strategy. In the follow-up research, more details can be considered to improve the autonomy of the tennis robot.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

ETHICS STATEMENT

Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

FUNDING

This work was supported by the Special Project "Research on the Development and Application of Physical Education in Colleges and Universities" of China Higher Education Society, Research on the influence of the evolution of national student physical health standard on the reform of physical education in colleges and universities (Grant No.: 21TYYB12) and Jiangxi Provincial Sports Bureau Sports Research Project, "Health China" led by the scientific theoretical basis and promotion of national fitness exercise and dance research (Grant No.: 202142).

REFERENCES

Alexopoulos, K., and Nikolakis, N. (2020). Digital twin-driven supervised machine learning for the development of artificial intelligence applications in manufacturing. *Int. J. Comp. Integr. Manufact.* 33, 429–439. doi: 10.1080/0951192X.2020.1747642

Aslan, M. F., and Durdu, A. (2020). Human action recognition with bag of visual words using different machine learning methods and hyperparameter optimization. *Neural Comp. Appl.* 32, 8585–8597. doi: 10.1007/s00521-019-04365-9

Faisal, A. I., Majumder, S., Mondal, T., Cowan, D., and Naseh, S. (2019). Monitoring methods of human body joints: state-of-the-art and research challenges. *Sensors.* 19, 2629. doi: 10.3390/s19112629

Gao, J., Ye, W., and Guo, J. (2020). Deep reinforcement learning for indoor mobile robot path planning. *Sensors* 20, 5493. doi: 10.3390/s20195493

Ghorbanzadeh, O., Blaschke, T., Gholamnia, K., Meena, S. R., and Tiede, D. (2019). Evaluation of different machine learning methods and deep-learning convolutional neural networks for landslide detection. *Remote Sensing* 11, 196. doi: 10.3390/rs11020196

Giles, B., and Kovalchik, S. (2020). A machine learning approach for automatic detection and classification of changes of direction from player tracking data in professional tennis. *J. Sports Sci.* 38, 106–113. doi: 10.1080/02640414.2019.1684132

Gurbuz, S. Z. (2019). Radar-based human-motion recognition with deep learning: promising applications for indoor monitoring. *IEEE Signal Proc. Magaz.* 36, 16–28. doi: 10.1109/MSP.2018.2890128

Haryanto, J. (2020). The relationship of concentration and eye-hand coordination with accuracy of backhand backspin serve in table tennis. *Int. J. Technol. Innov. Human.* 1, 51–56. doi: 10.29210/881701

Hou, C., Xu, Z., Qiu, W., Wu, R., Wang, Y., and Xu, Q. (2019). A biodegradable and stretchable protein-based sensor as artificial electronic skin for human motion detection. *Small* 15, 1805084. doi: 10.1002/smll.201805084

Jalal, A., and Kamal, S. (2019). Depth maps-based human segmentation and action recognition using full-body plus body color cues *via* recognizer engine. *J. Electr. Eng. Technol.* 14, 455–461. doi: 10.1007/s42835-018-00012-w

Jalal, A., and Quaid, M. A. K. (2019). A wrist worn acceleration based human motion analysis and classification for ambient smart home system. *J. Electr. Eng. Technol.* 14, 1733–1739. doi: 10.1007/s42835-019-00187-w

Jaruenpunyasak, J., García Seco de Herrera, A., and Duangsoithong, R. (2022). Anthropometric ratios for lower-body detection based on deep learning and traditional methods. *Appl. Sci.* 12, 2678. doi: 10.3390/app12052678

Kahanowich, N. D. (2021). Robust classification of grasped objects in intuitive human-robot collaboration using a wearable force-myography device. *IEEE Robot. Autom. Lett.* 6, 1192–1199. doi: 10.1109/LRA.2021.3057794

Krause, L. M., Buszard, T., Reid, M., and Pinder, R. (2019). Assessment of elite junior tennis serve and return practice: a cross-sectional observation. *J. Sports Sci.* 37, 2818–2825. doi: 10.1080/02640414.2019.1665245

Kunze, K. N., Karhade, A. V., Sadauskas, A. J., and Schwab, J. H. (2020). Development of machine learning algorithms to predict clinically meaningful improvement for the patient-reported health state after total hip arthroplasty. *J. Arthroplasty* 35, 2119–2123. doi: 10.1016/j.arth.2020.03.019

Li, Z., Zhang, R., and Lee, C. H. (2020). An evaluation of posture recognition based on intelligent rapid entire body assessment system for determining musculoskeletal disorders. *Sensors* 20, 4414. doi: 10.3390/s20164414

Liu, Y., Chen, P. H. C., and Krause, J. (2019). How to read articles that use machine learning: users' guides to the medical literature. *JAMA* 322, 1806–1816. doi: 10.1001/jama.2019.16489

Mohabatkar, H., and Ebrahimi, S. (2021). Using chou's five-steps rule to classify and predict glutathione S-transferases with different machine learning algorithms and pseudo amino acid composition. *Int. J. Peptide Res. Therap.* 27, 309–316. doi: 10.1007/s10989-020-10087-7

Neff, C., Mendieta, M., Mohan, S., Baharani, M., and Rogers, S. (2019). REVAMP 2 T: real-time edge video analytics for multicamera privacy-aware pedestrian tracking. *IEEE Internet Things J.* 7, 2591–2602. doi: 10.1109/JIOT.2019.2954804

Oudah, M., and Al-Naji, A. (2020). Hand gesture recognition based on computer vision: a review of techniques. *J. Imag.* 6, 73. doi: 10.3390/jimaging6080073

Pourdarbani, R., Sabzi, S., Kalantari, D., and Hernández-Hernández, J. L. (2020). A computer vision system based on majority-voting ensemble neural network for the automatic classification of three chickpea varieties. *Foods* 9, 113. doi: 10.3390/foods9020113

Sharma, M., and Kumar, N. (2021). Badminton match outcome prediction model using Naïve Bayes and Feature Weighting technique. *J. Ambient Intellig. Human. Comput.* 12, 8441–8455. doi: 10.1007/s12652-020-02578-8

Shi, Q. (2020). Fault diagnosis of an autonomous vehicle with an improved SVM algorithm subject to unbalanced datasets. *IEEE Trans. Indust. Electr.* 68, 6248–6256. doi: 10.1109/TIE.2020.2994868

Wang, M., and Zhang, Y. D. (2019). Human motion recognition exploiting radar with stacked recurrent neural network. *Digital Signal Proc.* 87, 125–131. doi: 10.1016/j.dsp.2019.01.013

Xiong, Q., Zhang, J., Wang, P., and Liu, D. (2020). Transferable two-stream convolutional neural network for human action recognition. *J. Manufact. Syst.* 56, 605–614. doi: 10.1016/j.jmsy.2020.04.007

Yang, L., Liu, Y., Yu, H., Fang, X., Song, L., and Li, D. (2021). Computer vision models in intelligent aquaculture with emphasis on fish detection and behavior analysis: a review. *Arch. Comput. Methods Eng.* 28, 2785–2816. doi: 10.1007/s11831-020-09486-2

Yunardi, R. T., Arifianto, D., and Bachtiar, F. (2021). Holonomic implementation of three wheels omnidirectional mobile robot using dc motors. *J. Robot. Control* 2, 65–71. doi: 10.18196/jrc.2254

Zhao, J., Liu, L., Wei, Z., Zhang, C., and Wang, W. (2019). R-DEHM: CSI-based robust duration estimation of human motion with WiFi. *Sensors.* 19, 1421. doi: 10.3390/s19061421

Zhao, L. (2020). Detection and recognition of human body posture in motion based on sensor technology. *IEEJ Trans. Electr. Electron. Eng.* 15, 766–770. doi: 10.1002/tee.23113

Zhu, M., Yi, Z., and Yang, B. (2021). Making use of nanoenergy from human–nanogenerator and self-powered sensor enabled sustainable wireless iot sensory systems. *Nano Today* 36, 101016. doi: 10.1016/j.nantod.2020.101016

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 11 May 2022
doi: 10.3389/fnbot.2022.820028

Application of Table Tennis Ball Trajectory and Rotation-Oriented Prediction Algorithm Using Artificial Intelligence

*Qiang Liu and Hairong Ding**

Shanghai Polytechnic University, Shanghai, China

The present work aims to accelerate sports development in China and promote technological innovation in the artificial intelligence (AI) field. After analyzing the application and development of AI, it is introduced into sports and applied to table tennis competitions and training. The principle of the trajectory prediction of the table tennis ball (TTB) based on AI is briefly introduced. It is found that the difficulty of predicting TTB trajectories lies in rotation measurement. Accordingly, the rotation and trajectory of TTB are predicted using some AI algorithms. Specifically, a TTB detection algorithm is designed based on the Feature Fusion Network (FFN). For feature exaction, the cross-layer connection network is used to strengthen the learning ability of convolutional neural networks (CNNs) and streamline network parameters to improve the network detection response. The experimental results demonstrate that the trained CNN can reach a detection accuracy of over 98%, with a detection response within 5.3 ms, meeting the requirements of the robot vision system of the table tennis robot. By comparison, the traditional Color Segmentation Algorithm has advantages in detection response, with unsatisfactory detection accuracy, especially against TTB's color changes. Thus, the algorithm reported here can immediately hit the ball with high accuracy. The research content provides a reference for applying AI to TTB trajectory and rotation prediction and has significant value in popularizing table tennis.

Keywords: artificial intelligence, machine learning, track recognition of table tennis, human motion recognition, support vector machines algorithm

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
Lidong Yang,
Inner Mongolia University of Science and Technology, China
Erika Zemková,
Comenius University, Slovakia

****Correspondence:***
Hairong Ding
hrding@sspu.edu.cn

Received: *22 November 2021*
Accepted: *21 February 2022*
Published: *11 May 2022*

Citation:
Liu Q and Ding H (2022) Application of Table Tennis Ball Trajectory and Rotation-Oriented Prediction Algorithm Using Artificial Intelligence. Front. Neurorobot. 16:820028. doi: 10.3389/fnbot.2022.820028

INTRODUCTION

Modern technologies like artificial intelligence (AI) have become the forefront of research with continuous science and technological advancement. AI is also known as machine intelligence and computer intelligence. As the name implies, this technology aims to intellectualize machines or computers like humans (Hu, 2020).

Artificial intelligence has seen two significant elements: summarization and logical deduction, regarded as the connectionism approach and the symbolism approach, respectively (Riguzzi et al., 2019; Park and Hainaut, 2020). Human beings process audio-visual signals based on cortical neural networks without thinking. This learning method is called the connectionism approach. Accordingly, connectionism scholars do not investigate the deep-seated learning process in their research but obtain the final result through machine learning algorithms learning a large number of data and methods autonomously. Human's mathematical derivation and proving theorem are based on solid subjective consciousness and axiomatic system, requiring conscious thinking and symbolic calculus. This learning method is called the symbolism approach (Lin et al., 2020). Symbolism scholars tend to design formulas to solve problems based on this definition. They investigate the deep-seated calculation process and obtain the final result through the complete process of machine learning (Al-Mukhtar and Al-Yaseen, 2020; Gomez-Gonzalez et al., 2020). The connectionism approach and symbolism approach are most common in the current AI field. Other algorithms are based on the extension of the two methods in all directions. A table tennis robot (TTR) is an advanced intelligent robot (IR) with comprehensive solid quality, consisting of vision, decision-making, and control systems. These robots can respond to external stimuli, thereby realizing man-machine confrontation in various scenes. TTR can assist professional player training well (Likitha, 2021) and is significant in popularizing table tennis sports. Additionally, IR is also of great significance (Carreras et al., 2020).

The innovation of this paper lies in the following aspects. First, the IR is used in table tennis training, and the table tennis ball (TTB) trajectory is predicted and calculated based on a deep learning algorithm. Secondly, a machine learning algorithm is proposed to identify motion states and the rotation and orbit of TTB. In short, AI technology is applied to table tennis training and competition to predict and determine the TTB trajectory accurately.

RELEVANT THEORIES AND METHODS

The Application of AI in the Field of Sports

With the continuous development of AI technology, IRs have been multi-functionalized and highly intellectualized. In particular, IRs have been extensively applied in many sports, including table tennis, badminton, basketball, and football. Among those sports IRs, TTR has some unique and delicate features worthy of in-depth exploration. For example, the TTB is very light and moves extremely fast, up to 5–20 m/s. Therefore, the TTR must be sensitive, accurate, and robust to lend itself well to train or play with professional players (Steiner et al., 2020). Thus, target identification and trajectory prediction of the table tennis ball (TTB) are incredibly complicated, becoming the key points and difficulties in the current research field (Zhang et al., 2020). The TTR-based TTB recognition depends on a vision system to analyze the hitting actions of the table tennis player and predict the real-time position and motion state of TTB. Thus, the TTR's vision system must have the ability to predict trajectory and plan actions for the target to ensure detection accuracy and real-time motion recognition (Zhao et al., 2021).

This paper studies the TTR from three aspects (Forghani, 2020): vision, decision-making, and control systems. Among them, the vision system distinguishes TTR from other sports IRs. The three procedures are interdependent and work collaboratively, each with different target detection and trajectory prediction tasks. First, the vision system of the TTR is an upgraded version of general machine learning vision systems and the eye to detect and track the real-time position and dynamic states of TTB. The vision system has a solid ability to track high-speed moving objects (Li, 2021; Zhang, 2021). Second, the decision-making system is the instruction-distributor of the three systems. After receiving the information transmitted by the vision system, the decision-making system needs to respond accordingly to predict the TTB trajectory (Gomis-Fons et al., 2021). Moreover, the vision system is also responsible for selecting the optimal hitting plan by screening TTB-hitting actions. Third, the control system or execution system is accountable for executing the instructions issued by the decision-making system (Oliveira et al., 2021; Payedimarri et al., 2021). Strength, speed, and accuracy are essential for TTB to be served or returned; thus, the trajectory prediction and timing must be precise.

Summing up, the research of TTR involves many fields, such as visual inspection (VI), intelligent decision-making, DL, and servo control. The present work mainly examines the software system of TTR to identify the motion state and trajectory of the TTB accurately.

Related Research on Table Tennis Robots

The research of table tennis robots originated in the 1980s. The first table tennis robot developed can only serve and does not have the function of confrontation with human beings. Therefore, the table tennis robot could only serve as a companion for athletes to send different trackballs. In the future, with the progress of research and technology, a table tennis robot that can catch the ball will be developed gradually.

The robot table tennis game rules were first formulated by the University of Portsmouth in the United Kingdom in 1983. The rules stipulated that the table tennis table was 2 meters long and 0.5 m wide, slightly smaller than the usual table tennis table. The robot developed by Gerhard Schweitzer of a Robotics Institute in Zurich, Switzerland, won the championship of the competition in 1988 and the Hong-Kong Robot Ping Pong Competition in 1992.

In 1987, the Alcatel-Lucent Bell Labs of AT&T Inc. in the United States intensively studied the mechanical system, vision system, and control system of the table tennis robot. They adopted a PUMA 260 manipulator with six-Degree-of-Freedom (DOF) in a robot, which is more flexible when hitting the ball. It is a real table tennis robot because it can use the vision system to judge the position of table tennis. It has successfully realized a man-machine match for nearly 20 rounds.

The Toshiba Corporation of Japan has developed a seven-DOF manipulator that can hit a table tennis ball against the wall by itself. Picasso, a robot developed by the Rochester Institute of Technology in the United States, uses a five-DOF manipulator. However, the visual system cannot quickly capture the trajectory of the TTB to meet the requirements of the man-machine rally because the robot uses a series manipulator. The University of Adelaide in Australia designed a six-DOF series manipulator by imitating PUMA. Although it is more flexible than the previous TTR, it can move in a relatively small range.

In Japan, Fumio Miyazaki of Osaka University has developed a four-DOF table tennis robot based on binocular vision. Sensors are installed on the opponent's racket and elbow to detect the rotation direction of the ball, which enables the robot to receive the ball from different angles. A bipedal humanoid robot developed by TOSY Robotics JSC in Vietnam participated in the Tokyo International Robot Exhibition in 2007. It can play table tennis against a human being with flexible and accurate actions, which has become the focus of public attention.

Most of the existing manipulator systems adopt a series structure characterized by significant moments of inertia and relatively strong design load. The manipulator is often designed to adapt to the corresponding industrial production environment. Although it is convenient for research to use this industrial manipulator as the hitting execution system of table tennis robots, its shortcomings are also evident. The real-time playing of table tennis against humans requires a high response speed from the robot. The robot needs to recognize and predict the trajectory of the TTB in a short time and then order the manipulator to hit the ball quickly. Series manipulators cannot meet this requirement. Scholars employ the parallel robot as the actuator of the table tennis robot system because of its flexible dynamic performance. The parallel manipulator reduces the moment of inertia of the manipulator and improves its flexibility.

In the robot system, the operation of the inverse solution of the hitting point is cumbersome, bringing a significant challenge to the operation speed of the whole system. After obtaining the three-dimensional coordinate information of the table tennis ball, the vision system predicts the coordinates of the hitting point. The table tennis system needs to carry out the inverse solution according to this coordinate and calculate the rotation angle of each joint point. The manipulator has countless postures to strike a certain point. The inverse solution process aims to find the most reasonable hitting method with a particular scheme. The actuator of the seven-DOF manipulator needs to calculate the rotation angles of the seven joints, respectively, requiring a tremendous amount of calculation. Therefore, many scholars choose a relatively simple mobile guide rail to avoid this problem. This actuator has a larger hitting space than the industrial manipulator, simplifying the calculation and the cumbersome trajectory planning process.

RESEARCH MODEL AND FRAMEWORK

AI-Based TTB Trajectory Prediction

A table tennis ball has the characteristics of high speed, small volume, light weight, and fast rotation, resulting in high requirements for the real-time capability and accuracy of the vision system. Traditional target detection for TTB mainly adopts color segmentation, contour search, or multi-sensor methods. They can quickly respond to detection with simple calculations. However, the surrounding environment quickly affects the detection accuracy, such as illumination and background, leading to low detection accuracy. For example, the color segmentation algorithm sets the detection threshold according to the TTB color. Trajectory prediction tasks are mainly completed through physical modeling. The TTB rotation model features high-order non-linearity. Traditionally, the linear approximation algorithm is often used for modeling, with a relatively large deviation accumulating with the iteration, resulting in low prediction accuracy and less extendibility.

Under the current AI era, fusing AI, and robotics is the general trend to extend the development space and research value of robots while bringing more research possibilities to the TTR. In particular, the DL methods can exert their advantages against the existing problems in traditional ways. Yet, DL-based detection networks involve vast amounts of calculation and training data, thus complicating the issues. Therefore, aiming at the shortcomings of conventional TTB detection and trajectory prediction methods, this study integrates a DL method with strong generalization ability and anti-interference ability with the vision system to study the TTB rotation and predict its trajectory explicitly.

As a cutting-edge technology in the AI field, intelligent TTR is a scorching research topic worldwide (Nataraj et al., 2021; Sun, 2021). The ultimate task of an intelligent TTR is to play table tennis with people and even assist in professional training. According to the above analysis, the vision, decision-making, and control systems are subsystems of the intelligent TTR. Given that TTB is small and fly fast, the vision system shoulders the primary task of recognizing the TTB's dynamic state quickly and accurately. In some high-level competitions worldwide, many top table tennis players play at speeds up to over 20 m/s, and the rate of 5 m/s in general. The length of an ordinary table is 2.74 m, so it takes <0.5 s for a TTB to fly across the table (Pezzo and Beckstead, 2020). Thus, a TTR must detect and analyze the TTB dynamic state within 0.5 s, including TTB identification, trajectory prediction, and action planning. A TTR should have high-speed processing and calculation capability. Image recognition and trajectory prediction become the primary tasks of the vision system. In short, the main research goal and difficulty of the vision system of TTR are to capture and analyze TTB accurately and in time.

In addition to real-time trajectory prediction and target recognition, the vision system needs to provide a timely and reasonable response for the subsequent decision-making and control systems. According to international research, most visual systems can be divided into monocular visual systems and binocular visual systems based on the camera number (Geffen et al., 2020; Muto et al., 2021). According to the installation position, visual systems can be divided into ontological and external systems. As their names suggest, ontological systems install the camera inside the robot, which moves with the robot (Tomasevic et al., 2019). The outward vision system fixes the camera outside the robot so that the camera can only be calibrated without moving.

Most TTRs employ the binocular vision system because it can determine the target's spatial position. Compared with the binocular system, the monocular vision system has lower costs and lower calibration difficulty. Still, it needs to project a target to determine the target's spatial position (Tkatek et al., 2020; Jammeli et al., 2021). To some extent, it increases the difficulty of the algorithm and increases the requirements for the environment. Therefore, monocular vision systems are not as common as binocular vision systems.

Humanoid robots usually put the camera of their vision system inside. In contrast, other robots with robotic arms put the visual systems outside because it is difficult to use the ontology vision system; after all, the vision system will change with the robot's movement, and the camera needs to be recalibrated. Such a dynamic process makes algorithm implementation more difficult. Therefore, robots with good market influence usually choose an external vision system.

The primary function of robot servo planning is to identify the TTB and predict its trajectory by positioning the dynamic TTB according to the characteristics of a TTB using the camera.

Here, the position-based visual servo control system is selected for TBR, as shown in **Figure 1**.

In **Figure 1**, the visual servo control system adjusts the internal and external parameters of the camera. The target's spatial pose is calculated according to the camera image. Another camera can photograph the TTB's edge, calculate the edge pose of the TTB through feature extraction and other operations, and compare it with the expected posture. The pose deviation of the TTB edge is calculated through two differentiations. Then, the pose deviation of each side of the TTB is transformed into the base coordinate system.

Correlation Between TTB Rotation and Trajectory Based on Neural Network

In a competition, TTB is often rotating. Thus, the critical point of trajectory prediction of the TTB lies in choosing prediction methods. Here, the rotating state of TTB is divided into two categories for detection. The first category is identifying the TTB by marking TTB. Once labeled, the system can directly calculate the rotation of TTB by calculating the movement relative to the center of TTB (Salvatore et al., 2021). In the second category, the TTB trajectory will be collected and analyzed, and then the dynamic state of TTB is reversed according to the trajectory. However, the research in this field is only based on the TTB rotation to judge its trajectory. Such technology is impossible to accurately judge the exact amplitude, direction, pose, and rotation speed (Lin et al., 2021).

The trajectory of the table tennis ball in the air is expressed by three kinematic parameters: spatial coordinates, velocity, and acceleration. In the whole process of ball movement, these parameters will change from time to time according to the different movement times of the ball (Glassman et al., 2021). Then, a trained deep neural network (DNN) is selected here. Nine TTB parameters are input into DNN to output the TTB landing point. In actual competitions, the player's prediction based on TTB trajectory is also the primary determinant of the ball's landing point. Accordingly, the DNN aims to calculate the landing point data.

The TTB trajectory analysis is mainly carried out through SIMI Motion in this experiment. This software system can analyze various sports and movements based on three-dimensional (3D) video, typical in sports technology analysis and teaching. SIMI Motion uses multiple cameras to synchronize the target motion, the multi-dimensional 3D frame for calibration, and manually marks the joint points. Meanwhile, SIMI Motion can obtain the moving object's two-dimensional (2D) and 3D data to calculate the coordinate, speed, acceleration, and angle between the marked points. This experiment uses SIMI Motion to synchronize two cameras to obtain the TTB trajectory data. First, the calibration is completed by photographing the multi-dimensional 3D frame of two cameras and manually marking the white dots on the 3D structure. Then, the synchronization light is used to synchronize the time of the two videos. Afterward, the SIMI Motion is used to manually mark and determine the spatial position of the TTB in each frame, thereby obtaining the 3D data of the TTB trajectory to calculate the TTB rotation speed, a vector with size and direction. Precisely, the high-speed camera captures the initial rotation track of the TTB transmitted by the TTR's serving module and calculates the number of image frames required for the LOGO to rotate one circle to determine the rotation speed of the TTB. The TTB's initial state (when TTR fires TTB) is recorded as the frame n_1 in the video, and the TTB state after one-circle rotation is denoted as the frame n_2. The frame rate of the high-speed camera is 3,000 frames/s; thus, the TTB rotation speed is $\frac{3000}{n_2-n_1}$ R/S. Subsequently, the rotation direction data is obtained through the TTR's service module. The TTR's service module serves the rotating ball through two pulleys and controls the size and direction of rotation by controlling the speed and direction of the upper and lower pulleys. This experiment tests the rotation type of service from the service modules. There are nine rotation types of service: topspin, backspin, left-sidespin, right-sidespin, left-side topspin, right-side topspin, left-side backspin, right-side backspin, and without spin, as detailed in **Table 1**.

As listed in **Table 1**, the TTR can only implement nine types of service due to the limitations of the service module. The horizontal angle of the pulley determines the TTB rotation direction. Because the outlet of the service module is horizontal, the fired TTB's initial velocity is in the horizontal direction. Then, the TTB speed can be obtained through SIMI Motion analysis. Additionally, the initial position of the ball outlet is known. Then, nine accurate initial values can be obtained, including the initial position x with direction coordinate (x); initial position y with direction coordinate (y); initial position z with direction coordinate (z); initial position x with direction velocity (v_x); initial position y with direction velocity (v_y), initial position z with direction velocity (v_z); initial position x with direction rotation size (ω_x); initial position y with direction rotation size ω_y; initial position z with direction rotation size ω_z. The origin of the coordinate system is set at the table tennis table's midpoint, where the service module is located. The abscissa extends along with the table horizontally, the vertical coordinate extends longitudinally, and the vertical coordinate is perpendicular to the table. A total of 171 effective balls are served by the service module, covering nine types of service, and nine initial and landing point values of all balls are obtained. The falling point data are the coordinates on the table, so the ordinate is 0. Through the experimental data collection, to get the accurate initial position coordinates, the accurate initial velocity (including speed and direction), and the accurate rotation velocity, the nine initial data are input into the DNN to output the precise landing point coordinates. Finally, the

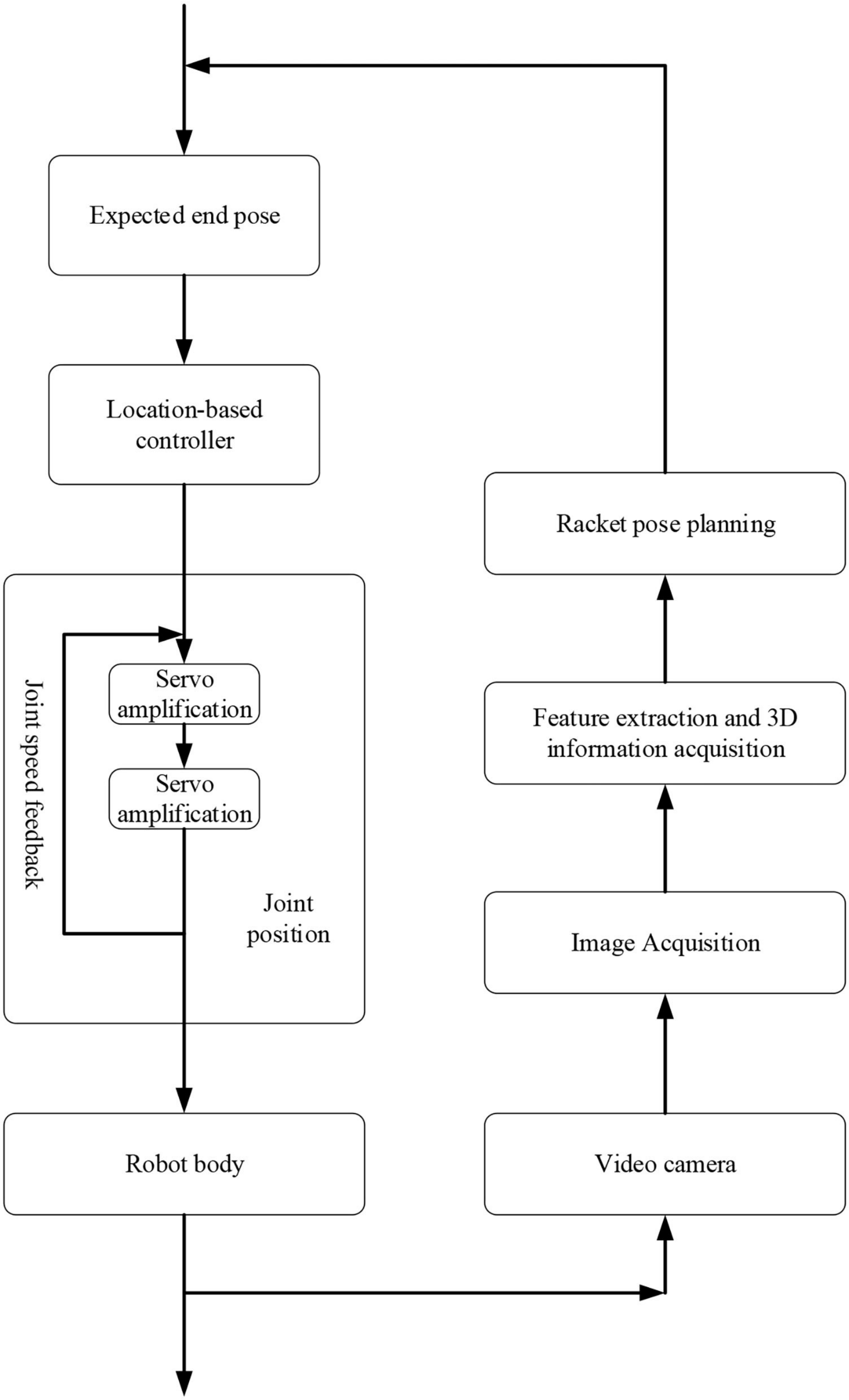


FIGURE 1 | Visual servo control system based on position.

TABLE 1 | Description of the table tennis robot (TTR)'s serve.

Type of service	Speed of upper and lower pulleys	Horizontal angle of the pulley
Topspin	The upper pulley is fast and the lower pulley is slow	90°
Backspin	The upper pulley is slow and the lower pulley is fast	90°
Without spin	The speed of the upper and lower pulleys are the same	90°
Left-Sidespin	The upper pulley is fast and the lower pulley is slow	0°
Right-Sidespin	The upper pulley is slow and the lower pulley is fast	0°
Left-Side topspin	The upper pulley is fast and the lower pulley is slow	−45°
Left-Side backspin	The upper pulley is slow and the lower pulley is fast	45°
Right-Side topspin	The upper pulley is fast and the lower pulley is slow.	45°
Right-Side backspin	The upper pulley is slow and the lower pulley is fast.	−45°

DNN algorithm explores the correlation between the input and output information.

In this experiment, Matlab pattern recognition is adapted for the trajectory prediction of TTB with different velocities and rotations.

The neuron nodes in the input layer have nine dimensions marked as "a." Neural nodes in the output layer have two numerical dimensions marked as "b." These are the abscissa and ordinate of TTB. The number of nodes in the hidden layer is determined after confirming the input and output neurons according to Equation (1).

$$F = \sqrt{0.43ab + 0.12b^2 + 2.54a + 0.77b + 0.35} + 0.51 \qquad (1)$$

The values of *ab* are substituted into Equation (1) to solve the number of hidden nodes as 6.

Then, the Non-linear Neural Network algorithms, such as Levenberg-Marquard (LM), Budgeted Rooted (BR), Backpropagation Neural Network, and Scaled Convergence Gradient (SCG) are used to fit the data using the Matlbe neural network toolkit, as displayed in **Figure 2**.

TTB Test Experiment Based on DL

The experiment begins after the preliminary preparations (Hildebrand et al., 2021; Zhang et al., 2021). The DNN used in the experiment is designed by Matlab toolkit, specified as follows.

Step 1. Collect data and import the input and output data into the database.

Step 2. Conduct simple data processing, including normalization, adjustment, and reconstruction. The regularization equation is:

$$M = \frac{m - m_{\min}}{m_{\max} - m_{\min}} \qquad (2)$$

The input data is disturbed before input to improve the generalization ability of neural networks and avoid the relative concentration of data in the same service spin mode.

Step 3. Construct the initial structures of input, output, hidden layer nodes, and the transfer function of the neural network.

Step 4. Set training time, target, error, and other parameters. Start training. Add weight correction parameters.

Step 5. Input the data to be tested after training.

Figure 3 illustrates the experimental process.

TTB Detection Based on Feature Fusion Network

FFN

The deeper a neural work is, the more times it scales down the original input image; thus, when locating small targets, some DNNs might get awkward performance, such as very low detection accuracy. In other words, every time the convolution kernel extracts image features, the feature map will shrink down by some ratio while the rich semantic information continues to be strengthened. Such operation is conducive to object detection and classification tasks, but the object's location information will be gradually discarded. Therefore, fusing the underlying feature information in the convolution downsampling process is necessary to enhance the network's ability to detect and locate small targets.

Earlier, neural networks need to input fixed-size images. Such operations as folding and flipping are often used in detecting photos with changed sizes, leading to information loss, which limits the accuracy of network recognition. In particular, feature pyramid networks (FPNs) can combine humble features lost in the original downsampling process. FPN will not increase model calculation substantially. On the contrary, it downsamples the semantically-rich upper feature layer in the top-down hierarchical network structure and then stacks and fuses the sampling results with the feature map of the same size, significantly improving the model's performance in small objects detection. Each layers' output can detect the type and position of objects. **Figure 4** displays the structure of FPN.

Two horizontally linked processes go through the FPN structure: bottom-up and top-down approaches. The bottom-up process is the forward propagation process when the feature map gradually shrinks with convolution kernel calculation. In contrast, the top-down process upsamples the feature map, and the horizontal link uses a 1*1 convolution kernel to fuse the feature map of the same size generated by bottom-up and top-down processes. In this way, the position details at the network

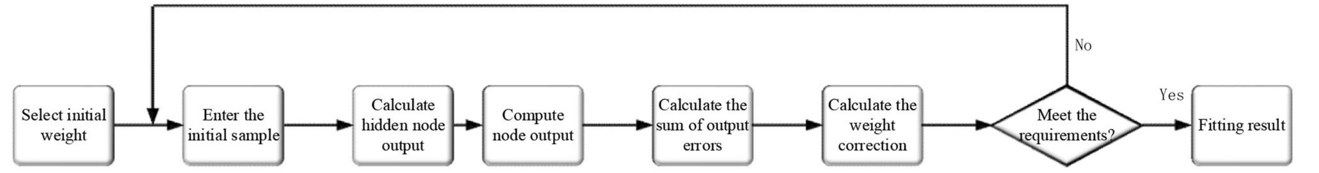


FIGURE 2 | Fitting process of neural networks.

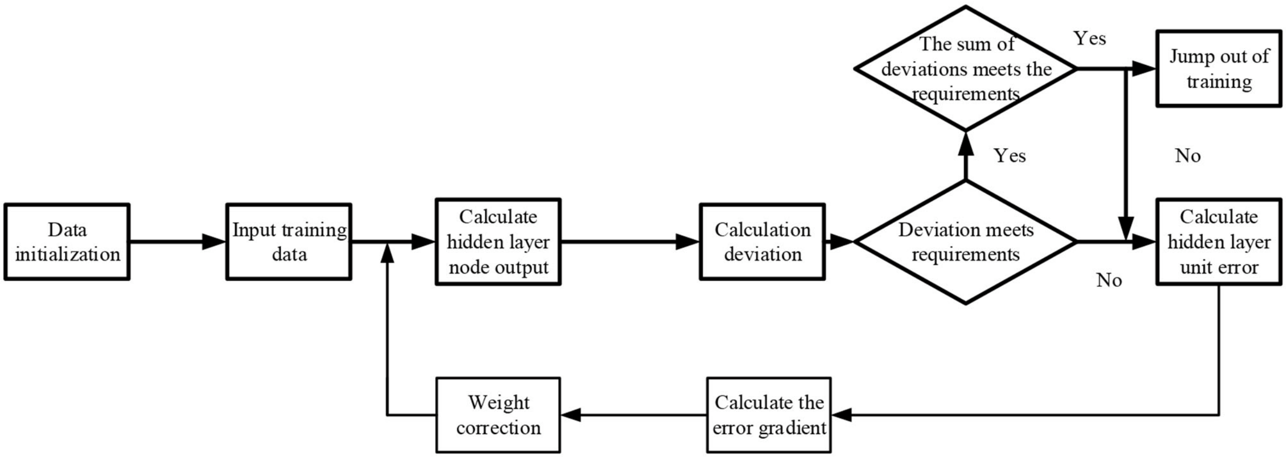

Data initialization
Input training data
Calculate hidden layer node output
Calculation deviation
Deviation meets requirements
Yes
The sum of deviations meets the requirements
Yes
Jump out of training
No
No
Calculate hidden layer unit error
Calculate the error gradient
Weight correction

FIGURE 3 | Table tennis ball (TTB) test based on DL algorithm.

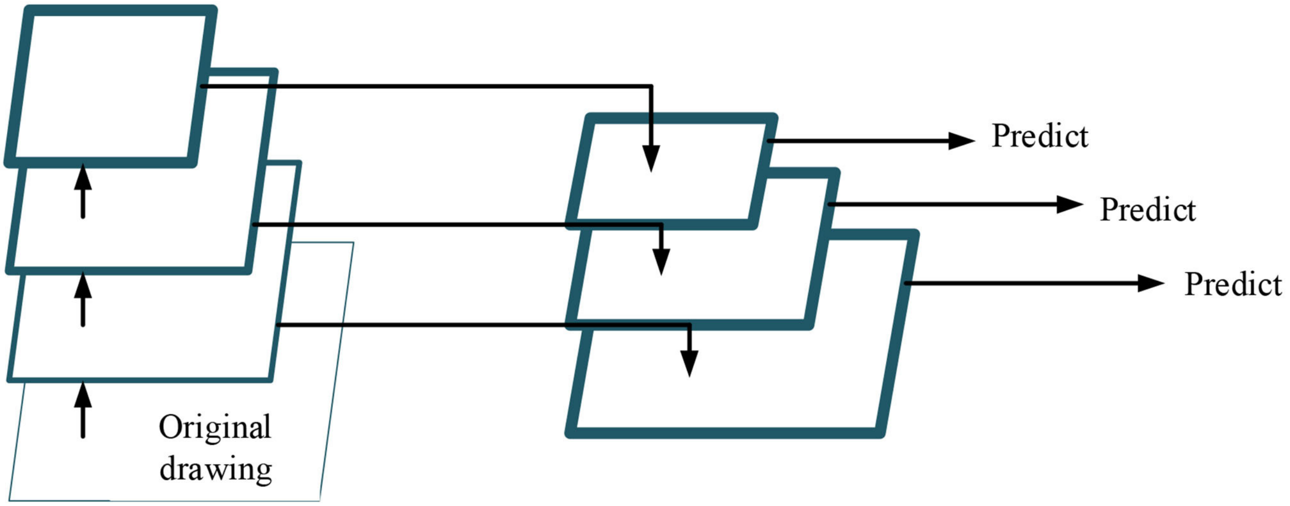


FIGURE 4 | Feature pyramid networks (FPN) structure.

bottom can train the network with accurate location information while simultaneously learning the target features, especially for small object detections. This paper improves FPN by adding a bottom-up connection, as shown in **Figure 5**.

A convolutional neural network (CNN) usually uses a single random feature layer (sometimes the last) to detect and locate targets. Each feature layer has its unique function for the target detection task. In **Figure 5**, the feature layer is followed by the adaptive pooling layer in area C. Then, the feature layers in B are merged into a feature layer of the same size. Then, the final feature layer of target detection is obtained by max-pooling feature fusion. Such a design can combine the feature

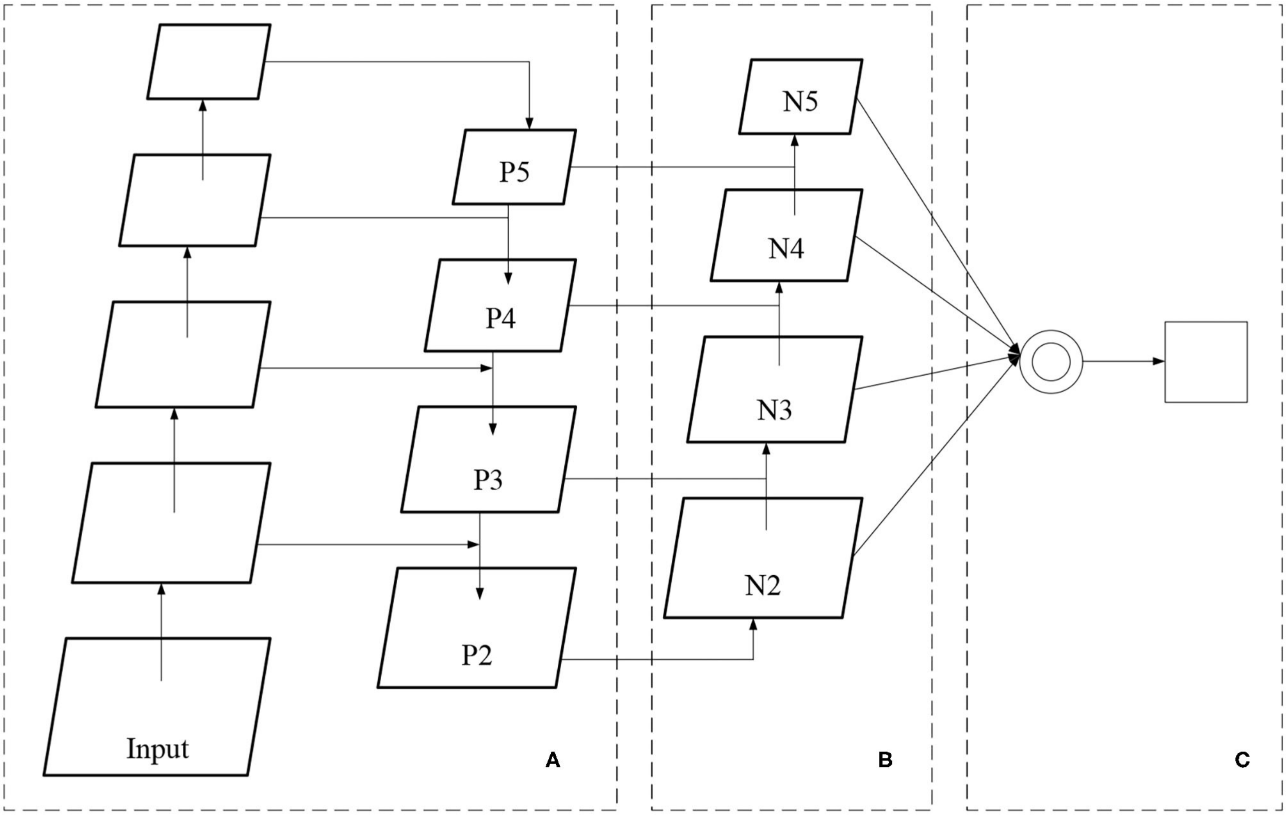


FIGURE 5 | Feature fusion network (FFN)'s structure.

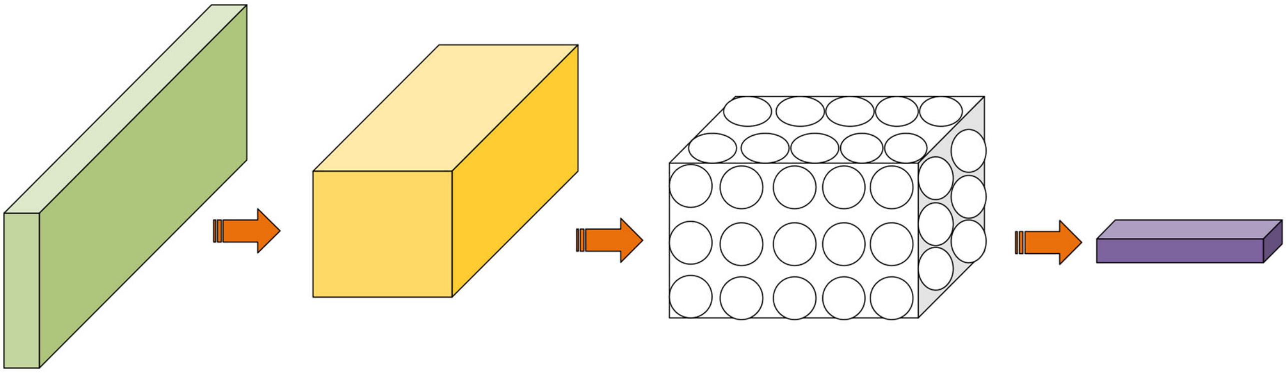


FIGURE 6 | Convolutional neural network (CNN)'s structure.

information of each feature map to significantly improve the position detection ability of the network to small targets.

Target Detection Network

Convolutional neural network is a supervised learning DNN, including convolution and pooling layers to extract input image features, an activation function to increase the non-linear ability of the network, and a fully connected layer to realize target detection and classification. At the same time, CNN can share weights, simplify network parameters, and avoid overfitting. **Figure 6** shows the structure of CNN.

The convolution layer is the most critical structure of CNN. It comprises convolution kernels of various sizes and depths. The network depth mainly refers to the number of convolution layers. During network initialization, the size and depth of the convolution layer will be set. Afterward, during network training, the network parameters repeatedly learn the data features, and backpropagation is used to optimize these parameters. The image

RGB value is constantly multiplied by the network weights to extract the image features during network transmission. The convolution layer can extract features and reduce the image dimensions. CNN employs valid padding (set to zero) to output images with the same dimensions as the original image. Different sliding steps can be set to reduce the image dimension. The smaller the sliding step size is, the fewer characteristic images are obtained. Generally, the richer the features to be extracted, the smaller the sliding step. Equation (3) indicates the output dimension of the convolution layer.

$$\begin{aligned} &W_2 = (W_1 - F + 2P)/S + 1 \\ &H_2 = (H_1 - F + 2P) \\ &D_2 = K \end{aligned} \tag{3}$$

In Equation (3), W_1 and H_1 represent the height and width of the input image. F is the size of the convolution kernel. P denotes the value of padding. S indicates the step size when the convolution kernel performs convolution operation. H_2 and are the length and width of the output dimension.

Following the convolution layer is generally the pooling layer whose scanning process is the same as that of the convolution layer. The pooling layer can reduce the resolution of the feature map, thereby simplifying the network parameters. Generally speaking, the pooling layer involves two operations: max pooling based on operation and mean pooling based on average. Max pooling is most commonly used and realized by taking a receiving domain's maximum value. Mean pooling adds all selected pixel values and then averages them.

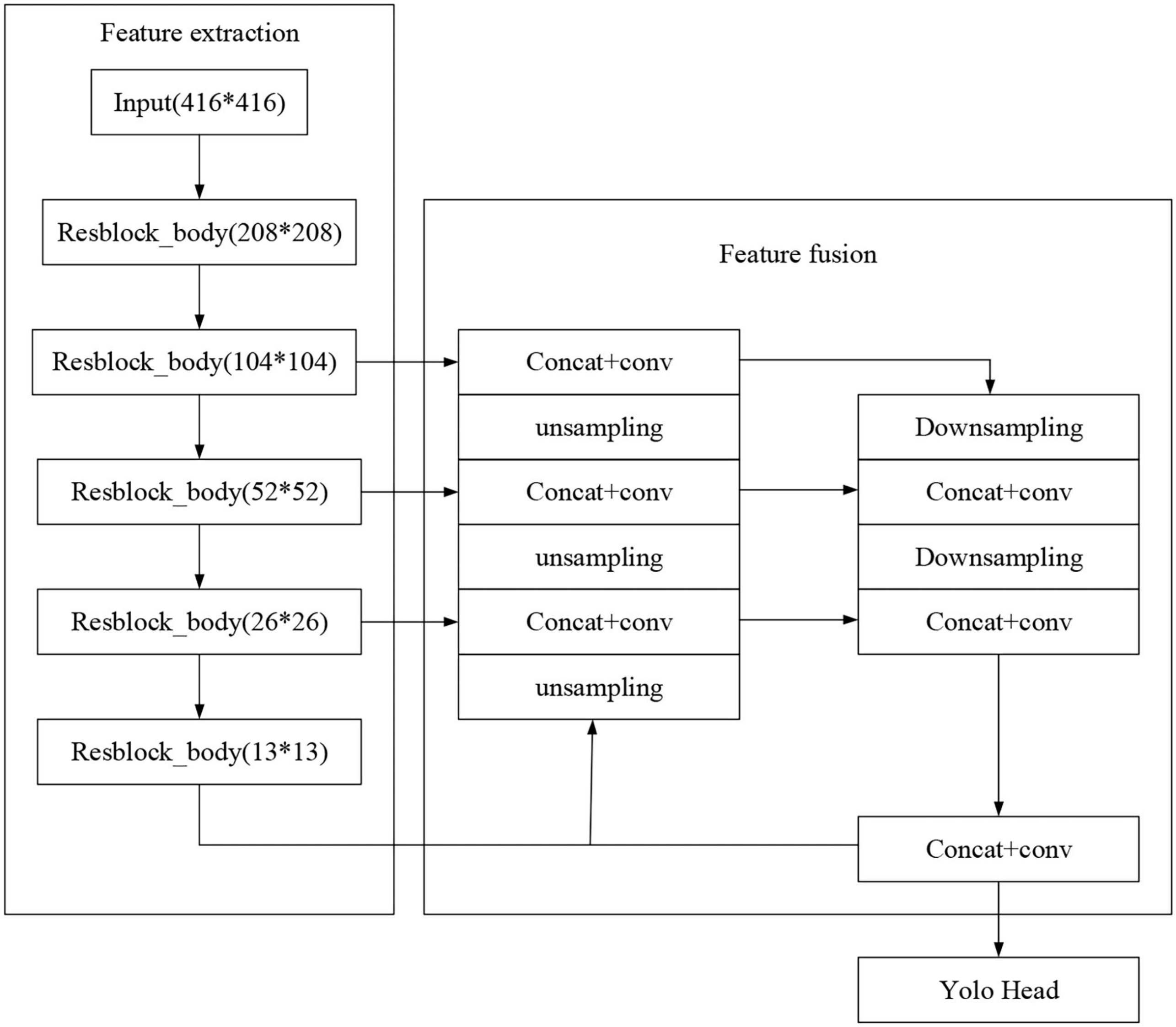


FIGURE 7 | Network structure diagram.

A normalization operation often follows the pooling layer to avoid the impact of nodes with large data values on classification. The purpose of normalization is to contain the input within an acceptable range [0, 1]. In CNN, the data normalization can be conducted before or after the fully connected layer or elsewhere, depending on the network structure.

Finally, the classification task is realized by the fully connected layer, which is mainly composed of convolution kernels of different sizes. The softmax layer can classify the fully connected layer's output. Additionally, a fully connected layer needs to optimize a substantial number of parameters, accounting for almost 80% of the network parameters. Some scholars have used the global average pooling to replace the fully connected layer, enhancing the network detection ability.

The first research on CNN can be traced back to the twentieth century. Common CNN structures include convolution, pooling, activation, and fully connected layers. Previous feature extraction networks mainly reduce the data dimension through a series of convolution downsampling and extract or sort out valuable features for subsequent use. In downsampling, the model captures and learns the object features. These semantically rich feature layers can enable the model to judge the image types. Still, the increase of the network depth also trades off the position information of the object in the original image, thus bringing great difficulties to the detection of small entities.

In the forward propagation of the network, Mean Square Error (MSE) is generally used to measure network loss. Suppose c classes and N training samples for a classification problem. Then, Equation (4) holds.

$$E^N = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{c}(t_k^n - y_k^n)^2 \tag{4}$$

In Equation (4), t_k^n represents the dimension k of the sample n. y_k^n means the k-th output of the network. The production of the multi-classification task often differs according to the different activation functions. Generally, only the output node corresponding to the input is positive, and the bits or nodes of other classes are 0 or negative. In the backpropagation of a sample, the error of the sample n is calculated according to Equation (5).

$$E^n = \frac{1}{2}\sum_{k=1}^{c}(t_k^n - y_k^n)^2 = \frac{1}{2}\left\| t^n - y^n \right\|_2^2 \tag{5}$$

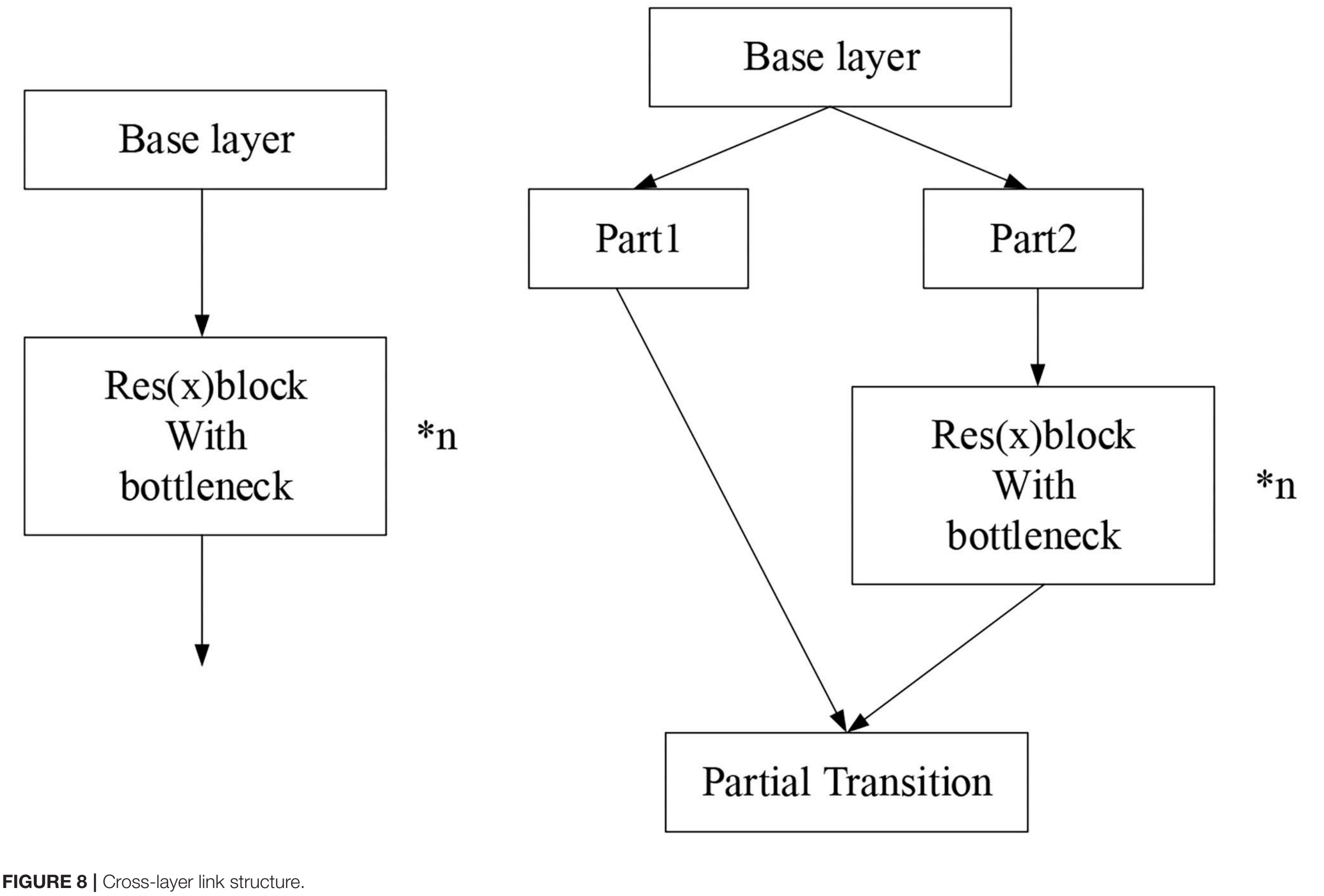


FIGURE 8 | Cross-layer link structure.

In the traditional fully connected neural network, it is necessary to calculate the partial derivative of the loss function about each weight of the network according to the rule of backpropagation. Here, *I* represents the current layer. Equation (6) describes the output of the current layer:

$$x^l = f(u^l), u^l = W^l x^{l-1} + b^l \tag{6}$$

There are various output activation functions, such as Sigmoid or Tanh. Sigmoid compresses the output to [0, 1], so the final output average ∼0. Therefore, if the mean of training data is normalized to 0 with variance 1, the convergence can be accelerated.

The backpropagation of CNN can be called the sensitivity basis of each neuron, meaning that the error changes as much as the basis changes. Equation (7) expresses the sensitivity basis.

$$\frac{\partial E}{\partial b} = \frac{\partial E}{\partial u}\frac{\partial u}{\partial b} = \delta \tag{7}$$

When $\frac{\partial u}{\partial b} = 1$, there is Equation (7); in other words, the sensitivity and error of the basis are equal to the derivative ($\frac{\partial E}{\partial u}$) of all inputs of a node. This reciprocal transformation allows errors in the upper layer to propagate back to the lower layer. The backpropagation process can be described as Equation (8).

$$\delta^l = (W^{l+1})^T \delta^{l+1} \mathrm{O} f'(u^l) \tag{8}$$

In Equation (8), O is to multiply each corresponding element in the matrix. Equation (9) expresses the sensitivity of neurons.

$$\delta^L = f'(u^L)\mathrm{O}(y^n - t^n) \tag{9}$$

Delta rule is generally used in weight updating. The input delta is used for amplification and contraction for each neuron input. Then, the input and the sensitivity of the *l*-th layer are cross-multiplied to represent the derivative of the error to the weight matrix of the *l*-th layer network. The final weight update

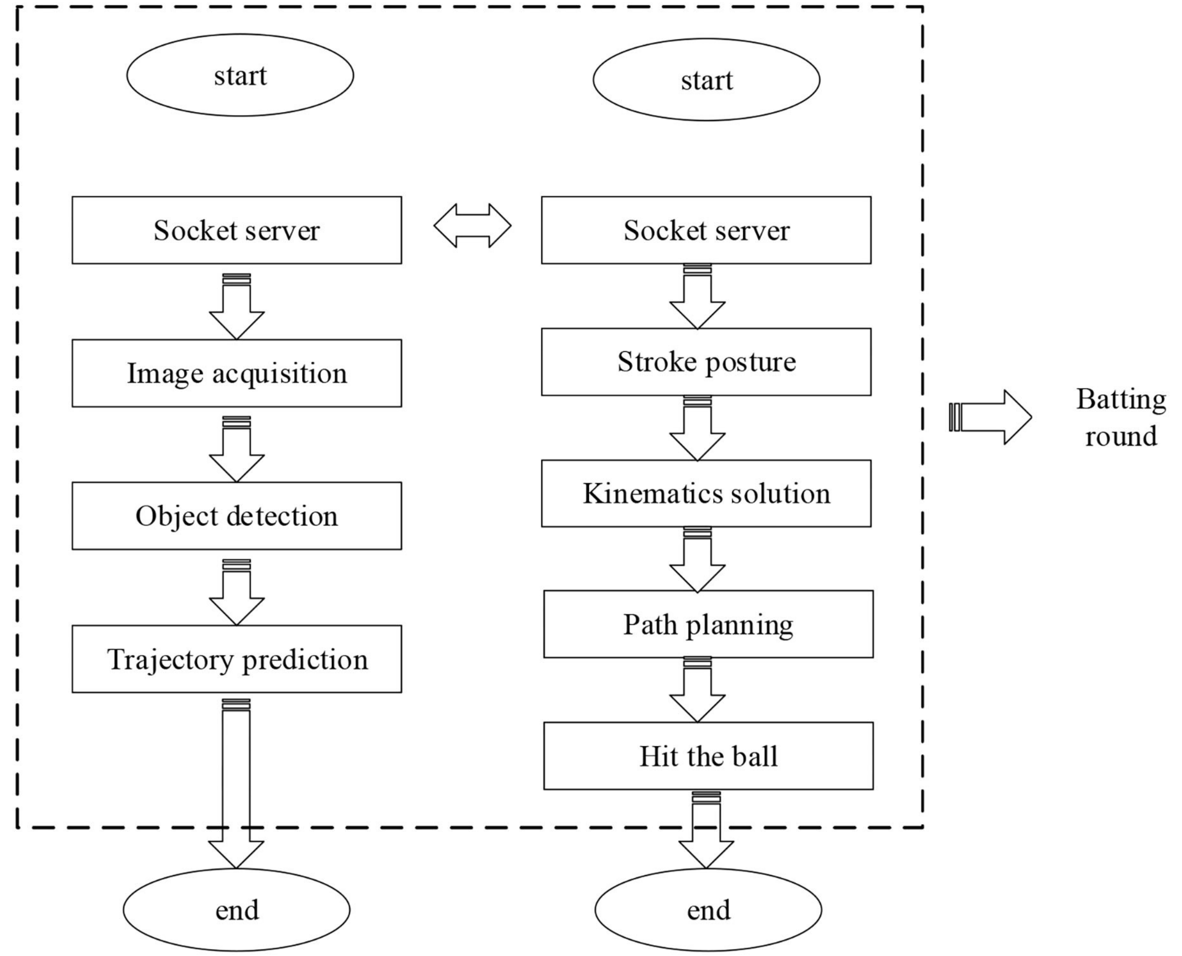


FIGURE 9 | Hitting flow of the table tennis robot (TTR) system.

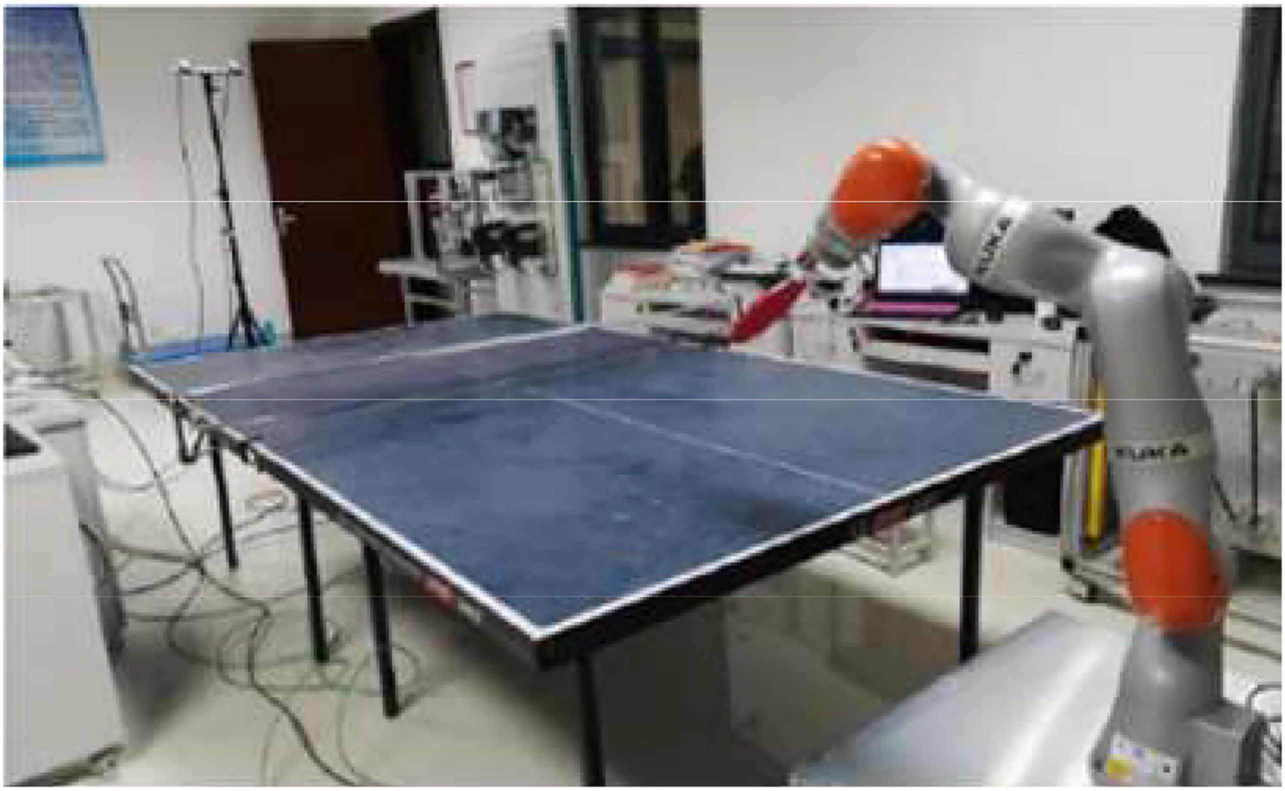


FIGURE 10 | Physical system of the 7-DOF table tennis robot.

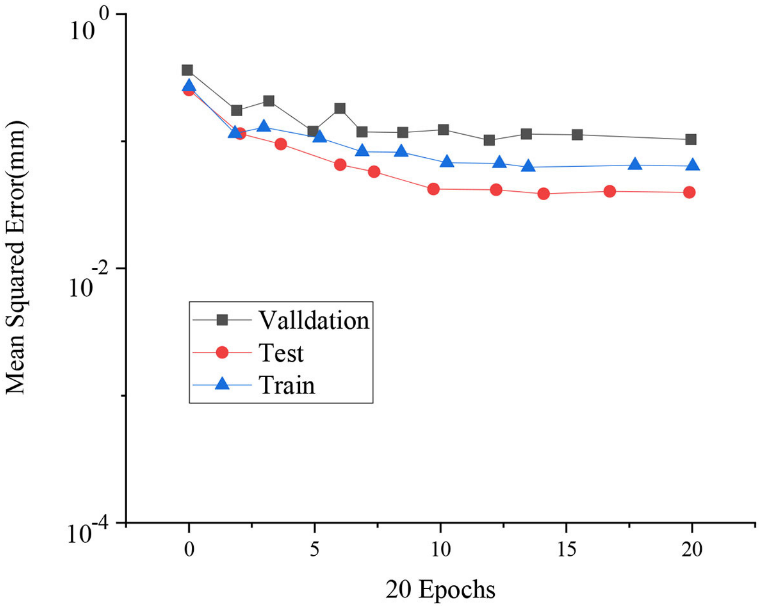


FIGURE 11 | MSE of fitting training based on LM algorithm.

needs to be multiplied by a negative learning rate, as shown in Equation (10):

$$\begin{aligned} \frac{\partial E}{\partial W^l} &= x^{l-1}(\delta^l)^T \\ \Delta W^l &= -\eta \frac{\partial E}{\partial W^l} \end{aligned} \tag{10}$$

In Equation (10), η is the specific learning rate corresponding to each weight. The operation of the convolution layer is generally referred to as downsampling. During downsampling, the image size is continuously reduced to make the abstract features of classification tasks more evident. Compared with downsampling, upsampling can enlarge the image resolution, and the resolution-enlarged image can exceed the original image in quality.

Interpolation is usually used for upsampling the input image, that is, inserting some new elements into the original image. The implementation methods are mainly divided into three categories.

(1) The nearest neighbor interpolation, also known as zero-order interpolation, uses the gray value of the image for interpolation. For a pixel in the original image, the gray value of the nearest input pixel value is inserted into the transformed image. The advantage of this method is that the calculation is simple and easy to implement, but the accuracy is low. It will cause contour or texture blurring.

(2) Deconvolution transforms the input low-dimensional features into high-dimensional features. However, deconvolution mainly restores the size of the image without completely restoring the original quality. For example, under a step size of 1, to convert a 2*2 feature image into its original 4*4 quality, zero padding will be performed on elements with padding = 2. Then a convolution kernel of 3*3 will be used for convolution operation. Finally, a 4*4 image is generated. Zero padding will be performed around each element when the step size is >1; then, the same operation with a step size of 1 will be served.

(3) Reverse pooling is mainly divided into max pooling and average pooling. The former needs to record the maximum value position in the feature map, and the other positions are filled with 0. The average pooling generally uses adjacent elements for padding.

The CNN-based target detection algorithms are mainly divided into candidate box-based two-stage and regression-based one-stage networks. In the two-stage network, candidate frame extraction and target detection are divided into two parts. Specifically, the target's candidate box needs to be extracted, based upon which the target detection is carried out. One-stage network cancels the target candidate box extraction of the two-stage network and features a single-step prediction. The one-stage network outputs a five-dimensional prediction result, including the object category, the center point position of the object, and the size of the prediction box, with a faster detection speed than a two-stage network.

Then, this study combines CNN with cross-layer link structures in the feature extraction network to solve the existing problems in the above methods. It constructs an FFN, as demonstrated in **Figure 7**.

The complex and extensive parameters result in the slow detection speed of neural networks. Therefore, reasonably lightening the network structure and streamlining network parameters is the key to improving the network detection speed. Accordingly, the present work combines the CNN and cross-layer link network in the feature extraction design. The input feature map is divided into two parts. One part is extracted through the residual network. The other part is directly concatenated and stacked with the output of

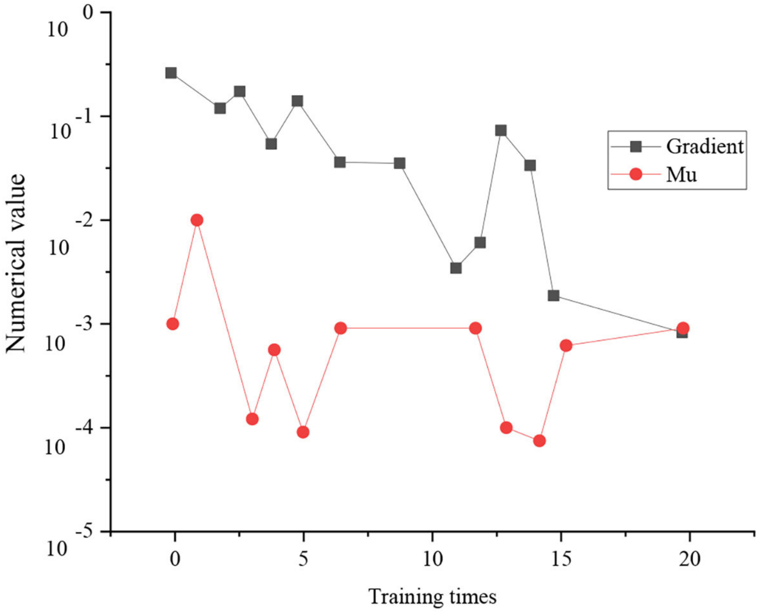


FIGURE 12 | Training state of neural network based on LM algorithm.

the residual network after feature integration through a 1*1 convolution kernel. In this way, a large concatenation residual network is constructed. Due to information loss during feature extraction by convolution kernel with a deepening network structure, such concatenation residual network can recover the lost information during network learning and strengthen feature extraction. The concatenated feature map will pass through a transition layer to prevent the network from learning repeated gradient information during backpropagation, optimizes the network gradient propagation, and reduces the convergence time of the network. **Figure 8** details the cross-layer link network.

The convolution kernel is set as follows. Res *block_body* is composed of 1*1 and 3*3 convolution kernels. The 1*1 convolution kernel reduces the network parameters by compressing (reducing) the dimensionality of the feature map, and the convolution kernel of 3*3 is used for feature extraction. Such a design can effectively streamline network parameters and increase its nonlinear ability. In forward propagation, each Res *block* stacks the input of this layer with the twice-convolutioned output. If the input is x, x + Res *block*(x) is the output. Such a structure constructs a basic residual block. Res *block_body* is the core part of building the whole network. Ordinary residual networks simply stack Res *block*. By comparison, the cross-layer link networks introduce an enormous residual edge Part 1 in the stacking process of fast residual. Part 2 continuously extracts the features of the input image, while Part 1 directly connects the input to the output of Part 2 with a small amount of processing.

Common activation functions include sigmoid, ReLU, tanh, leakyReLU, and Mish; Sigmoid can easily lead to gradient dispersion. When the ReLU is negative, the network neurons stop learning, and Mish will increase computation significantly. By contrast, leakyReLU is much more robust against these shortcomings. Therefore, this section adopts the leakyReLU as the activation function to simple calculation, preventing

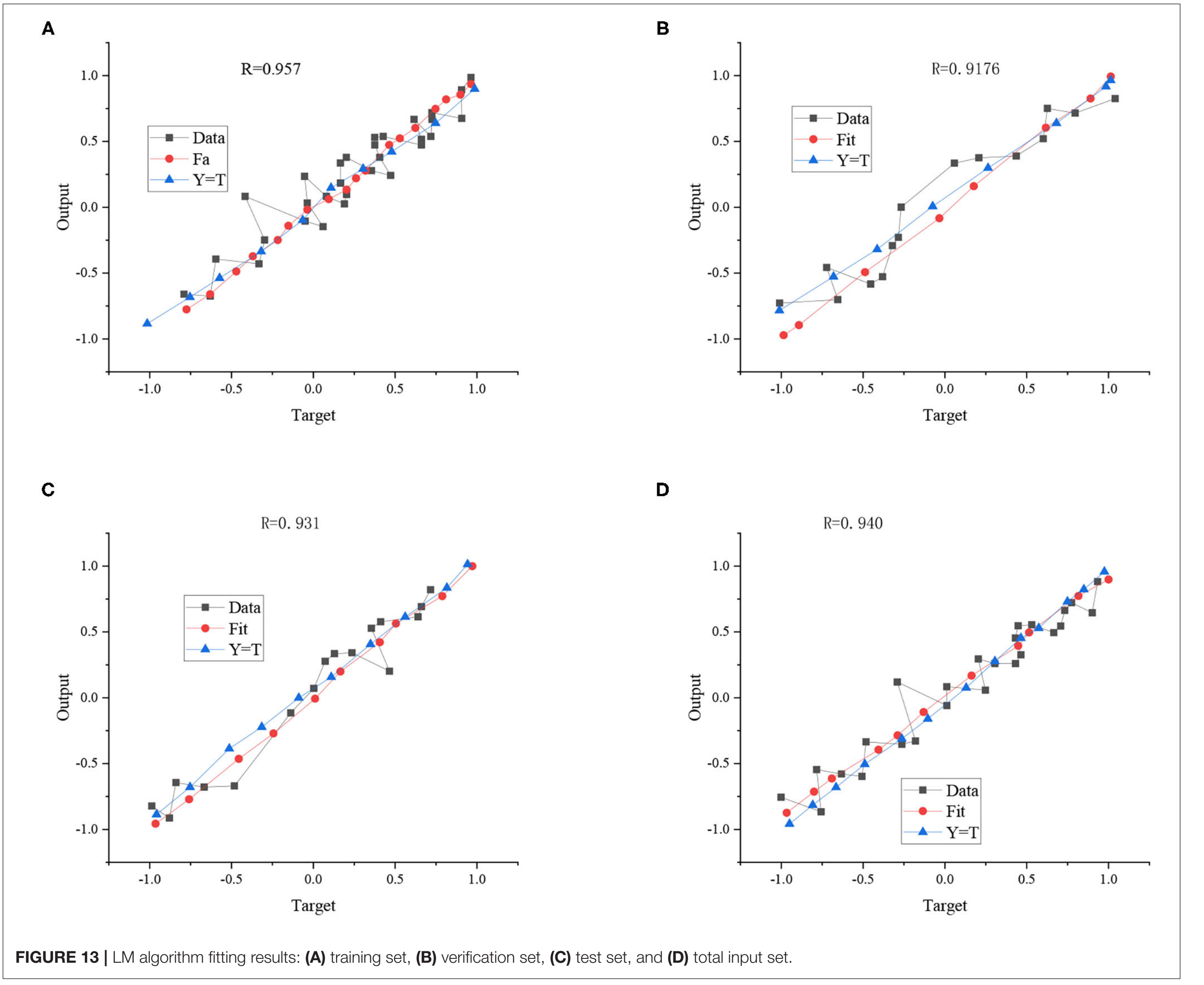


FIGURE 13 | LM algorithm fitting results: **(A)** training set, **(B)** verification set, **(C)** test set, and **(D)** total input set.

gradient explosion and dispersion. Equation (11) indicates the leakyReLU:

$$f(x) = \max(ax, x) \tag{11}$$

Design of the Physical Robot System

The seven-DOF KUKA manipulator system adopts a communication module of a 32-bit system. The vision system used here is designed in the 64-bit operating system. Therefore, the programs of the two parts should be designed into two modules separately. On the contrary, the two modules need to be connected to realize signal transmission in practice. The abstract layer encapsulates the Transmission Control Protocol/Internet Protocol (TCP/IP) family and only provides several interfaces to the application layer. It only needs to call these interfaces to realize real-time information

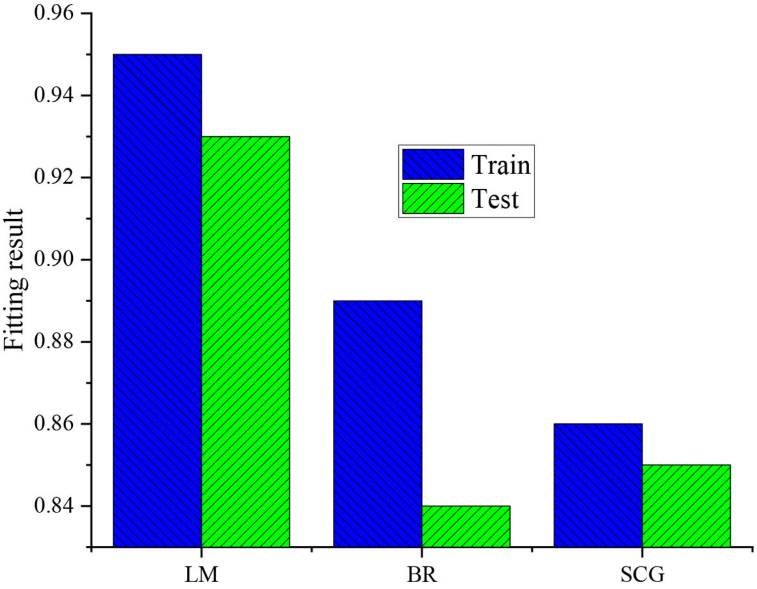


FIGURE 14 | Fitting comparison results.

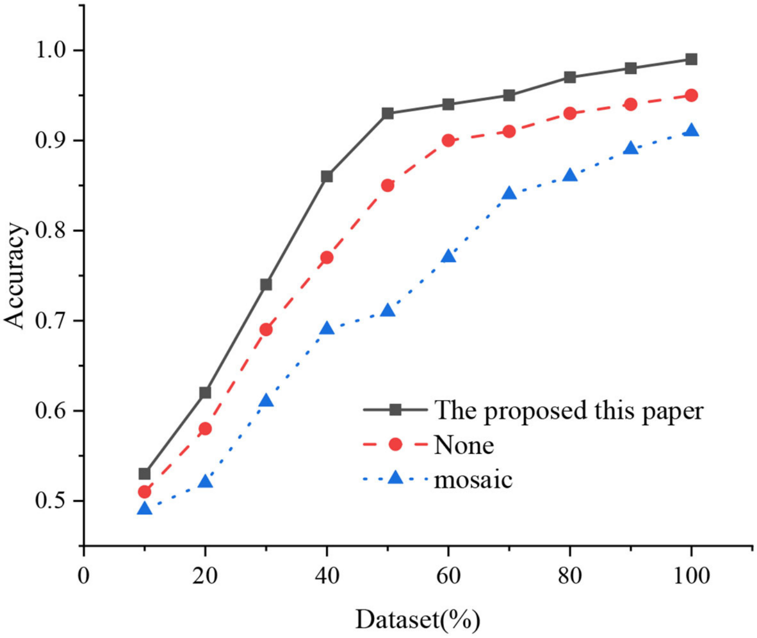


FIGURE 15 | Accuracy comparison after data enhancement.

communication when in use. **Figure 9** displays the TTR system designed here.

Figure 10 illustrates the physical design of the 7-DOF robot established here.

RESULTS AND DISCUSSION

Results of the Performance of DL-Based TTR

Figure 11 reveals the results of MSE of fitting training based on the LM algorithm.

The three curves in **Figure 11** represent the verification performance after different data sets are input into the model. The training time is 20, and the MSE of the test data set reaches the minimum (0.057) when trained 14 times. Overall, the error of the test set is smaller than the training set and the verification set.

Figure 12 displays the training state of the neural network based on the LM algorithm.

As shown in **Figure 12**, there are two indexes to judge the training state of neural networks, namely, the training gradient of the LM algorithm and Mu value. Obviously, with the increase of training times, the overall training gradient of neural networks tends to decline. After 20 times of training, the gradient comes to 0.001598. The second index is the Mu value, a parameter in the training model algorithm. According to the Mu value, the overall trend first increases with the training times and then declines. After 20 times of training, the Mu value is 0.001.

Figure 13 shows the fitting results of the LM algorithm.

In **Figure 13**, R represents the correlation coefficient (CC) between the expected and fitting results, and the value R ranges as $0 < R < 1$. Generally speaking, the closer the R-value is to 1, the closer the fitting result is to the expected result. The R values of the experimental training and test training results are 0.957 and 0.931, respectively, which shows that the ball landing point's initial velocity, rotation, and coordinates have a significant correlation.

BR algorithm and SCG algorithm are used for comparison to verify the superiority of the LM algorithm. **Figure 14** displays the comparison results.

Figure 14 illustrates that the fitting results of the LM algorithm are 0.95, while the fitting results of the BR algorithm and SCG algorithm are only 0.89 and 0.86. According to the test results, the *R*-value of the LM algorithm reaches 0.93, while the R values of the BR algorithm and SCG algorithm are only 0.84 and 0.85. Thus, no matter the training results or tests, the fitting condition of the LM algorithm is better than the BR algorithm and SCG algorithm. From the overall fitting situation, the fitting results of the three methods all exceeded 0.84. Therefore, even if different fitting algorithms are chosen, the CC between the expected and fitting results is very high, further showing that initial velocity, rotation, and coordinates of ball landing point have a solid correlation.

Experimental Results of TTB Detection Based on FFN

Subsequently, the experiment first uses Mosaic for data enhancement. Four images are spliced into one picture during network training and sent to network training. However, the detection effect of TTB is reduced rather than being improved as expected. The reason is probably that the size of TTB is scaled down after splicing, thus becoming more difficult for the network to obtain the TTB location and the decline of positioning accuracy. The TTB in each image is expanded three times without

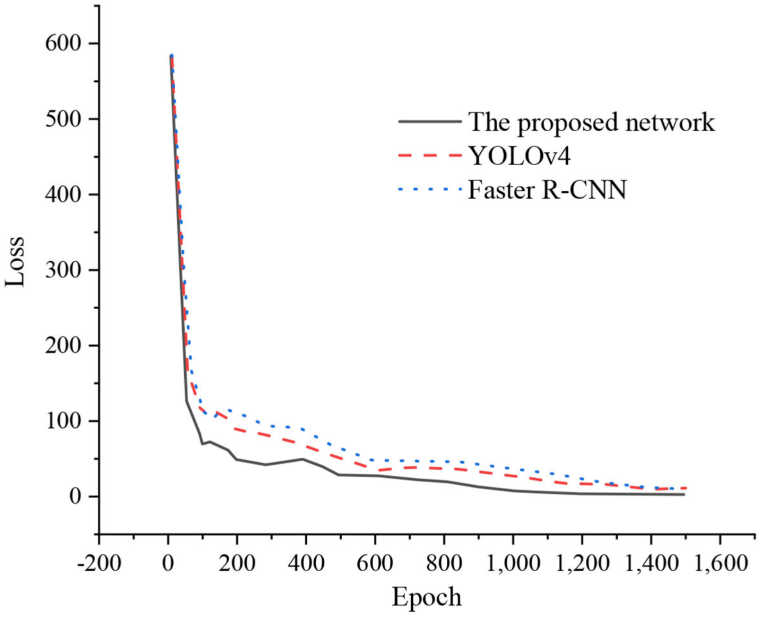


FIGURE 16 | Network loss comparison.

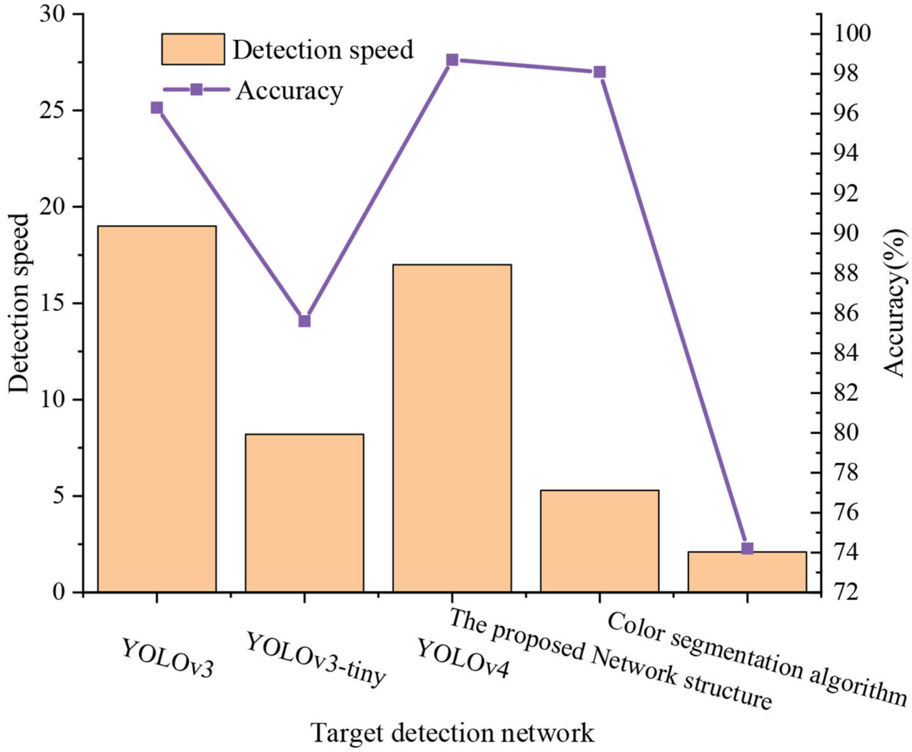


FIGURE 17 | Experimental results comparison.

changing the image resolution and diversified training samples inspired by Mosaic enhancement and combined with the target features and subject requirements of TTB. **Figure 15** illustrates the accuracy comparison after data enhancement.

This section builds a TTB-oriented target detection network using the Pytorch framework and Cuda to accelerate training. The dynamic calculation chart adopted by Pytorch can be changed and adjusted in real-time according to the actual calculation. At the same time, Pytorch also provides a toolkit for building a DL network with a clear and concise structure. It is a practical and efficient learning framework. In more than 8,000 data sets, this paper selects 7,000 images as the training set and the other 1,000 images as the test set. Then, **Figure 16** compares the network losses of the training and test sets.

After training, the network detection accuracy can reach over 98%, and the detection response can be down to 5.3 ms, thus meeting the requirements of the table tennis vision system. First, the experiment compares some of the latest one-stage target detection networks, YOLOv3, YOLOv3-tiny, and YOLOv4, using the same laboratory equipment and the same training data set and test set. **Figure 17** unveils the experimental results.

Figure 17 suggests that although the latest one-stage Yolo series models have achieved high accuracy in the TTB-oriented target detection task, the detection response is insufficient. Their network depth is more profound, and the amount of calculation and parameters are also more considerable, so they are not suitable for the research task of this study. The traditional color segmentation algorithm has advantages in detection response and can detect the target quickly, but the detection accuracy is not ideal. In particular, when the color of TTB changes, the detection ability will decline again. Therefore, the proposed TTB-oriented target detection network meets the table tennis real-time hitting and has very high accuracy.

Experimental Results of the 7-DOF Table Tennis Robot System

Based on the physical system, this study has carried out the hitting experiments of different rotating balls, including the ball without spin, topspin, backspin, left-sidespin, and right-sidespin. The ball speed is slow (4 m/s) and fast (6 m/s). **Table 2** provides the experimental results to analyze the feasibility and accuracy of this method in the physical system of TTR.

In **Table 2**, A represents the comparative experiment, 0 represents the ball without spin, 1 denotes topspin, 2 signifies left-sidespin, 3 represents backspin, and 4 represents right-sidespin. In the case of a fast ball (6 m/s), the success rate of hitting the ball without spin can reach 68.5%. It can still ensure a specific success rate in receiving slow table tennis with spin. The results prove that the rotating ball discrimination method based on human experience reported here can take into account the spin characteristics of table tennis and guarantee the hitting attitude

TABLE 2 | Hitting test results.

	Low speed					Fast speed				
Type	**0**	**1**	**2**	**3**	**4**	**0**	**1**	**2**	**3**	**4**
Number of trails	200	50	50	50	50	200	50	50	50	50
A success times	97	–	–	–	–	69	–	–	–	–
A success rate/%	48.5	–	–	–	–	34.5	–	–	–	–
B success times	158	36	34	37	29	137	24	21	24	20
B success rate/%	79	72	68	74	58	68.5	48	42	48	40

choice of the manipulator. However, the success rate of hitting the fast ball significantly decreases. Considering that the physical system of the seven-DOF KUKA mechanical arm used here has a slow control speed, the control system can only control the robotic arm to hit the ball through the slow mode. Consequently, the mechanical arm can only make effective hitting action after fast balls fly away from the table. Therefore, it is essential to improve the success rate of all kinds of balls in the rapid mode of a mechanical arm.

CONCLUSION

This study mainly calculates and predicts the rotation, trajectory, and velocity of TTB using several AI algorithms. Experiments have proved a close correlation among speed, spin, and landing point of TTB, which provides a solid foundation for the reverse rotation of the TTB trajectory. Specifically, the experiment compares the kernel function difference of several Machine Learning algorithms on the model prediction performance. Finally, the optimal fitting method and kernel function is determined. Additionally, the algorithm based on Kinect depth camera plans the player's hitting action and predicts the rotation and trajectory of TTB by identifying and classifying the dynamic state of TTB. The results demonstrate that Kinect and SVM algorithms can achieve satisfying recognition results and high recognition accuracy in TTB-oriented target recognition, trajectory prediction, and rotation prediction. The network structure reported here has an excellent performance in predicting TTB motion state and player motion recognition.

Still, there are some challenges and problems in the process. First, most experiments are still in the laboratory environment. Follow-up research will focus on live games and use camera videos to enhance the recognition accuracy of TTB rotation. Second, there are many interference factors in real competition applications. Solving the interference of external factors is a crucial problem to be solved in future research. Third, the experiment uses insufficient samples and a single experimental environment. Thus, it is necessary to add more examples from other environments to calculate the rotation more accurately and improve the recognition accuracy of TTB.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

ETHICS STATEMENT

The studies involving human participants were reviewed and approved by Shanghai Polytechnic University Ethics Committee. The patients/participants provided their written informed consent to participate in this study. Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

REFERENCES

Al-Mukhtar, M., and Al-Yaseen, F. (2020). Modelling water quantity parameters using artificial intelligence techniques, a case study Abu-Ziriq Marsh in south of Iraq. *IOP Conf. Ser. Mater. Sci. Eng.* 737, 012156. doi: 10.1088/1757-899X/737/1/012156

Carreras, J., Hamoudi, R. A., and Nakamura, N. (2020). Artificial intelligence analysis of gene expression data predicted the prognosis of patients with diffuse large B-Cell lymphoma. *Tokai J. Exp. Clin. Med.* 45, 37–48.

Forghani, R. (2020). Precision digital oncology: emerging role of radiomics-based biomarkers and artificial intelligence for advanced imaging and characterization of brain tumors. *Radiol. Imaging Cancer* 2, e190047. doi: 10.1148/rycan.2020190047

Geffen, O., Yitzhaky, Y., Barchilon, N., Druyan, S., and Halachmi, I. (2020). A machine vision system to detect and count laying hens in battery cages. *Animal* 14, 2628–2634. doi: 10.1017/S1751731120001676

Glassman, J., Humphreys, K., Yeung, S., Smith, M., Jauregui, A., Milstein, A., et al. (2021). Parents' perspectives on using artificial intelligence to reduce technology interference during early childhood: cross-sectional online survey. *J. Med. Internet Res*. 23, e19461. doi: 10.2196/19461

Gomez-Gonzalez, S., Prokudin, S., Schölkopf, B., and Peters, J. (2020). Real time trajectory prediction using deep conditional generative models. *IEEE Robot. Automat. Lett.* 5, 970–976. doi: 10.1109/LRA.2020.2966390

Gomis-Fons, J., Yamanee-Nolin, M., Andersson, N., and Nilsson, B. (2021). Optimal loading flow rate trajectory in monoclonal antibody capture chromatography. *J. Chromatogr. A* 1635, 461760. doi: 10.1016/j.chroma.2020.461760

Hildebrand, L. A., Pierce, C. J., Dennis, M., Paracha, M., and Maoz, A. (2021). Artificial intelligence for histology-based detection of microsatellite instability and prediction of response to immunotherapy in colorectal cancer. *Cancers* 13, 391. doi: 10.3390/cancers13030391

Hu, S. (2020). Research on data acquisition algorithms based on image processing and artificial intelligence. *Int. J. Patt. Recogn. Artif. Intell*. 34, 1–13. doi: 10.1142/S0218001420570050

Jammeli, H., Ksantini, R., Abdelaziz, F. B., and Masri, H. (2021). Sequential artificial intelligence models to forecast urban solid waste in the city of Sousse, Tunisia. *IEEE Trans. Eng. Manage*. 99, 1–11. doi: 10.1109/TEM.2021.3081609

Li, Y. (2021). Intelligent recognition system for the curves trajectory and track capture using artificial intelligence technology. *J Phys. Conf. Ser*. 1952, 042041. doi: 10.1088/1742-6596/1952/4/042041

Likitha, V. M. (2021). Artificial intelligence at health care industry Name-Likitha Vm Reg.No-20BEC0614 school of electronics and communication engineering VIT Vellore introduction. *Int. J. Hum. Arts Med. Sci*. 5, 39–42.

Lin, A., Kolossváry, M., Motwani, M., Išgum, I., Maurovich-Horvat, P., Slomka, P. J., et al. (2021). Artificial intelligence in cardiovascular imaging for risk stratification in coronary artery disease. *Radiol. Cardiothorac. Imaging* 3, e200512. doi: 10.1148/ryct.2021200512

Lin, H. I., Yu, Z., and Huang, Y. C. (2020). Ball tracking and trajectory prediction for table-tennis robots. *Sensors* 20, 333. doi: 10.3390/s20020333

Muto, M., Kondo, T., Yoshioka, M., Fukuyama, K., Suga, J., Sato, M., et al. (2021). From the standpoint of a core center hospital for cancer genome medicine-establishment of a provision system for wide-area precision oncology. Gan to kagaku ryoho. *Cancer Chemother*. 48, 866–872.

Nataraj, S. K., Pandiyan, P. M., Yaacob, S. B., and Adom, A. H. (2021). Intelligent robot chair with communication aid using TEP responses and higher order spectra band features. *Informatics* 17, 92–103. doi: 10.37661/1816-0301-2020-17-4-92-103

Oliveira, A. F., Afonso, M. Q., Azevedo, V. A., and Lemos, M. (2021). Bioinformatics and artificial intelligence approaches for unraveling evolutionary mechanisms in taxonomic groups of neurohypophysial hormones family. *Int. J. Swarm Intell. Evol. Comput*. 10, 211.

Park, S., and Hainaut, P. (2020). Editorial: dissecting the pathological bone marrow niche through single-cell omics and artificial intelligence. *Curr. Opin. Oncol*. 32, 137–138. doi: 10.1097/CCO.0000000000000604

Payedimarri, A. B., Concina, D., Portinale, L., Canonico, M., Seys, D., Vanhaecht, K., et al. (2021). Prediction models for public health containment measures on COVID-19 using artificial intelligence and machine learning: a systematic review. *Int. J. Environ. Res. Public Health* 18, 4499. doi: 10.3390/ijerph18094499

Pezzo, M. V., and Beckstead, J. W. (2020). Patients prefer artificial intelligence to a human provider, provided the ai is better than the human: a commentary on Longoni, Bonezzi and Morewedge (2019). *Judge. Dec. Mak*. 15, 443–445.

Riguzzi, F., Kersting, K., Lippi, M., and Natarajan, S. (2019). Editorial: statistical relational artificial intelligence. *Front. Robot. AI* 6, 68. doi: 10.3389/frobt.2019.00068

Salvatore, C., Interlenghi, M., Monti, C. B., Ippolito, D., Capra, D., Cozzi, A., et al. (2021). Artificial intelligence applied to chest X-ray for differential diagnosis of COVID-19 pneumonia. *Diagnostics* 11, 530. doi: 10.3390/diagnostics11030530

Steiner, D. F., Nagpal, K., Sayres, R., Foote, D. J., Wedin, B. D., Pearce, A., et al. (2020). Evaluation of the use of combined artificial intelligence and pathologist assessment to review and grade prostate biopsies. *JAMA Netw. Open* 3, e2023267. doi: 10.1001/jamanetworkopen.2020.23267

Sun, Y. (2021). Cloud edge computing for socialization robot based on intelligent data envelopment. *Comput. Electr. Eng*. 92, 107136. doi: 10.1016/j.compeleceng.2021.107136

Tkatek, S., Bahti, S., and Abouchabaka, J. (2020). Artificial-intelligence-for-improving-the-optimization-of-nphard-problems-a-review. *Int. J. Adv. Trends Comp. Sci. Eng*. 9, 7411–7420. doi: 10.30534/ijatcse/2020/73952020

Tomasevic, I., Tomovic, V., Ikonic, P., Rodriguez, J. M., Barba, F. J., Djekic, I., et al. (2019). Evaluation of poultry meat colour using computer vision system and colourimeter: Is there a difference? *Br. Food J*. 121, 1078–1087. doi: 10.1108/BFJ-06-2018-0376

Zhang, J. (2021). Automatic detection method of technical and tactical indicators for table tennis based on trajectory prediction using compensation fuzzy neural network. *Comput. Intell. Neurosci*. 2021, 1–12. doi: 10.1155/2021/3155357

Zhang, X., Zhang, K., Lin, D., Zhu, Y., Chen, C., He, L., et al. (2020). Artificial intelligence deciphers codes for color and odor perceptions based on large-scale chemoinformatic data. *GigaScience* 9, giaa011. doi: 10.1093/gigascience/giaa011

Zhang, Y., Zhou, Y. M., Liao, Z. H., Liu, G. Y., and Guo, K. C. (2021). Artificial intelligence-guided subspace clustering algorithm for glioma images. *J. Healthc. Eng*. 2021, 1–9. doi: 10.1155/2021/5573010

Zhao, H., Su, Z., Li, Q., Liu, F., and Liu, N. (2021). Real-time attitude propagation algorithm for high spinning flying bodies. *Measurement* 177, 109260. doi: 10.1016/j.measurement.2021.109260

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

ORIGINAL RESEARCH
published: 18 May 2022
doi: 10.3389/fnbot.2022.904017

R-STDP Spiking Neural Network Architecture for Motion Control on a Changing Friction Joint Robotic Arm

Alejandro Juarez-Lora, Victor H. Ponce-Ponce*, Humberto Sossa and Elsa Rubio-Espino*

Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico City, México

Neuromorphic computing is a recent class of brain-inspired high-performance computer platforms and algorithms involving biologically-inspired models adopting hardware implementation in integrated circuits. The neuromorphic computing applications have provoked the rise of highly connected neurons and synapses in analog circuit systems that can be used to solve today's challenging machine learning problems. In conjunction with biologically plausible learning rules, such as the Hebbian learning and memristive devices, biologically-inspired spiking neural networks are considered the next-generation neuromorphic hardware construction blocks that will enable the deployment of new analog *in situ* learning capable and energetic efficient brain-like devices. These features are envisioned for modern mobile robotic implementations, currently challenging to overcome the pervasive von Neumann computer architecture. This study proposes a new neural architecture using the spike-time-dependent plasticity learning method and step-forward encoding algorithm for a self tuning neural control of motion in a joint robotic arm subjected to dynamic modifications. Simulations were conducted to demonstrate the proposed neural architecture's feasibility as the network successfully compensates for changing dynamics at each simulation run.

Keywords: neuromorphic, robotics, reinforcement learning, STDP, reward modulation, control theory, applications

Edited by:
Jose De Jesus Rubio,
Instituto Politécnico Nacional (IPN),
Mexico

Reviewed by:
Luis Arturo Soriano,
Chapingo Autonomous University,
Mexico
Genaro Ochoa,
Instituto Tecnológico Superior de
Tierra Blanca, Mexico
Ricardo Balcazar,
Technological Institute of Higher
Studies of Coacalco (TESCO), Mexico

***Correspondence:**
Alejandro Juarez-Lora
jjuarezl2020@cic.ipn.mx
Victor H. Ponce-Ponce
vponce@cic.ipn.mx

Received: *25 March 2022*
Accepted: *14 April 2022*
Published: *18 May 2022*

Citation:
Juarez-Lora A, Ponce-Ponce VH, Sossa H and Rubio-Espino E (2022) R-STDP Spiking Neural Network Architecture for Motion Control on a Changing Friction Joint Robotic Arm. Front. Neurorobot. 16:904017. doi: 10.3389/fnbot.2022.904017

1. INTRODUCTION

Spiking neural networks (SNNs), also called the third generation of neural networks, represent a new design paradigm where some biological neural dynamics are replicated, with similar energy efficiency and *in situ* learning capabilities, as seen in living organisms, whereas hardware miniaturization is feasible. Neuromorphic computing (Saxena et al., 2018; Kendall and Kumar, 2020) emerges as an effort to create built-in neural hardware, emulating the neuronal impulsive-like electrical activity and *in-situ* synaptic learning in analog devices. Therefore, neuron dynamics have to be translated into circuit proposals to achieve these behaviors. As for the synapses, where learning occurs in biological brains, memristors are taking their role as the electrical element counterpart (Zhang et al., 2021). These devices, theorized by Leon Chua, relate flux with charge, resulting in a variable resistor. The conductivity is given by how much current has flowed between its ports in a determined period. Therefore, its conductance serves as the synaptic weight which can be tuned by applying current (Yue and Parker, 2019). At Zamarreño-Ramos et al. (2011), an exploration into how neurons and memristors can be interconnected as an array scheme to achieve large scale spiking systems, using synaptic time-dependant plasticity (STDP), is presented. Since then, several proposals have been presented. Recently, a memristor analog crossbar circuit is used to emulate a single layer perceptron for the MNIST image classification problem (Kim et al., 2021). In Shi et al. (2021), a circuit proposed to manage reward modulation is presented, setting the building blocks for implementation.

Cutting-edge neuromorphic implementations still demand going deeper into studying the neuron dynamics and plausible learning methods since the non-differentiable nature of the neuron dynamic doesn't allow the use of the well-known backpropagation synaptic weight adjustment; widely employed in ordinary artificial neural networks (ANN). Therefore, there are some open challenges to address before constructing high-performance neuromorphic devices, as well as encoding and decoding information techniques. According to Hu et al. (2022), learning algorithms used in SNNs are summarized in:

- Modified gradient-descendent-based algorithms: As neuron models are non-differentiable, some modifications are pertinent to achieve the classical backpropagation learning rule, employed in most of the ANNs, i.e., SpikeProp (Kheradpisheh and Masquelier, 2020).
- Algorithms using a spike train kernel: Where an error function is used to compute and update synaptic weights, using a spike train kernel, i.e., SPAN (Mohemmed et al., 2012).
- Algorithms using synaptic plasticity: Based on Hebbian learning, the synaptic weights tuning is given by the correlation of pre and post-synaptic spikes. In STDP, the modification of the neural strength connections is performed as the learning process occurs, as in biological brains (Hao et al., 2020).

Synaptic plasticity phenomena explain how learning is conducted in biological brains, enhancing conductivity between *neurons that fire together, wire together*, and deprecating those unused connections.

On other hand, information, usually shaped as an analog signal, has to be encoded into the neuron spike domain. The scientific community is still debating how information from the environment is converted into electrical neural activity. According to Dupeyroux et al. (2021), neuron spike coding methods can be classified into three categories.

- *Population encoding*: A group of n neurons, each one with different characterization (i.e., different τ_m, R_m, C_m), is set to be excited about an input current. As a result, at a given time-step, some neurons will spike faster than others. The characterization of neurons is made in such a way the domain of the input signal is distributed between the n neurons, using *tuning curves* (Voelker and Eliasmith, 2020).
- *Rate-based encoding*: One neuron is used to encode the variation of the input signal $\in [I_{min}, I_{max}]$. As larger an input signal gets, the spiking frequency of the neuron increases. A minimal input current traduces into a minimum spike frequency, inside a frequency interval $\in [f_{min}, f_{max}]$.
- *Temporal encoding*: Also called pulse coding, produces spikes according to a temporal change of the input signal. This is, if an input signal is constant, no spikes are produced, even if the signal is large. As soon the signal increases or decreases, spikes will be emitted.

While population encoding reaches the best performance, its efficiency is reduced, as it needs a huge amount of resources (neurons) in order to be implemented. Rate-based encoding has become the standard, but it presents the need to spike even with a zero input signal, increasing the power consumption. Besides, it can't encode negative values, as seen in Bing et al. (2019a), where a negative input signal has to be fed as its absolute value. The temporal encoding provides a time-based method, providing more information capacity per synaptic event, and it is supported by neurophysiological studies in auditory and visual processing in the brain (Guo et al., 2021). SNNs can send data encoded as the timing of spikes occurrences, allowing fast and low energy consumption hardware implementation, applicable to real-world robotics problems. Furthermore, SNNs are more prominent than non-spiking ANN as they profit from temporal stimulus information, referring to the precise timing of events that allows obtaining and processing of information.

Spiking neural networks on robotic design systems are a promising research topic as online learning, and huge computational capacities are commonly required in this field. For instance, at Chen et al. (2020), an SNN controls a 4-DOF (Degree of Freedom) manipulator arm using population encoding and a proposed learning rule. Bing et al. (2019b) use a reward-modulation learning rule to teach a differential robot how to track a path, using rate-based encoding to do conversion of visual input into spiking activity. A similar task is studied numerically by Bing et al. (2019b), controlling a snake's movement instead. Lu et al. (2021) achieve obstacle avoidance for an Ackerman-type mobile robot, using two neurons and two synapses, implemented on a digital development board. Bing et al. (2019a) achieve obstacle avoidance and goal-reaching for a differential robot, implementing separate neural control structures for each task. Over these articles, while control is achieved based on interaction with the environment, changing dynamics in the robot produced by weathering in the joints or unknown environmental perturbances are not addressed. A typical control strategy for a robotic open-chain manipulator requires re-tuning each time friction or mass on the robot changes, affecting performance. This article proposes an SNN architecture that learns how to reconstruct an input signal. Inspired by control theory, this structure is then used in a control loop, using the same input signals in a PID, but fed into the structure in order for the synaptic weights to evolve over changing dynamics on a 1-DOF robotic arm.

The document is structured as follows: Section 2 describes the control problem to be tackled, neuron and synapse dynamics, and how these are ensembled for a controlling proposal. Section 3 shows the simulation results of the proposed SNN

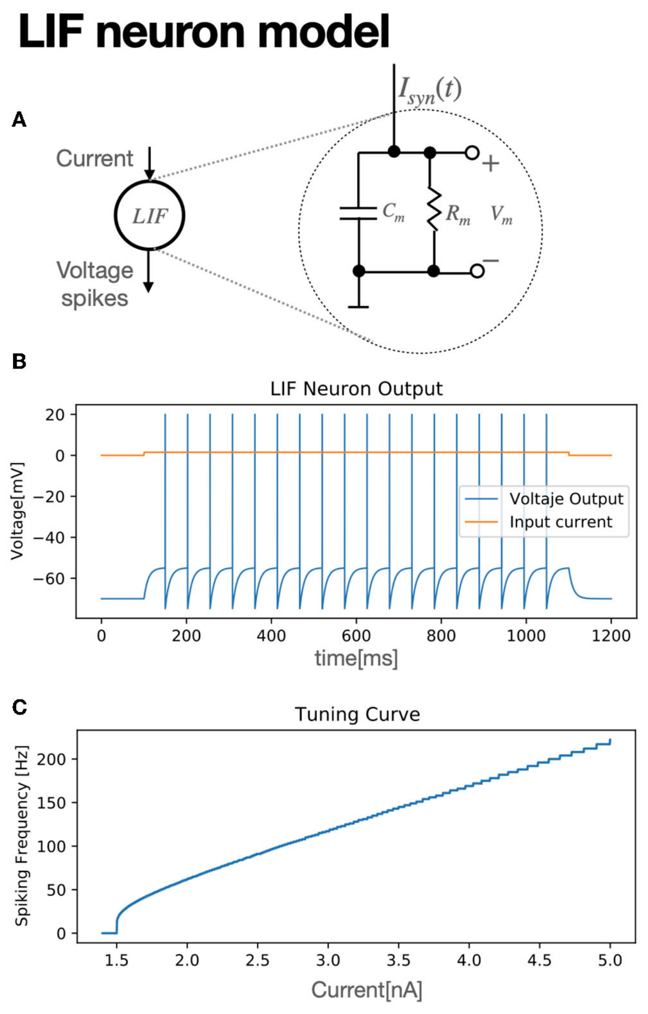


FIGURE 1 | LIF model **(A)** diagram of a spiking LIF neuron. **(B)** Spiking activity for a LIF neuron with $I_{syn} = 1.51\,nA$. **(C)** Tuning curve of the neuron model. Neuron parameters are in **Table 1**.

implemented in 1-link. Section 4 discusses results, advantages, and drawbacks, while at last, Section 5 is devoted to conclusion and future study.

2. MATERIALS AND METHODS

2.1. Control Problematic

According to Craig (1986), Lynch (2017), the dynamics of an open chain robotic manipulator can be written in joint space as:

$$\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) \tag{1}$$

Where $q = [\theta_1, \theta_2, ..., \theta_n]^T$ are the joint angles of the robotic arm with n DOFs, $M(\cdot)$ stands for inertia matrix terms, $C(\cdot)$ is the Coriolis's matrix and friction dynamics, $g(\cdot)$ are gravity compensation terms and $\tau = [\tau^1, \tau^2, ..., \tau^n]^T$ means the torque control for each joint. Typically, PID control strategies are the standard. Based on the desired state $x_d(t)$, a tracking error $q_e = q_d - q$ is defined, setting the control input $\tau(t)$ as:

TABLE 1 | Neuron and synapse modeling parameters.

Model	Parameters	Value
LIF neuron	Membrane resistance	$R_m = 10\ M\Omega$
	Membrane's capacitance	$C_m = 1\ nF$
	Time decay membrane	$\tau_m = 0.010\ s$
	Resting voltage	$E_l = -70\ mV$
	Reset voltage	$V_{reset} = -75\ mV$
	Spike voltage	$V_{spike} = 20\ mV$
	Threshold voltage	$V_{th} = -55\ mV$
RSTDP synapse	LTP scaling	$A_+ = 1$
	LTD scaling	$A_- = -1$
	Elegibity trace scale	$\tau_E = 0.010\ s$
	Min. Synaptic weight	$w_{min} = 1$
	Max. Synaptic weight	$w_{max} = 1,000$

$$\tau(t) = K_P\theta_e + K_i \int \theta_e(t)dt + K_d\dot{\theta}_e \tag{2}$$

At Equation (2), $K_P \in R^{n\times n}, K_d \in R^{n\times n}, K_i \in R^{n\times n}$ are the gain matrix for proportional, derivative, and integral control, which elements are zeros except in the diagonal. This strategy is the function of the tracking error, which on zero, there will be no control output. Consider:

$$\tau = \tilde{M}(q)\ddot{q} + \tilde{C}(q,\dot{q}) + \tilde{g}(q) \tag{3}$$

Here, $\tilde{M}, \tilde{C}, \tilde{g}$ represents our model representation of the plant, and it is perfect if $\tilde{M}(\ddot{q}) = M(\ddot{q})$, $\tilde{C}(q,\dot{q}) = C(q,\dot{q})$, and $\tilde{g}(q) = g(q)$. Therefore, if the control loop works on the estimation, it would work for the real model. Usually, in the development process of a robot controller, K_p, K_i, and K_d are tuned for initial $\tilde{M}, \tilde{C}$, and $\tilde{g}$. This becomes a problem as the robot's weathering modifies its dynamic properties, such as friction. Or perhaps, mass changes over time, as seen in biological limbs in living creatures.

2.2. Spiking Neural Network Modeling

In order to describe the proposed structure, a review of how a neuron generates spikes, how synapses store learning, and how to generate reward signals is presented.

2.2.1. Neuron Modeling

As an input stimulus is provided to the neuron cell, shaped as an input current, the membrane's potential v_m increases. Once it overpasses a threshold voltage v_{th}, the neuron produces a spike, and then it immediately resets its membrane potential to a reset voltage v_{reset}. The neuron cannot fire again until a certain refractory period has elapsed. Some differential equation models illustrate these neural dynamics with high biological plausibility but prohibitive computational cost such as Hodgin and Huxley or Izhikevich models (Izhikevich, 2004; Valadez-Godínez et al., 2020). Nonetheless, others with a lesser plausibility can compute the membrane potential with less effort degrading the accuracy,

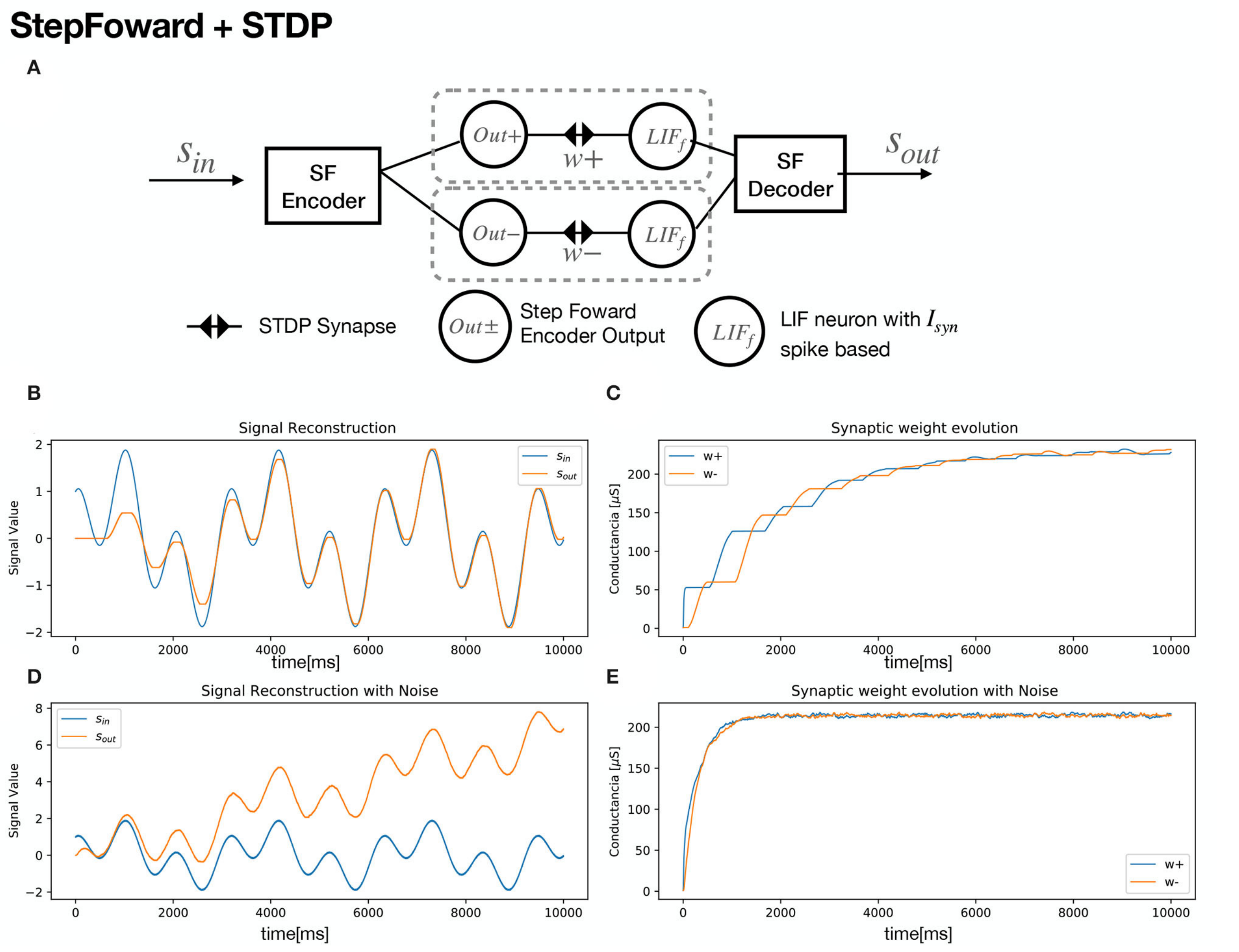


FIGURE 2 | Step Forward + STDP Neural structure. **(A)** Schematic of the structure. **(B)** Sinusoidal Signal $s_{in} = \sin(\omega t) + \cos(3\omega t)$ being reconstructed. **(C)** Synaptic Weight evolution. **(D)** Gaussian noise with $\sigma = 0.01$ was added to foster synaptic update. **(E)** Synaptic Weight evolution with added noise. Simulation Parameters are available at **Table 2**.

TABLE 2 | Step Foward + STDP Encoding simulation.

Model	Parameters	Value
Simulation parameters	Time step	$dt = 0.1ms$
	Signal angular velocity	$\omega = 2rad/s$
	Total time simulation	$10s$
SF encoder	Threshold	$s_{th} = 0.02$
	Initial base value	$s_b = 0$
SF decoder	Threshold	$s_{th} = 0.02$
	Initial base value	$s_b = 0$

but are still useful as a good model approximation, due that spikes generation with the same characteristics as biological neurons might not be necessary for circuit implementations, such as the *Leaky Integrate and Fire (LIF)* (Lu et al., 2021) model, given by:

$$\tau_m \frac{dv_m(t)}{dt} = -v_m(t) + E_l + R_m I_{syn} \tag{4}$$

At Equation (4), $v_m(t)$ represents the neuron's membrane potential, E_l is the resting potential of the neuron, R_m membrane resistance, $\tau_m = R_m C_m$ is the decay time for $v_m(t)$, being C_m the neuron's membrane capacitance. I_{syn} stands for the injected current to the neuron. Each time a spike arrives at the neuron, I_{syn} increases. On the other hand, if no spikes arrive at the neuron, the current decays. This phenomenon is described by LIF conductance-based model (Hao et al., 2020; Lu et al., 2021), composed of Equations (4 and 5):

$$\tau_m \frac{dI_{syn}}{dt} = -I_{syn} + C_m \sum_i^N w_{ij}\delta(t - t_i^f) \tag{5}$$

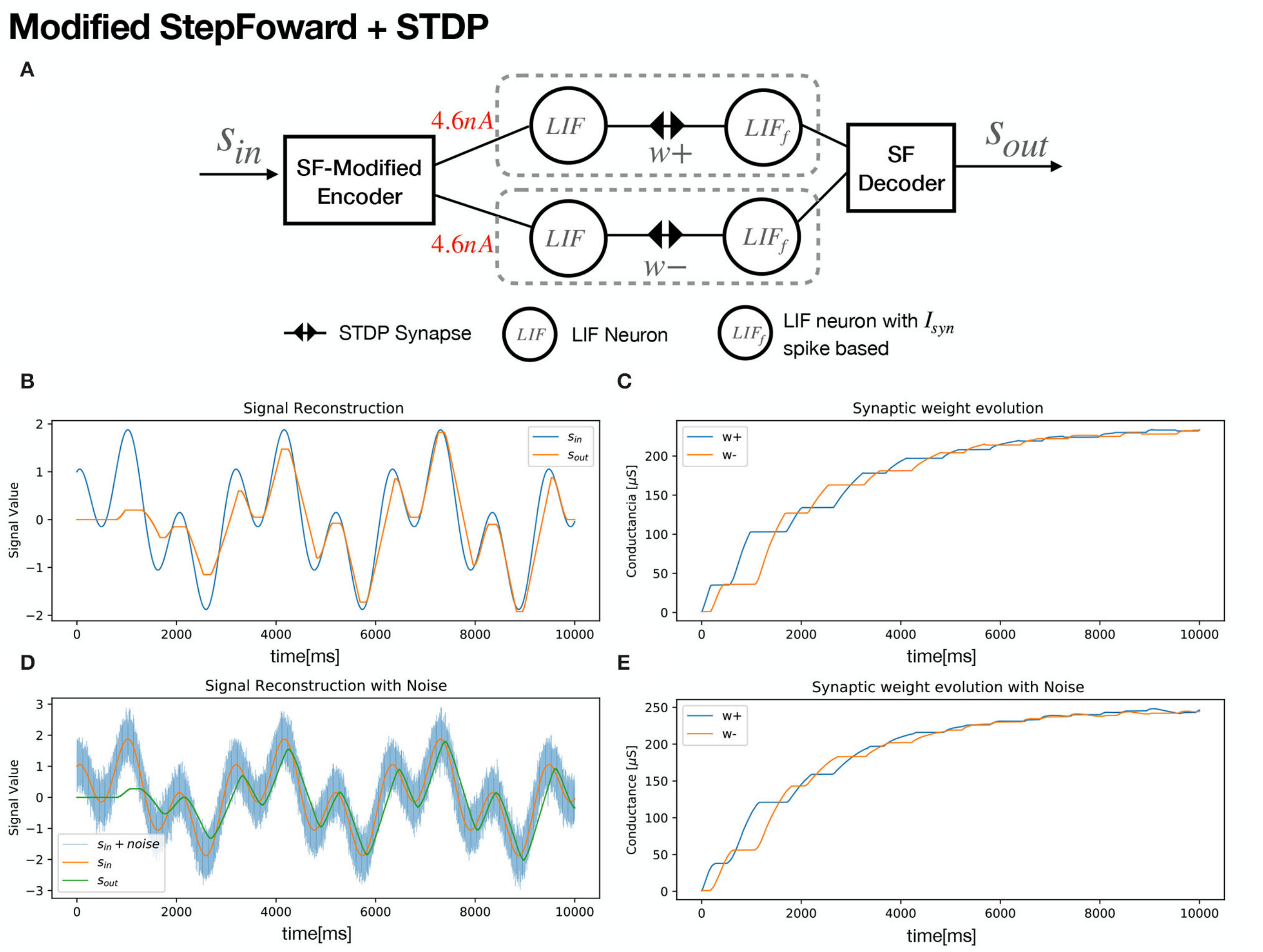


FIGURE 3 | Modified Step Forward + STDP Neural structure. **(A)** Schematic of the structure. **(B)** Sinusoidal Signal $s_{in} = \sin(\omega t) + \cos(3\omega t)$ being reconstructed. **(C)** Synaptic Weight evolution. **(D)** Gaussian noise with $\sigma = 0.3$ was added to foster synaptic update. **(E)** Synaptic Weight evolution with added noise. Simulation parameters are available at **Table 3**.

In Equation (5), w_{ij} is the synapse strength value between a presynaptic, $i-th$, neuron and a postsynaptic, $j-th$, neuron. As for each postsynaptic neuron, there can be N presynaptic neurons connected, t_i^f is then a vector with firing times from each of the N presynaptic neurons. δ is the Kronedecker delta function, which $\delta(x) = 1$ for $x = 0$ and $\delta(x) = 0$ for $x \neq 0$. Equation (5) assumes all presynaptic spikes have been produced at time t. For each time a new spike happens, $t_i^f = t$, therefore, $\delta(t - t_i^f) = 1$. Once the neuron threshold voltage v_{th} is over-passed, the neuron spikes, emitting a pulse of magnitude v_{spike}, then, the neuron resets to a reset potential $v = v_{reset}$ and it starts integrating again. **Figure 1A** shows the LIF structure model, while its spiking activity for a given fixed and variable input current is shown at **Figures 1B,C**, respectively.

TABLE 3 | Modified Step Foward + STDP Encoding simulation.

Model	Parameters	Value
Simulation parameters	Time step	$dt = 0.1\ ms$
	Signal angular velocity	$\omega = 2\ rad/s$
	Added gaussian noise	10%
	Total time simulation	10 s
SF encoder	Threshold	$s_{th} = 0.0005$
	Initial base value	$s_b = 0$
	Current output	$I_c = 4.6\ nA$
SF decoder	Threshold	$s_{th} = 0.025$
	Initial base value	$s_b = 0$

2.2.2. Synaptic Modeling

Once we define how neurons produce spikes, we will expose how synaptic strength is adjusted. STDP (Bing et al., 2019b) is an unsupervised learning algorithm (based on the Hebbian learning rule) for SNN. It describes how synaptic weights are strengthened or weakened according to neural spike activity, and

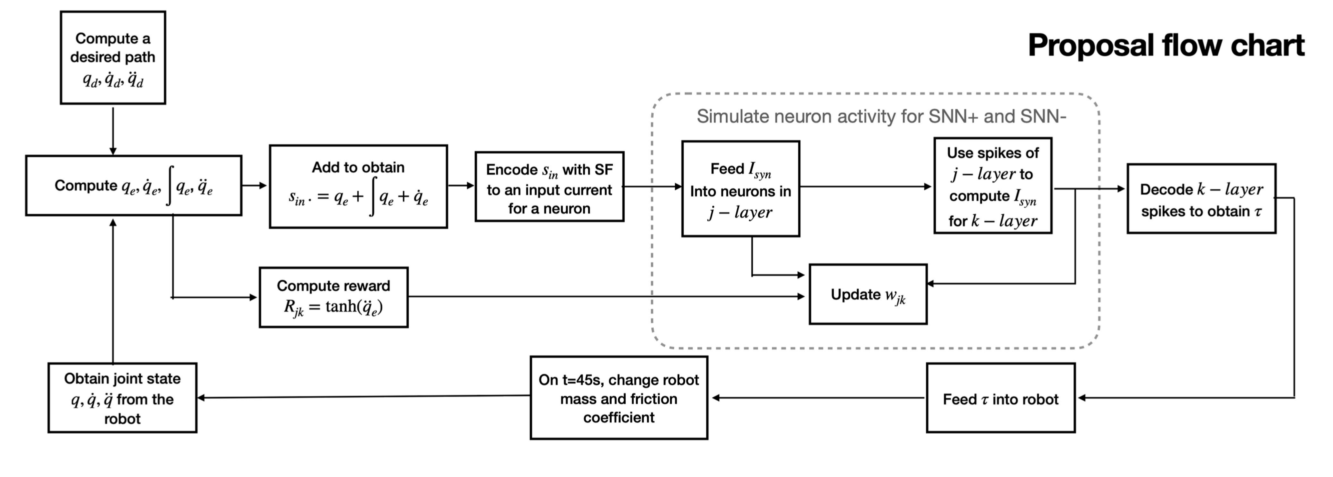

Proposal flow chart
Compute a desired path $q_d, \dot{q}_d, \ddot{q}_d$
Compute $q_e, \dot{q}_e, \int q_e, \ddot{q}_e$
Add to obtain $s_{in}. = q_e + \int q_e + \dot{q}_e$
Encode s_{in} with SF to an input current for a neuron
Simulate neuron activity for SNN+ and SNN-
Feed I_{syn} Into neurons in $j - layer$
Use spikes of $j - layer$ to compute I_{syn} for $k - layer$
Decode $k - layer$ spikes to obtain τ
Compute reward $R_{jk} = \tanh(\ddot{q}_e)$
Update w_{jk}
Obtain joint state $q, \dot{q}, \ddot{q}$ from the robot
On t=45s, change robot mass and friction coefficient
Feed τ into robot

FIGURE 4 | SNN-PID control proposal flow chart showing steps for implementation.

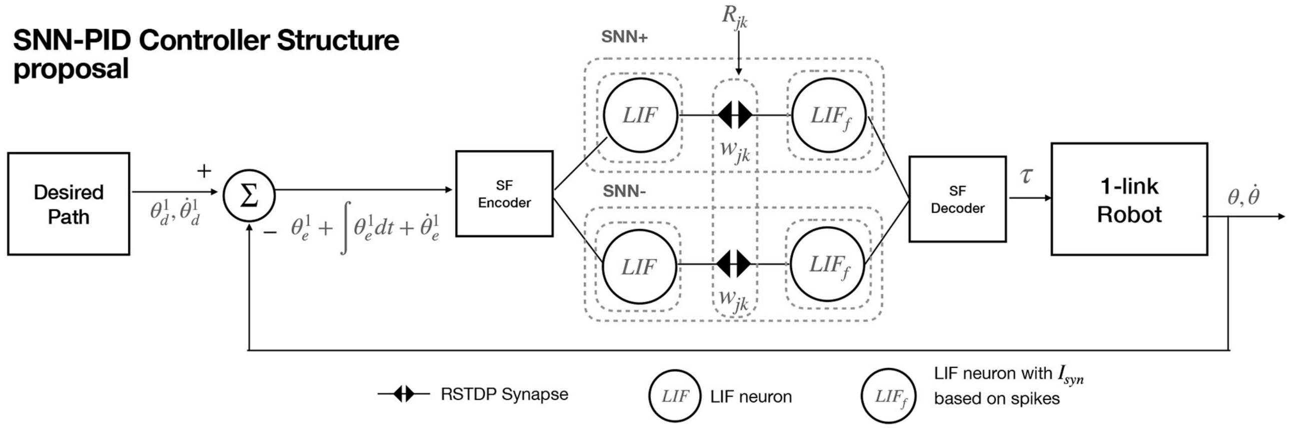

SNN-PID Controller Structure proposal
R_{jk}
SNN+
SNN-
LIF
LIF_f
w_{jk}
Desired Path
+
$\theta_d^1, \dot{\theta}_d^1$
Σ
−
$\theta_e^1 + \int \theta_e^1 dt + \dot{\theta}_e^1$
SF Encoder
SF Decoder
τ
1-link Robot
$\theta, \dot{\theta}$
RSTDP Synapse
LIF neuron
LIF neuron with I_{syn} based on spikes

FIGURE 5 | SNN-PID controller neural structure for 1-DOF robot pendulum.

it has demonstrated plausibility over conducted experiments in biological systems. First, a synaptic weight value is randomly assigned for each defined synapse. Then, the time difference between pre and post-synaptic firing times $\Delta t = t_{post} - t_{pre}$ is computed and it determines the rate of change Δw on the synaptic weight w as:

$$STDP(\Delta t) = \begin{cases} A_+ e^{-\Delta t/\tau_{post}} & \Delta t \geq 0 \\ A_- e^{-\Delta t/\tau_{pre}} & \Delta t < 0 \end{cases} \tag{6}$$

$$\dot{w} = \sum_{t_{pre}} \sum_{t_{post}} STDP(\Delta t) \tag{7}$$

Here, A_+, A_- are scaling constants depicting whether our synaptic weight has been incremented (*Long Term Potentiation* LTP) or decremented (*Long Term Depression* LTD). τ_{pre}, τ_{post} are positive and negative constants representing decay time. Once again, these equations imply all spikes have been produced. Since *neurons do not have a memory of all their fired spikes* (Bing et al., 2019a), Equation (6) can be rewritten in the function of the last firing time (Morrison et al., 2008; Gerstner et al., 2018). This results in the following expressions:

$$STDP(t - t_{pre/post}) = \begin{cases} A_+ \delta(t - t_{pre}) \\ A_- \delta(t - t_{post}) \end{cases} \tag{8}$$

As each time a presynaptic spike t_{pre} is produced, $t_{pre} = t$ and $\delta(t - t_{pre}) = 1$. With each postsynaptic spike, $t_{post} = t$ and $\delta(t - t_{post}) = 1$. Next, we define an eligibility trace E_{jk} for each synapse between a presynaptic neuron j and a post-synaptic neuron k as:

$$\dot{E}_{jk}(t) = -\frac{E_{jk}}{\tau_E} + STDP(t - t_{pre/post}) \tag{9}$$

This expression computes synaptic weight changing history, generated by the collected spikes. To control the sensitivity of the plasticity to delayed reward, an exponential $\tau_E = \tau_{pre} = \tau_{post}$ constant for $E_{jk}(t)$ is defined. This implies a symmetric learning rate for LTD and LTP (Taherkhani et al.,

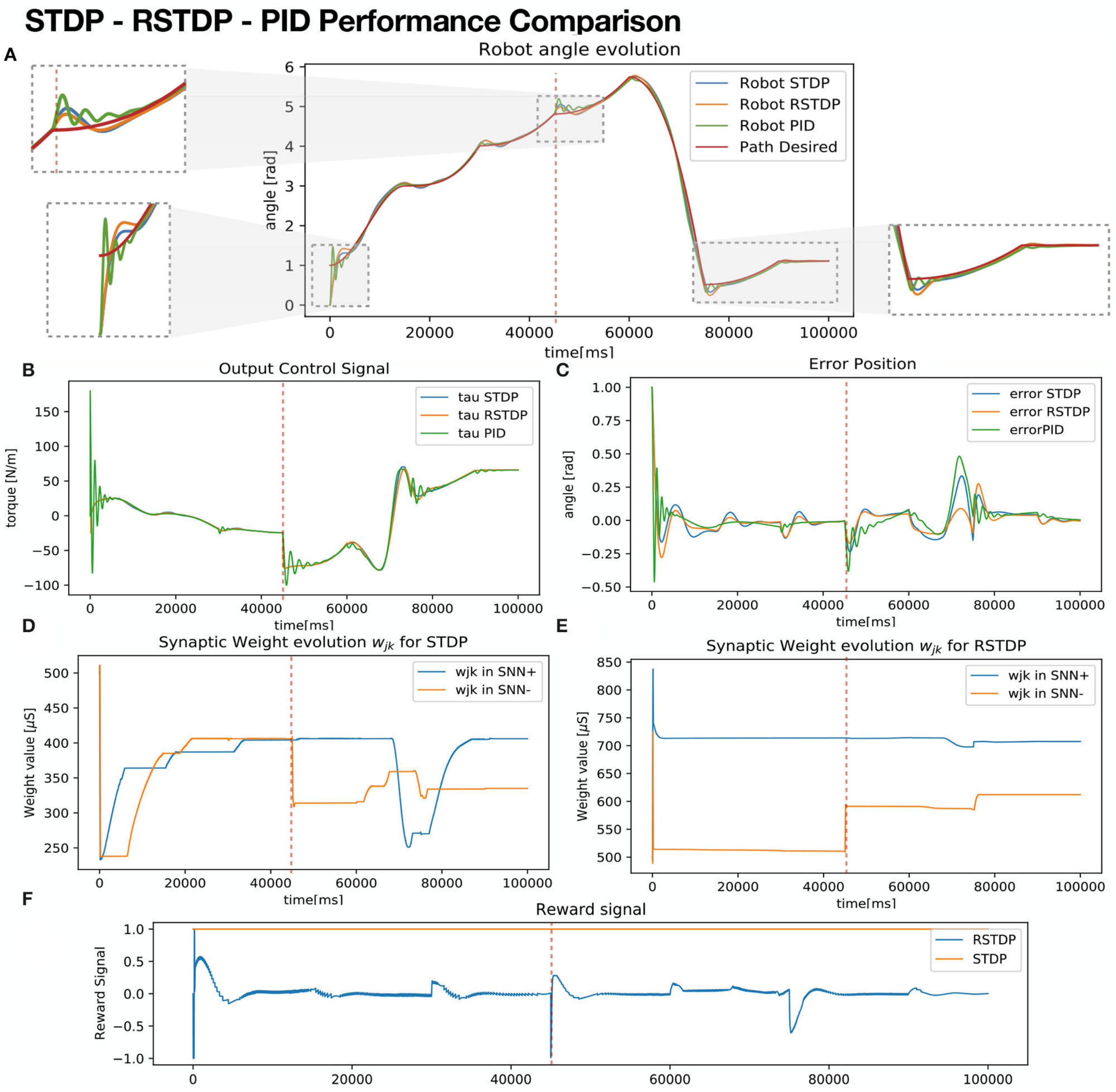


FIGURE 6 | 1-Link robot controller performance comparison. **(A)** Robot Angle evolution for each robot, SNN-RSTDP: with $R_{jk} = \tanh \ddot{q}_e$ (Yellow), SNN-STDP: proposal with $R_{jk} = 1$ (Blue), PID-Controller tuned with Ziegler Nichols (Green), compared against reference (Red). **(B)** Control output of each controller. **(C)** Error position: difference against the reference. **(D)** Synaptic weights evolution for constant reward robot (SNN-STDP). **(E)** Synaptic weight evolution for the rewarded robot (SNN-RSTDP). **(F)** Reward signal for both SNN proposals.

2020). Change in synaptic weights is obtained by integrating (Equation 9). The *reward modulated STDP*, or R-STDP learning rule model, integrates the reinforcement learning paradigm in SNNs, modifying the STDP algorithm based on dopamine effects for learning in biological brains (Framaux and Gerstner, 2016). Consider:

$$\dot{w}_{jk}(t) = R_{jk}(t) \times E_{jk}(t) \tag{10}$$

Here, $R_{jk}(t) \in R^{n\times 1}$ is a reward signal for the synapses between layer a $j - th$ and $k - th$ layer in a network, bounded inside $[-1, 1]$, which enables or disables synaptic modification (called learning), and it is defined by interaction with the environment as a function of an objective (i.e., the desired path, desired position, desired action). It is worth mentioning that, when $R_{jk} = 1$, the R-STDP rule equals STDP, as Equation (7) equals Equation (10). When R_{jk} is equal to 0, learning is inhibited.

Algorithm 1: Modified step forward encoding algorithm.

```
Data: input, threshold, base
Result: $Out_+(i)$, $Out_-(i)$
L = length(input);
for $i$ = 2:L do
    if $input(i) > base + threshold$ then
        base = base + threshold
        $Out_+(i) = 1$ ;
        $Out_-(i) = 0$ ;
    else if $input(i) < base + threshold$ then
        base = base - threshold
        $Out_+(i) = 0$ ;
        $Out_-(i) = 1$ ;
end
```

TABLE 4 | RMSE analysis result comparison for STDP-SNN, RSTDP-SNN, and PID controllers.

Signal	RSTDP	STDP	PID
Position	**0.101231**	0.12089	0.116812
Velocity	**0.116812**	0.21960	0.33481
Acceleration	0.43922	**0.400311**	1.49198

Best cases are in bold.

2.2.3. Encoding and Decoding Between Continuous and Spike Domains

Step forward encoding (SF henceforth), described in Kasabov et al. (2016) and Dupeyroux et al. (2021), is considered a temporal encoding algorithm, as it converts the variation of an input signal to spikes. The module for the step forward encoding contains two outputs ports Out_+, Out_-, and an input port, s_{in}, which is compared with a baseline value s_b. If the incoming signal is bigger than a certain predefined threshold value s_{th} (this is: $s_{in} > s_b + s_{th}$), then a spike will be produced over Out_+. On the contrary, if the signal has decreased ($s_{in} < s_b - s_{th}$), a spike will be produced in Out_-. As the spike's domain is always positive, the emitted spikes can be processed by SNNs representing positive and negative changes in value. The procedure herein is shown in **Algorithm 1**.

A neural structure proposal for exploiting SF encoding with SNN and STDP is shown in **Figure 2A**. An input signal is fed to an encoder and the decoded output signal tends to match the original, as the synaptic weights get updated (**Figure 2B**). For signal growth, learning in the $w+$ synapse occurs. Once the signal decreases, an update for negative synapse $w-$ begins (**Figure 2C**). In order to foster a quick synaptic weight adjustment in both synapses, Gaussian noise was added to the input signal s_{in} (**Figure 2D**), with a SD of $\sigma = 0.01$. As a secondary effect, accumulation in the decoder's output signal takes place, as seen in **Figure 2E**. In order to harness the low-pass filter dynamics of the LIF neuron model, a slight modification is proposed. Instead of spikes, a given current I_c is sent as the encoder outputs. **Figure 3** shows the same signal reconstruction obtained with the proposed modifications, setting $I_c = 4.6\,nA$ as current input to a LIF neuron which, according to its tuning curve (refer to **Figure 1C**), it would produce spikes at a frequency of 200 Hz. Signal reconstruction is achieved. Moreover, Gaussian noise with $\sigma = 0.3$ is added to the input signal, achieving signal reconstruction and filtering, as seen in **Figure 3D**.

2.3. Self Tuning SNN Controller Proposal

The objective is to create an SNN structure that enables the learning of the robot's dynamics and reconstructs the necessary torque control output based on Equation (2). In order to take advantage of synaptic plasticity properties, tuning PID control parameters K on the fly is performed. The procedure steps are shown in **Figure 4** and described in detail upnext. First, a module computes the desired path, using a cubic polynomial trajectory planning generation Algorithm (Craig, 1986; Spong et al., 2005), with initial and final points randomly defined between the joint's boundaries, and initial and final desired velocities set to zero. Next, $q_e, \int q_e dt, \dot{q}_e$ are computed and added; then, they are applied to an SF encoder module, which out is sent to an SNN processing positive changes (called SNN+), and another for negative changes (called SNN-). Both networks share the same structure, as SNN+ and SNN- are intended to process the necessary signal increments and decrements, respectively. These networks are composed of two layers $j-th$ and $k-th$ composed of n LIF neurons each (same amount of DOFs in the robot), modeled by Equations (4), (5). Between $j-th$ and $k-th$ layers, there are $w_{jk} \in R^{n\times n}$ synapse matrices with randomly initialized weight values between minimum and maximum synaptic values $[w_{min}, w_{max}]$. For all synapses, its value will be modified accordingly to Equations (8–10). Each output spike from the neurons of the $k-th$ layer in SNN+ and SNN- serves as input for the n SF decoders, which output corresponds to each torque input signal for the robot. The proposed structure is shown in **Figure 5**.

3. RESULTS

3.1. 1-DOF SNN Simulation Implementation

For a 1-link robot (a pendulum), its non-linear model has a shape like in Equation (1) and is given by:

$$\tau = ml^2\ddot{\theta} + mgl\sin(\theta) + ml^2k\dot{\theta} \tag{11}$$

Where m stands for the arm's weight, l its length, g is the gravity acceleration term, k is the viscous friction in the joint, $\theta \in [\theta_{max}, \theta_{min}]$ is the joint angle, τ represents torque in the robot's joint, acting as the input control signal to the system.

Figure 6 shows simulation results comparing performance between three controllers:

- SNN-STDP: proposed model with fixed reward signal $R_{jk} = 1$.
- SNN-RSTDP: proposed model with reward signal R_{jk} given as a function of acceleration error $\ddot{q}_e$.

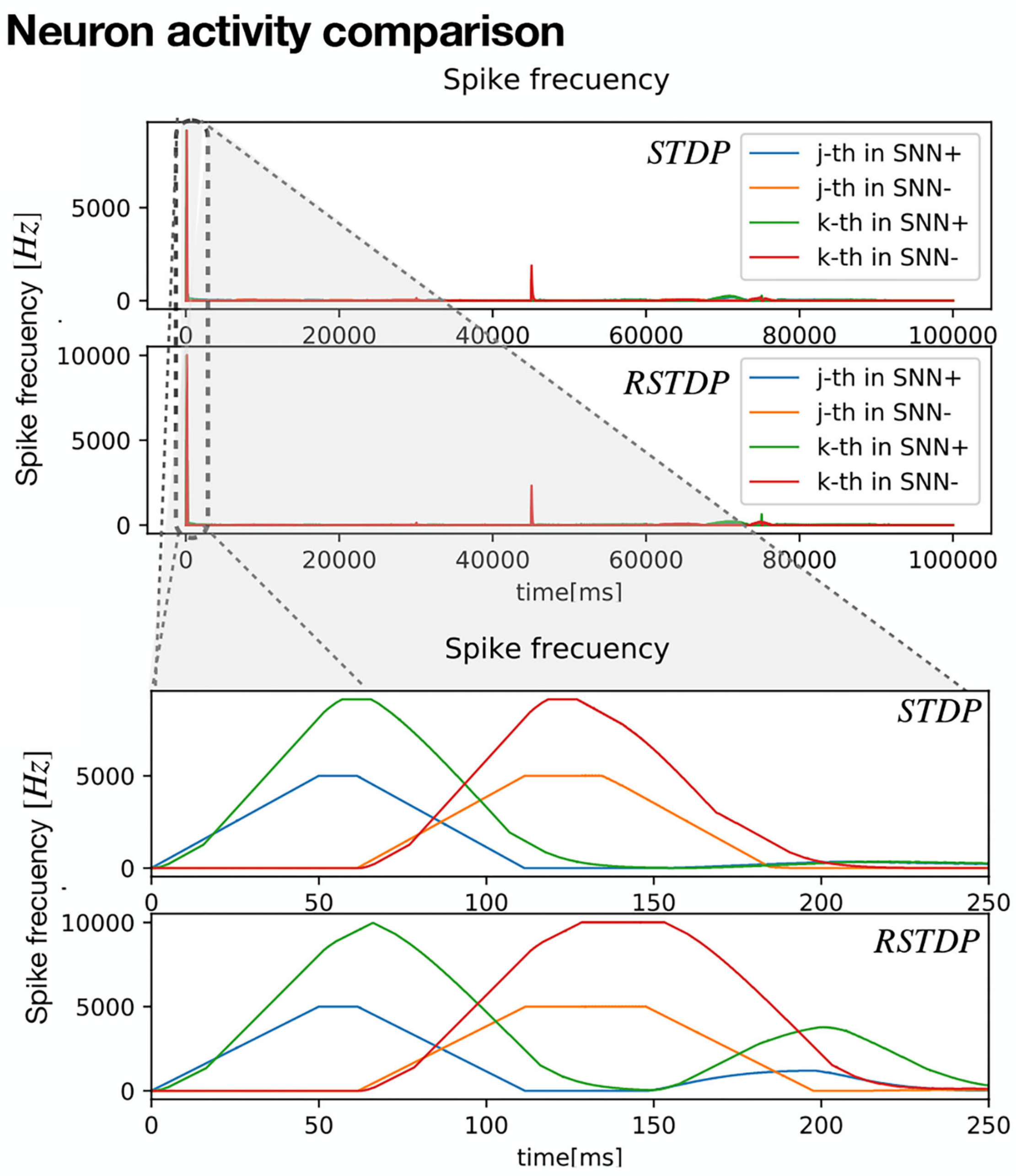


FIGURE 7 | Neuron activity for 1-link controller pendulum. First, $t = 100s$ of the entire control execution is shown, for SNN-STDP (Up) and SNN-RSTDP (down). A zoom at $t = 0ms - 250ms$ is shown at the highest neuron activity.

- A manually tuned PID controller, tuned by the Ziegler Nichols technique (Ogata, 2010)

For the SNN-RSTDP controller, a bounded reward signal delimited by $[-1, 1]$, is given next:

$$R_{jk} = \tanh(\ddot{q}_e) \tag{12}$$

For each episode of length $15s$, the desired path is computed, selecting initial and final positions randomly, but setting the final position of the current episode as the initial position for the next episode. The proposed SNN quickly tunes itself. At $t = 45s$ (refer to red dotted vertical line), the link's joint friction coefficient k and weight m increase to a new k_{new} and m_{new} values. From $t = 90s$ to $t = 100s$, $q_d = constant, \dot{q}_d = 0$.

In **Figures 6A,B**, it can be seen that robot angle runs evolve smoothly on both RSTDP and STDP controlled robots, in contradistinction from PID controlled robots, in which evolution oscillates more. Besides, in **Figure 6C**, it can be seen at the output control for the PID presents jittering, which in real scenarios would produce fatigue on the motor, decreasing its lifespan. In

TABLE 5 | 1-link robot simulation parameters.

Model	Parameters	Value
1-DOF robot parameters	Joint angle boundaries	$[\theta_{max}, \theta_{min}] = [0,2\pi]$
	Mass	$M = 1\ kg$
	New mass at $45s$	$M_{new} = 3\ kg$
	Longitude	$L = 2.5\ m$
	Gravity acceleration	$g = 9.8\ m/s^2$
	Initial viscous friction constant k	$0.1\ kg/s$
	New viscous friction constant at $45s$ k_{new}	$0.5\ kg/s$
PID controller	Proportional gain	$k_p = 180$
	Integral gain	$k_i = 50$
	Derivative gain	$k_d = 12.5$
Simulation parameters	Time step	$dt = 0.1\ ms$
	Number of episodes	6
	Length of an episode	$15s$
	Total time simulation	$100s$
SF encoder	Threshold	$s_{th} = 0.001$
	Initial base value	$s_b = 0$
	Output current	$I_c = 140nA$
SF decoder	Threshold	$s_{th} = 0.2$
	Initial base value	$s_b = 0$

order to analyze the variance of the tracking task, a *root mean squared error (RMSE)* metric signal (Petro et al., 2020), which is intended to be minimized, is defined as:

$$RMSE_q = \sqrt{\frac{\sum_{t=1}^{N}(q - q_d)^2}{N}} \tag{13}$$

Where N is the number of all timesteps along with the experiment. Similar values $RMSE_{\dot{q}}$, $RMSE_{\ddot{q}}$ can be obtained using velocities $\dot{q}$ and accelerations $\ddot{q}$ instead. **Table 4** shows the mean *RMSE* values for a hundred iterations with random desired trajectories, comparing the position, velocity, and acceleration for each of the three used controllers. It can be seen that RSTDP and STDP controllers achieve better performance, having lower RMSE values. $RMSE_{\ddot{q}}$ presents the worst metrics for the PID controller, explaining the jittering for the torque output, which is the function of the acceleration of the robot.

Figure 7 shows the spiking frequency of each neuron for both STDP and RTDP controllers. It can be seen that for the first $200ms$ (Refer to zoomed section), the frequency grows and drops quickly, as decoders send current to each input neuron according to the sensibility s_{th}. All the values used for RSTDP synapses, LIF neuron model, and SF encoding and decoding are depicted in **Table 1**. 1-link Robot simulation parameters are shown in **Table 5**. The simulation has been performed using Python3 scripts.

4. DISCUSSION

The utility of a neuromorphic controller operation for a 1-DOF robot capable of learning the changing dynamics has been experimentally demonstrated with results comparable with a standard control technique. The PID used for comparison is tuned using a pretty standard and popular procedure for industrial applications. It is an iterative process that intends to eliminate response oscillations based on select proper gains throughout multiple testing executions in the plant. The procedure ends when the responsible technician is pleased with the performance, making it as precise as its interpretation, and it has to be re-tuned each time the dynamics of the plant change.

Unlike the PID, our proposal eliminates the need for tuning procedures. Nonetheless, some issues have to be addressed. First, SNN parameters were selected to mimic biological brain systems, which can be modified to fit actual electrical circuit standards. For example, values of w_{min} and w_{max} were chosen arbitrarily, while they should be scaled to fit actual memristor conductance limit values.

A_+, A_-, which control LTD and LTP, play an important role in stability, as they control the learning rate of the system. Small values will result in slow convergence, while larger values will overshoot the output control signal. Value s_{th} for SF encoding will determine its sensitivity against the input signal, setting the amount of neural activity (spikes) as the response. For decoding, s_{th} determines output modification, as it has to be sufficiently large to scale the outgoing signal and sufficiently small to avoid overshoot and under-damping behavior. SF encoding also shows no neural activity for the SNN- stage for always increasing signals. I_c modules spiking frequency, as for higher values, output signal amplitude is affected too.

A stability analysis to determine proper LTD, LTP values, thresholds, and current inputs for encoding/decoding and learning rate values is needed. While it is a pending task, some challenges arise, as some system dynamics are not differentiable (LIF, SF models). Therefore, Lyapunov asymptotic stability analysis cannot be performed. However, some possible alternatives are proposing differentiable models of the neuron dynamics, defining the system on the frequency domain, or conducting Von Neumann stability studies.

On the other hand, noise then allows to update synaptic weights constantly, but the sensibility of the encoding is crucial, as for small s_{th} values, signal variation produces redundant neural activity, generating an accumulative error for decoding. Our proposal effectively used neuron dynamics as a filter, in an open loop. A possible alternative to use or implement alongside would be to use the *Moving-Window SF algorithm* instead. Similar to SF, starting from an initial baseline and threshold values, the baseline is updated differently as an average input signal for a time window. This corresponds to a median filter.

As this scheme proposal tackles fully actuated 1-DOF robotic manipulators, its usage in N-DOF has to be studied. SNN structure might be usable in under-actuated systems, but the SNN architecture must be modified. Hyper-redundant manipulators present a similar problem, as flexible robotic arms can be considered like infinite DOF systems. An infinite neural structure generation is problematic. Therefore, modifications have to be proposed in the future.

5. CONCLUSION

A self-tuning SNN architecture for a 1-DOF manipulator robot arm is proposed, based on a typical control scheme. Numerical simulation shows the feasibility and, in some cases, outperforms PID performance. The architecture also shows self-tuning properties on changing dynamics. From the control theory point of view, a neural structure with similar PID performance is described. Nevertheless, stability analysis is still pending, describing the relationship between spiking activity, current injection, learning rate, and coding velocity. Besides, explainable neural networks are possible, considering control loop architectures. However, neuron models, synapses, and coding/decoding modules should be implemented in analog circuit counterparts to achieve real-time computing scenarios with efficient energy consumption.

DATA AVAILABILITY STATEMENT

The datasets presented in this study can be found in online repositories. The name of the repository and accession number can be found below: Github, https://github.com/AlejandroJuarezLora/Frontiers-SNN.git.

AUTHOR CONTRIBUTIONS

AJ-L proposed, developed, programmed the neural control code and conducted simulation runs, and wrote the first draft of the manuscript. VP-P and HS proposed modifications to the SNN architectures. ER-E reviewed the code required for the PID controller experiments. All authors contributed to the conception and design of the study and manuscript revision, read, and approved the submitted version.

FUNDING

The authors would like to thank the economic support of the projects SIP 20210124, 20221780, 20211657, 20220268, 20212044, 20221089, 20210788, 20220226, and COFAA and CONACYT FORDECYT-PRONACES 6005.

ACKNOWLEDGMENTS

The authors would like to thank the support provided by Instituto Politécnico Nacional, Secretaría de Investigación y Posgrado, Comisión de Operación y Fomento de Actividades Académicas, and CONACYT-México for the support to carry out this research.

REFERENCES

Bing, Z., Baumann, I., Jiang, Z., Huang, K., Cai, C., and Knoll, A. (2019a). Supervised learning in snn via reward-modulated spike-timing-dependent plasticity for a target reaching vehicle. *Front. Neurorobot*. 13, 18. doi: 10.3389/fnbot.2019.00018

Bing, Z., Jiang, Z., Cheng, L., Cai, C., Huang, K., and Knoll, A. (2019b). "End to end learning of a multi-layered snn based on r-stdp for a target tracking snake-like robot," in *2019 International Conference on Robotics and Automation (ICRA)*, 9645–9651.

Chen, X., Zhu, W., Dai, Y., and Ren, Q. (2020). "A bio-inspired spiking neural network for control of a 4-dof robotic arm," in *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)* (Kristiansand: IEEE), 616–621.

Craig, J. J. (1986). *Introduction to Robotics: Mechanics and Control*. Reading, MA: Addison-Wesley Publishing Co. Inc.

Dupeyroux, J. (2021). A toolbox for neuromorphic sensing in robotics. *arXiv [Preprint]*. Available online at: https://arxiv.org/abs/2103.02751

Framaux, N., and Gerstner, W. (2016). Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules. *Front. Neural Circ*. 9, 85. doi: 10.3389/fncir.2015.00085

Gerstner, W., Lehmann, M., Liakoni, V., Corneil, D., and Brea, J. (2018). Eligibility traces and plasticity on behavioral time scales: experimental support of NeoHebbian three-factor learning rules. *Front. Neural Circ*. 12, 53. doi: 10.3389/fncir.2018.00053

Guo, W., Fouda, M. E., Eltawil, A. M., and Salama, K. N. (2021). Neural coding in spiking neural networks: a comparative study for robust neuromorphic systems. *Front. Neurosci*. 15, 638474. doi: 10.3389/fnins.2021.638474

Hao, Y., Huang, X., Dong, M., and Xu, B. (2020). A biologically plausible supervised learning method for spiking neural networks using the symmetric stdp rule. *Neural Netw*. 121, 387–395. doi: 10.1016/j.neunet.2019.09.007

Hu, T., Lin, X., Wang, X., and Du, P. (2022). Supervised learning algorithm based on spike optimization mechanism for multilayer spiking neural networks. *Int. J. Mach. Learn. Cybern*. doi: 10.1007/s13042-021-01500-8

Izhikevich, E. (2004). Which model to use for cortical spiking neurons? *IEEE Trans. Neural Netw*. 15, 1063–1070. doi: 10.1109/TNN.2004.832719

Kasabov, N., Scott, N. M., Tu, E., Marks, S., Sengupta, N., Capecci, E., et al. (2016). Evolving spatio-temporal data machines based on the neucube neuromorphic framework: design methodology and selected applications. *Neural Netw*. 78, 1–14. doi: 10.1016/j.neunet.2015.09.011

Kendall, J. D., and Kumar, S. (2020). The building blocks of a brain-inspired computer. *Appl. Phys. Rev*. 7, 011305. doi: 10.1063/1.5129306

Kheradpisheh, S. R., and Masquelier, T. (2020). Temporal backpropagation for spiking neural networks with one spike per neuron. *Int. J. Neural Syst*. 30, 2050027. doi: 10.1142/S0129065720500276

Kim, H., Mahmoodi, M. R., Nili, H., and Strukov, D. B. (2021). 4k-memristor analog-grade passive crossbar circuit. *Nat. Commun*. 12, 5198. doi: 10.1038/s41467-021-25455-0

Lu, H., Liu, J., Luo, Y., Hua, Y., Qiu, S., and Huang, Y. (2021). An autonomous learning mobile robot using biological reward modulate stdp. *Neurocomputing* 458, 308–318. doi: 10.1016/j.neucom.2021.06.027

Lynch, K. (2017). *Modern Robotics: Mechanics, Planning, and Control*. Cambridge, United Kingdom; New York, NY: Cambridge University Press.

Mohemmed, A., Schliebs, S., Matsuda, S., and And Kasabov, N. (2012). Span: spike pattern association neuron for learning spatio-temporal spike patterns. *Int. J. Neural Syst*. 22, 1250012. doi: 10.1142/S0129065712500128

Morrison, A., Diesmann, M., and Gerstner, W. (2008). Phenomenological models of synaptic plasticity based on spike timing. *Biol. Cybern*. 98, 459–478. doi: 10.1007/s00422-008-0233-1

Ogata, K. (2010). *Modern Control Engineering*. Boston, MA: Prentice-Hall.

Petro, B., Kasabov, N., and Kiss, R. M. (2020). Selection and optimization of temporal spike encoding methods for spiking neural networks. *IEEE Trans. Neural Netw. Learn. Syst*. 31, 358–370. doi: 10.1109/TNNLS.2019.2906158

Saxena, V., Wu, X., Srivastava, I., and Zhu, K. (2018). Towards neuromorphic learning machines using emerging memory devices with brain-like energy efficiency. *J. Low Power Electron. Appl*. 8, 34. doi: 10.3390/jlpea8040034

Shi, C., Lu, J., Wang, Y., Li, P., and Tian, M. (2021). "Exploiting memristors for neuromorphic reinforcement learning," in *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)* (Washington DC: IEEE), 1–4.

Spong, M. W., Hutchinson, S., and Vidyasagar, M. (2005). *Robot Modeling and Control*. Wiley. 146–189.

Taherkhani, A., Belatreche, A., Li, Y., Cosma, G., Maguire, L. P., and McGinnity, T. (2020). A review of learning in biologically plausible spiking neural networks. *Neural Netw*. 122, 253–272. doi: 10.1016/j.neunet.2019.09.036

Valadez-Godínez, S., Sossa, H., and Santiago-Montero, R. (2020). On the accuracy and computational cost of spiking neuron implementation. *Neural Netw*. 122, 196–217. doi: 10.1016/j.neunet.2019.09.026

Voelker, A. R., and Eliasmith, C. (2020). *Programming Neuromorphics Using the Neural Engineering Framework*. Singapore: Springer Singapore.

Yue, K., and Parker, A. C. (2019). "Analog neurons with dopamine-modulated stdp," in *2019 IEEE Biomedical Circuits and Systems Conference (BioCAS)* (Nara: IEEE), 1–4.

Zamarreño-Ramos, C., Camuñas-Mesa, L. A., Pérez-Carrasco, J. A., Masquelier, T., Serrano-Gotarredona, T., and Linares-Barranco, B. (2011). On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex. *Front. Neurosci*. 5, 26. doi: 10.3389/fnins.2011.00026

Zhang, X., Lu, J., Wang, Z., Wang, R., Wei, J., Shi, T., et al. (2021). Hybrid memristor-cmos neurons for *in-situ* learning in fully hardware memristive spiking neural networks. *Sci. Bull*. 66, 1624–1633. doi: 10.1016/j.scib.2021.04.014

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The handling editor JD declared a shared affiliation with the authors at the time of review.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 18 May 2022
doi: 10.3389/fnbot.2022.819550

Influence of Human–Computer Interaction-Based Intelligent Dancing Robot and Psychological Construct on Choreography

Liu Yang [1,2]*

[1] College of Arts, Hunan University of Arts and Sciences, Changde, China, [2] Faculty of Music and Performing Arts, Sultan Idris Education University, Tanjung Malin, Malaysia

To study the influence of Artificial Intelligence (AI) on dancing robots in choreography, this paper introduces the biped-humanoid robot-imagined choreography model alongside the Psychological Space Construction (Psychological Construct) and Human–Computer Interaction (HCI). The proposed model is based on deep learning and imitating human thinking and is capable of imagining new dance elements. Finally, simulation experiments are designed to verify the model's effectiveness. Dance professionals are invited to evaluate the robot-imagined dance posture. The results show that the proposed model can vividly imitate human dancers and imagine and create new dance movements. The average basic feature retention and innovation scores of 30 new dance elements imagined on the L_1 (head) are 7.29 and 7.64, respectively. By comparison, similar scores on 30 new elements in L_2 (upper-body) are 7.73 and 7.40, respectively. Therefore, the proposed intelligent robot-imagined choreography model can help the dancing robot choreograph more finely and improve the choreography efficiency. The research results have significant practical value for dance teaching.

Keywords: dance creation, Human–Computer Interaction, Artificial Intelligence, deep learning, dancing robot

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
Hsin-Te Wu,
National Ilan University, Taiwan
Jia-Lang Xu,
National Chung Hsing University, Taiwan
Chi Zhong,
Kyung Hee University, South Korea
Stanislav Makowski,
The City University of Warsaw, Poland

***Correspondence:**
Liu Yang
248169016@qq.com

Received: *21 November 2021*
Accepted: *11 April 2022*
Published: *18 May 2022*

Citation:
Yang L (2022) Influence of Human–Computer Interaction-Based Intelligent Dancing Robot and Psychological Construct on Choreography.
Front. Neurorobot. 16:819550.
doi: 10.3389/fnbot.2022.819550

INTRODUCTION

Dance, as an art, conveys people's emotional and ideal aesthetics through the relationship between body and spirit (Li, 2020). In the art of choreography, to pursue a better theme, the artistic works must appeal to choreographers' thoughts and emotions through a vague mental state in the psychological space. From the perspective of art, the Psychological Space Construction (Psychological Construct) is mainly affected by personal life experience, inner emotional experience, and cognitive activities. Meanwhile, dance is one of the art categories closest to psychology and linguistics. It takes the body as the material medium and entirely plays the role of body language through imagination and technical skills and connects human emotions. Besides, the body and psychology relationship is one of the most basic but delicate relations. Body language can convey and visualize psychological information through specific actions, the basic body language unit. Every action stirs a response from an individual's cognition. The image mapping in the human mind based on instantaneous thinking is an imaginary experience. As art forms flourish and human cognition deepens, new elements are constantly being injected into the psychological space and thus, enrich the source of chorographical innovation (Lei and Rau, 2021). Biped robot research involves multidisciplinary, including machinery, electronics, automation, control technology, and bionics. It has made high achievements in mechatronics. In particular, the gait analysis of the biped dancing robot can provide technical support for human research and prosthetics development when constructing a stable biped gait model. Dancing robots can also arouse teenagers' curiosity and enthusiasm for learning new technology by showing a stable gait and planning sports in the entertainment industry.

The technical level of biped robot gait research is not mature. Its stable gait and balance ability are still the research focus. Control and planning the robot's gait so that the robot's walking motion is always stable is of great practical importance. Barnes et al. (2021) observed that using robots to treat children with autism spectrum was promising and aroused interest in the research community. Children could interact with robots, such as dancing with robots and improve their concentration abilities. Kobayashi et al. (2022) proposed an exploratory study based on impromptu dance to explore the interaction between human dancers and mobile robots. A basic improvisation action algorithm was developed after many iterations with professional dancers. The robot trained in different dance styles created three unique original performances. Hua et al. (2021) researched the dancing robot, inputted the video content of dancing, estimated the human posture in the video with the help of the deep learning method, and obtained the coordinates and positions of the main points of the human body. Although there are many studies on dancing robots, there is a lack of research on humanoid dancing robots that can choreograph independently. Accordingly, the present work presents a method to provide a new direction for robot choreography.

The main contribution is to study the impact of Artificial Intelligence (AI) on dancing robots in choreography by introducing robot choreography and Psychological Construct. Most of the existing research focuses on robot dance's action design and research direction. Compared with the existing research, the innovation lies in establishing a choreography model that enables a bipedal humanoid robot to choreograph by imitating human thinking actively. Being able to choreograph beautiful dance moves using robots opens up new ideas for future research on dance creation.

RESEARCH METHODS

Robot Choreography and Psychological Construct

Dance experts often use familiar dance postures in choreography. Even similar dance postures exert different artistic tastes due to human dancers' individual charisma. Nevertheless, those uniformly programmed dancing robots' monotonous movements might easily get people disinterested after a few tryouts (Dong et al., 2021). Human–Computer Interaction (HCI) refers to the information exchange process between people and computers using a machine-understandable language to complete specific tasks in a certain interactive way. HCI studies the interactive relationship between system and user. The system can be machines or computerized systems and software. The HCI is realized through the external equipment with input/output functions or the corresponding software. Therefore, designers should give full play to the HCI of dancing robots to refine robot movements. The dancing robot-imagined choreography methods include random generation and mapping rules. In particular, the dancing robot-imagined model incorporates robot–human cooperation, dance posture learning, and robot-imagined choreography (Samosir and Widodo, 2020). Here, robot-imagined choreography refers to independently choreographing high-quality dance postures by the robot or through synchronized human–robot interaction. It is essentially the robot's intelligent behavior. High-quality dance can be defined explicitly through three features: maintaining the basic characteristics of human dance, innovation, and conforming to human aesthetics. The principle of robot-imagined interactive choreography is shown in **Figure 1**.

Figure 1 shows the choreography principle of the dancing robot with enhanced interactive learning. The system is constructed based on interactive Reinforcement Learning (RL). Then, by selecting dance actions with a high return, the robot will choreograph different dance moves considering human preferences. The Sarsa RL algorithm is used to update the dance action database in the choreograph system. Each action is given a unique cumulative return, and the Softmax action selection algorithm is used to select the appropriate dance actions to form dance works. At the same time, robots are encouraged to choose dance movements with higher returns over lower returns. Apparently, the Psychological Construct in choreography is a natural and intangible process of imaginative dance ideas. It is the psychological factory for processing dance works (Dou et al., 2021).

Personal experience, skills, cognition, and educational levels all contribute to artistic creation. However, without Psychological Construct, these elements are independent of each other and cannot lead to artic representations. Thus, material and spiritual (psychological) constructs are both essential for art. Like other art forms, choreography generates a series of psychological behaviors with complex interactions and relations, where the subject's imagination exerts the most active psychological function (Borovica, 2020). That is to say, a choreographer without artistic imagination can never choreograph aesthetic dance postures, nor can a positive and independent artistic dance image be formed.

From the perspective of art, the Psychological Construct is mainly affected by choreographers' experiences in real life. In the art of dance, to pursue a better theme, the artistic works must appeal to choreographers' thoughts and emotions through a vague mental state in the psychological space. For example, in the dance *A Couple Guarding A Sentry Box*, the second part involves a strong Psychological Construct by integrating real-life body language into dance moves. Thus, the dance has a solid real-life basis, reflecting the couple's life around the front line of fighting floods and disasters. Despite all the difficulties, the soldier's wife is brave and determined to support her husband: a People's Liberation Army (PLA) member, and share their joys and pains. Overall, the dance can well reflect ordinary people's lives and tendering emotions, thus infecting the audience. The two actors

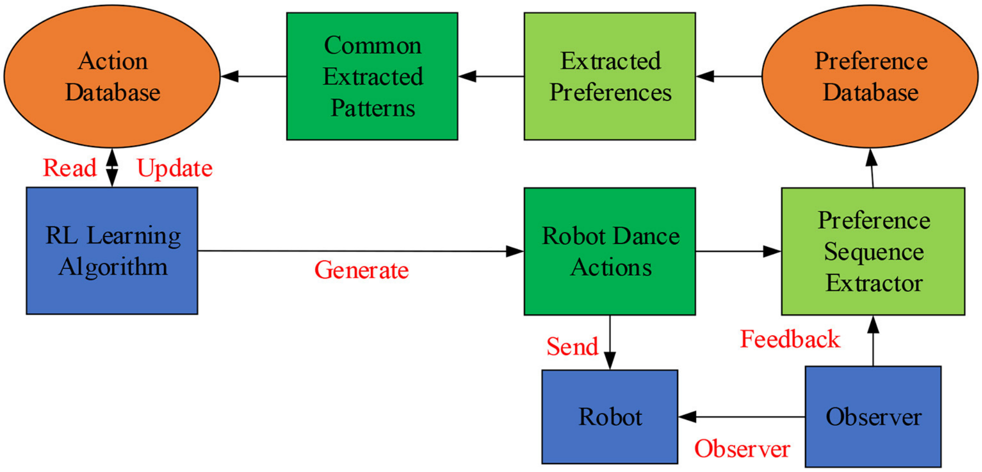


FIGURE 1 | Principle of dancing robot-imagined choreography with enhanced interactive learning.

accurately convey the characters' psychology through aesthetic dance moves, such as modesty, heartache, helplessness, comfort, coquetry, marital love, and other real-life scenes. Meanwhile, the director has consolidated the characters' emotions into each move and embeds their psychological rhythm into the dance connection. The process of the Psychological Construct is demonstrated in **Figure 2**.

Figure 2 is the mapping diagram of Psychological Construct during the performance of the dance *A Couple Guarding A Sentry Box*. Obviously, the dancers have successfully expressed the characters' emotions and psychology through dance moves. Such is the process of Psychological Construct.

Principle of Back Propagation Neural Network (BPNN)

Based on the achievements of DL in human keypoint recognition, the Three-dimensional (3D) position coordinates of human key points are obtained in the image by analyzing RGB images. BPNN is a multilayer Artificial Neural Network (ANN), a new research direction in the DL field. It is introduced into a broader concept of Machine Learning (ML) to help realize true AI. In particular, AI aims to understand the essence of intelligence and fabricate an intelligent machine with human intelligence. It involves multiple disciplines, such as robotics, language recognition, Image Recognition (IR), Natural Language Processing (NLP), and Expert Systems (ES). BPNN features error back-propagation and signals forward transmission. **Figure 3** sketches the structure of a three-layer ANN.

As detailed in **Figure 3**, generally, the ANN comprises three neuron layers: input layer, hidden layer, and output layer. The signal is fed into the input layer during the forward transmission, processed by the hidden layer, and outputted by the output layer. The lower-layer neurons' state is only affected by the state of neurons in the upper layer. Nevertheless, the error between the actual output and the expected result (if large enough) will be back-propagated to adjust the network threshold and weight. Such error back-propagation process will repeat itself until the predicted output of BPNN approach the expected output.

Biped Humanoid Robot-Imagined Choreography Model

In addition to adopting the idea of choreography, choreographers also follow the following creative rules. (1) Learn and master the core elements of some dance forms, such as the shape and position of hand and feet shape, position, and speed. New elements lay the foundation of spontaneous imagination. (2) All these elements are used as the basic dance, and the dance movement is innovated, conforming to human body aesthetics and design. (3) These dance movements are associated together with different combinations, fabricating an intact dance work. Based on the above BPNN research and the objective rules of human choreography, the biped robot can imitate human thinking when choreographing the independent dance. The specific flow is as follows. Learning → Memorizing → Imagining → Processing → Combining. In order to efficiently complete choreography independent of human thought on the biped robot platform, the model is implemented, as given in **Figure 4**.

The joint angle is given as the action command of the robot. The robot's inverse kinematics calculation and forward kinematics calculation involves the mutual conversion between the joint coordinates and the joint angle. Moreover, the angle must be adjusted in multiple steps. This complex angle adjustment is unnecessary when the robot cannot imitate some delicate motions due to the freedom and mechanical structure constraints.

Figure 4 displays the choreographic process by a humanoid robot. "Dance space representation" (Martinez Damia et al., 2021) represents all the concepts adopted, including related concepts of joints and limbs on top of the traditional

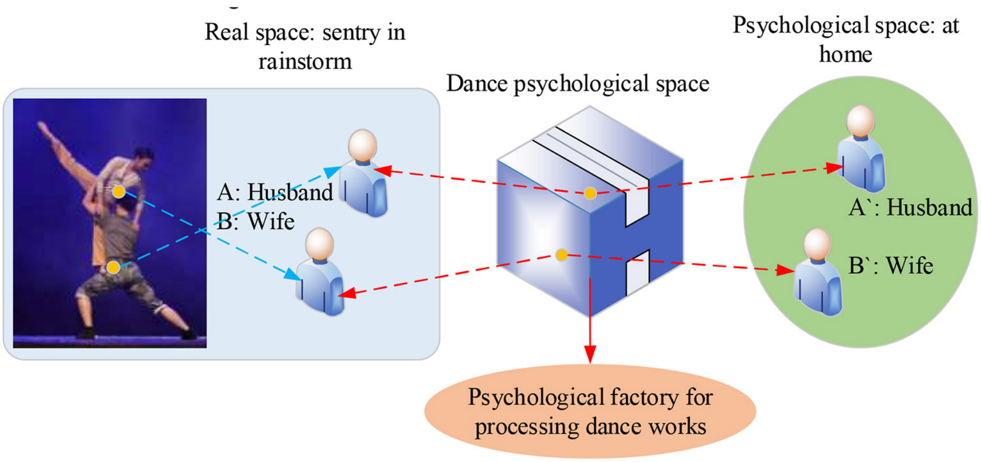

Real space: sentry in rainstorm
A: Husband
B: Wife
Dance psychological space
Psychological space: at home
A`: Husband
B`: Wife
Psychological factory for processing dance works

FIGURE 2 | Schematic diagram of psychological construction of dance *A Couple Guarding A Sentry Box*.

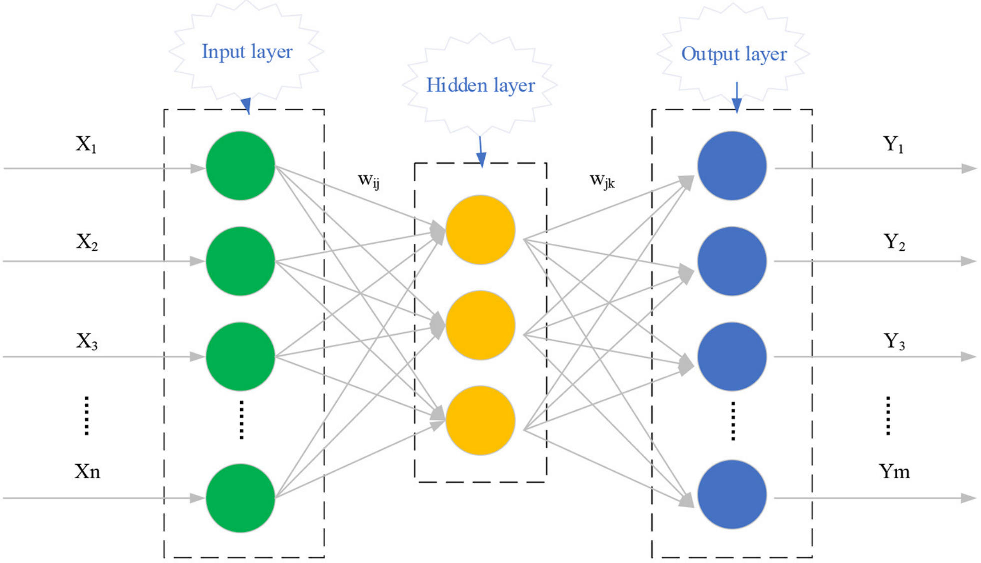


FIGURE 3 | Three-layer ANN structure.

dance-related concepts, such as dance elements and posture. Moreover, the "memory dance space" stores (memorizes) all objects. These objects incorporate the robot's learned dance elements and imagined new dance elements and postures, which can be exported as executive files or relational databases. From the operation process of the model, the robot first watches the basic dance elements displayed by dance experts and then learns these basic dance elements through imagination. Based on this, it expands the new dance elements through its independent imagination, which will be used as the basis for the "imagination" of the next dance. At the same time, machine learning is introduced to enable the robot to appreciate aesthetic dance moves as humans so that it realizes the initial imaginary dynamic dance posture. The robot randomly connects the dance elements of various body parts with the dance posture in line with human aesthetics. Finally, the dance posture corresponding to human aesthetics is associated with different combinations. Consequently, a robot's independently choreographed dance is created and mapped to the physical platform of a biped robot. Further, three methods are adopted to smooth the choreography process of humanoid robots: semi-interactive evolutionary calculation, release-constrained algorithm, and balance detection and balance compensation (Liu, 2020).

The dance representation space of the humanoid robot can be represented by HRDES = {K, B, L, D, DA, DN, DC}. The meaning and expression of each part are listed in **Table 1**.

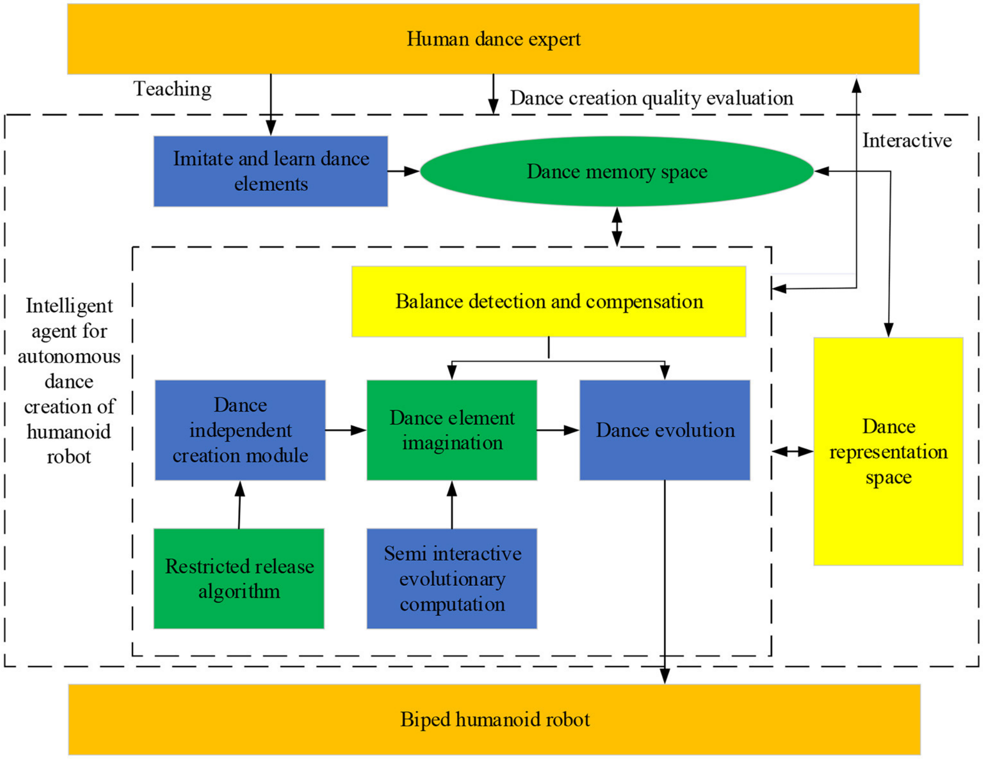

Human dance expert
Teaching
Dance creation quality evaluation
Interactive
Imitate and learn dance elements
Dance memory space
Intelligent agent for autonomous dance creation of humanoid robot
Balance detection and compensation
Dance independent creation module
Dance element imagination
Dance evolution
Dance representation space
Restricted release algorithm
Semi interactive evolutionary computation
Biped humanoid robot

FIGURE 4 | Framework of humanoid biped robot-imagined choreography model.

TABLE 1 | Humanoid robot dance representation space: HRDES.

Category	**K**	**B**	**L**	**D**
Meaning	Joint set	Domain set	Limb set	Constraint set
Representation	$K = \{K_1, K_2, \cdots, K_{\lvert K \rvert}\}$	$B = \{B_1, B_2, \cdots, B_{\lvert K \rvert}\}$	$L = \{L_1, L_2, \cdots, L_{\lvert L \rvert}\}$	$D = \{D_1, D_2, \cdots, D_{\lvert L \rvert}\}$
Other	K_i represents the *i*th joint of the humanoid robot.	B_i represents the domain corresponding to the *i*th joint.	L_i represents the *i*th limb of the humanoid robot.	D_i represents the constraint on the *i*th limb, where $D_i \in \{0, 1\}$.
Category	DA	DN	DC	
Meaning	Dance Collection	Dance posture	Dance routine	
Representation	$DA_i = \{V_{i,1}, V_{i,2}, \cdots, C_{i,\lvert L_i \rvert}\}$	$DN = \{DN_1, DN_2, \cdots, DN_{\lvert L \rvert}, B\}$	$DC = \{DC_1, DC_2, \cdots, DC_Y\}$	
Other	DA_i means the limb L_i's dance elements, and $V_{i,t}$ is the value of limb $V_{i,t}$ at the *i*th joint.	Dance posture includes all dance elements of humanoid robot members and their aesthetic evaluation value B	A robot dance routine consists of a dance sequence with a length of Y.	

Learning Basic Dance Elements and Imaging New Elements

Before independent choreography, the robot should first determine what dance postures to choreograph (Kashyap et al., 2020a). By defining dance postures, the robot can retain basic dance features of human dance moves (García and Diogo, 2020). HRDES indicates that dance elements are attached to limbs or expressed through specific dance postures, such as limb movement. Therefore, human limb movements must be manually classified to construct the limb movement set (*L*) before independent robot choreography. Then, the robot will learn all the dance elements demonstrated by human expert dancers alongside various dance postures. Different dance elements and postures combinations will be generated and stored in the robot dance memory space.

Next, the simulation learning will be implemented based on Posture Similarity Index (PSI), referring to literature (Saqlain et al., 2020). The robot's human dance elements learning flow is illuminated in **Figure 5**.

According to the learned dance elements, the robot's newly imagined elements align with the general motion perception system, so the robot-imagined choreography based on the release-constrained algorithm is reasonable (Kashyap et al., 2020b). The release-constrained algorithm first clarifies the direction of constraint; each robot member without constraints is imposed with constraints to retain the human dance features. The constraints on robots maintain the dance elements' original features and do not allow "imagination" expansion. **Figure 6** depicts the schematic diagram of the humanoid robot developing new dance elements "Imagination using the release-constrained algorithm.

As described in **Figure 6**, the corresponding robots can spontaneously "Imagine" new dance elements once the constraints are released. When all the constraints are released successively, all robot limbs without constraints are free to "image," thus innovating new dance elements while maintaining original dance features.

Every imagined dance element on limb L_i must be verified. The dance element DA_i in dance representation space is defined as a vector, and an n-dimensional space R is composed of n components. Thus, the dance element DA_i is the next point of space R. Then, to determine DA_i, a neighborhood $\mathrm{d_i}$ is first decided on the limb L_i, a parameter to balance the number of imagined new dance elements and the original dance elements. It is a positive integer ϵ [1, $\mathrm{MAX_i}$]. Suppose Dmin[1owerbound, upperbound] is the smallest field in the corresponding fieldset of all joints of limb L_i. In that case, $\mathrm{MAX_i}$ can be calculated by Equation (1):

$$MAX_i = \left[\frac{D_{\min} \cdot upperbpund - D_{\min} \cdot lowerbpund + 1}{2}\right] \tag{1}$$

The larger the neighborhood is, the fewer dance elements are in R, and the greater the difference between dance elements is, and vice versa. Therefore, an appropriate neighborhood d_i should be determined. All the neighborhood points (d_i) are similar, yet each is a unique dance element point. In R, two spatially distanced points and no less than $2d_i$ are deemed distinguishable dance elements. In Equations (2) and (3), $DA_{i,u}$ and $DA_{i,v}$ represent two dance elements on the same limb L_i, their PSI $[S\left(DA_{i,u}, DA_{i,v}\right)]$ is calculated by Equation (2), and Equation (3) judge whether the two elements are identical.

$$S\left(DA_{i,u}, DA_{i,v}\right) = EuclideanDistance(DA_{i,u}, DA_{i,v}) \tag{2}$$

$$Same(DA_{i,u}, DA_{i,v}) = \begin{cases} True\ if\ S\left(DA_{i,u}, DA_{i,v}\right) < 2d_i \\ False\ if\ S\left(DA_{i,u}, DA_{i,v}\right) \geq 2d_i \end{cases} \tag{3}$$

Next, a new dance element imagination algorithm is given based on the change of single joint value on limb L_i. As shown in **Algorithm 1**:

Algorithm 1 | Algorithm flow of the new dance element imagination based on the change of single joint value.

```
For (Select each limb $L_i$ in turn from the limb set L)
                    $Li = \{J_{i,1}, J_{i,2}, \cdots, J_{i,|L_i|}\}$
If $\mathrm{D_i} = 0$ Then
For (In limb $L_i$, select each learned basic dance element $DA_i$ in order)
                    $DA_i = (V_{i,1}, V_{i,2}, \ldots, V_{i,|L_i|})$
{
    If not repeat($DA_i$) Then
    The learned basic dance element $DA_i$ is added to $W$;
    For (Select each joint $K_{i,k}$ in order on the limb $L_i$)
                    $k \in [1, |L_i|]$
    {
        For (Select each value $V'$ on the domain $D_k$ [lowerbound,
upperbound] corresponding to joint $K_{i,k}$, with a step size of 2d)
        {
        Imagine and choreograph a new dance element
$DAi'(V_{i,1}, V_{i,2}, \cdots, V_{i,k-1}, V', V_{i,k+1}, \ldots, V_{i,|L_i|})$ on the
limb $\mathrm{L_i}$;
            If not repeat($DA_i$') Then
The imaginary new dance element $DA_i$' is added to $W$;
        }
    }
    If full Then
    Ends the imagination on the limb $L_i$ and jumps to the outer layer
of the cycle to select the next limb;
}
```

Model Experiment and Analysis

Subsequently, this section takes Tibetan tap dance as an experimental subject for independent humanoid robot-imagined choreography. The Tibetan tap dance features regular knee swings and footsteps (Hannah and Red, 2020). In particular, the experimental robot selects the most popular bipedal robot in AI research: the NAO robot (Robaczewski et al., 2021), which also sees wide applications in teaching, smart home, entertainment, and competition. Because of the superior processing mechanism for joint and body balance, NAO can present various body postures and is an ideal platform for studying biped robot dance. For model implementation convenience, NAO is used in the simulation environment. At the same time, Webots (Sun et al., 2022) are employed, a powerful environment development software for mobile robot modeling, programming, and simulation.

The experimental simulation environment is based on a personal computer. The software environment adopts Windows 7.0 Operating System (OS) and Webots 7.0 4.1 environments. The "memory dance space" is the Windows file system, where all instantiated objects are stored. The visual C++ platform is chosen for coding (Noor and Saad, 2021). In Webots, the hardware is configured with the robot NAO and the computer, which are used to visualize various objects (dance elements, dance posture, and overall dances).

The files in Windows OS, Webots environment, and Visual C++ are closely related, as manifested in **Figure 7**. There is one-way data transmission between Webots and Visual C++.

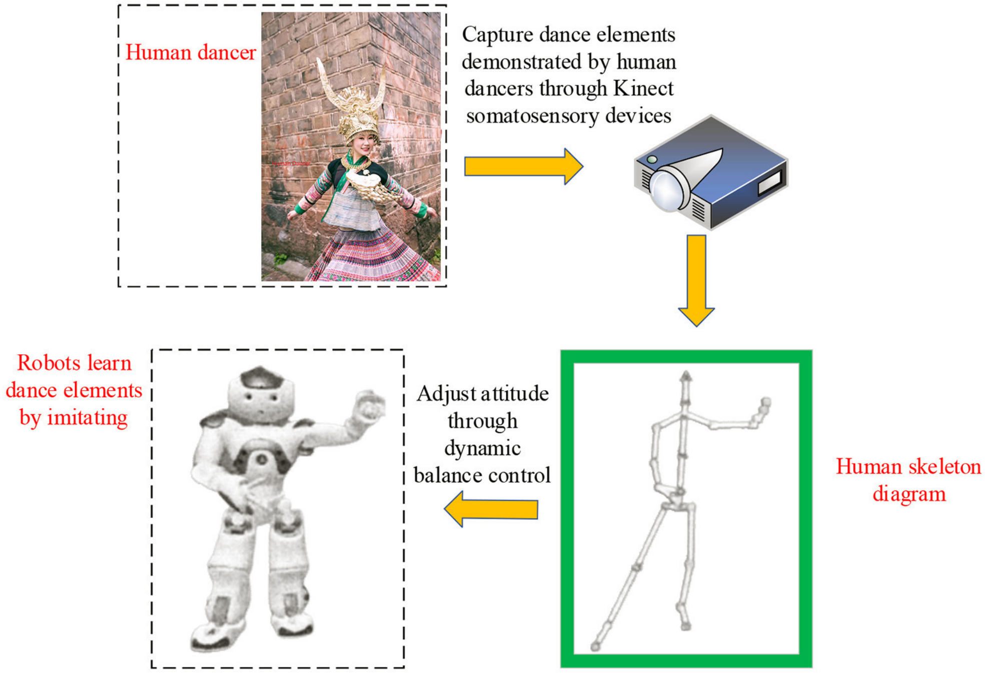

Human dancer
Capture dance elements demonstrated by human dancers through Kinect somatosensory devices
Robots learn dance elements by imitating
Adjust attitude through dynamic balance control
Human skeleton diagram

FIGURE 5 | Basic dance elements imitation based on PSI.

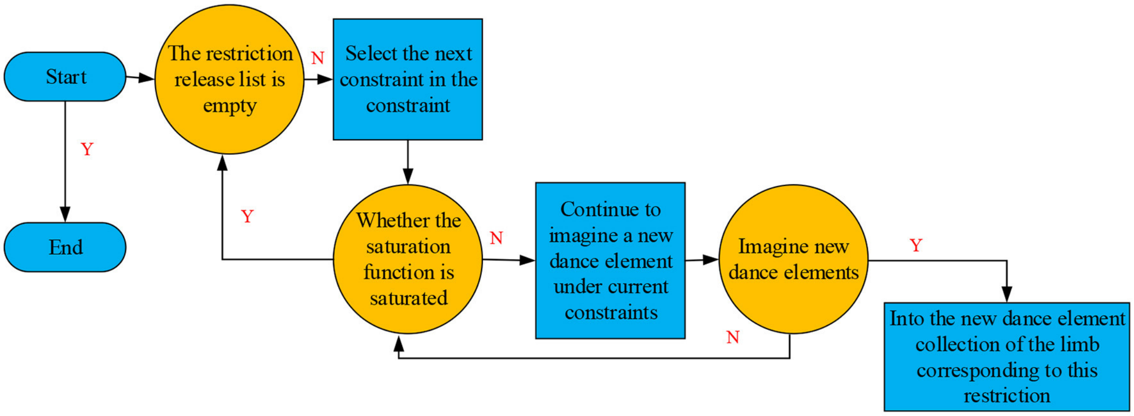

Start
The restriction release list is empty
N
Select the next constraint in the constraint
Y
End
Y
Whether the saturation function is saturated
N
Continue to imagine a new dance element under current constraints
Imagine new dance elements
Y
N
Into the new dance element collection of the limb corresponding to this restriction

FIGURE 6 | Algorithm flow of imagining new dance elements using the release-constrained algorithm.

Visual C++ generates the shared action control file to simulate NAO. Webots read the shared action files and display dance objects (dance posture, dance elements, and the overall dance) on the simulated NAO. The Windows Messages can synchronize the visual application Webots and C++. Because the dance memory space is implemented on Windows, Visual C++ must load and compile the whole implementation process of the dance object model. Thus, Visual C++ will repeatedly access the files in the dance memory space.

The proposed intelligent robot-imagined independent choreography model needs to learn the basic dance elements from dance professionals. This study chooses the Tibetan tap dance for the humanoid robot to imitate and learn. Human dancers participate in dance posture evaluation through semi-interactive evolutionary computing. Then, 10 ethnic dance professionals, who never used biped robots, are invited from Dance University to evaluate and analyze the robot-imagined dance postures. Nevertheless, they have rich experience in stage performance, choreography, and teaching. Everyone has a high aesthetic evaluation ability and aesthetic perception. To improve the dance posture evaluation effectiveness, these dance professionals are informed in advance that the dance style of the model robot is Chinese Tibetan tap dance.

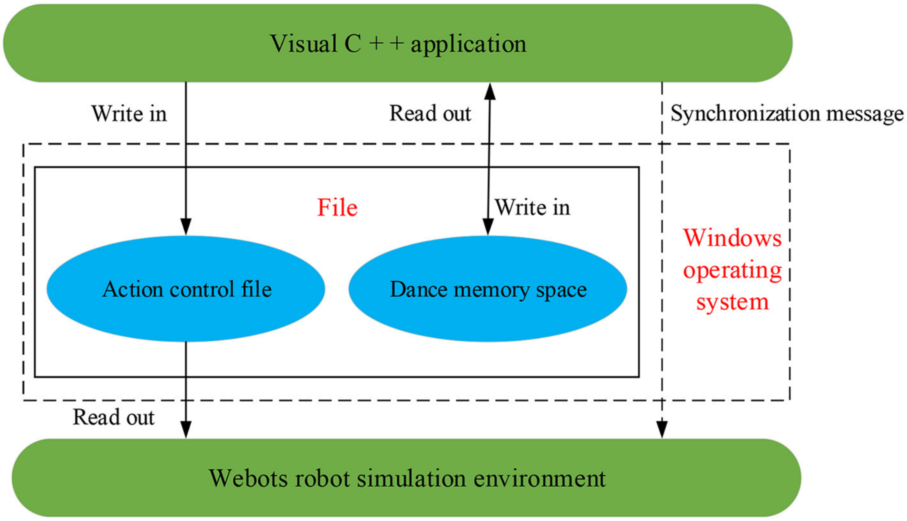


FIGURE 7 | Relationship among Windows, Webots, and Visual C++.

TABLE 2 | Comparison between the proposed model and IEC.

Models	Evaluation method	Final evolutionary outcome	Human load in HCI	Is there a leg dance?	Can the method retain the original features of dance?
Literature (Li et al., 2020) choreography model	Human evaluation	Optimal solutions (less)	Heavier	No	Cannot
The proposed model	Human + machine evaluation	Better solutions (more)	Lighter	Yes	Can

First, the dance representation space of the humanoid robot must be instantiated (Slot et al., 2020). According to the NAO technical manual, NAO is designed with 26 joints with corresponding motion ranges to determine groups K and B in the dance representation space. Because of the characteristics of tap dancing and the physical structure limitations of the NAO robot, the whole robot body is divided into three parts, namely, $L = \{L_1, L_2, L_3\}$, with the constraints $D = \{D_1, D_2, D_3\}$, as unfolded in **Table 2**. L_3 (lower-body) plays an essential role in body balance and reflects the features of Tibetan tap dance. Thus, constraints are imposed on limb L_3 to maintain the dance features, and $D_3 = 1$. In other words, it is not allowed to imagine dance postures on L_3 limbs. In Tibetan tap dance, human dancers often keep their hands relaxed. Accordingly, the NAO robot's left arm and right arm are modeled as the upper body L_2 (Upper-body), not independently. Compared with L_3, L_2, and L_l (head) are more flexible, so $D_1 = 0$ and $D_2 = 0$. Hence, L_2 and L_l can be imagined with new dance elements based on the learned dance elements on L_3. The joint set corresponding to limb L_1 is {HeadYaw, HeadPitch}. The joint set corresponding to limb L_2 is {LShoulderPitch, LShoulderRoll, LEbowYaw, LElbowRoll, LWristYaw, LHand, RShoulderPitch, RShoulderRoll, RElbowYaw, RElbowRoll, RWristYaw, RHand}. The joint set corresponding to limb L_3 is {LHipYawPitch, LHipRoll, LHipPitch, LKneePitch, LAnkiePitch, LAnkleRoll, RHipYawPitch, RHipRoll, RHipPitch, RKneePitch, RAnklePitch, RAnkleRoll}.

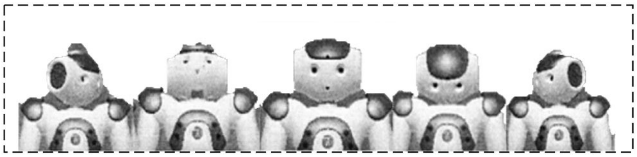


FIGURE 8 | Basic dance elements imitated on upper body L_2 (upper-body).

Instantiation of Dancing Robot HCI Interface

Based on the dance representation space of the instantiated humanoid robot, the PSI-based simulation learning method is adopted to learn the basic dance elements of human dancers. Thus, the robot's basic dance elements on three limbs (L_1, L_2, L_3) are collected and stored in the dance space. Specifically, seven basic dance elements are learned on the limb L_2, as given in **Figure 8**. Five basic dance elements are learned on the limb L_1, as shown in **Figure 9**. Another five basic dance elements have been learned on limb L_3, as depicted in **Figure 10**. Additionally, to enable the robot to imagine more new dance elements on limbs L_1 and L_2, the parameters are set as $d_1 = 1$ and $d_2 = 1$.

Then, multiple "excellent" dance postures are randomly picked out and are combined with different combinations for the robot to image new dance elements. Such is the whole process of independent robot-imagined choreography. **Figure 11** charts one dance posture imagined by the proposed robot-imagined choreography model.

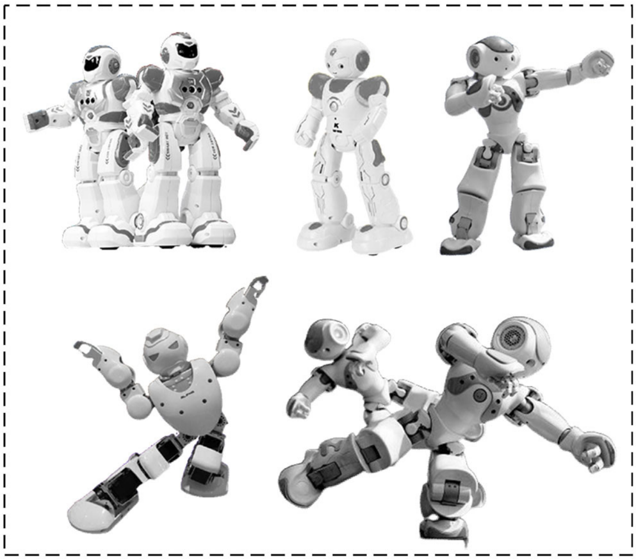


FIGURE 9 | Basic dance elements imitated on limb L_1 (head).

RESULTS AND DISCUSSION

Analysis and Evaluation of New Elements of Imaginative Dance

The experimental part mainly studies a kind of Tibetan tap dance. The proposed robot-imagined choreography model uses the release-constrained algorithm on L_1 and L_2. Altogether, 986 and 23,421 new dance elements are imagined on L_1 and L_2.

Then, 30 new dance elements on L_2 and 30 on L_1 are randomly selected for model evaluation. Ten expert dancers are invited to evaluate the innovation and basic features of the robot-imagined dance postures. Each expert is told to evaluate every dance element and score an integer between (Li, 2020; Martinez Damia et al., 2021) (1—the worst, and 10—the best). When all experts finish scoring, the lowest and highest scores are removed to avoid the negative impact of extreme data. The remaining scores are averaged as the final score of the dance element. **Figures 12**, **13** illustrate the evaluation results of robot-imagined new dance elements on L_2 and L_1. The y-axis represents the evaluated score of dance elements, and the x-axis is the number of dance elements. The two lines stand for the evaluation sores of dance innovation and retention of the basic dance features, respectively.

After calculation, the average basic feature retention score of 30 new elements on the L_2 is 7.73, and the average innovation score is 7.40. Those scores on L_1 are 7.29 and 7.64, respectively. In summary, the proposed robot-imagined choreography model

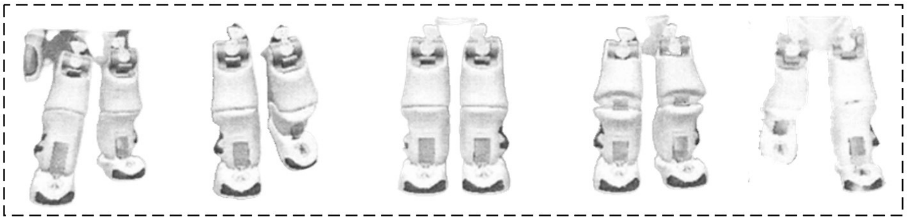


FIGURE 10 | Basic dance elements imitated on lower body L_3 (lower body).

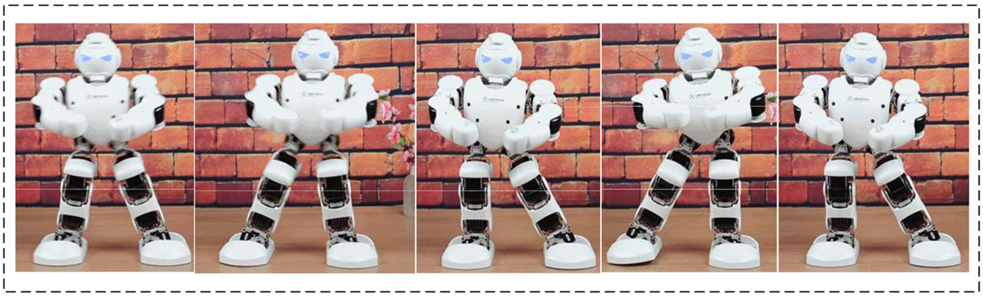


FIGURE 11 | An "Excellent" dance posture imagined by the humanoid robot.

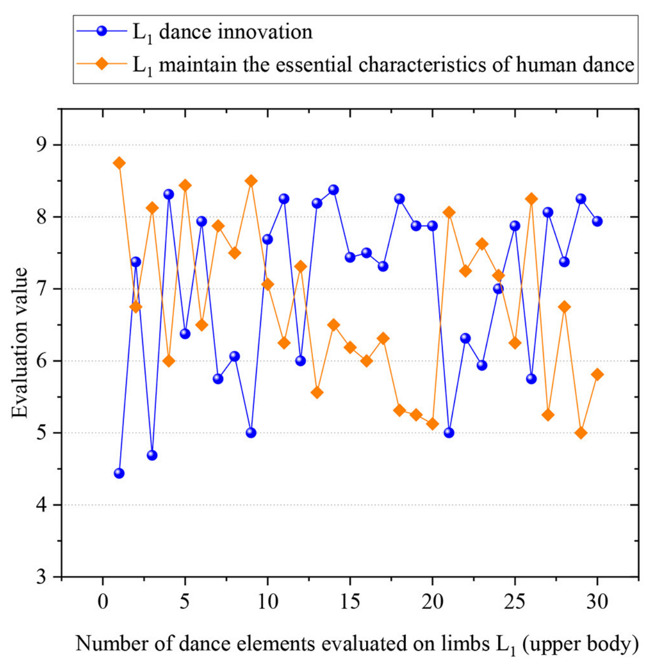


FIGURE 12 | Evaluation scores of robot-imagined 30 new dance elements on L_1.

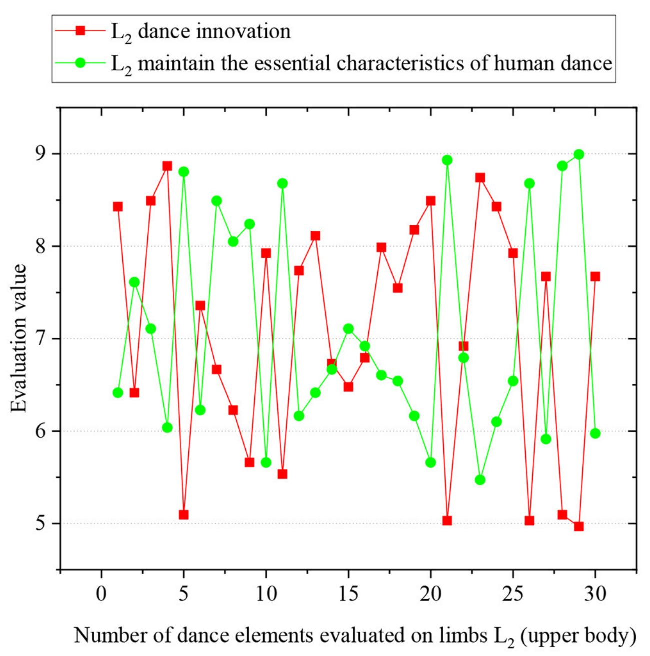


FIGURE 13 | Evaluation scores of robot-imagined 30 new dance elements on L_2.

can produce innovative dance elements while maintaining the basic dance features. Also, it comprehensively considers the contradiction between the basic dance features and innovation.

Comparison of Two Choreography Methods

Table 2 compares the Interactive Evolutionary Computing (IEC) method in literature (Li et al., 2020) with the proposed bipedal-robot-imagined choreography model.

In the proposed choreography model, the body of the robot NAO is divided into several members according to the features of the selected dance. Then, dance elements are extracted from different members. At the same time, any dance element, whether imitated or imagined, always retains the essential features of human dance. Therefore, the combined dance posture still retains the essential features of human dance. By comparison, the robot dance method based on IEC does not consider how to maintain the basic features of human dance.

Influence of the Dancing Robot on Choreography

As far as dance itself is concerned, dancing robots can broaden the expression of the art of dance. Interestingly, dance uses body language to express emotion and intention, with both artistic and decorative utilities (Lin et al., 2022). Currently, robots might not dance as flexibly and aesthetically as human dancers. Nevertheless, thanks to robots' increasingly intelligent humanoid features, robot-expressed dance will soon cause emotional resonance with the audience and influence the audiences' emotional expression. In the process of robot-expressed dance, factors affecting the audience's judgment also have particular research significance (Rifajar and Abdul, 2021). The sense of rhythm, the coherence of movement, and the symmetry of movement will affect the effect of dance, which requires reflection and research on the dance art itself. In addition to dancers' personalities, robot-expressed dance can also provide a research platform for analyzing and researching dance as an art (Li et al., 2020). Further, robot-expressed dance also improves people's understanding of dance itself. Robot-imagined choreography can stimulate authentic human choreography and enhance efficiency for professional dancers. Last, from a commercial point of view, the entertainment of dancing robots and their interaction with people make it a vital R&D field in the field of home service robots. During the HCI with dancing robots, people can be participants or evaluators (Peng et al., 2021). With human participation, the robot can imitate human action. At the same time, as a bystander, people can evaluate the robot's dance postures and feedback the information to the robot for self-improvement. Overall, as a popular form of performance, dance has the characteristics of rhythm, lyricism, and movement.

CONCLUSION

Dance language has both common characteristics and special laws as a language phenomenon. Robot dance is an exciting research field that has attracted worldwide attention and research.

First, this paper systematically reviews the research, summary, and classification model in the field of dancing robots. The innovation is the proposal of a new humanoid dancing robot model that can choreograph dance elements independently. The choreography refers to the human thinking mode. Specifically, it establishes a biped-humanoid robot to choreograph dance moves actively. Then, it invites human dance professionals to evaluate the robot-imagined dance posture to verify the feasibility and effect of the model. The 30 new dance elements imagined on the L_1 get an average basic feature retention and innovation scores of 7.29 and 7.64, respectively. The results show that the proposed independent robot-imagined choreography model is superior to other literature methods and is feasible and innovative. However, there are still some deficiencies. The proposed model is based on the synchronous change of the value of one joint and the value of two joints. Future works are expected to design a new algorithm for dance element imagination based on the value change of multiple joints and maintain human dance's innovation and basic features.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

ETHICS STATEMENT

The studies involving human participants were reviewed and approved by Sultan Idris Education University Ethics Committee. The patients/participants provided their written informed consent to participate in this study. Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

AUTHOR CONTRIBUTIONS

The author confirms being the sole contributor of this work and has approved it for publication.

REFERENCES

Barnes, J. A., Park, C. H., Howard, A., and Jeon, M. (2021). Child-robot interaction in a musical dance game: an exploratory comparison study between typically developing children and children with autism. *Int. J. Human-Comput. Interact*. 37, 249–266. doi: 10.1080/10447318.2020.1819667

Borovica, T. (2020). Dance as a way of knowing-a creative inquiry into the embodiment of womanhood through dance. *Leis. Stud.* 39, 493–504. doi: 10.1080/02614367.2019.1663442

Dong, R., Chang, Q., and Ikuno, S. (2021). A deep learning framework for realistic robot motion generation. *Neural Comput. Appl.* (2021) 2021, 1–14. doi: 10.1007/s00521-021-06192-3

Dou, X., Jia, H., and Li,. H. L. (2021). The linkage cultivation of creative thinking and innovative thinking in dance choreography. *Think Skills Creativ*. 1, 100–896. doi: 10.1016/j.tsc.2021.100896

García, J., and Diogo, S. (2020). Teaching a humanoid robot to walk faster through safe reinforcement learning. *Eng. Appl. Artif. Intell*. 88, 103–360. doi: 10.1016/j.engappai.2019.103360

Hannah, D., and Red, I. J. (2020). Vajranityam: phenomenological experience of Cham or lama dance as meditation. *J. Buddhist Stud.* 3, 1–17. doi: 10.1007/s42240-019-00043-0

Hua, J., Zeng, L., Li, G., and Ju, Z. (2021). Learning for a robot: deep reinforcement learning, imitation learning, transfer learning. *Sensors*. 21, 1278. doi: 10.3390/s21041278

Kashyap, A. K., Parhi, D. R., and Kumar, S. (2020b). Dynamic stabilization of NAO humanoid robot based on whole-body control with simulated annealing. *Int. J. Human. Robot*. 17, 2050014. doi: 10.1142/S0219843620500140

Kashyap, A. K., Parhi, D. R., Muni, M. K., and Pandey, K. K. (2020a). A hybrid technique for path planning of humanoid robot NAO in static and dynamic terrains. *Appl. Soft Comput*. 96, 106–581. doi: 10.1016/j.asoc.2020.106581

Kobayashi, T., Dean-Leon, E., Guadarrama-Olvera, J. R., Bergner, F., and Cheng, G. (2022). Whole-body multicontact haptic human-humanoid interaction based on leader-follower switching: a robot dance of the "Box Step". *Adv. Intell. Syst.* 4, 2100038. doi: 10.1002/aisy.202100038

Lei, X., and Rau, P. L. P. (2021). Effect of robot tutor's feedback valence and attributional style on learners. *Int. J. Soc. Robot.* 13, 1579–1597. doi: 10.1007/s12369-020-00741-x

Li, F. (2020) Study on the characteristics of dance art of minorities in West Hunan. *Front. Art Res.* (2020). 2, 44–50. doi: 10.25236/FAR.2020.020608

Li, J., Peng, H., Hu, H., Luo, Z., and Tang, C. (2020). Multimodal information fusion for automatic aesthetics evaluation of robotic dance poses. *Int. J. Soc. Robot*. 12, 5–20. doi: 10.1007/s12369-019-00535-w

Lin, V., Yeh, H. C., and Chen, N. S. (2022). A systematic review on oral interactions in robot-assisted language learning. *Electronics*. 11, 290. doi: 10.3390/electronics11020290

Liu, C., Zhang, T., Liu, M., and Chen, Q. (2020) Active balance control of humanoid locomotion based on foot position compensation. *J. Bionic Eng.*17, 134–147. doi: 10.1007/s42235-020-0011-x

Martinez Damia, S., Marzana, D., Alfieri, S., Pozzi, M., Marta, E., and Martinez, M. L. (2021). Psychological and structural barriers to immigrant community participation: The experience of peruvians in santiago de chile. *Am. J. Community Psychol.* 67, 456–469. doi: 10.1002/ajcp.12472

Noor, N. F. M., and Saad, A. (2021). Requirement elicitation techniques for a C-programming learning application. *J. Technol. Humans*. 2, 10–16. doi: 10.53797/jthkkss.v2i2.2.2021

Peng, H., Hu, J., Wang, H., Ren, H., Sun, C., Hu, H., and Li, J. (2021). Multiple visual feature integration based automatic aesthetics evaluation of robotic dance motions. *Information*. 12, 95. doi: 10.3390/info12030095

Rifajar, L., and Abdul, F. (2021). The path direction control system for lanange jagad dance robot using the MPU6050 gyroscope sensor. *Int. J. Robot Control Systems*. 1, 27–40. doi: 10.31763/ijrcs.v1i1.225

Robaczewski, A., Bouchard, J., Bouchard, K., and Gaboury, S. (2021). Socially assistive robots: the specific case of the NAO. *Int. J. Social Robot.* 13, 795–831. doi: 10.1007/s12369-020-00664-7

Samosir, A. S., and Widodo, N. S. (2020). Gyroscope and accelerometer sensor on the lanange jagad dance robot balance system. *Bullet. Ilmiah Sarjana Teknik Elektro*. 2, 51–58. doi: 10.12928/biste.v2i2.922

Saqlain, M., Jafar, N., Moin, S., Saeed, M., and Broumi, S. (2020). Single and multi-valued neutrosophic hypersoft set and tangent similarity measure of single-valued neutrosophic hypersoft sets. *Neutrosophic Sets Syst*. 32, 317–329. doi: 10.5281/zenodo.3723165

Slot, M., Peter, H., and Eric, L. (2020). A structured approach for the instantiation of digital twins. *Proc. CIRP*. 91, 540–545. doi: 10.1016/j.procir.2020.02.211

Sun, W., Tian, X., Song, Y., Pang, B., Yuan, X., and Xu, Q. (2022). Balance control of a quadruped robot based on foot fall adjustment. *Appl. Sci.* 12, 2521. doi: 10.3390/app12052521

Conflict of Interest: The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 20 May 2022
doi: 10.3389/fnbot.2022.860981

Applying Deep Learning-Based Human Motion Recognition System in Sports Competition

*Liangliang Zhang**

Academy of Sports and Leisure, Xi'an Physical Education University, Xi'an, China

The exploration here intends to compensate for the traditional human motion recognition (HMR) systems' poor performance on large-scale datasets and micromotions. To this end, improvement is designed for the HMR in sports competition based on the deep learning (DL) algorithm. First, the background and research status of HMR are introduced. Then, a new HMR algorithm is proposed based on kernel extreme learning machine (KELM) multidimensional feature fusion (MFF). Afterward, a simulation experiment is designed to evaluate the performance of the proposed KELM-MFF-based HMR algorithm. The results showed that the recognition rate of the proposed KELM-MFF-based HMR is higher than other algorithms. The recognition rate at 10 video frame sampling points is ranked from high to low: the proposed KELM-MFF-based HMR, support vector machine (SVM)-MFF-based HMR, convolutional neural network (CNN) + optical flow (CNN-T)-based HMR, improved dense trajectory (IDT)-based HMR, converse3D (C3D)-based HMR, and CNN-based HMR. Meanwhile, the feature recognition rate of the proposed KELM-MFF-based HMR for the color dimension is higher than the time dimension, by up to 24%. Besides, the proposed KELM-MFF-based HMR algorithm's recognition rate is 92.4% under early feature fusion and 92.1% under late feature fusion, higher than 91.8 and 90.5% of the SVM-MFF-based HMR. Finally, the proposed KELM-MFF-based HMR algorithm takes 30 and 15 s for training and testing. Therefore, the algorithm designed here can be used to deal with large-scale datasets and capture and recognize micromotions. The research content provides a reference for applying extreme learning machine algorithms in sports competitions.

Keywords: deep learning, human motion recognition, sports, recognition rate, convolutional neural network, data set

OPEN ACCESS

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
Yu-Sheng Su,
National Taiwan Ocean University, Taiwan
Alireza Souri,
Islamic Azad University, Iran

****Correspondence:***
Liangliang Zhang
chuyuan_520@163.com

Received: *24 January 2022*
Accepted: *11 April 2022*
Published: *20 May 2022*

Citation:
Zhang L (2022) Applying Deep Learning-Based Human Motion Recognition System in Sports Competition.
Front. Neurorobot. 16:860981.
doi: 10.3389/fnbot.2022.860981

INTRODUCTION

With the further penetrating of computer technology (CT) into the sports fields, more CT-empowered approaches are seeing applications in athletes' training, saving the workforce while sharing training experiences. Most commonly, CT can assist coaches and athletes in tactical formulation through video content analysis (VCA) (Jiang et al., 2021). In particular, VCA can often quickly identify the tactical information in the video, thereby improving the efficiency of analytical work. VCA mainly uses image processing technology. Due to the huge amount of information in competitive sports training and the high requirements for the processing ability of machines, the human motion recognition (HMR) method combined with deep learning (DL) is used chiefly in sports VCA (Wang Q. Z. et al., 2021). In terms of human detection, the research is abundant. Many methods have been proposed to quickly and accurately detect people in video images. However, only detecting people is far from enough due to the rising and varying application demands. In many scenarios, it is necessary to further perform motion recognition on the detected people. Therefore, the accurate and real-time HMR in the video image and positioning and motion analysis is vital in real-life scenarios. Specifically, HMR-related technologies are used in traffic scheduling, urban security, gymnastics rehearsal, and stage scene analysis. There are many scenes where target detection, positioning, and motion recognition can greatly improve work efficiency and reduce human resources and material consumption (Li et al., 2021). For example, in group gymnastics rehearsal, it is possible to evaluate the performance of individual members against given standards by detecting and analyzing their positions and movement. Such can improve the overall rehearsal efficiency. The athletes' technical movements are scored using HMR technologies in the Olympic gymnastics' competition. In some interactive games, versatile HMR methods are employed to present a better gaming experience for players. Specifically, virtual reality (VR) games can analyze and recognize the player's movements and intelligently identify the player's instructions. With the introduction of DL's concept, many scholars have devoted themselves to DL research and have made great progress and innovation.

Deep learning is a subcollection of machine learning (ML). It is a new research direction that mimics the human brain to enable machines to cluster data, learn features, and forecast with incredible accuracy. Simply put, it makes computers intelligent. DL is the representation and internal law of ML sample data. Interpreting the information obtained in the learning process help to realize the artificial intelligence (AI) training. The collected information includes images, texts, and sounds. In essence, DL is a kind of ML algorithm (Hsu et al., 2021), which has seen applications in many fields, including personalized technology and data mining (Sahu et al., 2021). Because of the superior processing ability in image understanding, DL algorithms are often used in the field of VCA. As a typical DL model, a convolution neural network (CNN) can realize a multiple-layer DL structure by convoluting and sampling the original image (Khaydarova et al., 2021). Thus, CNN has exerted excellent performance in visual target recognition (Sarma K. V. et al., 2021). In particular, CNN can extract complex patterns with high reference accuracy, suitable for image processing with spatial relationships, such as the DL applications in computer vision (CV) (Jin et al., 2021). CV technology mainly uses computers and cameras to capture, track, and measure the research object. Finally, combined with an AI algorithm, CV realizes automatic motion recognition of the research object. Meanwhile, CV technology solves many shortcomings of traditional human body recognition technology (Liang et al., 2021; Shen et al., 2021). The research of HMR covers multi-disciplinary knowledge, including AI, image processing, and pattern recognition (PR) (Zhang et al., 2021). The HMR algorithm based on multi-feature fusion (MFF) has become mainstream. So far, researchers have designed the HMR algorithm based on the depth-image and obtained a high computational efficiency. But the model performs poorly on micromotion recognition. Then, others have proposed a sequential deep belief network (SDBN)-based online HMR model to extend the deep belief network (DBN) model's recognition ability over static image recognition. However, the SDBN model also prolongs the training time and thus is less time-effective on large-scale datasets. Therefore, HMR design should factor in micromotion recognition performance apart from the time efficiency on large-scale datasets.

The present work will study the application of the HMR system in sports competitions. In particular, HMR in sports competition is tracking and recording human motions through some time-specific key motion points. Then, the key points are expressed by mathematical methods. The application of the HMR system is of great significance to developing competitive sports. Based on the DL algorithm, the present work uses the HMR system to analyze the sport's tactics in sports competitions with high efficiency and quality. Specifically, it introduces the research background of HMR, designs the algorithm considering large-scale datasets and micromotion recognition, and finally evaluates the algorithm's performance through simulation experiments. The innovation of the present work is to apply HMR under DL to the field of sports competition and design a sports video-oriented HMR algorithm using kernel extreme learning machine (KELM) multidimensional feature fusion (MFF) (hereafter, KELM-MFF-based HMR algorithm). The research content provides a reference for developing HMR in sports competition fields. The organizational structure is shown in **Figure 1**. The Introduction introduces the application background of DL in the field of HMR and proposes the research questions. The literature survey summarizes and analyzes the development of HMR. The HMR algorithm of DL using KELM-MFF is applied to sports competitions. Finally, the algorithm simulation is carried out.

LITERATURE SURVEY

The development of human body recognition technology began in the 1990s, and the traditional research direction of HMR is the identification, classification, and characterization of relevant information in sports competitions (Chen and Lee, 2021). Sports information representation includes the description of specific movements of the human body, the detection of spatiotemporal information points in videos, and the tracking and recognition of motion-intense trajectory (IT) (Shieh et al., 2021). There are many ways to characterize specific human motions, such as combining multiple camera videos from different angles. Accurate HMR can be achieved by estimating the motion posture of the human body (Gao et al., 2022). HMR model can be implemented by focusing on specific human parts' motion states. Then, motions can be positioned by the constraint of the tree structure and specific motion information (Hu et al., 2022). Meanwhile, the spatiotemporal and graphical models can be combined to build an integrated HMR framework (Low et al., 2022). Generally, detection tools must be used to detect spatiotemporal information points in the video and

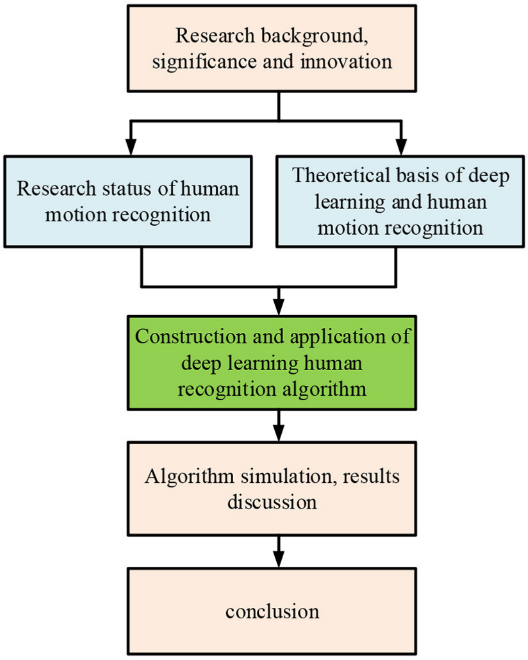

Research background, significance and innovation
Research status of human motion recognition
Theoretical basis of deep learning and human motion recognition
Construction and application of deep learning human recognition algorithm
Algorithm simulation, results discussion
conclusion

FIGURE 1 | The organizational structure.

describe their features, such as filters and three-dimensional detectors. Common feature descriptors for human motions in VCA include optical flow histogram and directional gradient histogram (Pardos et al., 2022). The first step of tracking and recognizing the motion trajectory is to pre-process and sample the video, then track the motion, and finally get multiple data representation images (Sharif et al., 2022). Afterward, the characterized human motion information needs to be identified and classified. The traditional research of HMR is mainly based on human skeleton information (Miao and Liu, 2021). Studies have shown that motion history point cloud can also describe and recognize actions.

The CV-based method requires simple equipment and is convenient to deploy. It is the main method to study HMR at this stage. It is mainly divided into top–down and bottom–up detection methods. The top–down detection method directly uses the existing detector to estimate the posture of a single person every time. Thus, the detection time is directly proportional to the number of people detected. With the increase of the target human in the image, the detection time of each image also increases, wherein the bottom–up method can separate the target human in the complex image. This method does not directly use the correlation information of other body parts and the global information of others in the image. However, the efficiency is not significantly improved, and the final local correlation needs large amounts of calculation. For example, Chen et al. (2021b) proposed a bottom–up method to associate some detection candidates with a single human body. However, the final detection time was relatively long. Liu (2022) combined the image-pairing score detection method with ResNet, significantly improving the calculation efficiency. However, it took minutes to detect each image, a far cry from real-time detection (Liu, 2022). Detecting the joint points of human posture is a single frame-oriented method. However, motion recognition analyzes the sequential posture set, featuring time-spatial characteristics. Thivel et al. (2022) believed that superimposing and calculating the motion silhouette of the human body could get the motion energy map and motion history map. They matched the two feature maps with the template to realize motion recognition (Thivel et al., 2022). Bu et al. (2022) used scale-invariant feature transform (SIFT) feature to describe motion trajectory. They then used hidden Markov model (HMM) for HMR (Bu et al., 2022). There is also research on the skeleton points-based HMR. These methods are relatively simple with a relatively low recognition rate. Choi et al. (2022) introduced the concept of "entropy." They proposed an HMR model based on motion energy using a dynamic time warping algorithm to realize HMR (Choi et al., 2022).

As from the past studies, the DL algorithm helps improve the HMR algorithm's efficiency on large-scale datasets. However, the micromotion-oriented HMR algorithm needs more in-depth research to analyze sports tactics better. In particular, the present work uses KELM to combine the manual features of improved dense trajectory (IDT) with the DL features. As such, the proposed KELM-MFF-based HMR algorithm has both advantages of manual features and DL features and can interpret human motion in sports videos from multiple angles.

TECHNICAL BACKGROUND AND DESIGN OF THE PROPOSED KELM-MFF-BASED HMR ALGORITHM

Development of HMR Technology

At present, the most popular HMR system is based on a two-CNN structure (Wang and Feng, 2021), where two CNNs are combined, one for cutting out the action image and the other for inputting the original image. The two-CNN fusion structure reduces the network parameters and accelerate the training speed (Zhang X., 2021). Additionally, some research combines spatial and temporal dimensions of CNNs for HMR. A total of two parallel frames are used to build the CNN. Alternatively, a professional camera is used to accurately recognize the human motion in the video in combination with the long-term recurrent convolutional network (LRCN) (Chen et al., 2021a). **Figure 2** displays (Kim S. U. et al., 2021) the content of HMR based on CNN.

Figure 2 implies that with the development of the 3D neural network, 3D CNN sees applications in HMR. Research directions in CNN-based HMR include video frame number, time sequence, region, and other influencing factors.

The proposed KELM-MFF-based HMR algorithm comprises a display screen, power supply (PS), controller system, and posture sensor. The system PS is a set of lithium batteries charged by the management module. The voltage stabilizing module provides appropriate PS voltage for the main controller and peripherals. The posture sensor collects the original data and sends them to the microcontroller unit (MCU). Afterward, the

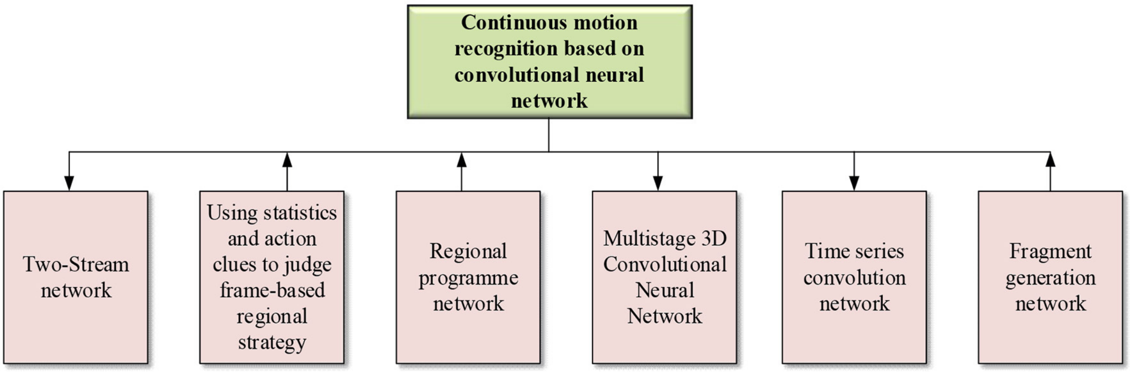

Continuous motion recognition based on convolutional neural network
Two-Stream network
Using statistics and action clues to judge frame-based regional strategy
Regional programme network
Multistage 3D Convolutional Neural Network
Time series convolution network
Fragment generation network

FIGURE 2 | Motion recognition based on CNN.

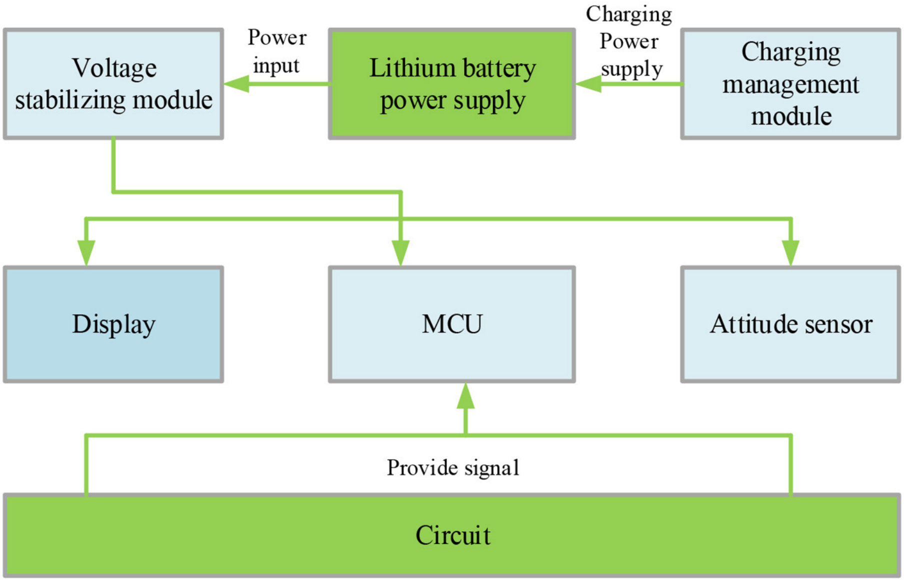

Voltage stabilizing module
Power input
Lithium battery power supply
Charging Power supply
Charging management module
Display
MCU
Attitude sensor
Provide signal
Circuit

FIGURE 3 | HMR system.

MCU sends the processed data results to the screen. The specific process is profiled in **Figure 3**.

Design of HMR Algorithm Under DL

The single hidden-layer feedforward neural network (FNN) can be solved by extreme learning machine (ELM) theory, which is more straightforward than other theories. Therefore, the present work selects the ELM to design the HMR algorithm (Hao et al., 2022). Next, the ELM model classifies a certain number of training samples (Su et al., 2021) and outputs as the minimization. The ultimate purpose is to minimize the training error.

Combined with the literature knowledge, Equation (1) gives the compatible expression of KELM.

$$f\left(x_j\right) = \left[P\left(x_j, x_1\right) \ldots P\left(x_j, x_n\right)\right]^T \left(\frac{I}{C} + P\right)^{-1} T \tag{1}$$

In Equation (1), C, x_j, and T are the regularization parameter, the training error vector, and the real motion classification. P indicates the kernel function, $j = 1, \ldots, n$. Equation (1) calculates the significance of the classification attribute of the analyzed motion training video.

Dong et al. (2022) found that information fusion helped to improve the algorithm's performance. Based on this, **Figure 4** divides the KELM into two parts for analysis.

In **Figure 4**, KELM first fuses the manual feature kernel and the DL feature kernel and predicts the manual, DL, and fusion feature kernels' score vectors (scores). Then, a neural network-trained classifier classifies the scores.

Equation (2) (Kang et al., 2021) shows the manual and DL features fusion process.

$$P\left(x_i, x_j\right) = b(x_i)b^T(x_j) \tag{2}$$

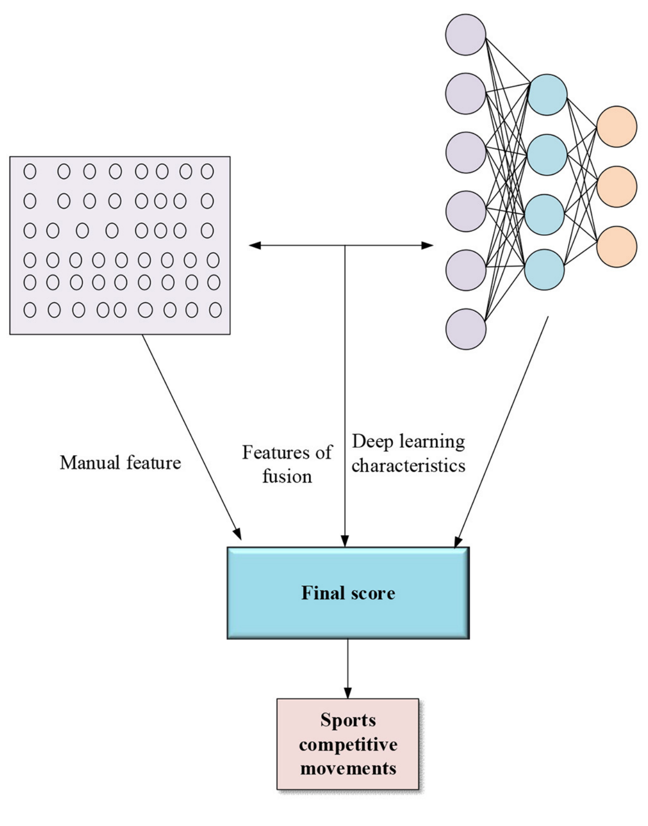


FIGURE 4 | KELM structure.

Equation (2) expresses the common manual and DL features in the video, where $P(x_i, x_j)$ are the different elements of p. The manual and DL feature scores are averaged as the fusion feature scores. Then, scores of fusion, manual, and DL features are estimated.

Subsequently, the three kernel score vectors are used for the fusion operation. The neural network structure calculates the kernel matrix based on the feature scores. Equation (3) is used to estimate the feature scores (Sedmidubsky et al., 2021).

$$P\left(q_i, q_j\right) = \exp(-\frac{\left\|q_i - q\right\|^2}{\sigma^2}) \tag{3}$$

Equation (3) is a square exponential kernel expression, where q means the video prediction score, p stands for the Gaussian element, and σ denotes the free parameter.

Further, the proposed KELM-feature fusion-based HMR algorithm is implemented using CNN and manual features. The manual features are coded by the IDT descriptor, including absolute motion features of pixels, description of static features, relative motion features of pixels, and trajectory (Lang et al., 2021). The IDT descriptor uses the Fisher vector and involves a huge amount of data (Kim T. et al., 2021).

Against training data spillover, this section proposes a new mechanism using the principal component analysis (PCA) for the IDT descriptor. The PCA-based new mechanism sets Gaussian element P to 256 to train the model and trains the dataset to 25,600 subsets randomly sampled. Finally, the Fisher vector of the IDT descriptor is obtained. Here, the linear kernel of the descriptor is designed independently, and the descriptive kernel of manual features is solved by Equation (4) (Sarma M. et al., 2021).

$$P_b = \frac{1}{n_d} \sum\nolimits_{i=1}^{n_d} P_i \tag{4}$$

Equation (4) expresses the kernel matrix of manual features, where n_d denotes a descriptor set to 4 types. They are pixel absolute motion features, descriptive static features, relative motion features, and trajectory.

The design of DL features is completed by organizing and processing descriptors. The descriptors of DL features are set to 4,096-dimensional video descriptors, and finally, the kernel matrix PD is obtained by processing (Liu and Ji, 2021).

Equation (5) calculates fusing manual features and DL features.

$$P = \frac{P_d + P_b}{2} \tag{5}$$

Equation (5) is mainly expressed by fusing the average values.

Characteristics of DL in the Field of Sports Competition

In this section, DL is introduced in the fuzzy judgment of micromotion in sports videos. **Figure 5** illustrates the main structure.

Figure 5 indicates that the DL feature extraction is carried out through two dimensions of video: time and red–green–blue (RGB) channels, which will be described later.

Here, the video's time template is used for HMR, the weighted harmonic value of motion information difference, and statistical data at different frames. Equation (6) (Zhang X. Q., 2021) illustrates the specific expression.

$$TT = (\frac{1}{250}) \sum\nolimits_{i=2}^{n} v_1 \cdot r(i) \tag{6}$$

The weighted harmonic value in Equation (6) varies between 0 and 250. v means the weight value, n stands for the number of video frames, and r denotes the motion information of each frame.

Equation (7) is obtained by transforming Equation (6).

$$TT = \sum\nolimits_{i=2}^{n} (\frac{v_1}{255}) \cdot r(i) \tag{7}$$

Equation (7) is transformed with fuzzy membership function (MF) to generate Equation (8) (Wang et al., 2021b).

$$TT = \sum\nolimits_{i=2}^{n} \lambda(i) \cdot r(i) \tag{8}$$

In Equation (8), λ denotes a fuzzy MF, and λ ϵ (0-1). Equation (8) expresses that the weight and fuzzy MF directly affect the

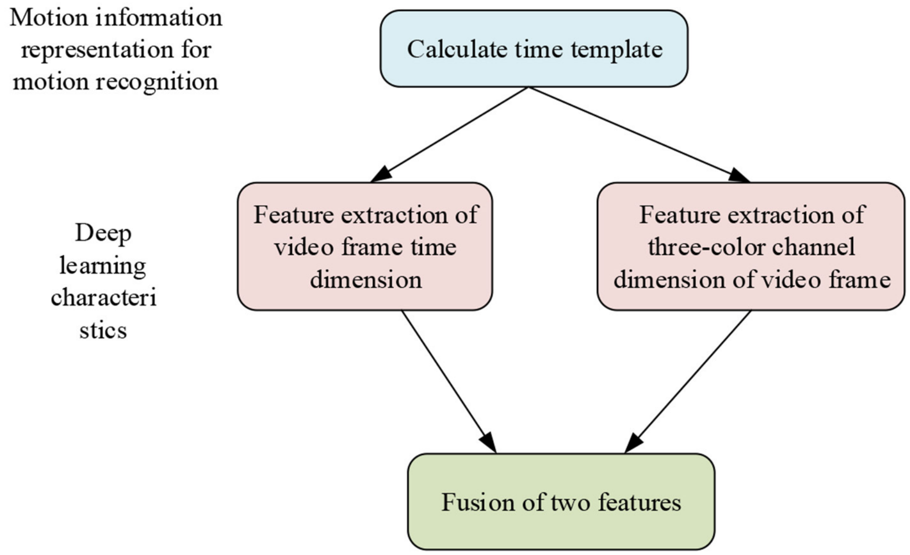

Motion information representation for motion recognition
Calculate time template
Deep learning characteri stics
Feature extraction of video frame time dimension
Feature extraction of three-color channel dimension of video frame
Fusion of two features

FIGURE 5 | DL in the field of sports competition.

TABLE 1 | DL parameter settings.

DL	Specific parameters
CNN	2 × 2 feature kernels in the pooling layer and 5 × 5 feature kernels in the convolution layer
Tri-color channel	RGB mode

significance of sports information. Equations (9–12) display the fuzzy MFs designed here.

$$\lambda 1\,(\mathrm{i}) = 1 \tag{9}$$

$$\lambda 2\,(\mathrm{i}) = \frac{\mathrm{i}}{\mathrm{n}} \tag{10}$$

$$\lambda 3\,(\mathrm{i}) = 1 - \frac{\mathrm{i}}{\mathrm{n}} \tag{11}$$

$$\lambda 4\,(\mathrm{i}) = \begin{cases} \frac{2i}{n},\ 0i \le \frac{n}{2} \\ 2 - \frac{2i}{n},\ \frac{n}{2}i \le n \end{cases} \tag{12}$$

In Equations (9–11), $i \epsilon$ (0, n).

Notably, the membership degree (MD) of the four fuzzy MFs is given in the Section Results.

The CNN is used to describe the DL features of motion information. As mentioned above, CNN is used to learn information features based on time templates. **Table 1** lists the parameters of the DL algorithm set.

Table 1 signifies that the architecture of the CNN adopted is 5C-2s-5c-2s, where 2s indicates that the number of feature kernels under the maximum pooling layer is 2 × 2. 5c indicates the number of feature kernels under the convolution layer, which is 5 × 5. Tri-color channel mode refers to RGB mode, applied to SVM to recognize competitive sports motions (Chen K. Y. et al., 2021).

Simulation Experiment of the Proposed KELM-MFF-Based HMR Algorithm

Subsequently, this section evaluates the proposed KELM-MFF-based HMR algorithm. The experimental sample adopts two kinds of video datasets. The first dataset contains large amounts of low-resolution data, which is used to test the proposed algorithm's large-scale data processing ability. The second dataset has high-resolution micromotion samples. It tests the proposed algorithm's micromotion recognition ability. In this way, sports micromotions in competitive sports can be accurately identified.

Experimental Dataset

The datasets used include the (University of Central Florida (UCF) 101 and NATOPS datasets. UCF 101 dataset is collected on the Internet, with high complexity and obvious background clutter. UCF 101 contains 13,320 video clips with 101 action categories. At the same time, this article determines three training set-test set partition schemes. The test dataset selects seven video sequences from 25 groups for each partition scheme. The other 18 video sequences are selected for training.

NATOPS video dataset contains high-resolution images. The motion recognition accuracy of the algorithm designed in this article is evaluated by small hand movements in 24 sports fields. Some gesture movements also include handshape changes. The dataset can be used to evaluate the recognition rate of the action recognition algorithm. The video dataset has a high resolution of 320 × 240; overall, 20 categories are designed. Each category includes 24 small hand movements and 24 × 20 actions altogether. Then, the first five categories are selected to test the algorithm. The last 10 categories are used to train the algorithm.

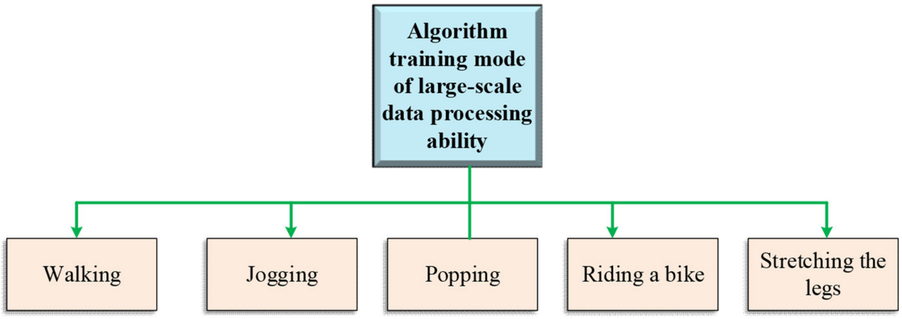

Algorithm training mode of large-scale data processing ability
Walking
Jogging
Popping
Riding a bike
Stretching the legs

FIGURE 6 | Training mode of large-scale data processing ability algorithm.

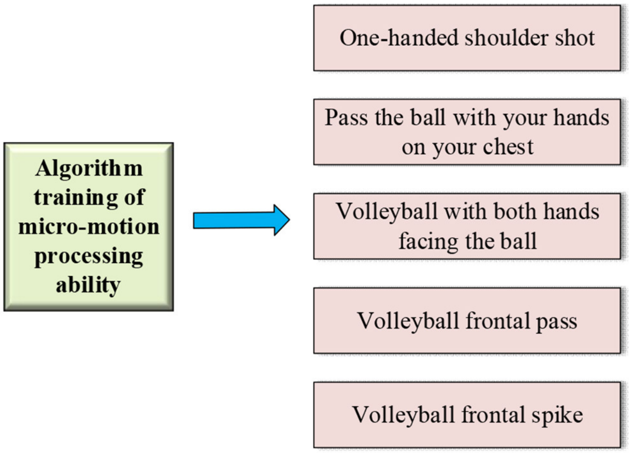

Algorithm training of micro-motion processing ability
One-handed shoulder shot
Pass the ball with your hands on your chest
Volleyball with both hands facing the ball
Volleyball frontal pass
Volleyball frontal spike

FIGURE 7 | The proposed KELM-MFF-based HMR algorithm's training mode of micromotion processing ability.

In the first dataset, the recognition performance of the proposed algorithm is evaluated by comparing it with other algorithms. The algorithms involved include an action recognition algorithm in the context of Converse3D (C3D), an action recognition algorithm in the context of the combination of motion information and SVM (SVM-MFF), an action recognition algorithm in the context of CNN, an action recognition algorithm in the context of IDT, and action recognition algorithm in the context of CNN + optical flow (CNN-T). The algorithm recognition performance evaluation under the second dataset selects 64 × 48 frames as the time template. It extracts the features of the four fuzzy FM functions in Equations (9–12) and then compares them with other algorithms. A total of 6 microhand motion recognition algorithms are involved in this experiment.

Recognition Rate of the Proposed KELM-MFF-Based HMR Algorithm

The general training framework of the proposed KELM-MFF-based HMR algorithm on the large-scale dataset is outlined in **Figure 6**.

As signified in **Figure 6**, the large-scale data processing ability of the proposed KELM-MFF-based HMR algorithm is mainly studied through the motions of jogging, walking, cycling, and stretching legs.

The general training framework of the proposed KELM-MFF-based HMR algorithm on the second dataset is portrayed in **Figure 7**.

As in **Figure 7**, the second dataset aims to train the micromotion processing ability of the proposed KELM-MFF-based HMR algorithm and focus on hand motions. Therefore, the experiment selects the videos of professional basketball and volleyball games, which both have specific requirements for players' ball-handling skills and thus involve many microhand motions.

Algorithm Feature Fusion Strategy

The feature fusion strategy is divided into two parts: early feature fusion and late feature fusion. The first dataset is mainly used for experimental analysis. **Figure 8** displays the main flow of feature fusion.

From **Figure 8**, the early fusion means feature fusion before classifier, and the later fusion is the feature fusion after classifier. More precisely, the early fusion strategy fuses the feature kernel before the classifier classifies the features. By comparison, the late fusion strategy first fuses the score vectors of each feature. It then classifies the score vector to achieve motion classification. Then, the influence of different feature fusion strategies on the performance of the HMR algorithm is evaluated on the UCF101 dataset. The proposed KELM-MFF-based HMR algorithm is compared with other kernel-based MFF HRM algorithms. Notably, the experiment mainly compares the performance of the proposed KELM-MFF-based HMR algorithm with the SVM-MFF-based HMR algorithm in terms of recognition rate under different feature fusion strategies.

Finally, the time efficiency of the proposed KELM-MFF-based HMR algorithm is evaluated on the UCF 101 dataset with the SVM-MFF-based HMR algorithm. The experimental environment is configured with an Intel i7 3.3 GHz CPU and 16GB RAM.

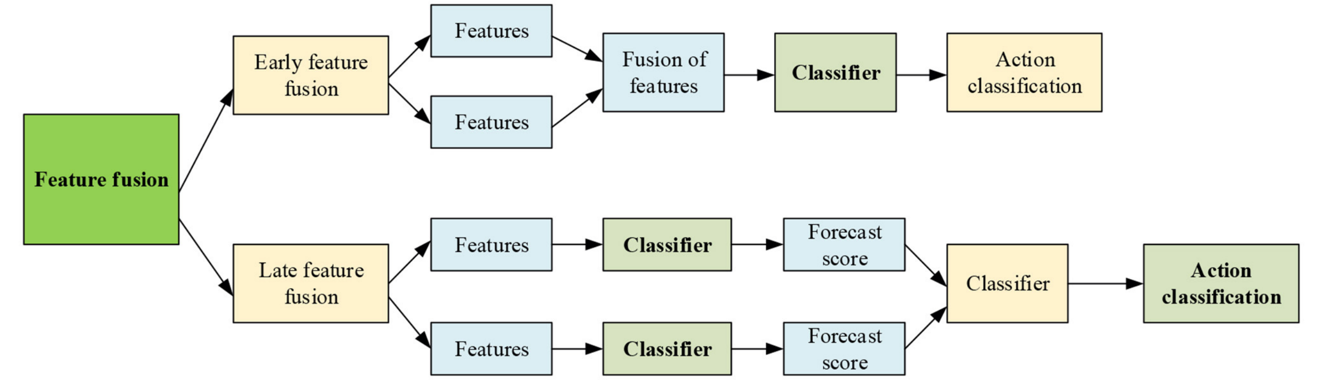


FIGURE 8 | Block diagram of feature fusion.

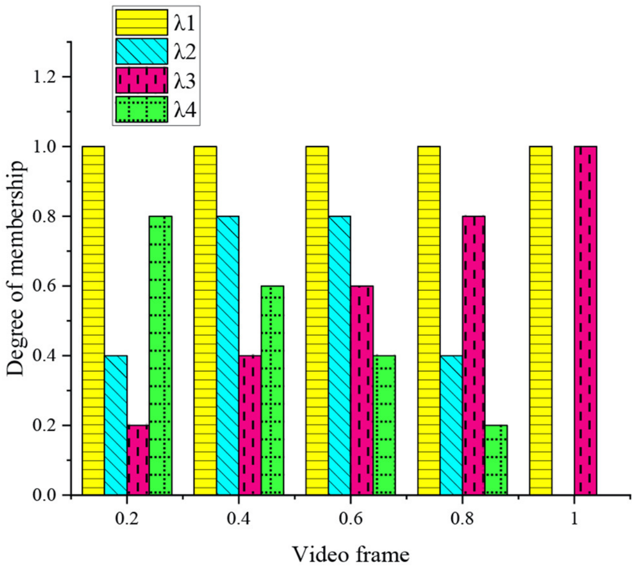


FIGURE 9 | MD of fuzzy MF.

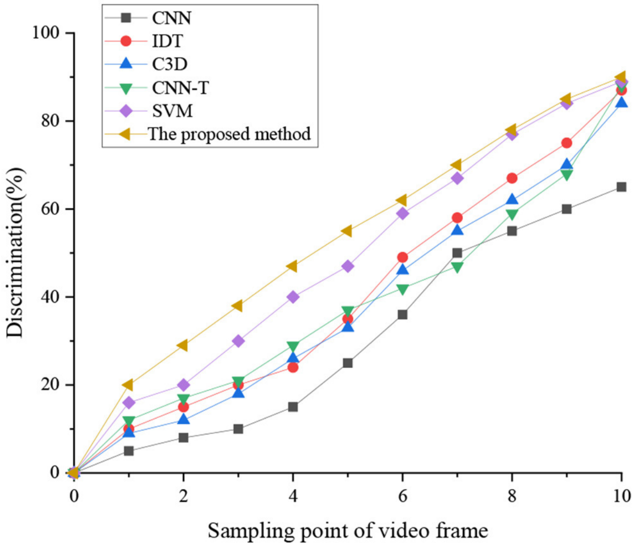


FIGURE 10 | Comparison of large-scale data processing capabilities of different algorithms.

ANALYSIS OF SIMULATION RESULTS

Comparison of Results of Recognition Rate of Different Algorithms

The MD of the fuzzy MF in the previous section is counted in **Figure 9**.

Figure 9 implies that λ2, λ3, and λ4 represent the beginning, middle, and end regions of sports competitive video in the time dimension. λ1 will not change with the change of video frame. λ1 is mainly used to calculate the energy in sports competitions, which is not affected by time and space. λ2 increases linearly with the change of video frame, which is mainly used to calculate the historical image of sports competitions and is significantly related to the number of video frames. λ3 and λ2 change in opposite directions, and λ3 is significantly correlated with the number of video frames. λ4 assigns the highest significance in the middle area.

Next, the performance of several HMR algorithms on large-scale data is comparatively analyzed, including the C3D-based HMR, SVM-MFF-based HMR, CNN-based HMR, IDT-based HMR, and CNN + optical flow (CNN-T) HMR. The results are signaled in **Figure 10**. The present work has not considered the hyperparametric adjustment and only sets the learning rate to 0.01 for all algorithms.

Figure 10 implies that different algorithms have different recognition rates on different superimposed motions. The recognition rate of the proposed KELM-MFF-based HMR algorithm is higher than that of the other five algorithms. The recognition rates of other algorithms at 10 sampling points are sorted from high to low as SVM-MFF-based HMR, CNN-T,

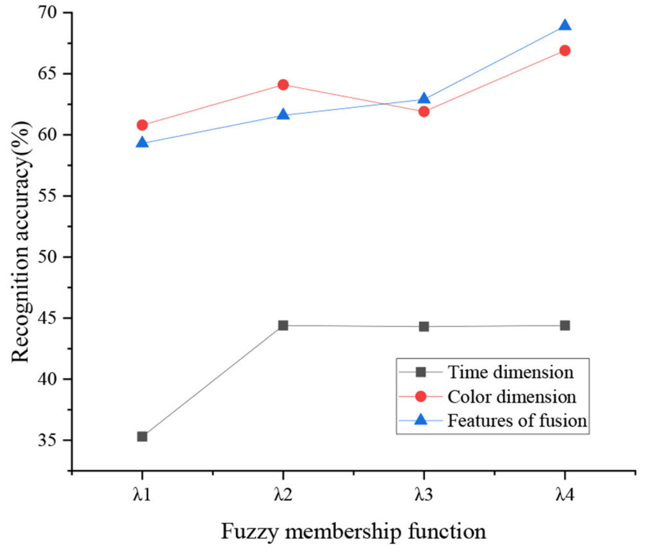


FIGURE 11 | Comparison of micromotion processing capabilities of different algorithms.

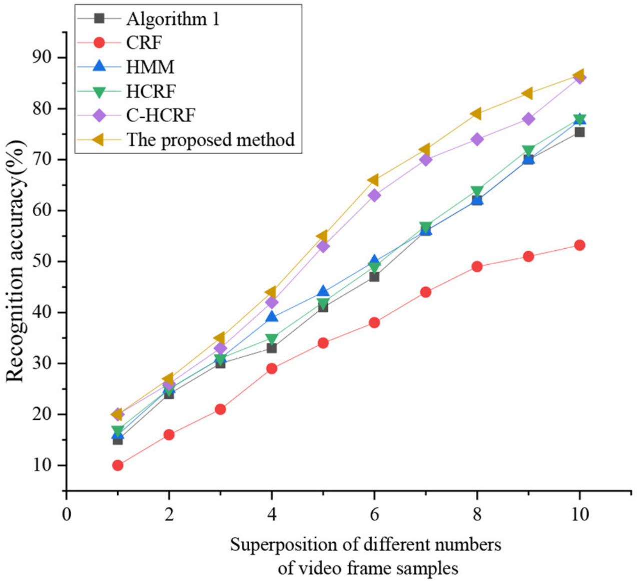


FIGURE 12 | Comparison of accuracy of micromotion recognition with different algorithms.

IDT, C3D, and CNN. Probably, it is because SVM-MFF-based HMR and CNN-T are MFF algorithms. By comparison, IDT, C3D, and CNN are single feature recognition. The conclusion can be drawn that the recognition ability of the MFF algorithm is better than that of a single feature recognition algorithm. The recognition ability of SVM-MFF-based HMR is not much different from the proposed KELM-MFF-based HMR algorithm. Presumably, the reason is that the SVM-MFF-based HMR algorithm adds a sports information mechanism using a time template to the SVM algorithm, improving the recognition rate. So far, numerous pieces of the literature have shown the MFF algorithm's advantages. Additionally, the present work results are consistent with Tanaka et al. (2022) latest research results. The recognition ability of MFF is stronger than that of a single feature recognition algorithm. The difference is that many comparison models are used in the present work (Tanaka et al., 2022).

Figure 11 denotes the recognition of different dimension features by fuzzy MF.

Figure 11 illustrates that the feature recognition rate of RGB features is higher than that of the time feature, by up to 24%. The recognition rate of the time dimension is low because the microhand motions are easy to overlap in movement. The first fuzzy MF has a lower recognition rate. The second, third, and fourth MFs have a better recognition rate because they represent the significance of the beginning, middle, and end of the video sequence. Overall, the accuracy of the RGB color feature is better than that of the time feature because microhuman motions are more likely to overlap. In this case, the recognition of color feature is higher. In the human body recognition research on the phenomenon of human body overlap, the recognition rate of the proposal of Santos et al. (2022) is consistent with the present work.

Afterward, the last three fuzzy MFs are fused. The comparison is made between the proposed KELM-MFF-based HMR algorithm and the SVM-MFF-based HMR algorithm in recognizing micromotions, as sketched in **Figure 12**.

As **Figure 12** displays, the proposed KELM-MFF-based HMR algorithm has the highest recognition rate of the other five algorithms. The proposed KELM-MFF-based HMR algorithm fuses motion information with RGB features. It has more advantages in capturing micromotion than the other five algorithms. Hidden Markov algorithm (HMM), algorithm 1, and conditional random field (CRF) belong to a single recognition algorithm, so the recognition rate is low. By comparison, the hidden conditional random field (HCRF) and the continuous hidden conditional random field (C-HCRF) use video sequences. They have a higher recognition rate because they belong to multidimensional recognition. Apparently, the proposed KELM-MFF-based HMR algorithm can be used to deal with micro-HMR in sports videos. Compared with the latest research results of Varshney et al. (2022), the experimental accuracy of the present work is higher, indicating the superiority of fusing motion information and color features. However, the analysis of color fusion in the literature is more in-depth than the present work, so the result findings are more convincing than the present work. Thus, the present work will also do more in-depth research on color fusion in the future.

Comparison of Feature Fusion Strategies of Different Algorithms

Figure 13 compares the recognition rate between different fusion strategies of the SVM-MFF-based HMR and the proposed KELM-MFF-based HMR.

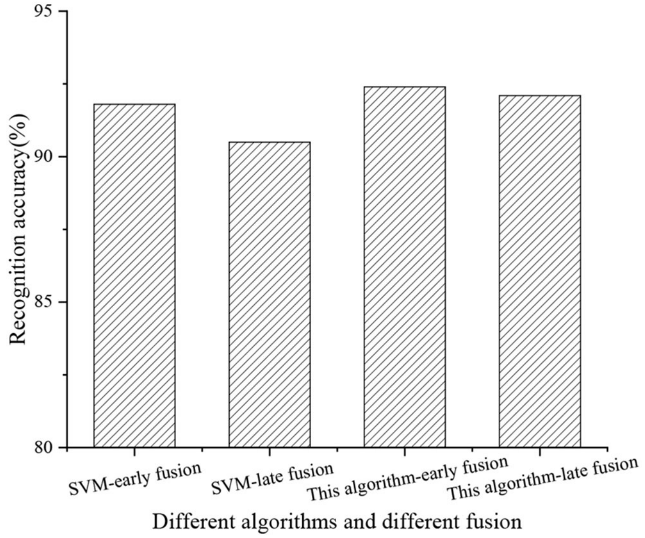


FIGURE 13 | Comparison of recognition rate under different algorithm feature fusion strategies.

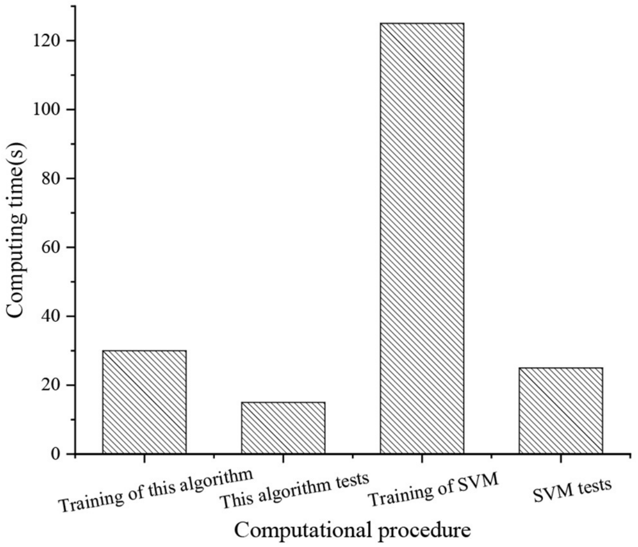


FIGURE 14 | Comparison of time efficiency of different algorithms.

Figure 13 signifies the recognition rate. Apparently, the proposed KELM-MFF-based HMR algorithm is higher than the SVM-MFF-based HMR algorithm in both the early and late feature fusion stages, reaching 92.4 and 92.1%. The SVM-MFF-based HMR algorithm has reached 91.8 and 90.5%. The conclusion draws that the recognition rate is higher when features are fused earlier than later under both algorithms.

Comparison of Time Efficiency of Different Algorithms

The time efficiency of the proposed KELM-MFF-based HMR algorithm is compared with that of the SVM-MFF-based HMR algorithm, as plotted in **Figure 14**.

Figure 14 signifies that the proposed KELM-MFF-based HMR algorithm takes a shorter time to train and test than the SVM-MFF-based HMR, only 30 and 15 s for training and testing. By comparison, the SVM-MFF-based HMR algorithm takes 125 and 25 s. Thus, the time efficiency of the proposed KELM-MFF-based HMR algorithm is much higher than that of the SVM-MFF-based HMR algorithm. Therefore, the proposed KELM-MFF-based HMR algorithm can be used to deal with large-scale datasets. Bhatia et al. (2022) also have observed that the KELM is suitable for processing large-scale datasets. At present, there are few researches on large-scale dataset processing in sports. The advantage of the present work is applying the proposed KELM-MFF-based HMR algorithm to the field of sports.

DISCUSSION

In improving HMR, the recognition rate of the proposed algorithm on large-scale data is more than 86%, higher than that of the SVM, CNN-T, IDT, C3D algorithm, and CNN algorithms. The recognition rate of the CNN-T algorithm is also high, which intuitively shows the superiority of the MFF algorithm. With the rapid development of science and technology, the combinatorial algorithm has become the first choice of current researchers. The MFF algorithm can improve the performance of a single algorithm and make up for the limitations of a single algorithm. The latest research by Kyaw et al. (2022) shows that the MFF algorithm is a critical way to solve practical problems. HMR is inseparable from color recognition. The accuracy of color recognition of the proposed algorithm is affected by the phenomenon of human action overlap. It is hoped to strengthen the research on improving the recognition rate of action overlap in the future. For HMR, the proposed KELM-MFF-based HMR has the highest accuracy among the five comparison algorithms, mainly because this algorithm integrates RGB color features and motion information. The research of Yang and Zou (2022) suggests that the integration of RGB color features plays a vital role in recognition algorithms. The present work verifies that the ELM is suitable for processing large-scale video datasets through the time efficiency comparison of different algorithms. The finding provides data support for applying the ELM algorithm in video recognition fields.

CONCLUSION

Following a review of the HMR system using the DL algorithm, the present work studies the application of HMR systems in sports competitions. After background introduction, a KELM-MFF-based HMR algorithm is designed to improve traditional algorithms' poor performance against large-scale data and micromotions in sports videos. Then, a simulation experiment is designed to evaluate the performance of the proposed KELM-MFF-based HMR algorithm. The research findings corroborated that the proposed KELM-MFF-based HMR algorithm can be used to solve two problems in the current algorithm. (1) The DL features of human motions in the video sequence are analyzed through the time template to assign different significance to different time domains of the motion information. (2) The time template of the video sequence is inputted into the CNN to learn the feature set of sports motions. The manual and DL features are complementary and describe the human motions in videos from different angles. The research content provides a reference for applying the DL algorithm in sports competitions. There are still some deficiencies in the article. The second experimental dataset (NATOPS video dataset) only involves the professional motions in basketball and volleyball without adding other sports. Meanwhile, the analysis of color characteristics is not deep enough. The present work does not optimize the hyperparameters of the model. The follow-up research can combine the common sports actions into a new dataset for a more comprehensive analysis. There is a need to deepen the research on color characteristics and increase the hyperparameter setting and recognition rate of human actions for more convincing research results. After further improvement, it is expected to be applied to college sports events.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

ETHICS STATEMENT

The studies involving human participants were reviewed and approved by Xi'an Physical Education University Ethics Committee. The patients/participants provided their written informed consent to participate in this study. Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

AUTHOR CONTRIBUTIONS

The author confirms being the sole contributor of this work and has approved it for publication.

REFERENCES

Bhatia, Y., Bari, A. S. M., Hsu, G. S. J., and Gavrilova, M. (2022). Motion capture sensor-based emotion recognition using a bi-modular sequential neural network. *Sensors.* 22, 403. doi: 10.3390/s22010403

Bu, D., Guo, S., and Li, H. (2022). sEMG-Based motion recognition of upper limb rehabilitation using the improved yolo-v4 algorithm. *Life.* 12, 64. doi: 10.3390/life12010064

Chen, H. Y., and Lee, C. H. (2021). Deep learning approach for vibration signals applications. *Sensors.* 21, 3929. doi: 10.3390/s21113929

Chen, K. Y., Chou, L. W., Lee, H. M., Young, S. T., Lin, C. H., and Zhou, Y. S. (2021). Human motion tracking using 3d image features with a long short-term memory mechanism model—an example of forward reaching. *Sensors.* 22, 292. doi: 10.3390/s22010292

Chen, P., Guo, S., Li, H., Wang, X., Cui, G., Jiang, C., et al. (2021a). Through-wall human motion recognition based on transfer learning and ensemble learning. *IEEE Geosci. Remote Sens. Lett.* 19, 1–5. doi: 10.1109/LGRS.2021.3070374

Chen, P., Wang, X., Wang, M., Yang, X., Guo, S., and Jiang, C. (2021b). Multi-view real-time human motion recognition based on ensemble learning. *IEEE Sens. J.* 21, 20335–20347. doi: 10.1109/JSEN.2021.3094548

Choi, S. H., Park, K. B., Roh, D. H., Lee, J. Y., Mohammed, M., and Ghasemi, Y. (2022). An integrated mixed reality system for safety-aware human-robot collaboration using deep learning and digital twin generation. *Robot. Comput. Integr. Manufact.* 73, 102258. doi: 10.1016/j.rcim.2021.102258

Dong, B., Zhang, Z., Shi, Q., Wei, J., Ma, Y., and Xiao, Z. (2022). Biometrics-protected optical communication enabled by deep learning–enhanced triboelectric/photonic synergistic interface. *Sci. Adv.* 8, eabl9874. doi: 10.1126/sciadv.abl9874

Gao, H., Guo, F., Zhu, J., Kan, Z., and Zhang, X. (2022). Human motion segmentation based on structure constraint matrix factorization. *Sci. China Inform. Sci.* 65, 1–2. doi: 10.1007/s11432-020-2967-3

Hao, Z., Niu, J., Dang, X., and Qiao, Z. (2022). WiPg: contactless action recognition using ambient wi-fi signals. *Sensors.* 22, 402. doi: 10.3390/s22010402

Hsu, K. T., Guan, S., and Chitnis, P. V. (2021). Comparing deep learning frameworks for photoacoustic tomography image reconstruction. *Photoacoustics.* 23, 100271. doi: 10.1016/j.pacs.2021.100271

Hu, K., Ding, Y., Jin, J., Weng, L., and Xia, M. (2022). Skeleton motion recognition based on multi-scale deep spatio-temporal features. *Appl. Sci.* 12, 1028. doi: 10.3390/app12031028

Jiang, X., Liang, S. J., He, X. Y., Ziegler, A. D., Lin, P., and Pan, M. (2021). Rapid and large-scale mapping of flood inundation *via* integrating spaceborne synthetic aperture radar imagery with unsupervised deep learning. *Int. Soc. Photogr. Remote Sens. J. Photogram. Remote Sens.* 178, 36–50. doi: 10.1016/j.isprsjprs.2021.05.019

Jin, G. Y., Sha, H. Y., Feng, Y. H., and Huang, J. (2021). GSEN: an ensemble deep learning benchmark model for urban hotspots spatiotemporal prediction. *Neurocomputing.* 455, 353–367. doi: 10.1016/j.neucom.2021.05.008

Kang, S., Jang, M., and Lee, S. (2021). Identification of human motion using radar sensor in an indoor environment. *Sensors.* 21, 2305. doi: 10.3390/s21072305

Khaydarova, R., Mouromtsev, D., Fishchenko, V., Shmatkov, V., Lapaev, M., and Shilin, I. (2021). ROCK-CNN: distributed deep learning computations in a resource-constrained cluster. *Int. J. Embedded Real-Time Commun. Syst.* 12, 14–31. doi: 10.4018/IJERTCS.2021070102

Kim, S. U., Jang, H., Im, H., and Kim, J. (2021). Human motion reconstruction using deep transformer networks. *Pattern Recogn. Lett.* 150, 162–169. doi: 10.1016/j.patrec.2021.06.018

Kim, T., Park, J., Lee, J., and Park, J. (2021). Predicting human motion signals using modern deep learning techniques and smartphone sensors. *Sensors.* 21, 8270. doi: 10.3390/s21248270

Kyaw, K. S., Limsiroratana, S., and Sattayaraksa, T. A. C. (2022). A comparative study of meta-heuristic and conventional search in optimization of multi-dimensional feature selection. *Int. J. Appl. Metaheur. Comput.* 13, 1–34. doi: 10.4018/IJAMC.292517

Lang, Y., Hou, C., Ji, H., and Yang, Y. (2021). A dual generation adversarial network for human motion detection using micro-Doppler signatures. *IEEE Sensors J.* 21, 17995–18003. doi: 10.1109/JSEN.2021.3084241

Li, M. F., Lin, H., Chen, J. S., Diaz, J. M., Xiao, Q., Wang, F., et al. (2021). swFLOW: a large-scale distributed framework for deep learning on Sunway TaihuLight supercomputer. *Inform. Sci.* 570, 831–847. doi: 10.1016/j.ins.2020.12.079

Liang, Y. Y., Peng, W., Zheng, Z. J., Silvén, O., and Zhao, G. (2021). A hybrid quantum–classical neural network with deep residual learning. *Neural Networks.* 143, 133–147. doi: 10.1016/j.neunet.2021.05.028

Liu, Q. (2022). Human motion state recognition based on MEMS sensors and Zigbee network. *Comput. Commun.* 181, 164–172. doi: 10.1016/j.comcom.2021.10.018

Liu, Y., and Ji, Y. (2021). Target recognition of sport athletes based on deep learning and convolutional neural network. *J. Intellig. Fuzzy Syst.* 40, 2253–2263. doi: 10.3233/JIFS-189223

Low, W. S., Chan, C. K., Chuah, J. H., Tee, Y. K., Hum, Y. C., and Salim, M. I. M. (2022). A review of machine learning network in human motion biomechanics. *J. Grid Comput.* 20, 1–37. doi: 10.1007/s10723-021-09595-7

Miao, A., and Liu, F. (2021). Application of human motion recognition technology in extreme learning machine. *Int. J. Adv. Robotic Syst.* 18, 1729881420983219. doi: 10.1177/1729881420983219

Pardos, A., Menychtas, A., and Maglogiannis, I. (2022). On unifying deep learning and edge computing for human motion analysis in exergames development. *Neural Comput. Appl.* 34, 951–967. doi: 10.1007/s00521-021-06181-6

Sahu, A. K., Sharma, S., Tanveer, M., and Raja, R. (2021). Internet of things attack detection using hybrid deep learning model. *Comput. Commun.* 176, 146–154. doi: 10.1016/j.comcom.2021.05.024

Santos, F. G. C., Oliveira, D. D. S., Passos, A. L., Pires, R. G., Santos, D. F. S., and Valem, L. P. (2022). Gait recognition based on deep learning: a survey. *ACM Comput. Surv. (CSUR).* 55, 1–34. doi: 10.1145/3490235

Sarma, K. V., Harmon, S., Sanford, T., Roth, H. R., Xu, Z. Y., and Tetreault, J. (2021). Federated learning improves site performance in multicenter deep learning without data sharing. *J. Am. Med. Inform. Assoc.* 28, 1259–1264. doi: 10.1093/jamia/ocaa341

Sarma, M. S., Deb, K., Dhar, P. K., and Koshiba, T. (2021). Traditional Bangladeshi sports video classification using deep learning method. *Appl. Sci.* 11, 2149. doi: 10.3390/app11052149

Sedmidubsky, J., Elias, P., Budikova, P., and Zezula, P. (2021). Content-based management of human motion data: survey and challenges. *IEEE Access.* 9, 64241–64255. doi: 10.1109/ACCESS.2021.3075766

Sharif, M. I., Khan, M. A., Alqahtani, A., Nazir, M., Alsubai, S., and Binbusayyis, A. (2022). Deep learning and kurtosis-controlled, entropy-based framework for human gait recognition using video sequences. *Electronics.* 11, 334. doi: 10.3390/electronics11030334

Shen, C. G., Wang, C. C., Huang, M. H., Xu, N., van der Zwaag, S., and Xu, W. (2021). A generic high-throughput microstructure classification and quantification method for regular SEM images of complex steel microstructures combining EBSD labeling and deep learning. *J. Mater. Sci. Technol.* 93, 191–204. doi: 10.1016/j.jmst.2021.04.009

Shieh, C. S., Lin, W. W., Nguyen, T. T., Chen, C. H., Horng, M. F., and Miu, D. (2021). Detection of unknown DDoS attacks with deep learning and gaussian mixture model. *Appl. Sci.* 11, 5213. doi: 10.3390/app11115213

Su, M. C., Tai, P. T., Chen, J. H., Hsieh, Y. Z., Lee, S. F., and Yeh, Z. F. A. P. (2021). Motion recognition algorithm based on depth sensors. *IEEE Sens. J.* 21, 16990–16996. doi: 10.1109/JSEN.2021.3079983

Tanaka, R., Nozaki, S., Goshima, F., and Shiraishi, J. (2022). Deep learning versus the human visual system for detecting motion blur in radiography. *J. Med. Imag.* 9, 015501. doi: 10.1117/1.JMI.9.1.015501

Thivel, D., Corteval, A., Favreau, J. M., Bergeret, E., Samalin, L., and Costes, F. (2022). Fine detection of human motion during activities of daily living as a clinical indicator for the detection and early treatment of chronic diseases: the E-Mob project. *J. Med. Internet Res.* 24, e32362. doi: 10.2196/32362

Varshney, N., Bakariya, B., Kushwaha, A. K. S., and Khare, M. (2022). Human activity recognition by combining external features with accelerometer sensor data using deep learning network model. *Multimedia Tools Appl.* 2, 1–20. doi: 10.1007/s11042-021-11313-0

Wang, Q. Z., Ma, W. Q., and Liu, G. (2021). SieveNet: decoupling activation function neural network for privacy-preserving deep learning. *Inform. Sci.* 573, 262–278. doi: 10.1016/j.ins.2021.05.054

Wang, S., An, Q., Li, S., Zhao, G., and Sun, H. (2021b). Wiring effects mitigation for through-wall human motion micro-Doppler signatures using a generative adversarial network. *IEEE Sens. J.* 21, 10007–10016. doi: 10.1109/JSEN.2021.3057592

Wang, Y. W., and Feng, F. (2021). Reliability enhancement algorithm of human motion recognition based on knowledge graph. *Int. J. Distrib. Syst. Technol.* 12, 1–15. doi: 10.4018/IJDST.2021010101

Yang, G., and Zou, W. X. (2022). Deep learning network model based on fusion of spatiotemporal features for action recognition. *Multimedia Tools Appl.* 13, 1–22. doi: 10.1007/s11042-022-11937-w

Zhang, N. N., Ying, S., Ding, W. P., Zhu, K., and Zhu, D. (2021). WGNCS: a robust hybrid cross-version defect model *via* multi-objective optimization and deep enhanced feature representation. *Inform. Sci.* 570, 545–576. doi: 10.1016/j.ins.2021.05.008

Zhang, X. (2021). Application of human motion recognition utilizing deep learning and smart wearable device in sports. *Int. J. Syst. Assur. Eng. Manag.* 12, 835–843. doi: 10.1007/s13198-021-01118-7

Zhang, X. Q. (2021). Application of convolution network model based on deep learning in sports image information detection. *E3S Web Conf.* 233, 02024. doi: 10.1051/e3sconf/202123302024

Conflict of Interest: The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 26 May 2022
doi: 10.3389/fnbot.2022.829445

Application of Deep Convolution Network Algorithm in Sports Video Hot Spot Detection

Yaling Zhang[1], Huan Tang[2], Fateh Zereg[3]* and Dekai Xu[4]

[1] School of Management, Beijing Sport University, Beijing, China, [2] College of Sports, Leisure and Tourism, Beijing Sport University, Beijing, China, [3] Department of Theory and Methodology of Football, Chengdu Sport University, Chengdu, China, [4] Institute of Physical Education, Hoseo University, Asan-si, South Korea

Sports videos are blowing up over the internet with enriching material life and the higher pursuit of spiritual life of people. Thus, automatically identifying and detecting helpful information from videos have arisen as a relatively novel research direction. Accordingly, the present work proposes a Human Pose Estimation (HPE) model to automatically classify sports videos and detect hot spots in videos to solve the deficiency of traditional algorithms. Firstly, Deep Learning (DL) is introduced. Then, amounts of human motion features are extracted by the Region Proposal Network (RPN). Next, an HPE model is implemented based on Deep Convolutional Neural Network (DCNN). Finally, the HPE model is applied to motion recognition and video classification in sports videos. The research findings corroborate that an effective and accurate HPE model can be implemented using the DCNN to recognize and classify videos effectively. Meanwhile, Big Data Technology (BDT) is applied to count the playing amounts of various sports videos. It is convinced that the HPE model based on DCNN can effectively and accurately classify the sports videos and then provide a basis for the following statistics of various sports videos by BDT. Finally, a new outlook is proposed to apply new technology in the entertainment industry.

Keywords: deep convolutional neural network, human motion recognition model, big data technology, hot spot detection, sports video

OPEN ACCESS

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
Samuel Spevack,
Exponent, United States
Eva H. Dulf,
Technical University of Cluj-Napoca, Romania

***Correspondence:**
Fateh Zereg
doctorfateh@mail.ru

Received: *05 December 2021*
Accepted: *07 March 2022*
Published: *26 May 2022*

Citation:
Zhang Y, Tang H, Zereg F and Xu D (2022) Application of Deep Convolution Network Algorithm in Sports Video Hot Spot Detection. Front. Neurorobot. 16:829445. doi: 10.3389/fnbot.2022.829445

INTRODUCTION

Today, the Chinese are enjoying an ever affluent material life as the domestic economy boosts, with which their pursuit for spiritual wellbeing rises sharply. Especially with the increasingly mature computer and networking technology (CNT), online resources have become primary resort of people for spiritual pursuit and substantially enhance entertainment industry of China. Thus, it is not uncommon for Chinese citizens to watch live sports broadcasts or sports videos to escape the fast-paced and competitive study, life, and work.

The deep convolutional neural network (DCNN) algorithm is particularly apt for image recognition (IR). Once, feature extraction (FE) has hindered the further development of IR techniques, given the complexity of image data over other common data types, such as texts. At that time, image features were mainly represented through artificial means that are based on understanding or estimation of researchers.

Such a situation has not seen a fundamental change until the emergence of CNN, almost tailor-made for automatic FE for images over traditional machine learning (ML). Thus, CNN has dramatically simplified many emerging technologies, such as IR and Pattern Recognition (PR) (Sulam et al., 2019; Maiorino et al., 2021). In particular, big data technology (BDT) uses Big Data to provide analytical solutions to complex practical problems. The so-called big data refers to a collection of structured, semistructured, and unstructured data sets that are so voluminous for conventional data processing algorithms to tackle appropriately that include the acquisition, storage, management, and analysis of big data. Big data features massive scale, rapid flow, diverse types, and low-value density (Romanowski, 2019; Tolan et al., 2020).

In the Big data era, deep learning (DL) has become one of the most favored technologies in various industrial applications, on which extensive research has been conducted. For example, Singh et al. (2021) proposed a novel pointwise convolution design using spatial input information. Specifically, they extracted and refined the input spatial context information on two scales (Singh et al., 2021). Finally, a time-series feature vector for classification trained the support vector machine (SVM). Sandula et al. (2021) constructed a new camera motion classification framework based on the hue-saturation-intensity (HSI) model to compress block motion vectors. The designed framework sends the input to the inter-frame block motion vector decoded by the compressed stream to estimate its size and direction and assign the motion vector direction to hue and the motion vector size to saturation under a fixed Intensity. Then, the HSI distribution was converted to Red, Green, Blue (RGB) images. After that, CNN was used for supervised learning to identify 11 camera motion modes that include seven pure camera motion modes and four hybrid camera modes. The results showed that the recognition accuracy of this method for 11 camera modes reached over 98% (Sandula et al., 2021). Rajesh and Muralidhara (2021) designed a reconstruction loss based on new driving and used the implicit multivariate Markov random field regularization method to enhance local details. They used a multi-column network to propagate the local and global information from the context to the target coated area (Rajesh and Muralidhara, 2021). Churchill et al. (2020) applied the original data from a single diagnosis (the electron cyclotron emission-oriented imaging diagnostic data) from the Tokamak. They designed a neural network (NN) architecture for the prevalent interruption prediction problem: the fusion energy Tokamak (Churchill et al., 2020). Jamali et al. (2021) built a timeliness and computational efficiency CNN structure and combined it with other mainstream NN and ML algorithms: random forest (RF), Gaussian network, and Bayesian optimization Tree. As a result, the training time was significantly shortened (Jamali et al., 2021). Chen et al. (2020) employed DL in video recognition and put forward a DCNN model for human motion recognition (HMR) (Chen et al., 2020). Minhas et al. (2019) devised a track and field video-oriented shot classification method using an eight-layer AlexNet, including five convolution layers and three fully connected layers. The shot was divided into a long shot, medium shot, close-up shot, and off-site shot. The standardization and screening layer response on the feature map improved overall training and validation performance on different databases (Minhas et al., 2019). Ramesh and Mahesh (2020) analyzed the performance of the pre-training network, used the AlexNet for FE, image classification (IC), and transfer learning (TL), and compared its performance with the DCNN with a user-defined layer on the same dataset. Consequently, the performance of DCNN was improved (Ramesh and Mahesh, 2020). To sum up, DCNN has been applied in various aspects, the most important of which is used for data processing and machine vision recognition, which can greatly improve the accuracy of HMR in the video. In particular, AlexNet has gained enormous technological advances through the research of experts and scholars on CNN technology in different fields. Generally, an AlexNet is a CNN structure with five convolution layers and three fully connected layers, with two parallel graphics processing units (GPUs) for computation acceleration. Meanwhile, the upper and lower information can interact at the third convolution layer and fully connected layer. AlexNet extends the basic principles of CNN to deeper and wider networks. Concerning the current research topic, the sports video's hotspot detection, it is found that different types of CNNs generate great differences in prediction accuracy. Therefore, the present work will optimize the hotspot detection model combined with BDT to empower the scoring system effectively.

Thereupon, the present work proposes a human pose estimation (HPE) for sports video classification, recognition, and detection using DCNN and BDT. The available relevant literature mostly uses the DCNN algorithm to extract the features of sports images and can correctly identify the character poses, only with relatively low recognition accuracy. By comparison, the present work innovatively employs a Haar-like FE algorithm for image data screening and enhancement, thus dramatically improving the accuracy of HPE, reducing video delay, and enhancing the efficiency of athlete motion capture. Thus, a sports video recognition and classification model based on BDT and DCNN is implemented, and the simulation experiment is carried out. The innovation lies in optimizing the traditional NN model by combining BDT with DCNN. Afterward, by comparing single HPE algorithms and fusion models, the most appropriate combination is selected to implement the proposed athlete-oriented HPE model, increasing video pose recognition accuracy and efficiency. As a result, a new perspective is proposed for computer technology (CT) to promote the development of the entertainment industry. Innovatively, the first-identification-and-then-recognition method is employed to address the HMR problem in sports video sequences. The target i algorithm (TIA) is primarily used to detect poses of athletes in sports videos. Then, the TIA is used to identify category of athletes of motion behaviors in the video sequence. The value and excellence of present research provide a more effective method to detect the pose of the athletes in sports video and offer a new outlook for the related research of high-precision HPE in the future.

CONSTRUCTION OF SPORTS VIDEO CLASSIFICATION AND RECOGNITION SYSTEM

Big Data Technology

Big data refers to extra-large-scale datasets that cannot be captured, managed, and processed by conventional software tools within finite iterations. It is a voluminous, high-growth-rated, diversified information asset that requires a new processing mode with more robust decision-making (DM) power, insight and discovery power, and process optimization ability (Singh et al., 2019). **Figure 1** shows the basic architecture of Big Data.

Generally speaking, before the final data report review or data-based algorithm prediction, BDT will go through the following processing links: data collection, data storage, data processing, and data application.

Here, the primary sources of experimental data are (a) background data of sports videos from major online platforms, (b) rating data of various television (TV) sports channels, and (c) viewing data of various broadcast channels during major sports events (Opotowsky and Webb, 2020; Saju and Ravimaran, 2020).

Image Classification and FE

Figure 2 signifies the flow of an ordinary TIA.

Select the Bounding Box

When such target information as the number, size, and type in a picture is unknown, it is necessary to ensure two aspects of the algorithm design: (a) the bounding boxes for FE include as many items to be inspected as possible and (b) the bounding boxes for FE are as few as possible (Chavez-Badiola et al., 2020).

Three methods are available for selecting bounding boxes: Sliding Window Algorithm (SWA), Selective Search (SS), and Region Proposal Network (RPN). SWA reflects the detailed features of images in the sliding window. Global FE methods mainly include principal component analysis (PCA), gray level-gradient co-occurrence matrix (GLCM), frequency domain analysis method, scale-invariant feature transform (SIFT), and Haar-like features. Remarkably, the present experiment chooses the RPN method for bounding box selection with the following implementation steps. First, pictures are used as initial input for SS, according to which the bounding boxes are further determined. Then, the pictures are segmented into smaller images with unique features, similarities, and gradients used for splicing the bounding boxes into large blocks. Finally, the size of bounding boxes is determined according to the blocks. The RPN method reduces bounding box extraction from images when compared with the SWA. Meanwhile, RPN uses the CNN algorithm to extract bounding boxes and generates dense boxes by a fixed scale. Further, CNN can classify and point out the positions of the boxes so that the final bounding boxes could contain all objects, thus minimizing the number of bounding boxes (Abu Hasan et al., 2020; Budiman and Sugiarto, 2020; Tarsitano et al., 2021). **Figure 3** illustrates the RPN algorithm flow.

Feature Extraction

Feature Extraction technology is the core of computer vision and is the basis of IR and IC. FE involves both bottom-layer features

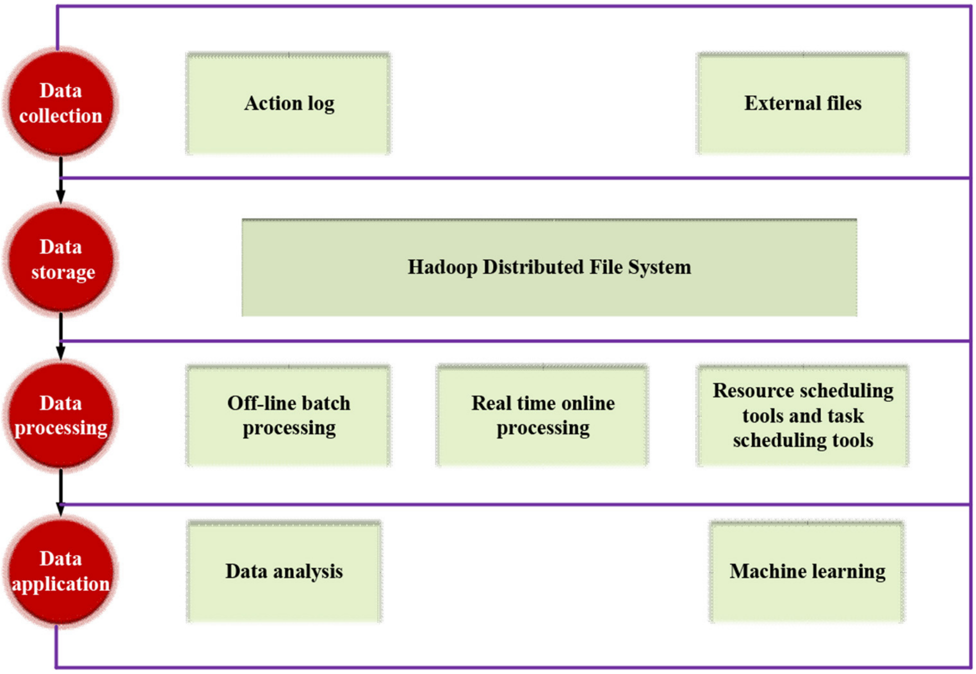

Data collection
Action log
External files
Data storage
Hadoop Distributed File System
Data processing
Off-line batch processing
Real time online processing
Resource scheduling tools and task scheduling tools
Data application
Data analysis
Machine learning

FIGURE 1 | Basic architecture of Big Data.

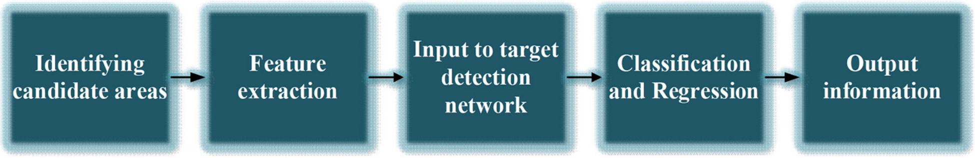

Identifying candidate areas
Feature extraction
Input to target detection network
Classification and Regression
Output information

FIGURE 2 | Detection flow of the target identification algorithm (TIA).

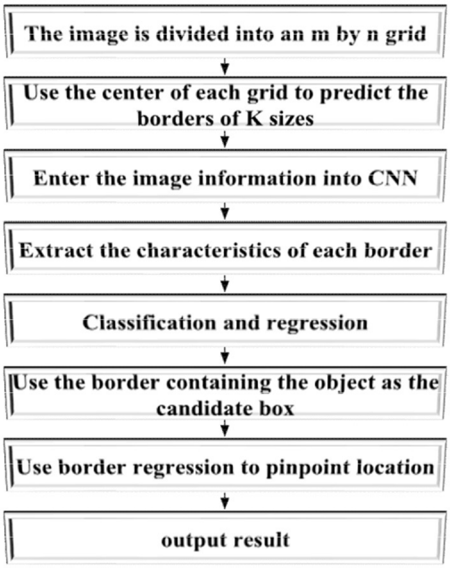

The image is divided into an m by n grid
Use the center of each grid to predict the borders of K sizes
Enter the image information into CNN
Extract the characteristics of each border
Classification and regression
Use the border containing the object as the candidate box
Use border regression to pinpoint location
output result

FIGURE 3 | Bounding box selection by Region Proposal Network (RPN) method.

(image color and texture) and advanced features (including image generalization and abstraction). Information, such as size and quantity, helps in visualizing asset management (Khaleefah et al., 2020; Saad and Hirakawa, 2020). These features are listed as follows.

a. Image color: A coordinate system can be established based on the RGB three-hue format and calculation. RGB can adjust the vector size to express different image colors as vector components. RGB three-hue format can create color distribution histogram, color space coding, and image hash features.
b. Image texture refers to the distribution of image hue and light intensity in a certain area, including the GLCM.
c. Image edge: It summarizes the outline of image content. Standard FE algorithms include Canny or Sobel operator edge detection.
d. Image transformation: The time-to-frequency domain transformation can depict the image details, where the statistical and content features are separated.
e. Image depth: The CNN algorithm is used for post-extraction of standard and deep image features, widely used to describe image features at the semantic and bottom feature levels.

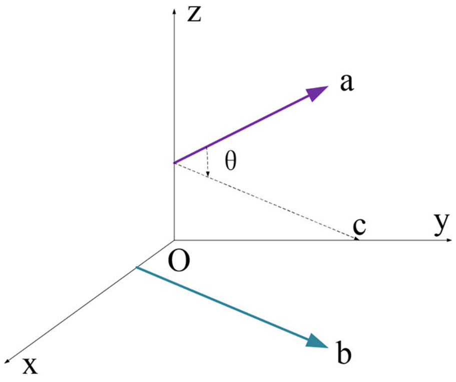


FIGURE 4 | Schematic diagram of the cross product.

Feature Matching

Correlation between image features can be different. The proposed algorithm calculates the image correlation by defining the distance between two features (Gururaj and Tunga, 2020; Sugiarto et al., 2020; Desai et al., 2021), as detailed below:

a. Euclidean Distance (ED): Equation (1) counts the distance (ED) between two feature vectors in the Euclidean space.

$$d_{xy} = \sqrt{\sum_{k=1}^{K} (x_k - y_k)^2} \tag{1}$$

In Equation (1), x and y are all feature vectors, K stands for the feature vector dimension, and d_{xy} refers to the ED. The larger the d_{xy} is, the larger the correlation between two points and vice versa.

b. Cross product. This parameter describes the consistency of vector direction and value in a different dimension, expressed as Equation (2) and displayed in **Figure 4**.

$$\left|\overrightarrow{a} \times \overrightarrow{b}\right| = \left|\overrightarrow{a}\right| . \left|\overrightarrow{b}\right| . \sin\theta \tag{2}$$

Where $\overrightarrow{a}$ and $\overrightarrow{b}$ symbolize two vectors of a point. θ equals to the space angle between two vectors and vector C is the parallel spatial line of $\overrightarrow{b}$. Therefore, the angle between $\overrightarrow{a}$ and C equals that of the $\overrightarrow{b}$ and C.

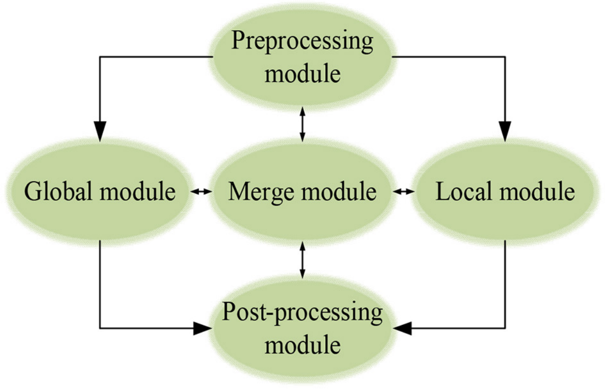


FIGURE 5 | The modular expression framework.

c. Cosine distance. It indicates the angular distance, namely, the cosine value of the angle between two eigenvectors. The larger the value is, the more similar the two eigenvectors are and vice versa.

Expression Framework of HPE

Human Pose Estimation means estimating the coordinates and position of human joints relative to each part in a given image. Here, a single-lens camera is used for HPE. This section designs an expression framework to describe the whole image and express information-abundant local features effectively. Therefore, a rough human body model is implemented initially. Then, optimization is made on the local features in all directions. SWA can scan the whole image with pre-determined sliding window parameters in local feature modeling. A 220 × 220-pixel sliding window extracts the local features. Before a full-effect expression framework can be established, it is necessary to pre-process the input images so that DCNN can accept (Wang et al., 2019; Lv and Liang, 2020; Abid et al., 2021), as illuminated in **Figure 5**.

Pre-Processing and Post-Processing Module

The pre-processing module is used for data adjustment and enhancement during model training. In contrast, the post-processing module extracts' joint information of the athlete during training and competition and feedback the information.

Data Enhancement

Then, only minor changes need to be made to the existing data set to obtain more data, such as flipping, translation, or rotation. CNN can classify objects robustly even if they are placed in different directions. After translation, the viewpoint, size, or illumination (or a combination of the three) remain unchanged. Images collected from the internet will have different sizes. Since fully connected layers are fundamental design in most NNs, image size must be fixed before inputting the network. Popular data enhancement technologies operate pictures in the following ways: scaling, translation, rotation, flipping, adding noise, illumination adjustment, and perspective transformation.

Image recognition relies on BDT, and many training samples must be prepared. Accordingly, to weigh the shortage of training samples, the present work chooses to enhance the sample data, minimizing model overfitting or distortion from insufficient training. Then, a human body structure with K joints is expressed by Equation (3), where L will be used as the initial input of joints:

$$L = (l_1, \cdots, l_i \cdots), (i \in \{1, \cdots, K\}) \tag{3}$$

In (3), $l_i = (x_i, y_i)$ is the coordinates of the i-th human joint.

Where D represents the position information, L is the actual value of human action. The original labeled sample is expressed as (D, L), which is expressed as $(D, \overline{L})$ after the expression framework is adjusted. D_x^r and D_y^r denote the after-rotation abscissa and ordinate, respectively.

Data enhancement performs multiple operations on the original image. Then, to describe the coordinates after rotation, the top left corner of the image is set to be 0 point and its coordinates to be (1,1). From this point on, the Y-axis is set to top-to-bottom, the X-axis left-to-right. The picture size is (D_x, D_y). The rotation angle is φ; clockwise in the positive direction. $l_i{}^r = (x_i{}^r, y_i{}^r)$ is the coordinate of the rotated human joint i. D^r is the data of the rotated image.

a. Rotation: as in Equations (4–7).

$$x_i^r = (x - \frac{D_x}{2}) \cos\varphi - (y - \frac{D_y}{2}) \sin\varphi + \frac{D_x^r}{2} \tag{4}$$

$$y_i^r = (y - \frac{Dy}{2}) \cos\varphi - (x - \frac{Dx}{2}) \sin\varphi + \frac{D_y^r}{2} \tag{5}$$

$$D_x^r = (D_y - D_x \tan\varphi) \sin\varphi + D_x \cos\varphi \tag{6}$$

$$D_y^r = (D_x - D_y \tan\varphi) \sin\varphi + D_y \cos\varphi \tag{7}$$

The rotation angle is used to obtain amounts of training samples randomly.

b. Translation: Different points on the image are selected to translate the human body, by which the minimum clipping area can be solved by Equations (8) and (9).

The rotation angle is random to obtain many training samples.

$$C_{top-left} = (\min(x_1^r, \cdots, x_K^r), \min(y_1^r - y_K^r) \tag{8}$$

$$C_{bottom-right} = ((\max(x_1^r, \cdots, x_K^r), \max(y_1^r - y_K^r)) \tag{9}$$

Where $C_{top-left}$ is the coordinate of the leftmost pixel in the clipping region, and $C_{bottom-right}$ is at the bottom right. min and max represent the minimum and the maximum values in the sequence, x_K^r and y_K^r refer to the abscissa and ordinate after conversion.

c. Scaling
d. Horizontal turning: The coordinates after horizontal turning are expressed as Equations (10) and (11).

$$x_j^f = D_x^S - x_i^S \tag{10}$$

$$y_j^f = y_i^S \tag{11}$$

In Equations (10) and (11), x_j^f refers to the inverted abscissa, y_j^findicates the inverted ordinate, D_x^S represents the data of the inverted image, (x_j^f, y_j^f) indicates the coordinate value of the inverted pixel, and j stands for the joint corresponding to the original joint i.

HPE and Evaluation

The Precision-Recall (P-R) curve is used to evaluate the detection effect. Equations (12) and (13) calculate the P and R values.

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

For a test, the relationship between the predicted value and the actual value has four situations as signified in the following: TP represents that the actual value of the positive sample is true, FP represents that the predicted value of the negative sample is true, TN represents that the predictive value of the positive sample is true, and FN represents that the prediction of negative sample is false. Pindicates precision and R denotes recall.

Deep Convolutional Neural Network

Deep Learning

In recent years, DL has been a relatively new and popular research direction in the ML field. DL enables computers to learn and summarize the internal laws of various data (sound, image, and other data) and finally power the computer to analyze and learn like humans. At present, many significant breakthroughs have been made in DL (Cocos and Fiks, 2019). At first, in 2006, DL appeared as a new research direction in ML and gradually been applied to various industries. In 2012, Stanford University took the lead in building a training model called DNN. Then, using 16,000 Central Processing Units (CPUs), DNN technology has made a great breakthrough in voice and IR applications. In 2016, alpha dog, an artificial go (Chinese "Weiqi") software was developed based on DL, defeated Li Shishi, the top go game master in the world. Since then, many well-known high-tech companies worldwide have begun to invest tremendous material resources and talents in DL and setup relevant research institutes.

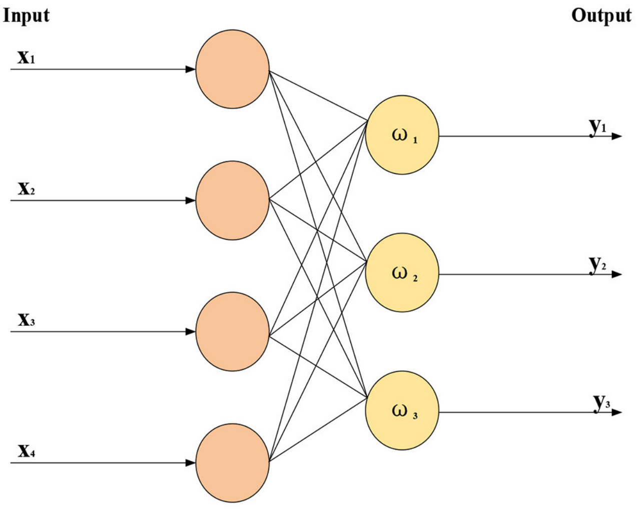


FIGURE 6 | Structure of simple neural network.

Machine learning technology studies how computers simulate or realize learning behaviors of humans to learn new knowledge, rewrite existing data structures, and improve program performance. From a statistical perspective, ML is used to predict data distribution, model available data, and then predict new data, requiring testing and training data to conform to the same distribution. Its basic feature is to imitate the mode of information transmission and processing between neurons in the human brain. Computer vision and Natural Language Processing (NLP) are the most notable applications of ML. Obviously, "DL" is strongly related to "neural network" in ML. "Neural network" is also its main implementation algorithm and means. Thus, "DL" can also be called an "improved neural network" algorithm. The idea of DL is to simulate human neurons. Each neuron receives information and transmits it to all adjacent neurons after processing. **Figure 6** sketches the structure of a simple NN.

AlexNet: A DCNN

The basic principle of AlexNet has been applied to many other deep network structures. The leading new technologies used by AlexNet are as follows:

a. Rectified linear activation function (ReLU) is successfully used as the activation function (AF) of CNN, and its effect is verified to exceed that of Sigmoid function in deep NN. Thus, ReLU can solve the gradient dispersion problem of the Sigmoid function. Although the ReLU was proposed a long time ago, it has not seen further application until the emergence of AlexNet (Shorten and Khoshgoftaar, 2019).
b. During training, Dropout is used to ignore some neurons randomly to avoid model overfitting. Dropout has been discussed in a separate paper, and AlexNet has put it into

practice and proved its effect through practice. In AlexNet, Dropout is mainly used in several last fully connected layers.

c. Overlapping maximum pooling (Max pooling) is introduced into CNN. Before AlexNet, CNN used average pooling. By comparison, AlexNet introduces Max pooling to avoid the blurring effect of average pooling. Additionally, AlexNet proposes to scale down the step size than the pooling core so that there will be overlap and coverage between the outputs of the pooling layer, which improves the richness of features.

d. A local response normalization (LRN) layer is proposed to create a competition mechanism for local neurons. The value with a larger response becomes relatively larger and inhibits other neurons with minor feedback. As a result, it enhances the model generalization. In training many chain data, once the distribution of each batch of training data is different (batch gradient decline), the network must learn to adapt to different distributions in each iteration, which will significantly reduce the training speed.

Figure 7 displays the AlexNet-based DCNN structure, and **Table 1** lists its detailed parameters.

The proposed AlexNet-based DCNN has five convolution layers, followed by the Max-pooling layer for down-sampling and three fully connected layers. The last layer is the SoftMax output layer, which has 1,000 nodes and corresponds to 1,000 image-classification options in ImageNet atlas. The network middle volume dividing base layer is calculated by two independent operations, which is beneficial to GPU parallelization and reduces calculation.

Two-dimensional (2D) CNN comprises a 2D convolution layer and a 2D pooling layer. Since the 2D CNN does not process object motion information in the time dimension between images, it cannot detect video sequences. Researchers proposed three-dimensional (3D) CNN to better extract video features by adding a time dimension to the convolution kernel (namely, the 3D convolution kernel) to obtain time-domain and spatial domain information. Thus, 3DCNN can achieve better recognition results. Three dimensional CNN forms a cube by stacking multiple consecutive frames. In this cube structure, each feature map in the convolution layer is connected to multiple adjacent consecutive frames in the previous layer to capture motion information.

The improved CNN model has three 3D convolution layers, three 3D pooling layers, one fully connected layer, one SoftMax layer, and two Flatten layers, as shown in **Figure 8**. The size of the 3D convolution kernel is [3 × 3 × 3], where "3 × 3" represents the spatial dimension, the third "3" is the time dimension, the step size is "1," and padding is "1." The number of convolution kernels from the first convolution layer to the sixth convolution layer is 64, 128, 256, 256, 512, and 512, respectively. Each convolution layer is connected to the pooling layer, and Max pooling is used throughout the network structure. The Max pooling can effectively eliminate the estimated mean shift caused by the parameter error of the convolution layer. The filter size of the first pooling layer is [1 × 2 × 2], and the size of the remaining filters is [2 × 2 × 2]. After the Dropout and Flatten layers, SoftMax is used to classify and get the class output. **Figure 8** denotes an improved 3DCNN structure.

Multi-Instance Learning (MIL)

According to the ambiguity of training data, the research in this field can be roughly divided into three learning frameworks: supervised learning, unsupervised learning, and Reinforcement Learning (RL). Supervised learning marks the samples, while unsupervised learning does not need to mark them, so the learning model has great ambiguity. MIL can be considered the fourth learning framework juxtaposed with the three traditional learning frameworks. MIL can be described as follows: suppose that each datum in the training data set is a package, each package is a set of examples, each package has a training tag, and the examples in the package are tagless. Then, one positively marked example will make the whole package positive; all examples within a negatively tagged package are negative. Noticeably, here the tagging operation is in relation to the sample training. Differently put, the training operation only labels the package, not the examples, but the example labels do exist. Meanwhile, samples can be positive or negative and are used for further sample classification. All the sample tags are given in supervised learning but are unknown in MIL. By comparison, all examples are tagless in unsupervised learning, while the package in MIL is

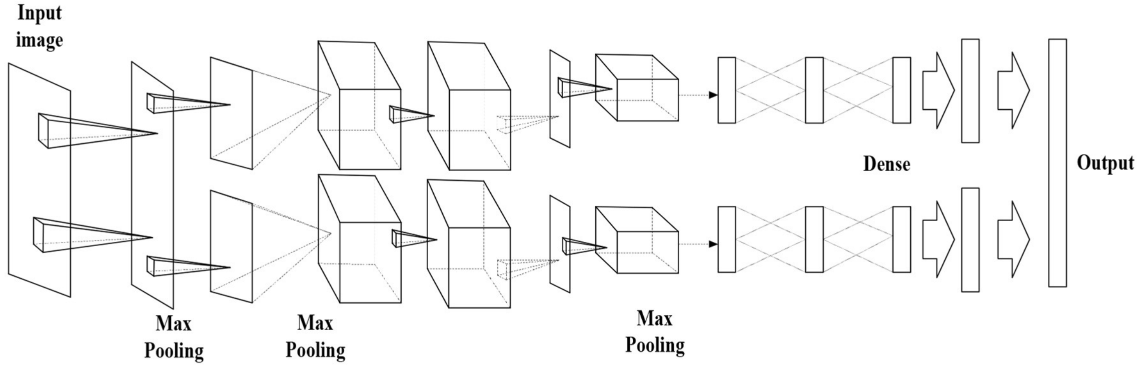


FIGURE 7 | Deep Convolutional Neural Network (DCNN) structure.

TABLE 1 | Detailed AlexNet-based Deep Convolutional Neural Network (DCNN) parameters.

Input image data (size: 227 * 227 * 3)		
Convolution layer 1	11 * 11 convolution kernel a, number = 48, step size = 4 Activate function (relu) Pooling layer (kernel size = 3, stride = 2) Standardization	11 * 11 convolution kernel b, number = 48, step size 4 Activate function (relu) Pooling layer (kernel size = 3, stride = 2)
Convolution layer 2	Convolution layer size = 5*5, number = 128, step size = 1 Activate function (relu) Pooling layer (kernel size = 3, stride = 2) Standardization	Convolution layer size = 5*5, number = 128, step size = 1. Activate function (relu) Pooling layer (kernel size =3, stride = 2)
Convolution layer 3	Convolution kernel size = 3 * 3, number = 192, step size = 1 Activate function (relu)	Convolution kernel size = 3 * 3, number = 192, step size 1. Activate function (relu)
Convolution layer 4	Convolution kernel size = 3 * 3, number = 192, step size = 1 Activate function (relu)	Convolution kernel size = 3 * 3, number = 192, step size 1. Activate function (relu)
Convolution layer 5	Convolution kernel size = 3*3, number = 192, step size = 1 Activate function (relu) Pooling layer (kernel size = 3, stride = 2)	Convolution kernel size = 3*3, number = 192, step size = 1. Activate function (relu) Pooling layer (kernel size = 3, stride = 2)
Fully connected layer 6	2,048 neurons Dropout	2,048 neurons. Dropout
Fully connected layer 7	2,048 neurons Dropout	2,048 neurons. Dropout
Fully connected layer 8	1,000 neurons	

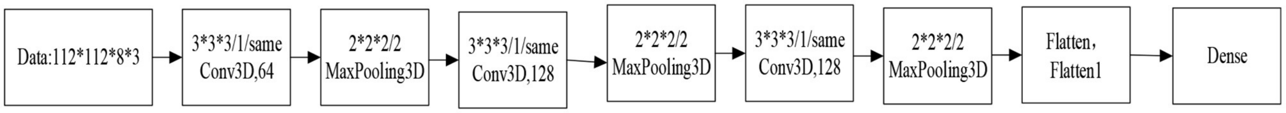


FIGURE 8 | Structure of the improved three-dimensional Convolutional Neural Network (3DCNN).

tagged. However, one characteristic of MIL is that it widely exists in the real world and has great potential application prospects.

Weakly Supervised Learning (WSL)

Supervised learning technology builds prediction models by learning many labeled training samples, which has succeeded in many fields. However, due to the high cost of data annotation, it is not easy to obtain strong supervision information, such as all truth labels, in many tasks. However, due to the lack of formulated labels, the performance of unsupervised learning in practical application is often very limited. To solve this problem, relevant researchers put forward the concept of WSL that can reduce the workload of manual marking and introduce supervised human information to improve the performance of unsupervised learning to a great extent.

Weakly Supervised Learning is relative to supervised learning. Unlike supervised learning, the data labels in WSL are allowed to be incomplete. That is, only part of the data in the training set has labels, and the rest or even most of the data are unlabeled. In other words, supervised learning is indirect. That is, the ML signal is not directly assigned to the model but indirectly transmitted to the ML model through some guiding information. In short, WSL covers a wide range. In some sense, tag learning can be regarded as WSL as long as the annotation information is incomplete, inaccurate, or imprecise. WSL mainly includes Semi-Supervised Learning (SSL), TL, and RL.

RESEARCH MODEL AND FRAMEWORK

Algorithm Model

The recognition method based on the strength feature metric is used here. **Figure 9** displays its recognition and detection steps. General human behavior recognition algorithm: spatio-temporal graph convolutional network (ST-GCN) refers to recognizing human behavior in the videos, namely, reading video. According to the types of actions and the processing tasks, the behavior recognition tasks in various cases are slightly different. First, compare and distinguish the two groups of concepts:

The steps of image content identification and detection are as follows:

The first step is to use the faster region CNN (FR-CNN) target detection model trained on Microsoft Common Objects in Context (MSCOCO) general dataset to detect human features. Part of the FR-CNN structure comes from the convolution layer of pre-training, and the other part is its unique structure. The training process is as follows: (a) initialize the network with the convolution layer of pre-training and carry out training. After training, the convolution layer of pre-training and its unique structure will be updated and (b) initialize the FR-CNN with the pre-training convolution layer. Note that the pre-training convolution layer is the same as the first step. The trained network is used to calculate the unique value, and then the updated suggestion is given to FR-CNN. Then, the model training continues to update the convolution layer of pre-training and the network structure of FR-CNN. In steps a and b, the FR-CNN is initialized with the same pre-training convolution layer and then trained independently. Therefore, the update of the pre-training convolution layer must be different after training, which means that the pre-training convolution layer is not shared. (c) The convolution layer pre-trained in step b is used to initialize the network and train the network for the second time. Notably, the convolution layer of pre-training should be locked this time. Differently put, the pre-trained convolution layer will remain unchanged during the training process. However, the unique structure will be changed, so the pre-training convolution layer will always be consistent with the pre-trained FR-CNN structure in step c. (d) Keep the pre-trained convolution layer in step c unchanged, initialize FR-CNN, and train FR-CNN for the second time.

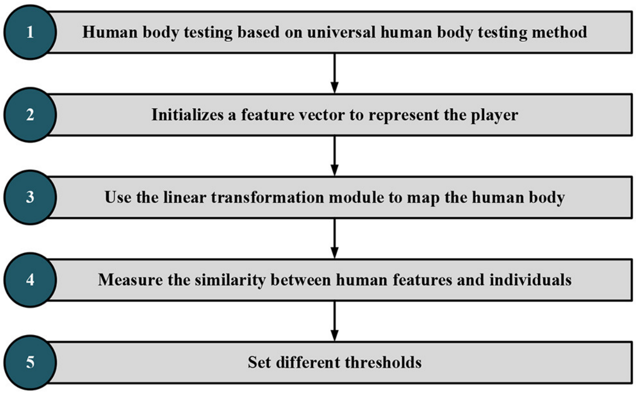

1
Human body testing based on universal human body testing method
2
Initializes a feature vector to represent the player
3
Use the linear transformation module to map the human body
4
Measure the similarity between human features and individuals
5
Set different thresholds

FIGURE 9 | Identification and detection steps.

In the second step, an initial vector represents main trunk of the athlete, and the SWA is used for FE. A rectangular box with multiple scales is specified in advance. The sliding window slides from left to right and from top to bottom according to a particular step from the upper left corner of the image. Each sliding position is reserved as a bounding box. Increasing the rectangular box type and reducing the sliding step allow the bounding box to put down all detected objects. The linear transformation module is trained by the MIL method. When it is known that the image contains object categories, the candidate frame of the image is extracted first. These bounding boxes form a negative package for an image that does not contain a target object. These bounding boxes form a positive package for an image containing a target object. The goal of MIL is to determine which bounding box in the positive package is the target object. After continuous MIL between positive and negative packets, the category of bounding boxes is finally determined to complete object localization. There is no need to label the position and category of the boundary.

In the third step, the linear transformation module represents the human trunk features. The feature is extracted on the feature map of the target detection model according to the human trunk detection frame, and the linear transformation module is trained in the form of WSL. In this paper, RL is selected, a typical WSL algorithm. Unlike supervised learning, RL needs to try to find the results of each pose, and there is no training data to tell

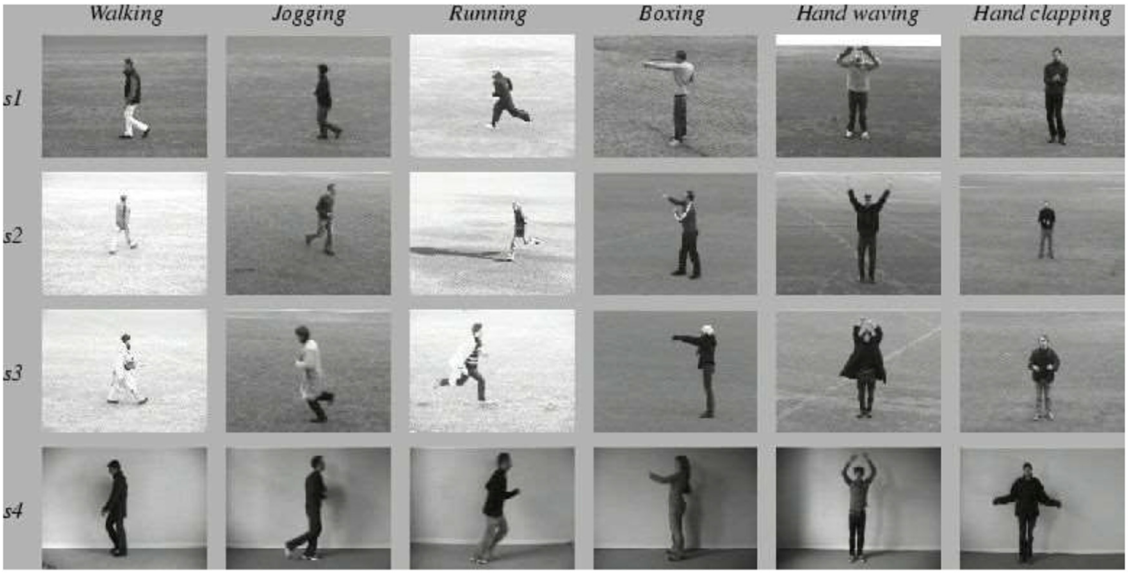


FIGURE 10 | KTH datasets.

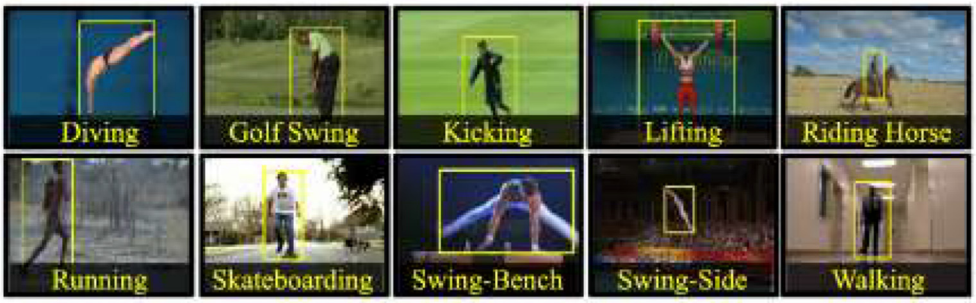


FIGURE 11 | UCF Sports datasets.

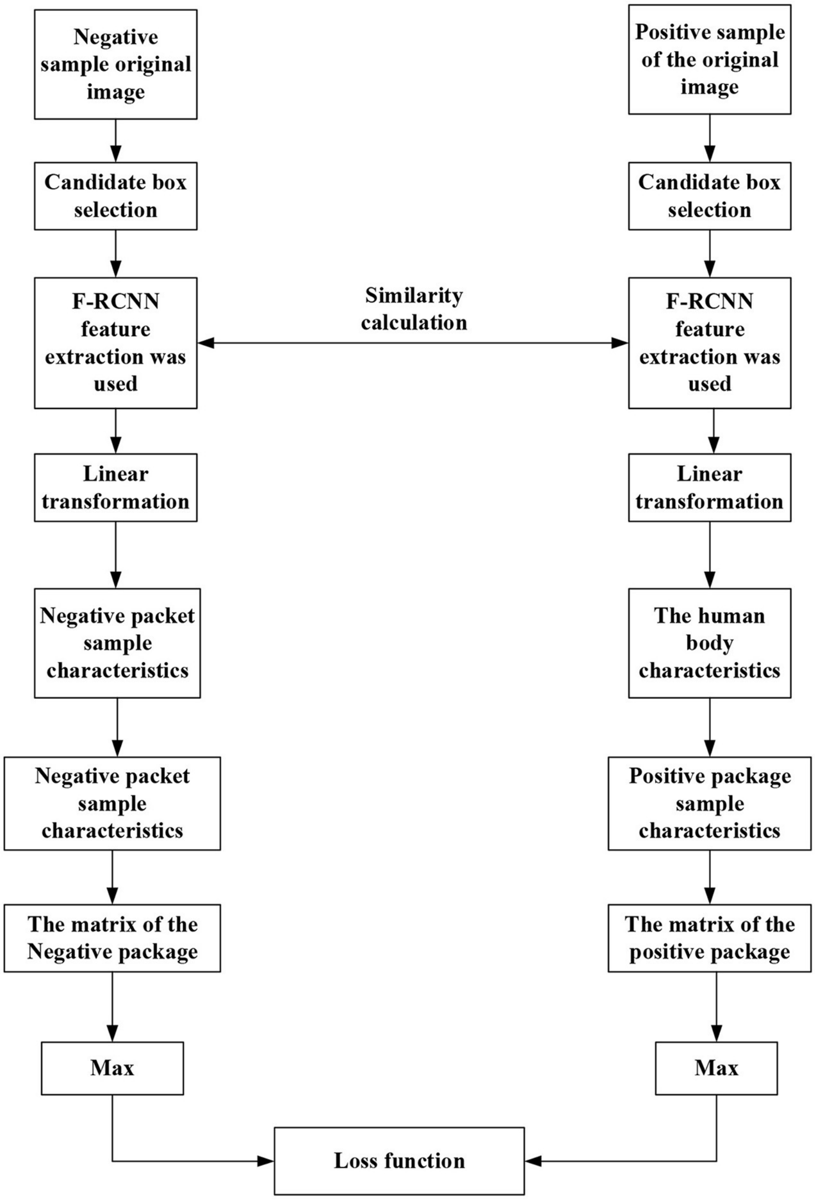

Negative sample original image
Candidate box selection
F-RCNN feature extraction was used
Linear transformation
Negative packet sample characteristics
Negative packet sample characteristics
The matrix of the Negative package
Max
Similarity calculation
Positive sample of the original image
Candidate box selection
F-RCNN feature extraction was used
Linear transformation
The human body characteristics
Positive package sample characteristics
The matrix of the positive package
Max
Loss function

FIGURE 12 | Target detection frame of algorithm.

the machine which pose to estimate. However, an appropriate reward function can make the ML model learn the corresponding strategies under the guidance of the reward function. The goal of RL is to study how to learn a behavior strategy to maximize the cumulative reward in interaction with the environment. In short, RL keeps trying in the training process, deducting points if the decisions are wrong and reward points if otherwise to get the best decision in each state environment.

The fourth step is to map human features into the feature space of the athlete to measure the similarity between human features and athlete features. First, the video is segmented into frames and then the general target detector based on FR-CNN is used to detect the people in each frame. Afterward, the detection results of the athletes are distinguished by the trained example feature measurement model. Then, the multi-frame detection result fusion strategy based on video time-domain context is used to optimize the detection results of each frame. Further, according to the detection frame of the athlete, the image is enlarged, cut to a certain extent, and sent to the single-person pose detector. Finally, the detection results of the single HPE module are mapped back to the original image to obtain the complete athlete-oriented HPE of each frame.

Fifth, to effectively distinguish whether there is inter-frame mutation and local mutation, it is necessary to calculate the inter-frame similarity and local similarity and set the threshold to distinguish. Meanwhile, to calculate the similarity of images, there is a need to extract the image features. Specifically, this paper adopts the Haar-like feature method to compare positions of athletes between multiple frames. The threshold is set to distinguish athletes from non-athletes, and the detection results are optimized based on the video time-domain context information.

As in **Figure 9**, image frames with "someone" are classified as positive packets and frames without "someone" as negative packets. The detection frame extracts the image features that are then transformed into the same domain space. The maximum feature similarity is taken as the positive packet similarity. Following a similar procedure, the negative packet similarity is determined. **Figure 9** draws the target detection frame of the algorithm (Ma et al., 2019; Afrasiabi et al., 2021).

Description of Experimental Data Sets

KTH dataset was released in 2004, which contains six types of human behaviors: walking, jogging, running, boxing, waving, and applause. Each type of behavior is performed by 25 people in four different scenarios (outdoors, outdoors with scale changes, outdoors with clothing changes, and indoors) multiple times, fixed with cameras. The database has a total of 2,391 video samples with 25 Frames Per Second (FPS), a 160 × 120 resolution, and a 4-s average length, as portrayed in **Figure 10**.

The UCF Sports action library is mainly composed of 13 conventional sports actions. This data set consists of a series of actions collected from various sports activities, which are usually broadcast on TV channels, such as British Broadcasting Corporation (BBC) and Entertainment and Sports Programming Network (ESPN) (Kumar et al., 2021). These video sequences are obtained from various material websites, as outlined in **Figure 11**. **Figure 12** shows the target frame detection algorithm.

Preparation of Simulation Experiment

The single-stage-based YOLOv3 detection algorithm model, two-stage-based FR-CNN, Anchor-free, CornerNet, and Merge Soring algorithm are selected as the control group. The model recognition is tested on the MSCOCO dataset, and the largest feature map is selected for FE. Finally, experiments are carried out using the extracted football video and volleyball training video data.

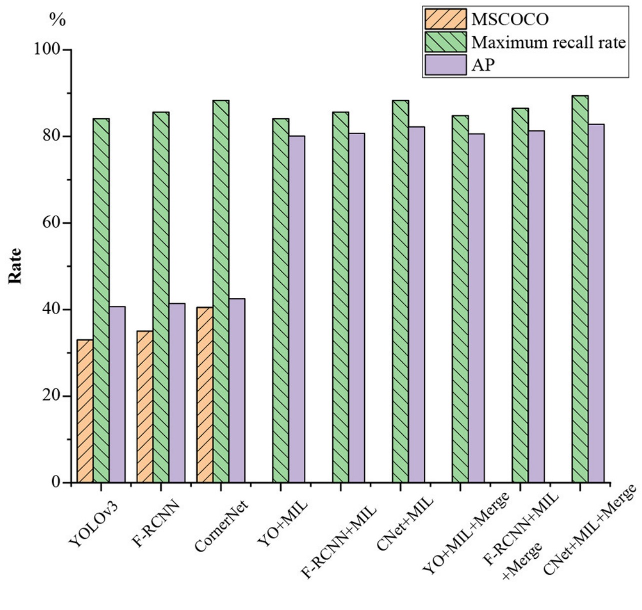


FIGURE 13 | Football video detection results.

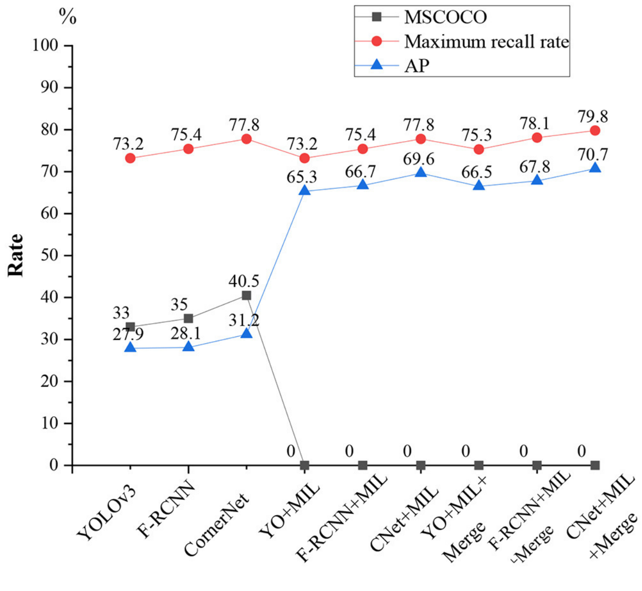


FIGURE 14 | Volleyball video detection results.

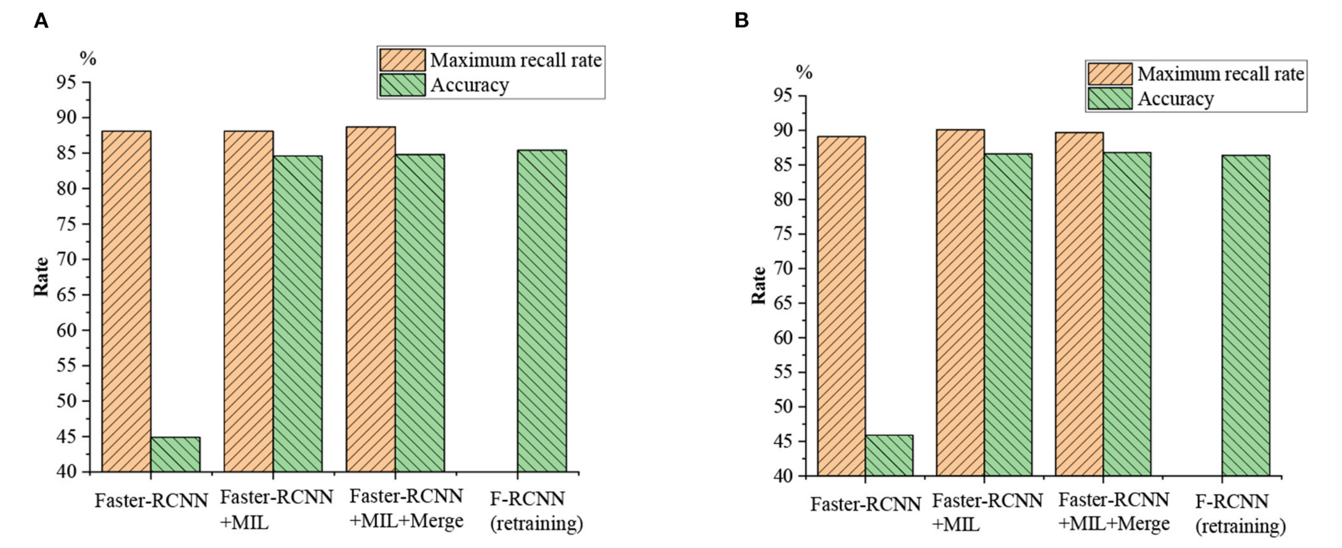


FIGURE 15 | Comparative experiment of mixed sports video recognition. **(A)** Retraining rate as 0.01; **(B)** retraining rate as 0.001.

TABLE 2 | Performance comparison of the same Human Pose Estimation (HPE) algorithm.

Model name	Deep network computing time	Joint point matching time	Athlete pose matching time	Complete test time	Accuracy
FR-CNN	147 ms		10 ms	157 ms	86.1%
Alphapose	158 ms			203 ms	78.3%
Pifpaf	258 ms	102 ms	10 ms	443 ms	79.5%
Local Pifpaf	213 ms	302 ms		294 ms	79.6%

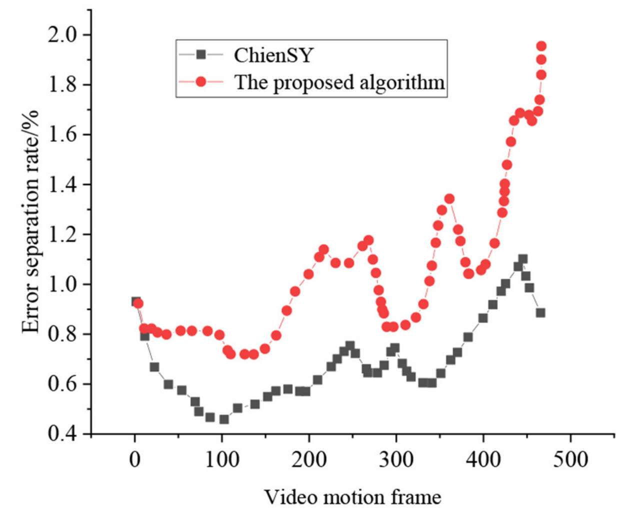


FIGURE 16 | Test results of false separation rate.

Experiment 1: Different diving training videos of a sports college are collected as the experimental objects, with a resolution of 342 × 2,200.

Experiment 2: The database uses the KTH action library and UCF Sports action library. The experiment uses the proposed method to identify the confusion matrix of the two experimental action libraries. The KTH library trains the models using 35 athletes' sports training videos, tests the effect of the model on one person for 36 rounds of cross-detection, and averages the accuracy of pose recognition. Each type of video in the UCF Sports database is randomly selected as a testing video. The remaining video is used as a training video for 50 rounds of cross-validation. The pose recognition accuracy is averaged.

Experiment 3: The proposed performance of the model is compared to the top-down athlete-oriented HPE model based on AlphaPose, the bottom-up athlete-oriented HPE model based on PifPaf, and the athlete-oriented HPE model based on PifPaf with local spatial constraints. Then, 300 football and volleyball training video frames are extracted to generate a training set with 200 pictures and a test set with 100 pictures, respectively. Altogether, 600 non-athlete images are extracted and clustered into a training set that contains 400 images and a testing set that contains 200 images (Wang et al., 2021).

EXPERIMENTAL DESIGN AND PERFORMANCE EVALUATION

The Effectiveness of Feature Metrics for Detection Results

The proposed validity of the algorithm is tested on the football and volleyball video datasets through comparative analysis with YOLOv3, F-RCNN, CornerNet, YOLOv3 + the proposed

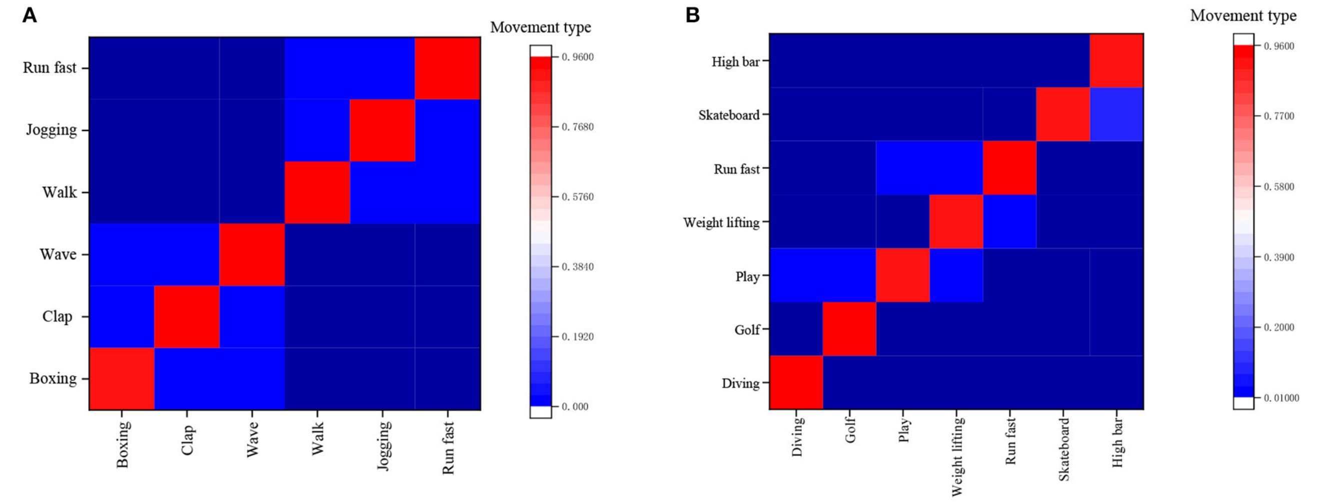


FIGURE 17 | Recognition results of the confusion matrix. **(A)** Results of KTH confusion matrix, **(B)** Results of UCF Sports confusion matrix.

algorithm, F-RCNN + the proposed algorithm, CornerNet + the proposed algorithm, YOLOv3 + the proposed algorithm + Merge, and CornerNet + the proposed algorithm + Merge. **Figures 13**, **14** plot the recognition accuracy, maximum recall, and accuracy of several models on the MSCOCO dataset.

As in **Figures 13**, **14**, although there are differences in the effectiveness of various models on different motion data sets, the differences are relatively low in HPE accuracy, and the maximum recall differences are the lowest. Additionally, the maximum recall of individual YOLOv3, FR-CNN, the Anchor-free, and CornerNet algorithms is normal. However, the average accuracy on MSCOCO and AP is only 20–40%, so using a general detection algorithm alone for sports video detection and recognition is not feasible. However, the average accuracy is greatly improved when the feature metric detection method is added to the general model; the maximum recall and average accuracy are both improved to over 65%. The maximum recall of football videos is over 80%, while the maximum recall is generally over 70% for volleyball videos.

Performance Comparison Experiment

Different kinds of mixed sports videos are used for identification and contrast experiments, and the results are displayed in **Figure 15**. The performance of the proposed athlete-oriented HPE model is compared with that of the mainstream HPE model, as plotted in **Table 2**.

Figure 15 suggests that the FR-CNN model algorithm shows higher recognition accuracy on mixed sports videos than on a single sports video dataset, which may be related to the previously established data set. After retraining, the recognition accuracy of the FR-CNN algorithm has increased from <45–85%, with significant improvement. The maximum recall for the other two matching algorithms is not much different from that of a single FR-CNN model algorithm. In comparison, the recognition accuracy of the FR-CNN model algorithm before and after retraining with the mixed sports videos is not much different (from 40% to a bit higher). Therefore, the proposed athlete-oriented HPE algorithm has presented the best maximum recall and accuracy.

Figure 16 signifies that the separation results obtained by the ChienSY method have obvious cavity problems, and the results obtained by the proposed HPE algorithm are complete and transparent. The false separation rate of the proposed method is about 1%, while that of the ChienSY method is about 3.4%. Due to biased parameter interference in the camera static motion model, the error separation rate of the ChienSY method shows a significant increasing trend in the later stage. In contrast, the error separation rate of the proposed algorithm does not show significant changes, indicating that the proposed method can accurately separate sports videos.

Figure 17A gives the identification results of the KTH confusion matrix by the proposed method, and **Figure 17B** presents the identification results of the UCF Sports confusion matrix by the proposed method.

Figure 17A concludes that the proposed method can separate the first and last three types of actions in the KTH database. Still, the recognition accuracy for walking, jogging, and fast running is relatively low, mainly because these three human motions are hard to discern from one another in terms of the involved body joints' motion amplitude. **Figure 17B** implies that it is basically the same as the recognition accuracy of various actions in the confusion table of the UCF Sports database. It is low of the variance of recognition accuracy of various actions in the database. **Table 2** corroborates that the athlete-oriented HPE model's estimation time is mostly wasted on the NN, while no time has been spent on some modules. This is because the sensitivity of the research experiment is millisecond (ms), and the time span <1 ms is all denoted as 0 ms. In conclusion, the proposed FR-CNN algorithm has shown the fastest detection speed, as small as 157 ms, and its accuracy is much higher than the other three mainstream models, reaching 86.1%.

CONCLUSION

Combined with BDT and DCNN algorithm, this study proposes an athlete-oriented FR-CNN-based HPE model to automatically recognize human actions and sports types from sports videos and finally realize sports video classification and hot spot detection. Concretely, BDT is mainly used to search the training set and testing set of sports video data. Then, the HPE model is realized by the DCNN algorithm, and the feature recognition algorithm is established to recognize the motion of the athlete. Finally, the proposed HPE model is trained with mixed motion videos. Final experiments demonstrate that by combining BDT with DCNN algorithms, the proposed HPE model effectively recognizes human motions in sports videos and can score accurately.

Although the expected objectives have been achieved, there are still some limitations. The compatibility of the model is insufficient, so future research work will focus on further improving the compatibility of the model and optimizing the code. (2) Due to the inability to obtain complete television and network video ratings and program information, the model may not accurately identify complex motion movements. Therefore, future research work will focus on further collecting relevant data, constructing reasonable data sets, understanding the actual situation and collecting data, and constructing data sets to optimize the model.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

ETHICS STATEMENT

The studies involving human participants were reviewed and approved by Beijing Sport University Ethics Committee. The patients/participants provided their written informed consent to participate in this study. Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

REFERENCES

Abid, M., Khabou, A., Ouakrim, Y., Watel, H., Chemcki, S., and Mitiche, A. (2021). Physical activity recognition based on a parallel approach for an ensemble of machine learning and deep learning classifiers. *IEEE Access.* 6, 223–229. doi: 10.3390/s21144713

Abu Hasan, M. D. B., Ahmad, Z. A. B., Leong, M. S., and Hee, L. M. A. (2020). Removal technique using stochastic subspace-based image feature extraction. *J. Imag.* 6, 10–11. doi: 10.3390/jimaging6030010

Afrasiabi, S., Mohammadi, M., Afrasiabi, M., and Parang, B. (2021). Modulated Gabor filter based deep convolutional network for electrical motor bearing fault classification and diagnosis. *Sci, Meas Technol.* 15, 1–4. doi: 10.1049/smt2.12017

Budiman, F., and Sugiarto, E. (2020). Image feature extraction of numbers and letters using matrix segmentation. *Scie. Visualizat.* 12, 96–101. doi: 10.26583/sv.12.1.11

Chavez-Badiola, A., Flores-Saiffe Farias, A., Mendizabal-Ruiz, G., Garcia-Sanchez, R., Drakeley, A. J., and Garcia-Sandoval, J. P. (2020). Predicting pregnancy test results after embryo transfer by image feature extraction and analysis using machine learning. *Sci. Rep.* 10, 4394–4395. doi: 10.1038/s41598-020-61357-9

Chen, Y., Hu, S., Mao, H., Deng, W., and Gao, X. (2020). Application of the best evacuation model of deep learning in the design of public structures. *Int. Visual Comp.* 102, 975–977. doi: 10.1016/j.imavis.2020.103975

Churchill, R. M., Tobias, B., and Zhu, Y. (2020). Deep convolutional neural networks for multi-scale time-series classification and application to tokamak disruption prediction using raw, high temporal resolution diagnostic data. *Phys. Plasma.* 27, 062510. doi: 10.1063/1.5144458

Cocos, A., and Fiks, A. G. (2019). Reply to comment on: Deep learning for pharmacovigilance: recurrent neural network architectures for labeling adverse drug reactions in Twitter posts. *J. Am. Med. Inform. Assoc.* 172, 1122–1131. doi: 10.1093/jamia/ocy192

Desai, P., Pujari, J., and Sujatha, C. (2021). Impact of multi-feature extraction on image retrieval and classification using machine learning technique. *SN Computer Sci.* 2, 69–71. doi: 10.1007/s42979-021-00532-9

Gururaj, C., and Tunga, S. (2020). AI based feature extraction through content based image retrieval. *J. Comput. Theor. Nanosci*. 2, 45–47. doi: 10.1166/jctn.2020.9018

Jamali, A., Mahdianpari, M., Brisco, B., Granger, J., Mohammadimanesh, F., and Salehi, B. (2021). Wetland mapping using multi-spectral satellite imagery and deep convolutional neural networks: a case study in newfoundland and labrador, Canada. *Can. J. Remote. Sens.* 2, 1–18. doi: 10.1080/07038992.2021.1901562

Khaleefah, S. H., Mostafa, S. A., Mustapha, A., and Nasrudin, M. F. (2020). Review of local binary pattern operators in image feature extraction. *Indones. J. Electr. Eng. Comput. Sci.* 19, 23–26. doi: 10.11591/ijeecs.v19.i1.pp23-31

Kumar, B. S., Raju, S. V., and Reddy, H. V. (2021). Human action recognition using novel deep learning approach. *Mater. Sci. Eng.* 1042, 12031–12039. doi: 10.1088/1757-899X/1042/1/012031

Lv, Z. H., and Liang, Q. (2020). Deep belief network and linear perceptron based cognitive computing for collaborative robots. *Appl. Soft Comput.* 10, 300. doi: 10.1016/j.asoc.2020.106300

Ma, S., Cai, W., Liu, W., Shang, Z., and Liu, G. (2019). A lighted deep convolutional neural network based fault diagnosis of rotating machinery. *Sensors.* 19, 169–173. doi: 10.3390/s19102381

Maiorino, A., Del Duca, M. G., Tomc, U., Tušek, J., Kitanovski, A., and Aprea, C. (2021). A numerical modelling of a multi-layer LaFeCoSi Active Magnetic Regenerator by using Artificial Neural Networks. *Appl. Therm. Eng.* 197, 117375. doi: 10.1016/j.applthermaleng.2021.117375

Minhas, R. A., Javed, A., Irtaza, A., Mahmood, M. T., and Joo, Y. B. (2019). Shot classification of field sports videos using alexnet convolutional neural network. *Appl. Sci.* 9, 342–346. doi: 10.3390/app9030483

Opotowsky, A. R., and Webb, G. D. S. (2020). Small effects, big data, big anxiety: our evolving understanding of low-risk congenital heart disease. *Circ. Cardiovasc. Interv*. 13, 79–82. doi: 10.1161/CIRCINTERVENTIONS.120.009350

Rajesh, B., and Muralidhara, B. L. (2021). Image inpainting via generative multi-column with the aid of deep convolutional neural networks. *Base University Working Papers* 4, 425–426.

Ramesh, M., and Mahesh, K. A. P. (2020). A performance analysis of pre-trained neural network and design of CNN for sports video classification. *2020 International Conference on Communication and Signal Processing (ICCSP)*. 6, 28–30. doi: 10.1109/ICCSP48568.2020.9182113

Romanowski, A. (2019). Big data-driven contextual processing methods for electrical capacitance tomography. *Trans. Ind. Inform.* 3, 1609–1618. doi: 10.1109/TII.2018.2855200

Saad, E., and Hirakawa, K. (2020). Improved photometric acceptance testing in image feature extraction tasks. *J. Electron. Imag.* 29, 1–4. doi: 10.1117/1.JEI.29.4.043012

Saju, C. J., and Ravimaran, S. A. (2020). A tool to extract name entity recognition from big data in banking sectors. *Int. J. Web Serv.* 17, 18–39. doi: 10.4018/IJWSR.2020040102

Sandula, P., Kolanu, H. R., and Okade, M. (2021). CNN-based camera motion classification using HSI color model for compressed videos. *Signal Image Video Process.* 14, 12–14. doi: 10.1007/s11760-021-01964-9

Shorten, C., and Khoshgoftaar, K. M. (2019). Survey on image data augmentation for deep learning. *J. Big Data Vol.* 6, 60–63. doi: 10.1186/s40537-019-0197-0

Singh, A., Garg, S., Kaur, K., Batra, S., and Kumar, N. (2019). Fuzzy-folded bloom filter-as-a-service for big data storage in the cloud. *Trans. Ind. Inform.* 15, 2338–2348. doi: 10.1109/TII.2018.2850053

Singh, P., Mazumder, P., and Namboodiri, V. P. (2021). Context extraction module for deep convolutional neural networks. *Pattern Recognit.* 1, 284–293. doi: 10.1016/j.patcog.2021.108284

Sugiarto, B., Gojali, E. A., Herlan, H., and Lestari, P. A. (2020). A smartphone-based of wood identification using image feature extraction. *J. Comput. Sci. Eng.* 4, 9–15. doi: 10.29303/jcosine.v4i1.311

Sulam, J., Aberdam, A., Beck, A., and Elad, M. (2019). On multi-layer basis pursuit, efficient algorithms and convolutional neural networks. *Trans. Pattern Anal. Mach. Intell.* 26, 141–144. doi: 10.1109/TPAMI.2019.2904255

Tarsitano, F., Bruderer, C., Schawinski, K., and Hartley, W. G. (2021). Image feature extraction and galaxy classification: a novel and efficient approach with automated machine learning. *Mon. Notices Royal Astron. Soc.* 4, 945–949. doi: 10.1093/mnras/stac233

Tolan, N. V., Parnas, M. L., Baudhuin, L. M., Cervinski, M. A., Chan, A. S., and Holmes, D. T. (2020). "Big Data" in laboratory medicine. *Clini. Chem.* 2, 452–453. doi: 10.1373/clinchem.2015.248591

Wang, Y., Liang, Z., and Cheng, X. (2021). Fast target tracking based on improved deep sort and YOLOv3 fusion algorithm. *Int. Conf. Pioneering Comp. Sci. Engineers and Edu.* 2021, 10. doi: 10.1007/978-981-16-5940-9_27

Wang, Z., Hou, Y., Jiang, K., Zhang, C., Dou, W., Huang, Z., et al. (2019). A survey on human behavior recognition using smartphone-based ultrasonic signal. *IEEE Access.* 99, 1–2. doi: 10.1109/ACCESS.2019.2931088

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 30 May 2022
doi: 10.3389/fnbot.2022.865693

Green Supply Chain Optimization Based on BP Neural Network

*Huan Wang**

College of Economics and Management, Hubei University of Automotive Technology, Shiyan, China

With the emergence and development of the Back Propagation neural network (BPNN), its unique learning, generalization, and non-linear characteristics have been gradually excavated and fully applied in the field of prediction. To improve the economic and green benefits of enterprises, the BPNN algorithm is applied to the green supply chain assisted by intelligent logistics robots. The BPNN algorithm can be used to output the characteristics of different information and optimize the green supply chain according to the input parameters and the influencing factors in the network. Firstly, an evaluation index system is established for selecting suppliers, which includes 4 first-level indicators: operational indicators, economic indicators, green indicators, social indicators, and 14 corresponding secondary indicators. Secondly, the evaluation indicator system is modeled through the BPNN. Finally, using the BPNN model, a supply chain enterprise's selection of cooperative enterprises in Xi'an is taken as the research object and simulation. Finally, the output results of the five alternative enterprises are 0.77, 0.75, 0.68, 0.72, and 0.65, respectively. The enterprise with the highest output results is selected as the cooperative enterprise and the enterprise with the second highest output results as an alternate. The green supply chain model based on the proposed BPNN is scientific and effective through specific simulation experiments. It has certain reference significance for the relevant issues related to subsequent optimization of the green supply chain.

Keywords: Back Propagation neural network algorithm, network model, green supply chain, intelligent logistics robot, artificial intelligence

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
Cherian Samuel,
Indian Institute of Technology (BHU), India
Wei-Che Chien,
National Dong Hwa University, Taiwan

****Correspondence:***
Huan Wang
wangh@huat.edu.cn

Received: *30 January 2022*
Accepted: *25 April 2022*
Published: *30 May 2022*

Citation:
Wang H (2022) Green Supply Chain Optimization Based on BP Neural Network.
Front. Neurorobot. 16:865693.
doi: 10.3389/fnbot.2022.865693

INTRODUCTION

With the rapid development of technology and the continuous progress of society, as an important driving force for a new round of scientific and technological revolution and industrial transformation, artificial intelligence (AI) has become an extremely important technological content at present. Under the background of made in China 2025 and the continuous transformation and upgrading of technology in the manufacturing field, the production level and the application efficiency of China's manufacturing have been greatly improved. The widespread use of AI technology has not only brought revolutionary changes to traditional manufacturing, but also affected the transition of modern manufacturing to a smarter, more modular trend (Brito et al., 2020). With the development of AI technology, the robot industry has also developed from programmable robots and sensory robots to intelligent robots, which has become a trend of development at present and in the future (Staal et al., 2020). With the application of intelligent robots in the logistics industry, the operation efficiency of the logistics system has been greatly advanced. Intelligent logistics robots have gradually demonstrated their advantages of speed and convenience in logistics, warehousing, transportation and other fields. While technology and the economy are developing rapidly, "green development" has also become a hot topic in society, and all walks of life are transforming to green.

Ahmet and Alk (2020) believed that supply chain management (SCM) was an important part of reducing costs and increasing profits for most companies in a competitive enterprises environment, and the success of enterprises is directly related to the performance of the supply chain. The introduction of the green SCM theory also provides a new way of thinking for the logistics industry to realize green transformation, so that enterprises can choose the right partners and jointly complete the green transformation. While ensuring the economic benefits of enterprises, they also harvest green benefits. Shao et al. (2019) considered that the idea of the green supply chain could provide a decision-making basis for the bidding and procurement process, and proposed that the green supply chain mainly included three levels: external environment, corporate strategy, and inventory level (Shao et al., 2019). At present, the domestic green supply chain is still in its infancy, and there are fewer studies on the optimization of the existing green supply chain. With the emergence and development of the Back Propagation neural network (BPNN), its unique learning, generalization, and non-linear characteristics have been gradually excavated and fully applied in the field of prediction. The BPNN algorithm can output the characteristics of differentiated information according to different input parameters and various influencing factors in the network, and optimize the green supply chain of the logistics industry. The innovation of this research is to use the predictability of BPNN to predict the choice of different supply chain enterprises, thus achieving the optimal choice.

To improve the economic and green benefits of enterprises, the BPNN algorithm is applied to the green supply chain. Firstly, an evaluation index system is established for selecting suppliers. Secondly, the system is modeled through BPNN. Finally, the BPNN model is used to simulate the selection of partners by a supply chain enterprise in Xi'an, which verifies that the proposed index system model is scientific and effective.

THE MODEL OF GREEN SUPPLY CHAIN USING THE BPNN

Construction of Evaluation Index System of Green Supply Chain

The concept of supply chain first appeared in manufacturing, which refers to the whole process of manufacturing enterprises purchasing raw materials, producing and manufacturing, supplying to distributors, transferring to retailers, and finally to consumers. With the gradual deepening of the research on manufacturing operation mode, the academic community combines supply chain with supply management and expresses it as a relationship between suppliers and manufacturing enterprises. Nowadays, manufacturers, supply enterprises, transportation enterprises, distributors, information flows and consumers are all included in the supply chain. In the whole process of SCM, the green supply chain relies on its advanced scientific concepts and technical means, takes economic benefits, social benefits and environmental protection benefits as management goals, and carries out high-efficiency and low-cost overall control of logistics, fund flow and information flow of the entire supply chain (Gao et al., 2021).

The green supply chain is a closed-loop system with four main features: (1) The main goal of traditional SCM is to improve the profitability of enterprises in the supply chain, and the green supply chain not only aims to improve enterprise efficiency, but also adds two new goals: social and environmental benefits. In practice, the three benefits may be contradictory, and the realization of one benefit improvement will often lead to the decline of the other benefit. Green SCM is to comprehensively consider the three benefits and make the three develop together. (2) Traditional supply chain is the product of enterprise organization, and the main body of the green supply chain is more complex. Government intervention, preferences of consumer groups, etc. will have an impact on it. Due to the complexity of the main body in the supply chain, its behavioral goals are also more diversified, thus promoting the common development of multiple benefits. (3) The life cycle of products in the traditional supply chain includes design, raw material procurement, product manufacturing, sales, logistics and after-sales service, while the green supply chain also includes the recovery of waste materials to form a closed loop to minimize energy consumption and reduce environmental hazards. (4) The main factor of traditional SCM is the relationship between subjects, but the green supply chain also includes technical requirements in addition to the relationship between subjects (Cui et al., 2021).

The formation of the green supply chain can be divided into internal and external aspects. The specific principle of formation is shown in **Figure 1**.

In **Figure 1**, the internal factors that make up the green supply chain include internal driving drive and automatic force. The internal driving force includes economic benefits, social benefits and environmental benefits. The green supply chain improves economic benefits by reducing costs, reducing waste, promoting the market, and improving brand benefits, improving social benefits by improving product security, saving energy and increasing employment, and improving environmental benefits through improving the efficiency of the use of material resources and controlling the entry of highly polluting substances into the supply chain. The automatic forces include self-organization, self-evolved and self-coordinated. The subjects involved in the green supply chain share resources, restrict each other, and also promote each other. When a subject in the supply chain behaves abnormally, the supply chain will adjust itself in time (Fallahpour et al., 2020; Meager et al., 2020).

The external factors that constitute a green supply chain include security force and external pressure. The security force includes technological innovation, institutional innovation, and industrial structure adjustment. Through correct guidance and appropriate incentives, the government promotes the green upgrading of the supply chain, establishes effective

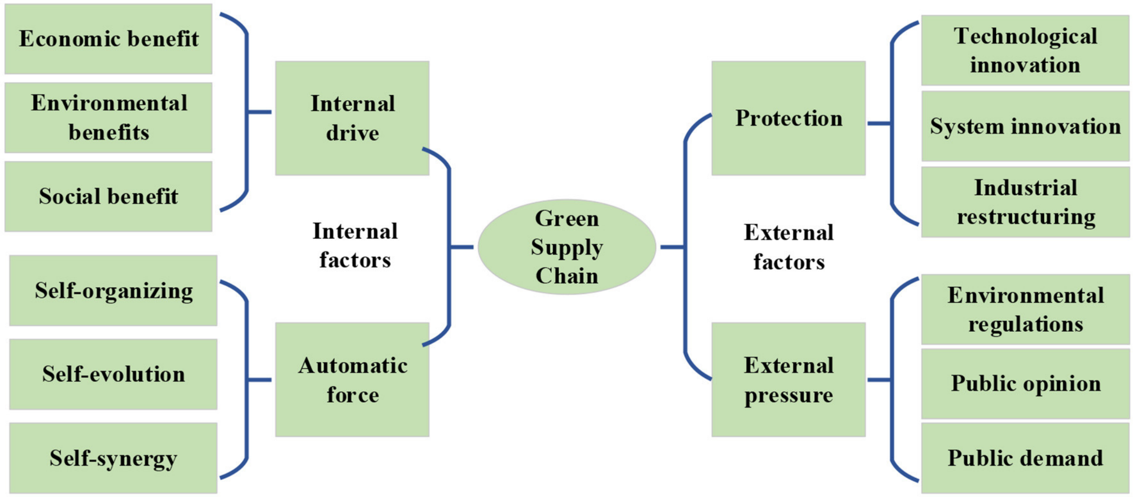


FIGURE 1 | The formation principle of the green supply chain.

environmental regulations, builds standardized incentive mechanisms, guarantees the due benefits of technological innovation of enterprises, makes macro-adjustments to the industrial structure of enterprises, and formulates appropriate industry development plans to achieve the green upgrade of the supply chain (Midya et al., 2021). The external pressure includes environmental regulations and standards, public opinion orientation and public demand. The introduction of relevant regulations and standards can promote the green upgrade of the supply chain, and public opinions on environmental pollution and the use of toxic and hazardous materials can also improve the environmental protection concepts of corporate managers and consumers. With the improvement of public consumption capacity, consumers will also buy products for the green premium of products, and they will be more inclined to choose corporate products with a strong sense of social responsibility, which also ensures the demand for products in the green supply chain (Khan et al., 2021).

According to the characteristics and formation principle of the green supply chain, the indicator system of the green supply chain is set. The operational indicators, economic indicators, green indicators and social indicators are taken as the first-level indicators of the indicator system of the green supply chain. Among them, operational indicators include five secondary indicators, namely delivery advance, response speed, delivery accuracy, production flexibility and order completion rate. Economic indicators include four secondary indicators, and they are cost reduction rate, destocking level, sales profit margin, and capital turnover rate, respectively. Green indicators include two secondary indicators, namely energy utilization rate and waste resource recovery rate, respectively. Social indicators include three secondary indicators, namely social welfare investment, enterprise reputation level, and employee and customer satisfaction. The specific content is shown in **Figure 2**.

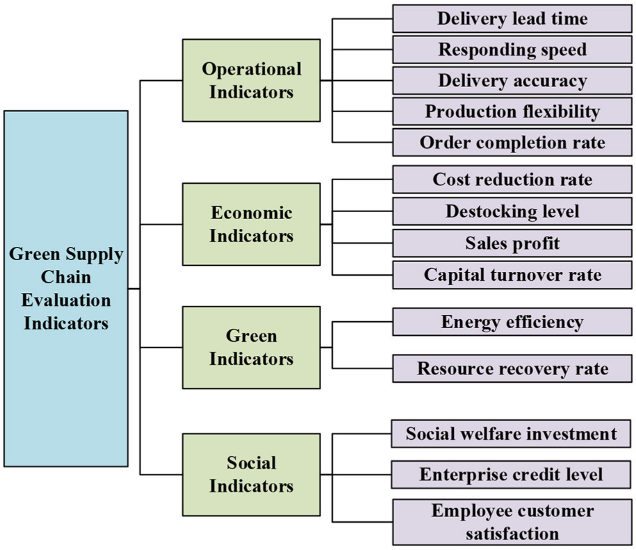


FIGURE 2 | The evaluation indicator system of the green supply chain.

In **Figure 2**, the first-level indicators of the evaluation indicator system of the green supply chain are four categories: operational indicators, economic indicators, environmental indicators and social indicators, and the corresponding secondary indicators are a total of 14 categories. Each indicator is independent and does not overlap with each other, but it affects, restricts and interacts with each other. By applying these indicators to BPNN, a green supply chain evaluation model based on BPNN can be constructed.

The BPNN Model

For building a green supply chain evaluation model based on BPNN, it is necessary to understand the meaning of BPNN. The concept of the BPNN was first proposed by a group of scientists led by Rumelhart and McClelland in 1986, and it was widely used. The BPNN includes the input layer, the output layer, and the hidden layer. Neurons in the same layer are not connected to each other, and only neighboring upper and lower neurons can connect to each other (Wu et al., 2019). The BPNN can generate different output information by inputting different input data to meet all training sets as much as possible (Tang and Yu, 2021). The structure is shown in **Figure 3**.

In the BPNN, the hidden layer may be only one layer or multiple layers, each layer may have one or more neurons. But the theory has proved that the BPNN with only one layer of the hidden layer can approximate arbitrary non-linear continuous functions. So in most cases, the BPNN with only one layer of the hidden layer is used. In the input layer, the number of neurons is the same as the number of input parameters, and the number of neurons in the output layer is the same as the number of output parameters (Zhou et al., 2019, 2020).

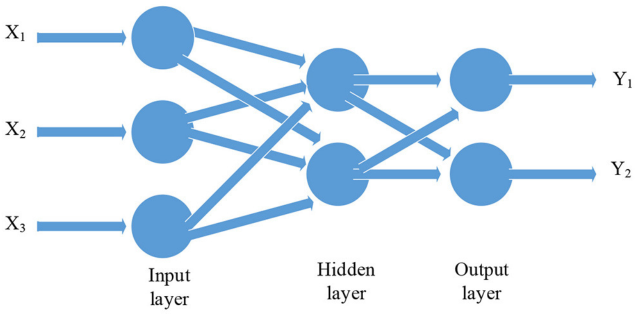


FIGURE 3 | The structure of the BPNN.

Generally speaking, the excitation function between the hidden layer and the output layer is the Sigmoid function, which includes two forms, namely the Log-Sigmoid function and the Tan-Sigmoid function. The expression of the Log-Sigmoid function is shown in Equation (1):

$$f(x) = \frac{1}{1+e^{-x}} \tag{1}$$

The expression of the Tan-Sigmoid function is shown in Equation (2):

$$f(x) = \frac{1-e^{-x}}{1+e^{-x}} \tag{2}$$

As the most frequently applied excitation function in the BPNN, the Sigmoid function "extrudes" the input of the upper function and outputs it into a range, thereby completing arbitrary non-linear mapping from input to output (Sakaki et al., 2019).

Neural networks (NNs) also have the function of autonomous learning, and their learning methods can be divided into supervised learning and unsupervised learning. Supervised learning can make the output value of each group of NNs as close as possible to the actual output value. Because the BPNN trains the NN with the input and output of the training set, the number of hidden layers, learning rate, threshold and weights of neurons can be adjusted. Unsupervised learning can only adjust the weights between neurons, and there is no actual output value, so it is impossible to compare the output of the NN with the actual output, so it is impossible to adjust the parameters.

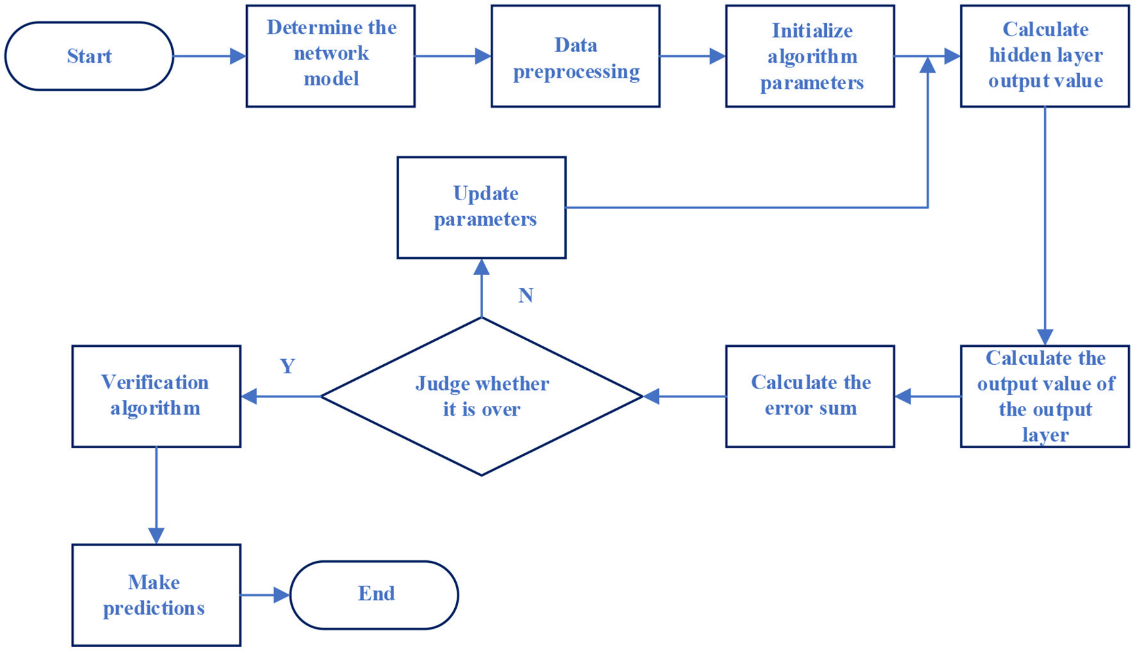

Start
Determine the network model
Data preprocessing
Initialize algorithm parameters
Calculate hidden layer output value
Update parameters
N
Y
Verification algorithm
Judge whether it is over
Calculate the error sum
Calculate the output value of the output layer
Make predictions
End

FIGURE 4 | The operation flow of the BPNN model.

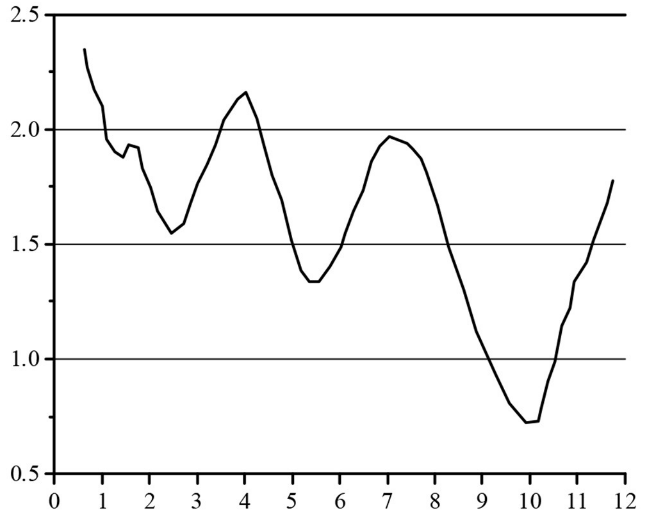


FIGURE 5 | The function curve of the sum of errors.

When using the model of the BPNN, there are the following steps: first, the output information is calculated according to the input information, then the threshold and weight are reversely updated between the two adjacent levels according to the error between the actual output and the expected output. When learning, the standard BPNN algorithm uses the descent method of the error function. When the sum of squared errors between the actual output and the expected output is minimized, the learning process ends (Peng et al., 2018; Hirschfeld et al., 2020; Carvalho and Plastino, 2021).

Because the BPNN with only one layer of the hidden layer can approximate arbitrary non-linear continuous functions, resulting in most NN models that only use one layer of the hidden layer, so this time NN models with only one layer of the hidden are also used as examples. a is the number of input layer neurons, b is the number of output layer neurons, and c is the number of hidden layer neurons. Y_{mn} is the weight of the mth input layer neurons to the nth hidden layer neurons. Q_{sn} is the weight of the s-th hidden layer neurons to the nth output layer neurons. x_{m1} to x_{ma} is an input parameter of mth group data. y_{m1} to y_{mb} is the b output parameter of mth group data. Y_{n0} is the threshold of the nth hidden layer neuron. Q_{s0} is the threshold of the sth output neuron. The application process of the BPNN model is shown in the equation.

Step 1: The number of neurons is determined. The number of neurons in the input layer is the same as the number of input values. The number of neurons in the output layer is the same as the number of output parameters. The number of hidden layer neurons is usually estimated by the experience formula, and then the number is determined by temptation. Two experience formulas are shown in Equations (3) and (4):

$$p = \sqrt{a+b} + \varepsilon \quad (0 \leq \varepsilon \leq 10) \tag{3}$$

$$p = \log_2 a \tag{4}$$

Because the secondary index of the chain evaluation index of this green supply is 14, the number of neurons in the input layer is 14, the number of neurons in the output layer is 1, and the value is 0–10 in turn for ε. The speed of convergence of the NN processing data is tested, and the number of hidden layers is 8. The model of the BPNN is finally set to 14*8*1.

Step 2: Data pre-processing. Firstly, the input and output of the BPNN are determined, secondly, the corresponding training set is built, and finally, the processing of the data is normalized. Generally, the data is normalized into intervals [0,1] or intervals [−1,1], as shown in Equation (5) to (7):

$$\mathrm{x_{new}} = \frac{\mathrm{x - x_{min}}}{x_{\mathrm{max}} - \mathrm{x_{min}}} \tag{5}$$

$$\mathrm{x_{mid}} = \frac{\mathrm{x_{max} - x_{min}}}{2} \tag{6}$$

$$\mathrm{x_{new}} = \frac{\mathrm{x - x_{mid}}}{0.5\,(\mathrm{x_{max} - x_{min}})} \tag{7}$$

Equation (5) is to normalize the data to [0,1], Equations (6) and (7) are to normalize the data to [−1,1].

Step 3: Parameters of the input network model. Input weight Y_{mn}, weight Q_{sn}, learning rate η, thresholds, incentive function, maximum recursive times and setting target error.

Step 4: Calculate the output of the hidden layer. First, the weighted sum of the data is calculated by the input layer in the NN, and then the excitation function is used to "compress it" and finally, the output value of the hidden layer is obtained, as shown in Equation (8):

$$\mathrm{K_m} = \mathrm{f}\left(\sum\nolimits_{\mathrm{k}=1}^{\mathrm{a}} \mathrm{Y_{km}x_k} + \mathrm{Y_{m0}}\right) \tag{8}$$

In the equation, K_{m} is the output value of the mth neuron of the hidden layer. f() is the excitation function, as shown in Equations (1) or (2).

Step 5: Calculate the output value of the output layer neurons, as shown in Equation (9):

$$\mathrm{y_i} = \mathrm{f}\left(\sum\nolimits_{\mathrm{k}=1}^{\mathrm{n}} \mathrm{K_kQ_{km}} + \mathrm{Q_{m0}}\right) \tag{9}$$

Step 6: Calculation error. The output error is calculated according to the output value of the BPNN model and the actual output value. The sum of squares of errors is shown in Equation (10):

$$\mathrm{E} = \sum\nolimits_{\mathrm{m}=0}^{\mathrm{N_t}} \sum\nolimits_{\mathrm{n}=1}^{b} \left[\mathrm{t_n^{(m)}} - \mathrm{y_n^{(m)}}\right]^2 \tag{10}$$

In Equation (10), E represents the sum of squares of errors. N_t indicates the total number of training samples. m shows the mth training sample. n is the nth output neuron, and b means the number of output neurons. $y_n^{(m)}$ is the nth network model output of the mth training sample and $t_n^{(m)}$ is the nth actual output of the mth training sample.

Step 7: Distinguish whether the training is over. Distinguish whether the BPNN model training is over or not by the following situations:

(1) Given error maximum > error E.

TABLE 1 | Questionnaire of self-evaluation indicators of the green supply chain.

No.	Problem description	Totally disagree	Broadly disagree	Largely agree	Broadly agree	Totally agree
1	Customers are very satisfied with the quality of the product					
2	Customers are very satisfied with the price of the product					
3	Customers are very satisfied with the timeliness of delivery					
4	Customers are very satisfied with the accuracy of delivery					
5	The needs of customers are satisfied for different batches and different product combinations					
6	The company always pays attention to the market and responds quickly					
7	The company can successfully solve the temporary increase in orders					
8	The company can meet orders of different batches at any time					
9	The company keeps inventory to a minimum while maintaining customer demand					
10	The company's transportation costs are reduced to a minimum					
11	The company minimizes loss of the product					
12	The company can make accurate forecasts of inventory levels					
13	There is no problem with the turnover of the company's funds					
14	The company's control of sales profits is very strict					
15	The company can rationally use equipment and tools to improve efficiency					
16	The company saves or recycles recyclable resources					
17	The benefits within the company are very good					
18	Employees are very satisfied with the company's salary and benefits					
19	The company has a certain reputation in the society					
20	The company and its partners can achieve mutual benefit and win-win results					

(2) The number of recursions > the set parameter values.

(3) When the training error E is basically unchanged for many consecutive times, it has converged to the minimum value.

If one of the above conditions is met, skip the next step, otherwise, go to the next step.

Step 8: First, adjust the weights and thresholds in the NN, and then jump back to Step 4. The BPNN algorithm is used to learn according to the gradient descent method of error functions. The partial derivatives of E relative thresholds and weights should be calculated first, as shown in Equations (11) to (14):

$$\frac{\partial \mathrm{E}}{\partial \mathrm{Q_{sn}}} = \sum\nolimits_{\mathrm{p}=1}^{N_t} \left[\mathrm{y_s^{(p)}} - \mathrm{t_s^{(p)}}\right] \varphi\left(\sum\nolimits_{\mathrm{m}=1}^{\mathrm{a}} \mathrm{Y_{nm}x_m} + \mathrm{Y_{n0}}\right) \tag{11}$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{Q_{s0}}} = \sum\nolimits_{\mathrm{p}=1}^{\mathrm{N_t}} \left[\mathrm{y_s^{(p)}} - \mathrm{t_s^{(p)}}\right] \tag{12}$$

$$\begin{aligned}\frac{\partial \mathrm{E}}{\partial Y_{\mathrm{nm}}} = &\sum\nolimits_{\mathrm{p}=1}^{\mathrm{N_t}} \sum\nolimits_{\mathrm{s}=1}^{\mathrm{b}} \left[\mathrm{y_s^{(p)}} - \mathrm{t_s^{(p)}}\right] \varphi\left(\sum\nolimits_{\mathrm{m}=1}^{\mathrm{a}} \mathrm{Y_{nm}x_m} + \mathrm{Y_{n0}}\right)\\ &\left[1 - \varphi\left(\sum\nolimits_{\mathrm{m}=1}^{\mathrm{a}} \mathrm{Y_{nm}x_m} + \mathrm{Y_{n0}}\right)\right] x_m\end{aligned} \tag{13}$$

$$\begin{aligned}\frac{\partial E}{\partial \mathrm{Y_{n0}}} = &\sum\nolimits_{\mathrm{p}=1}^{\mathrm{N_t}} \sum\nolimits_{s=1}^{\mathrm{b}} \left[y_s^{(p)} - t_s^{(p)}\right] \varphi\left(\sum\nolimits_{\mathrm{m}=1}^{a} \mathrm{Y_{nm}x_m} + \mathrm{Y_{n0}}\right)\\ &\left[1 - \varphi\left(\sum\nolimits_{\mathrm{m}=1}^{\mathrm{a}} \mathrm{Y_{nm}x_m} + \mathrm{Y_{n0}}\right)\right]\end{aligned} \tag{14}$$

In the equations, E represents the sum of squares of errors. N_t indicates the total number of training samples. m shows the mth training sample. n is the nth output neuron, and b means the number of output neurons. The thresholds Y_{n0}, Q_{s0} and weights Y_{nm}, Q_{sn} are modified, as shown in Equations (15) to (18):

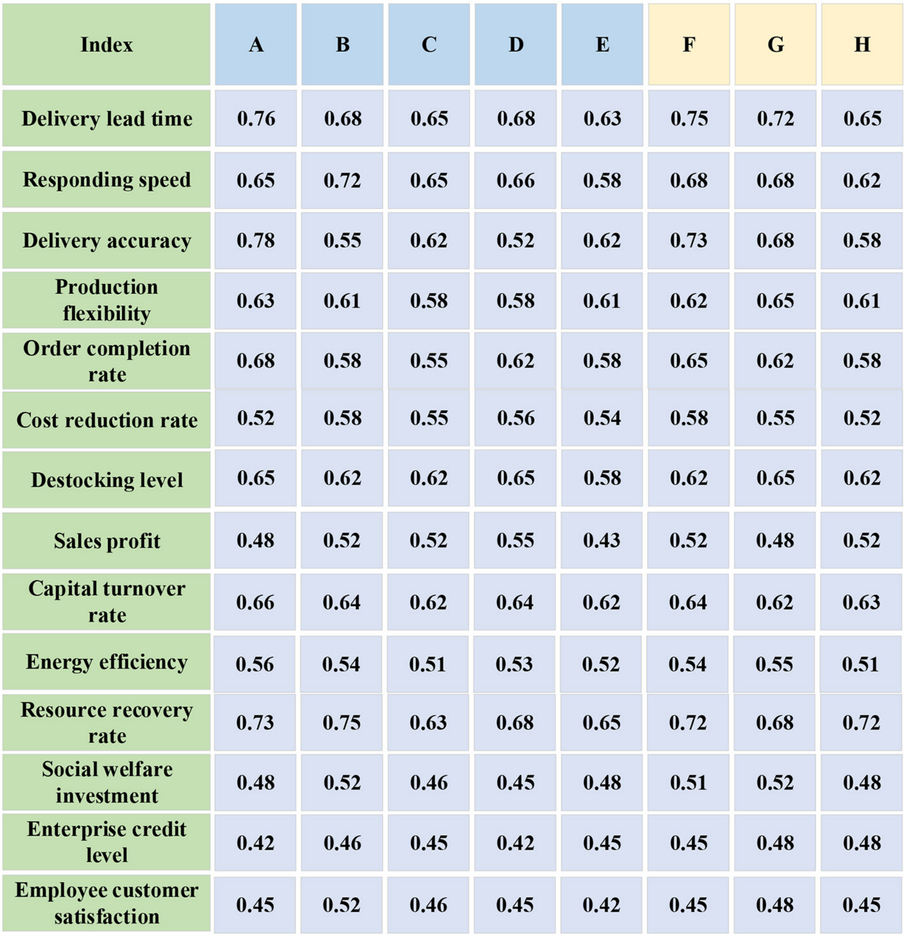
$$\mathrm{Q_{sn}^{new}} = \mathrm{Q_{sn}^{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Q_{sn}}} \tag{15}$$

$$\mathrm{Q_{s0}^{new}} = \mathrm{Q_{s0}^{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Q_{s0}}} \tag{16}$$

$$\mathrm{Y_{nm}^{new}} = \mathrm{Y_{nm}^{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Y_{nm}}} \tag{17}$$

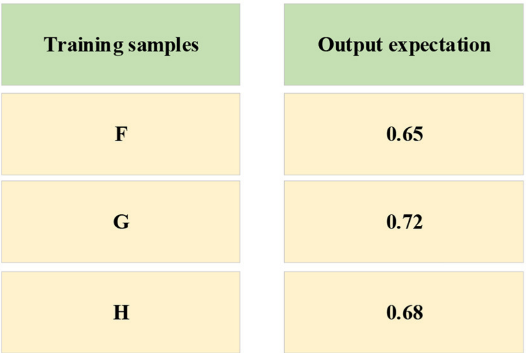
$$\mathrm{Y_{n0}^{new}} = \mathrm{Y_{n0}^{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Y_{n0}}} \tag{18}$$

Step 9: Verify the feasibility of the algorithm. The BPNN model that has been trained is used to calculate the output according to the input of the training set, and then the output value of the NN model with the actual output value are compared and calculated. Finally, the error is calculated and the rationality of its structure is analyzed.



Index	A	B	C	D	E	F	G	H
Delivery lead time	0.76	0.68	0.65	0.68	0.63	0.75	0.72	0.65
Responding speed	0.65	0.72	0.65	0.66	0.58	0.68	0.68	0.62
Delivery accuracy	0.78	0.55	0.62	0.52	0.62	0.73	0.68	0.58
Production flexibility	0.63	0.61	0.58	0.58	0.61	0.62	0.65	0.61
Order completion rate	0.68	0.58	0.55	0.62	0.58	0.65	0.62	0.58
Cost reduction rate	0.52	0.58	0.55	0.56	0.54	0.58	0.55	0.52
Destocking level	0.65	0.62	0.62	0.65	0.58	0.62	0.65	0.62
Sales profit	0.48	0.52	0.52	0.55	0.43	0.52	0.48	0.52
Capital turnover rate	0.66	0.64	0.62	0.64	0.62	0.64	0.62	0.63
Energy efficiency	0.56	0.54	0.51	0.53	0.52	0.54	0.55	0.51
Resource recovery rate	0.73	0.75	0.63	0.68	0.65	0.72	0.68	0.72
Social welfare investment	0.48	0.52	0.46	0.45	0.48	0.51	0.52	0.48
Enterprise credit level	0.42	0.46	0.45	0.42	0.45	0.45	0.48	0.48
Employee customer satisfaction	0.45	0.52	0.46	0.45	0.42	0.45	0.48	0.45

FIGURE 6 | The data of testing and training sample.



Training samples	Output expectation
F	0.65
G	0.72
H	0.68

FIGURE 7 | The output value of training sample.

Step 10: After the algorithm training is over, the completed NN model is used to solve the problem. The operation flow of the BPNN model is shown in **Figure 4**.

The reason why the BPNN model can be used so widely is due to its following characteristics:

(1) Good non-linear mapping ability. In the BPNN model, the threshold of each neuron, the weights between neurons, etc., can be stored as specific information.
(2) Distributed storage information. In the BP neural network model, the threshold of each neuron, the weights between neurons, etc. can be expressed as stored as specific information.
(3) Autonomous learning ability. The BPNN model will calculate the output according to the input of the training set, and then compare it with the actual output to reversely update the threshold and weights in the model until the network model gradually stabilizes.
(4) Synchronization. All neurons in the BPNN can receive and process information separately, so neurons in the same layer can calculate the received data synchronously, and then transmit the calculated results to the lower neurons together.
(5) Good fault tolerance. Even if the BPNN model is partially damaged or the information received is lost, it can work without much impact (Huang et al., 2020; Sze et al., 2020; Samek et al., 2021).

The BPNN has the above advantages and solves many problems. However, with people's continuous research on it, it is found

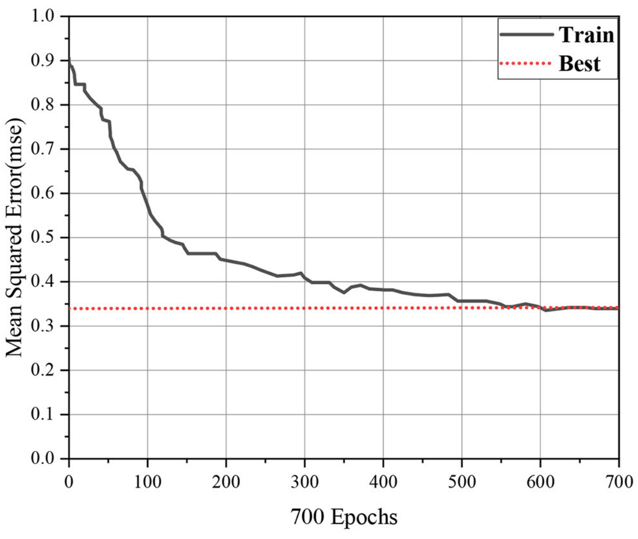
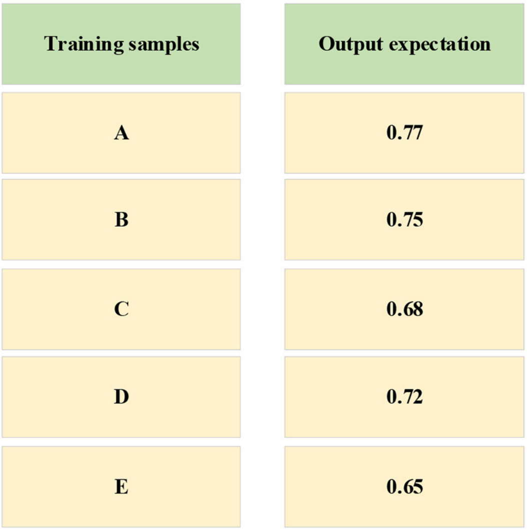


FIGURE 8 | The convergence diagram of the target error of the BPNN.



Training samples	Output expectation
A	0.77
B	0.75
C	0.68
D	0.72
E	0.65

FIGURE 9 | Results of the output.

that there are many shortcomings a. It is mainly reflected in the following aspects:

(1) The convergence speed of the sum of errors is very slow during training. To maintain stability, the learning rate is a small fixed value, which leads to the convergence speed of the sum of errors is very slow during BPNN training.

(2) The sum of errors can easily enter the local minimum value. The function diagram of the sum of errors is a continuous but uneven curve, which contains N local minimal values and an overall minimal value. The purpose of model training is to find the overall minimal value of the sum of errors. In the process of training, it starts from a point in the function curve of the sum of errors, and moves to a minimum value in the descending direction. This minimum value may be the overall minimum value or a local minimum value, so the sum of errors is easy to enter the local minimum value in the process of training (Alarifi et al., 2020). The details are shown in **Figure 5**.

(3) The training time of the BPNN is long. Because the number of layers of the hidden layer and the number of neurons are not fixed, even if the number of layers is generally only used one layer, however, the experience formulas should still be used to calculate the valuation to determine the number of neurons in the hidden layer, and then train them many times to test the most suitable number of neurons. Therefore, the training time of the BPNN is long.

(4) The NN model is unstable. When the BPNN model gradually stabilizes through training, and then inputs new training data, the previous threshold and weights cannot be applied, and the NN model cannot be applied, so the new training data has to be combined with the previous training data for retraining. Therefore, the BPNN model is unstable (Zielonka et al., 2020).

(5) There is an endless cycle in the NN. When the Sigmoid function is used as an incentive function, if the learning rate or weight is too large, the derivative of the Sigmoid function will tend to be 0, and the correction values of the threshold and weights will also tend to be 0. At this time, the NN will fall into an endless cycle.

In view of the above situation, the adaptive adjustment method of the learning rate and the additional momentum method are generally used to improve it. The additional momentum method is used to adjust the weights and thresholds, as shown in Equations (19) to (22):

$$\Delta \mathrm{Q}_{\mathrm{sn}}^{\mathrm{new}} = \alpha \Delta \mathrm{Q}_{\mathrm{sn}}^{\mathrm{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Q}_{\mathrm{sn}}} \tag{19}$$

$$\Delta \mathrm{Q}_{\mathrm{s0}}^{\mathrm{new}} = \alpha \Delta \mathrm{Q}_{\mathrm{s0}}^{\mathrm{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Q}_{\mathrm{s0}}} \tag{20}$$

$$\Delta \mathrm{Y}_{\mathrm{nm}}^{\mathrm{new}} = \alpha \Delta \mathrm{Y}_{\mathrm{nm}}^{\mathrm{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Y}_{\mathrm{nm}}} \tag{21}$$

$$\Delta \mathrm{Y}_{\mathrm{n0}}^{\mathrm{new}} = \alpha \Delta \mathrm{Y}_{\mathrm{n0}}^{\mathrm{old}} - \eta \frac{\partial \mathrm{E}}{\partial \mathrm{Y}_{\mathrm{n0}}} \tag{22}$$

α is a momentum factor with a value (0,1); η is a learning rate with a value (0,1); and E represents the sum of squares of the training error. Due to the influence of momentum factors, when the training drops to a local minimal value, the minimal value is still searched until the overall minimal value is reached (Huang et al., 2019). η will affect the training speed and model effect. The automatic adjustment method of the learning rate can deal with this problem:

$$\eta_{k+1} = \begin{cases} 1.04\eta_k, E_{k+1} < E_k \\ 0.8\eta_k, E_{k+1} > 1.05E_k \\ \eta, \text{ other} \end{cases} \tag{23}$$

k shows the kth algorithm recursive. When the sum of squares of the training error is less than the sum of the recursive squares of the previous algorithm, the learning rate can be appropriately improved. When it is >1.05 times the sum of the recursive error squares of the previous algorithm, the learning rate should be appropriately reduced. In other cases, the learning rate should remain unchanged.

VERIFICATION OF EXPERIMENTAL DATA AND SIMULATION OF MODEL

Verification of Experimental Data

A supply chain enterprise in Xi'an as an example is taken to select suitable supply chain partners for it. The alternative collaborators are A-E. According to the existing indicator system, the BPNN model is established. The number of neurons in the input layer is 14, the number of neurons in the hidden layer is 8, and the number of neurons in the output layer is 1. The final output value of the model is the final evaluation result, and the priority is given to the high score.

The data are collected in the form of questionnaires. The questionnaire adopts the Likert five-point scale. Each question ranges from totally disagree to totally agree, representing 1-5 points, respectively. The specific content of the questionnaire is shown in **Table 1**.

In **Table 1**, the problems contain the actual problems shown by all 14 secondary indicators. The questionnaire is distributed to the management of the alternative partner, 20 copies per enterprise, and finally, the sample data of the test partner is obtained by sorting out the statistics of the questionnaire. In addition, F, G and H enterprises in the same industry are selected to investigate as training samples. The test sample data and training sample data are normalized after they are obtained. The final data is shown in **Figure 6**.

The training data is input into the NN and the expected output is shown in **Figure 7**.

Simulation and Analysis of Model

In **Figure 5**, the data of the three enterprises F, G, and H are brought into the NN as the input value, and the output value in **Figure 6** is taken as the target. When inputting the training function, the target error is 0.01. After 700 times of training, the requirements of the error are met, and the training is completed. The convergence diagram of the target error of the BPNN is shown in **Figure 8**.

The test data of the five enterprises A-E are brought into the trained BPNN model, and the expected value is obtained as **Figure 9**.

Figure 9 indicates that enterprise A has the highest evaluation, with a score of 0.77, which can be used as a supply chain partner, and enterprise B ranks second in the evaluation, with a score of 0.75, which can be used as an alternative enterprise for supply chain partners. The evaluations of C, D and E enterprises are 0.68, 0.72 and 0.65, respectively. The evaluation score is lower than that of company A and company B, so it can be given up in this selection of the supply chain partner. Finally, this experiment selected the optimal cooperative enterprise for the supply chain through the BPNN.

BPNN has strong self-learning and self-improvement capabilities. In other words, the quality and quantity of training samples brought in during the BPNN training process largely determine the output quality of the evaluation model. Therefore, to make enterprises choose more suitable supply chain partners, enterprises should use the evaluation in the process of supplier selection and the evaluation in the whole process of contract performance to establish a complete supplier evaluation database. Meanwhile, the BPNN is continuously trained as a sample to improve the supplier's BPNN selection model. And enterprises should also do a good job in the training of staff, carry out special training courses, professional lectures, experience sharing sessions, etc., to improve staff's understanding of the BPNN system and practical skills. On the basis of systematic training for the existing personnel, the introduction of high-level talents should also be carried out in a targeted manner, and the talent structure should be adjusted.

CONCLUSION

The BPNN algorithm is used to output various information features and optimize the green supply chain according to the input parameters and the different influencing factors in the network. Firstly, by combining the management practice of supply chain enterprises, an evaluation index system of the green supply chain is established, and then the system is combined with the BPNN algorithm to build a BPNN model. Secondly, a supply chain enterprise in Xi'an as an example is taken to select suitable supply chain partners for it, the relevant data of the five alternative enterprises is input into the trained NN. Finally, the output results are 0.77, 0.75, 0.68, 0.72, and 0.65, respectively. The enterprise with the highest output results is selected as the cooperative enterprise. Through specific experiments, the scientific and effectiveness have been proved. Due to some limitations, the involved evaluation indicators are not comprehensive enough. In the future, the scope of research will be expanded, the evaluation indicators of the green supply chain will be added, and the number of influencing factors and neurons will be added. It has certain reference significance for the relevant issues related to subsequent optimization of the green supply chain.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

The author confirms being the sole contributor of this work and has approved it for publication.

REFERENCES

Ahmet, C., and Alk, A. (2020). An integrated open-loop supply chain network configuration model with sustainable supplier selection: fuzzy multi-objective approach. *SN Appl. Sci.* 2, 1–15. doi: 10.1007/s42452-020-2200-y

Alarifi, A., Tolba, A., Al-Makhadmeh, A., and Said, W. (2020). A big data approach to sentiment analysis using greedy feature selection with cat swarm optimization-based long short-term memory neural networks. *J. Supercomput.* 76, 4414–4429. doi: 10.1007/s11227-018-2398-2

Brito, T., Queiroz, J., Piardi, L., Fernandes, L. D. A., and Leitao, P. A. (2020). Machine approach for collaborative robot smart manufacturing inspection for quality control systems. *Proc. Manuf.* 51, 15–18. doi: 10.1016/j.promfg.2020.10.003

Carvalho, J., and Plastino, A. (2021). On the evaluation and combination of state-of-the-art features in twitter sentiment analysis. *Artif. Intell. Rev.* 54, 1887–1936. doi: 10.1007/s10462-020-09895-6

Cui, H. Y., Huang, Z. X., Yüksel, S., and Diner, H. (2021). Analysis of the innovation strategies for green supply chain management in the energy industry using the QFD-based hybrid interval valued intuitionistic fuzzy decision approach. *Renew. Sust. Energy Rev.* 143, 110844. doi: 10.1016/j.rser.2021.110844

Fallahpour, A., Wong, K. Y., Rajoo, S., and Mardani, A. (2020). An integrated fuzzy carbon management-based model for suppliers' performance evaluation and selection in green supply chain management. *Int. J. Fuzzy Syst.* 22, 712–723. doi: 10.1007/s40815-019-00759-y

Gao, S., Qiao, R., Ming, K. L., Li, C., Xia, L. (2021). Integrating corporate website information information into qualitative assessment for benchmarking green supply chain management practices for the chemical industry. *J. Clean. Prod.* 311, 127590. doi: 10.1016/j.jclepro.2021.127590

Hirschfeld, H., Swanson, K., Yang, K., Barzilay, R., and Coley, C. W. (2020). Uncertainty quantification using neural networks for molecular property prediction. *J. Chem. Inf. Model.* 60, 3770–3780. doi: 10.1021/acs.jcim.0c00502

Huang, C., Long, X., and Cao, J. (2020). Stability of antiperiodic recurrent neural networks with multiproportional delays. *Math. Methods Appl. Sci.* 43, 6093–6102. doi: 10.1002/mma.6350

Huang, F., Zhang, X., Zhao, Z., Xu, J., and Li, Z. (2019). Image-text sentiment analysis via deep multimodal attentive fusion. *Knowl Based Syst.* 167, 26–37. doi: 10.1016/j.knosys.2019.01.019

Khan, N. U., Anwar, M., Li, S., Khattak, K. S. (2021). Intellectual capital, financial resources, and green supply chain management as predictors of financial and environmental performance. *Environ. Sci. Pollut. Res.* 28, 19755–19767. doi: 10.1007/s11356-020-12243-4

Meager, S., Kumar, V., Ekren, B., and Paddeu, D. (2020). Exploring the drivers and barriers to green supply chain management implementation: a study of independent UK restaurants. *Proc. Manuf.* 51, 1642–1649. doi: 10.1016/j.promfg.2020.10.229

Midya, S., Roy, S. K., and Yu, V. F. (2021). Intuitionistic fuzzy multi-stage multi-objective fixed-charge solid transportation problem in a green supply chain. *Int. J. Mach. Learn. Cybernet.* 12, 699–717. doi: 10.1007/s13042-020-01197-1

Peng, H., Ma, Y., Li, Y., Cambria, E. (2018). Learning multi-grained aspect target sequence for chinese sentiment analysis. *Knowl Based Syst.* 148, 167–176. doi: 10.1016/j.knosys.2018.02.034

Sakaki, M., Ueno, T., Ponzio, A., Harley, C. W., and Mather, M. (2019). Emotional Arousal Amplifies Competitions Across Goal-Relevant Representation: A Neurocomputational Framework. *Cognition* 187, 108–125. doi: 10.1016/j.cognition.2019.02.011

Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., and Müller, K. R. E. (2021). Networks and beyond: a review of methods and applications. *Proc. IEEE.* 109, 247–278. doi: 10.1109/JPROC.2021.3060483

Shao, L., i,. G., Zhang, S., and Green Supply Chain, L. (2019). Behavior and business performance: evidence from China. *Technol. Forecast. Soc. Change* 144, 445–455. doi: 10.1016/j.techfore.2017.12.014

Staal, A. S., Salvatierra, C. G., Albertsen, D. D., Mahendran, M., and Bgh, S. (2020). Towards a collaborative omnidirectional mobile robot in a smart cyber-physical environment. *Proc. Manuf.* 51, 193–200. doi: 10.1016/j.promfg.2020.10.028

Sze, V., Chen, Y., Yang, T., and Emer, J. S. (2020). Efficient processing of deep neural networks. *Synth. Lect. Comput. Archit.* 15, 1–341. doi: 10.2200/S01004ED1V01Y202004CAC050

Tang, S., and Yu, F. (2021). construction and verification of retinal vessel segmentation algorithm for color fundus image under BP neural network model. *3E J Supercomput.* 77, 1–15. doi: 10.1007/s11227-020-03422-8

Wu, Z., Jiang, C., Conde, M., Deng, B., and Chen, J. (2019). Hybrid improved empirical mode decomposition and BP neural network model for the prediction of sea surface temperature. *Ocean Sci.* 15, 349–360. doi: 10.5194/os-15-349-2019

Zhou, S., Shen, C. Y., Zhang, L., Liu, N. W., He, T. B., Li, Y., et al. (2019). Dual-optimized adaptive Kalman filtering algorithm based on BP neural network and variance compensation for laser absorption spectroscopy. *Opt. Express* 27, 31874–31888. doi: 10.1364/OE.27.031874

Zhou, X., Hu, Y., Liang, W., Ma, J., and Jin, Q. V. (2020). Enhanced anomaly detection for industrial big data. *IEEE Trans. Indus. Inform.* 17, 3469–3477. doi: 10.1109/TII.2020.3022432

Zielonka, A., Sikora, A., Wozniak, M., Wei, W., and Bai, Z. (2020). Intelligent internet of things system for smart home optimal convection. *IEEE Trans. Indus. Inform.* 17, 4308–4317. doi: 10.1109/TII.2020.3009094

Conflict of Interest: The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 09 June 2022
doi: 10.3389/fnbot.2022.865403

Intervention Effect of Rehabilitation Robotic Bed Under Machine Learning Combined With Intensive Motor Training on Stroke Patients With Hemiplegia

Guangliang Liu[1]*, *Haiqin Cai*[2] *and Naruemon Leelayuwat*[1]

[1] Graduate School of Khon Kaen University, Khon Kaen, Thailand, [2] College of Music, Gannan Normal University, Ganzhou, China

It was aimed to discuss the effect of bed-type rehabilitation robots under machine learning combined with intensive motor training on the motor function of lower limbs of stroke patients with hemiplegia. A total of 80 patients with stroke hemiplegia were taken as the subjects, who all had a course of treatment for less than 6 months in the Rehabilitation Medicine Department of Ganzhou Hospital. These patients were divided into the experimental group (40 cases) and the control group (40 cases) by random number method. For patients in the control group, conventional intensive motor training was adopted, whereas the conventional intensive motor training combined with the bed-type rehabilitation robot under machine learning was applied for patients in the experimental group. *Fugl-Meyer Assessment of Lower Extremity* (FMA-LE), *Rivermead Mobility Index* (RMI), and *Modified Barthel Index* (MBI) were used to evaluate the motor function and mobility of patients. The human–machine collaboration experiment system was constructed, and the software and hardware of the control system were designed. Then, the experimental platform for lower limb rehabilitation training robots was built, and the rehabilitation training methods for stroke patients with hemiplegia were determined by completing the contact force experiment. The results showed that the prediction effect of back-propagation neural network (BPNN) was better than that of the radial basis neural network (RBNN). The bed-type rehabilitation robot under machine learning combined with intensive motor training could significantly improve the motor function and mobility of the lower limbs of stroke patients with hemiplegia.

Keywords: stroke, hemiplegia, intensive motor training, machine learning, bed-type lower limb rehabilitation robot

Edited by:
Mu-Yen Chen,
National Cheng Kung
University, Taiwan

Reviewed by:
Pejman Hosseinioun,
Islamic Azad University, Iran
Rupert Ortner,
G.Tec Medical Engineering Spain
S.L., Spain

****Correspondence:***
Guangliang Liu
guangliangli@kkumail.com

Received: *29 January 2022*
Accepted: *19 April 2022*
Published: *09 June 2022*

Citation:
Liu G, Cai H and Leelayuwat N (2022) Intervention Effect of Rehabilitation Robotic Bed Under Machine Learning Combined With Intensive Motor Training on Stroke Patients With Hemiplegia.
Front. Neurorobot. 16:865403.
doi: 10.3389/fnbot.2022.865403

INTRODUCTION

Stroke, also known as cerebrovascular accident, shows high morbidity, high mortality, high disability, and high recurrence (Pan, 2018; D'Ancona et al., 2020). At present, the incidence of stroke exceeds that of tumors and heart diseases in China, and it has become the world's second and the China's first most fatal disease (Luney et al., 2020; Xia et al., 2021). Hemiplegia is the most common sequelae of stroke. It has been surveyed that more than 50% of patients with hemiplegia suffer from lower limb spasm, and the severity of spasm increases over time (Suri et al., 2018; Tomida et al., 2019; Tsuchimoto et al., 2019; Park et al., 2021). Stroke patients with hemiplegia have a high incidence of lower limb motor dysfunction, which is difficult to recover with a poor prognosis. It seriously affects the independence of the patients' daily life and social activities (Yang et al., 2021).

From the overall rehabilitation process after stroke, it is difficult to recover from hemiplegia of the lower limbs, which not only takes a long time but also costs a lot of money; the treatment effect is limited (Alawieh et al., 2018; Uwatoko et al., 2020; Tang et al., 2021). With the rapid development of modernization, informatization, and intelligent technology, high-performance rehabilitation robots have emerged. Bed-type rehabilitation robots have been gradually used in clinical research on the rehabilitation of lower limbs of stroke patients with hemiplegia (Zhang et al., 2021). The bed-type rehabilitation robot, as an emerging physical therapy technology for the treatment of lower limb dysfunction of stroke patients with hemiplegia, can provide high-precision and high-repeatability training. At present, the gait training robot Rehoambulator (Calabrò et al., 2016) developed by American HealthSouth Medical Company has been productized and popularized to the market by Motorika Company. Its two separate mechanical legs are fixed on the frame: one leg only has two degrees of freedom of hip joint and knee joint, and the two connecting rods on the leg are driven by the motor to drive the big leg and the small leg to reciprocate. Erigo (Sarabadani Tafreshi et al., 2016), an early rehabilitation training system for nerve injury developed and promoted by Swiss company Hocoma, consists of a rehabilitation bed with an adjustable angle and a stepping system to help lower limb rehabilitation, which can realize early intensive rehabilitation training.

Nowadays, machine learning-based algorithms can train the human body model directly, which makes the prediction and recognition of human intentions more accurate in human–computer collaboration. The algorithm based on machine learning captures, learns, and predicts human actions by visual sensors to identify the operator's intention, so as to improve the coordination between patients and robots (Cha et al., 2021). Therefore, developing a lower limb rehabilitation robot with independent intellectual property rights, simple structure, low cost, and convenient operation will be of great significance to the development of the rehabilitation medical robot industry in China.

It was innovated based on the intensive motor training that the bed-type rehabilitation robot under machine learning was used to perform lower limb rehabilitation training for stroke patients with hemiplegia. The effect on lower limb motor function of stroke patients with hemiplegia was observed in this work.

MATERIALS AND METHODS

Study Subjects

A total of 80 patients with stroke hemiplegia were selected to be the research subjects, and they all had a course of treatment for less than 6 months in the Rehabilitation Medicine Department of Ganzhou Hospital. All patients met the stroke diagnostic criteria established by the cerebrovascular disease academic conference (Hellmich et al., 2020). They were divided into the experimental group (40 cases) and the control group (40 cases) by random number method, and their data were evaluated, trained, and analyzed by 3 physicians. There were 38 men and 42 women, with an average age of 48.52 ± 11.46 years. This work was approved by the Medical Ethics Committee of Ganzhou Hospital, and the patients and their families understood the research situation and signed informed consent forms.

The following were the inclusion criteria. With transcranial magnetic resonance imaging examination, the patients met the diagnostic criteria for stroke combined with hemiplegia. It was the first onset of the patients, Brunnstrom stage of the affected lower limb is above stage II, and the course of the disease was less than half a year. The patients and their families accepted and cooperated with the experiments.

Exclusion criteria were as follows. Patients were in progress with cerebrovascular diseases. Patients had serious heart, liver, lung, and other organ damages. Patients suffered from severe cognitive dysfunction and sensory aphasia. Patients had other major mental illnesses. Patients had diseases that did not allow them to complete the lower limb motor training, such as thrombosis of the lower limbs, joint swelling, and joint stiffness. Patients had other diseases that might lead them and their families to be unable to cooperate, or they were unwilling to participate.

Collection of the Patients' Clinical Data

Clinical data of all research subjects were collected, including the name, age, race, place of residence, education level, stroke type, course of disease, hemiplegic side, history of atrial fibrillation, history of coronary heart disease, history of diabetes, history of smoking, and history of drinking. After treatment, the patients were followed up for 8 weeks, and the total recovery time of the two groups was compared.

Examination and Evaluation of Motor Function and Mobility

Fugl-Meyer Assessment Lower Extremity (FMA-LE) (Madhoun et al., 2020), *Rivermead Mobility Index* (RMI) (Lim et al., 2019), and *Modified Barthel Index* (MBI) (Taghizadeh et al., 2020) were adopted jointly to test and evaluate patients' motor function and mobility. The subjective state of the examination process and the environment had a certain impact, so it was necessary to be guided in accordance with the unified instructions. The hints that exceed the specified range were eliminated, to create a relaxing, comfortable, and quiet evaluation environment.

Fugl-Meyer Assessment Lower Extremity score is internationally recognized as the most standard and most widely used method for evaluating stroke combined with hemiplegia, with high sensitivity and reliability. It is mainly used to evaluate autonomous, separated, and independent movements related to coordinated movements. It could be used for accurate quantitative assessment of lower limb motor function of patients with hemiplegia. There are 34 evaluation items, and each item is scored to be 0–1 points. There are three levels for the score, as 0 point means the patient was incapable of activities completely, 1 point means the patient could complete part of the activities, and 2 points mean the patient could complete the activities normally. The total score for the normal motor function is 34. The higher the total score, the better the recovery of motor function.

The RMI was used to evaluate the transferability of patients. The scale covers 15 items in total, including turning over on the bed, sitting balance, independent standing, and independent walking indoors. The results were obtained through inquiry except for assisted observation was performed in going up and downstairs and running. The total score is 15 points, and the score ranges from 0–1 point for each item. Then, 0, 1, and 2 points mean the patient is unable to complete, able to complete partially, and able to complete normally, respectively. The higher the score, the better the transferability.

The MBI was used to evaluate the patient's daily mobility. The MBI covers 10 items, such as eating, dressing, going up and downstairs, transferring, and walking. The scores are graded into four levels of 0, 5, 10, and 15 points, with a total score of 100. Then, 100 points mean there is no need for dependence, 60–99 points mean light dependence, 41–59 points mean moderate dependence, and less than 40 points mean severe dependence.

Treatment Methods

For patients in the control group, conventional intensive motor training was given. According to the situation of motor dysfunction in patients, the appropriate intensive motor training method was chosen.

In the experimental group, conventional intensive motor training was combined with the bed-type rehabilitation robot under machine learning. The training of the intelligent rehabilitation robot was completed by professional therapists. Bed-type rehabilitation robot can provide patients with a maximum weight of 135 kg, accommodate patients with a maximum height of 2 m, dynamic weight support: 0–85 kg, dynamic weight support range: 0–18 cm, treadmill speed range: 1–3.2 km/h, treadmill speed accuracy:+/−0.1 km/h, as shown in **Figure 1**. The therapists instructed the patients to complete the isolating movements with the uninhibited lower limb before training. Bed-type rehabilitation robot training was as follows:

During the training, the patients were required to look at the front horizontally. The patients were asked to adjust their body posture and maintain the symmetry of their body posture as looking themselves in the mirror. They should extend the knee actively in the middle of the support phase and fully extend the hip joint at the end of the support phase. The virtual mode and training parameters of the rehabilitation robot were set as follows. The treatment time was 30 min, the rising angle of the bed was 70–80°, and the training pace was 1.24–1.78 km/h. Throughout the training process, the therapists tried not to help as much as possible. If the patients had negative emotions, lack of concentration, and so on, the therapists should promptly encourage and remind the patients to participate in the training actively. The training time for walking was 30 min per day, and the total time was 45 min (including robot setting, patients' preparation, training time of walking, and getting out of the bed after training). It should be trained 5 times a week, and 2 weeks were spent to complete. In the training process, if the blood pressure of patients exceeded 180/110 mmHg, the heart rate exceeded 75% of the age-standard heart rate, or they had headaches, nausea, or other adverse symptoms, the training had to be stopped immediately.

Human–Machine Coordination Experimental System Model

After the human–machine collaboration started, it was expected that the rehabilitation robot could cooperate with the human limb movement, to speed up the response speed of the robot. The dynamic function of the human–machine coordinated motion model was expressed as Equation (1).

$$Ma + Bv + Gv = F_1 + F_2 \tag{1}$$

In Equation (1), M, B, G, a, and v represented system inertia, damping, stiffness, the end acceleration of the robot, and the speed and the desired end speed, respectively. F_1 and F_2 represented the force of the operator and the force of the rehabilitation robot, respectively. The human–machine coordinated motion model was a load with a mass of m that was coordinated by humans and robots. It was necessary to consider the inertia, damping, and stiffness of the system comprehensively, as shown in **Figure 2**, in which F_3 was the force of the object. In the process of human–machine collaboration, the force contributed by the robot needed to exceed the force exerted by the operator, thus reducing the burden on the operator.

Construction of Human–Machine Coordinated Motion Experiment System Under Machine Learning

The principle of operator intention recognition under machine learning mainly consists of two parts. One part is offline learning, in which the data are collected and classified using a fuzzy method, and the samples are trained in the neural network. The other part is online execution, including information acquisition and the prediction of back-propagation neural network (BPNN), as shown in **Figure 3**. The characteristics of fuzzy classification were utilized to perform cluster analysis on the required data in the offline part. Then BPNN was applied to train the data samples. The model constructed in the offline part was used to predict the speed information of the operator during the process of robot identification, learning, and prediction of operator information in the online execution part. The information predicted by the BPNN was inputted into the robot in advance, to make the robot follow the operator for collaborative tasks.

Back-propagation neural network was composed of an input layer, a hidden layer, and an output layer. The number of network layers was determined by the hidden layer. The input layer node entered the input quantity, the input layer node output quantity was the input quantity of the hidden layer, and the hidden layer output quantity was taken as the input quantity of the output layer. The final output is worked out as shown in **Figure 4**. The BPNN shows strong learning ability, adaptability, and high fault tolerance. During the training process, the difference between the output and the expected value was adjusted to

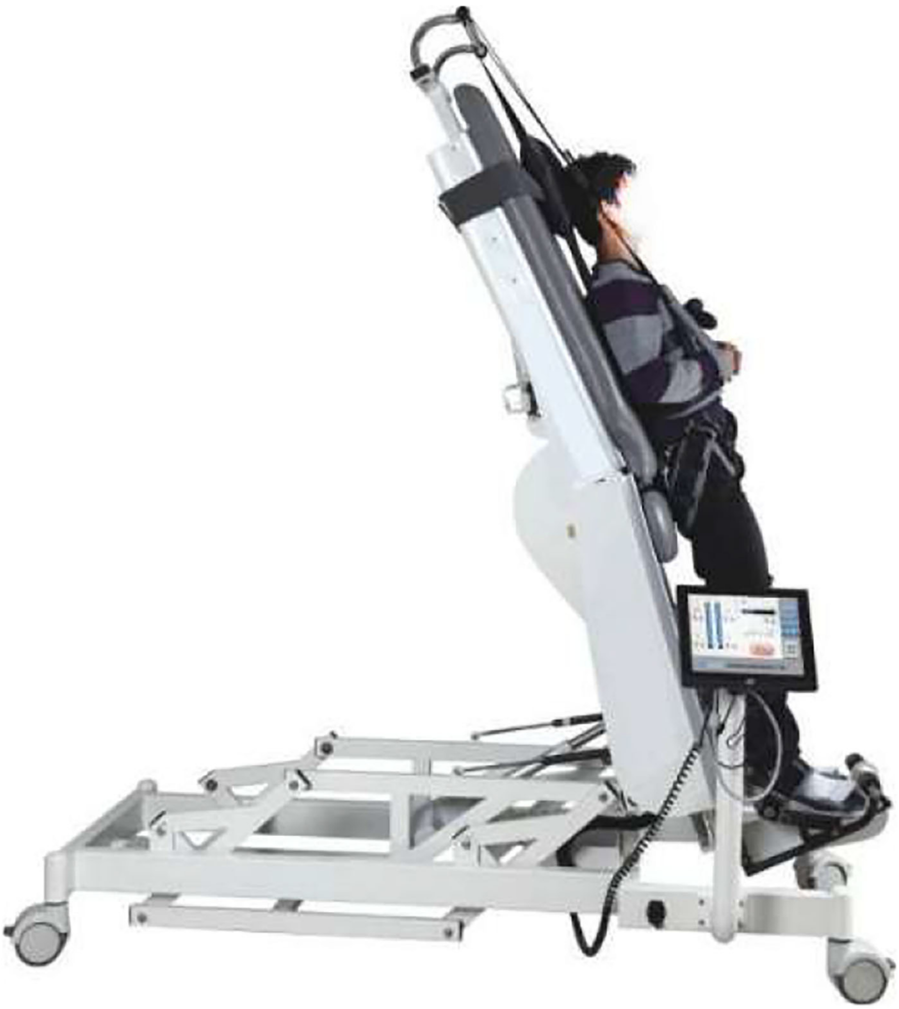


FIGURE 1 | Bed-type lower robot.

change the parameters for self-adjustment. The three-layer neural network could complete the work well of prediction, linear approximation, recognition, etc., so it was chosen and applied.

Due to the slow convergence of the BPNN and the difficulty of determining the hidden layer and the number of nodes, it was aimed to improve the limitations of the BPNN.

On the one hand, the momentum term was added. Since the BPNN did not consider the direction of the previous gradient when modifying the weight or threshold, the stability was poor and the convergence was slow. Therefore, the momentum term could be added for correction, which was expressed as Equation (2) and Equation (3).

$$\Delta W\left(k\right) = \gamma\left[(1-z)\,G\left(k\right) + z\Delta W\left(k-1\right)\right] \tag{2}$$

$$G\left(k\right) = \frac{\partial E}{\partial W\left(k\right)} \tag{3}$$

In Equations (2) and (3), Δ*W(k)* represented the weight at the moment, Δ*W(k-1)* represented the weight at the previous moment, and *G(k)* represented the negative gradient function at moment *k*. γ was the learning rate and z was the momentum factor.

On the other hand, the step size was changed. The convergence speed of the network was mainly determined by the learning rate γ. If γ was too small, the convergence speed became slower. If γ was too large, the system would be unstable. The step size of seat could be optimized, and the Equations (4) and (5) were as follows.

$$\delta = sgn\left[G\left(k\right)G\left(k-1\right)\right] \tag{4}$$

$$\gamma\left(k\right) = 2^{\lambda}\gamma\left(k-1\right) \tag{5}$$

In Equations (4) and (5), $sgn\left(\cdot\right)$ represented the sign function, *G(k)* and *G(k-1)* represented the negative gradient function at moments *k* and *k-1*, respectively. γ was still the learning rate.

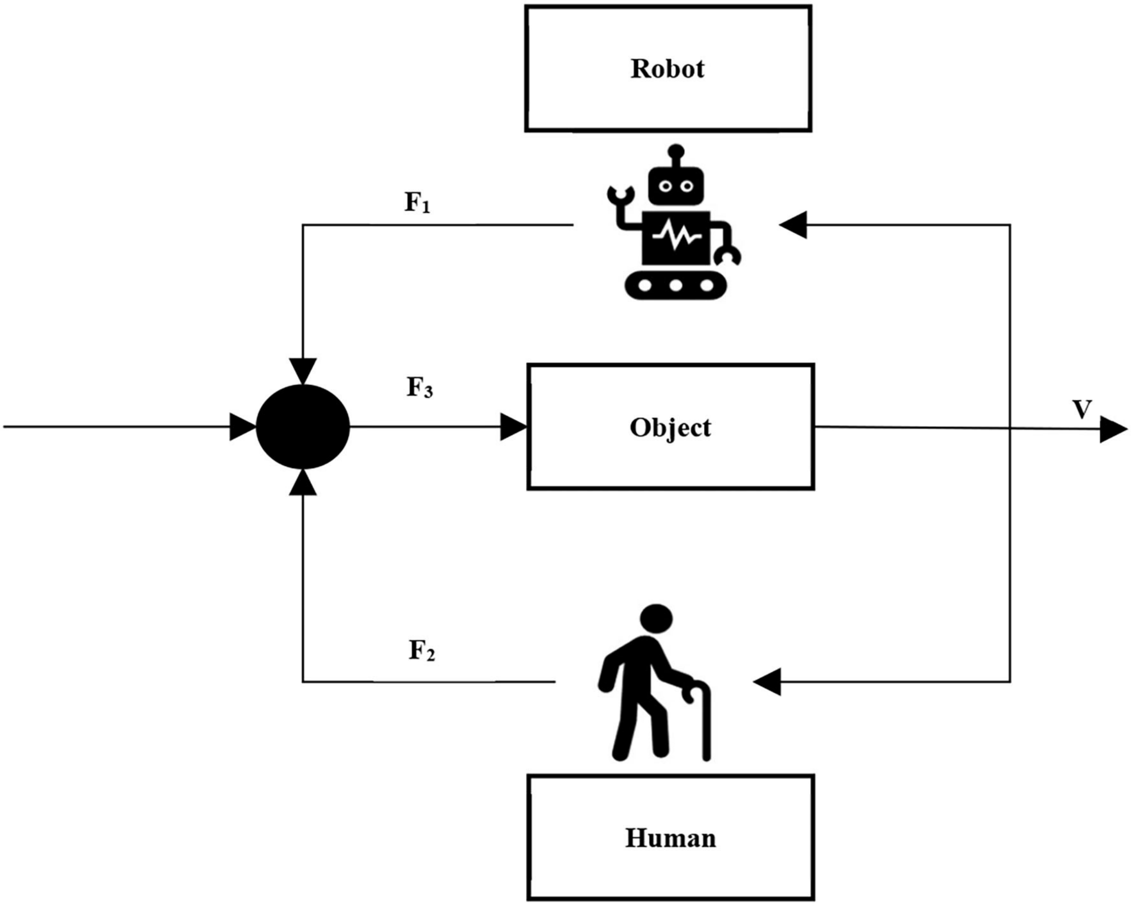


FIGURE 2 | Human–machine coordinated motion model.

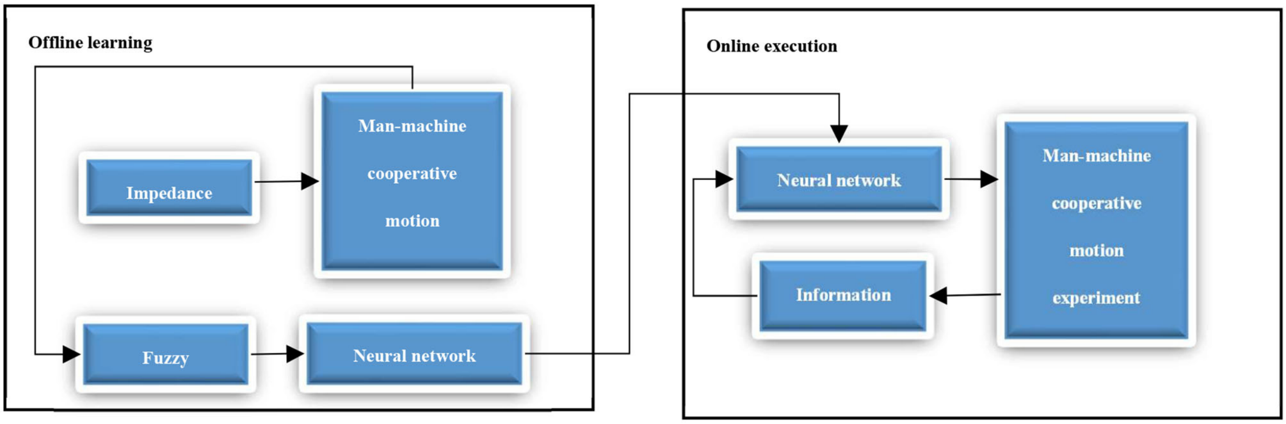


FIGURE 3 | Schematic diagram of operator intention recognition under machine learning.

The hardware of the human–machine coordinated motion experiment system included industrial control computer, force information acquisition module, servo control system, and motion execution module. **Figure 5** is a simplified diagram of the human–machine coordinated motion experiment system. The force information acquisition module was made up of a six-dimensional force sensor and the matched data acquisition card, and the six-dimensional force sensor was to collect the

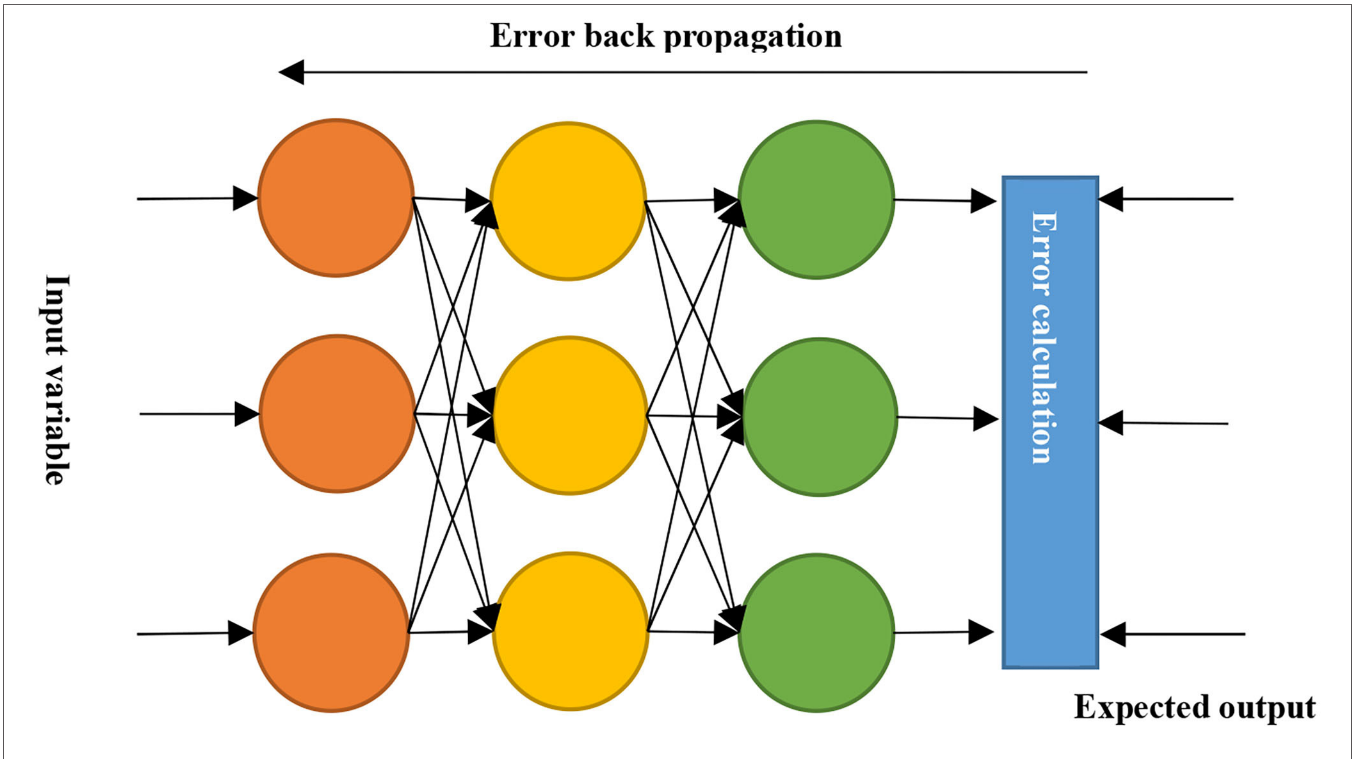


FIGURE 4 | Schematic diagram of the basic structure of BPNN.

interaction force between human and machine. The servo control system consisted of a motion control card and a servo driver. **Figure 6** shows the hardware structure of the system.

The robot control program was compiled by SMC Basic Studio software, which supported online programming and debugging of Basic language and G code, with rich programming languages and concise programs. In addition, the software supported the function demonstration and debugging of the motion controller, mainly including IO monitoring, uniaxial motion function test, zero-return motion function test, psychomotor vigilance test (PVT), and DA/PWM output function test. The control software can control the related motion parameters and IO parameters of the controller by the user through the upper computer, which was convenient to operate. The multi-task mode set by Basic language was adopted in the control program design.

When the trajectory data of human walking joints were analyzed, it was necessary to prevent the vibration of the motor caused by the uneven speed from affecting the wearing comfort of patients and the life of the motor, so as to ensure the characteristics and smoothness of the motion curve of the collected trajectory. Multi-order sine trigonometric function is used to fit the collected discrete data. The fitting function is as follows:

$$f\left(t\right)=\sum_{i=1}^{n}A_i\sin(x_it+b_i) \tag{6}$$

In Equation (6), n represented the order of sine trigonometric function, A_i represented amplitude modulation coefficient, x_i represented frequency coefficient, and b_i represented offset.

$$\min\|t\|_2^2=\sum_{i=0}^{l}[(g\left(t\right)-f(t))]^2 \tag{7}$$

In Equation (7), $g\left(t\right)$ represented the collected discrete trajectory points, and l referred to the fitting order.

Sample Data Collection

The data were collected with the human–machine coordinated motion experiment system, as shown in **Figure 7**. First, the tension and pressure sensor used in the system and the alternating current servo motor representing the robot were initialized. The zero-return operation opportunity was set on the end handle, to ensure that the collected end position information of the robot was consistent. When sample data were collected, the robot in the one-degree-of-freedom human–machine collaboration system was controlled by impedance control. The operator used the movements in motor spaces restricted by each speed as much as possible to cover the one-degree-of-freedom human–machine collaboration data space, so that the samples had diversity. The operating frequency of the human–machine coordinated motion experiment system was 1 kHz during the data collection process, and the sampling frequency was 60 Hz. During the data collection, data in 5 s were randomly selected, and the number of repeated samplings was 3 times.

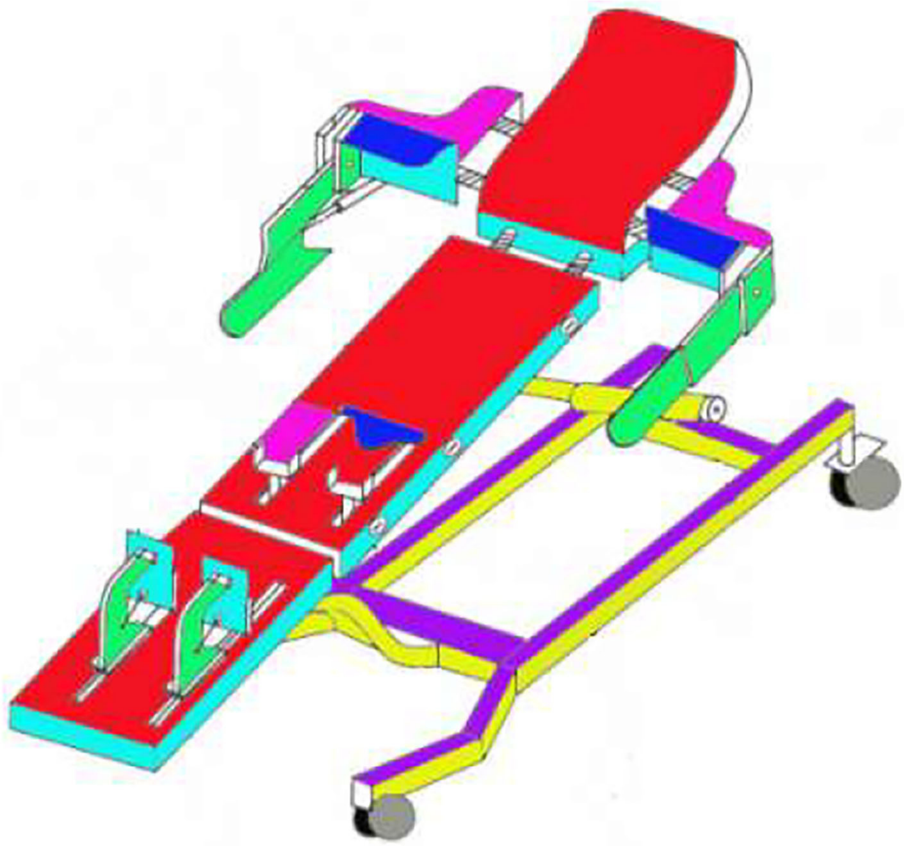


FIGURE 5 | Schematic diagram of the human–machine coordinated motion experiment system.

Simulation Experimental Verification

A simulation experiment was performed to verify the effectiveness of the above experiments and to prevent the diversity of verification equipment and the multi-degree-of-freedom motion control coupling. The above-mentioned human–machine coordinated motion experiment system was applied, and the BALM-3 tension and pressure sensor (Honeywell, China) was used to detect the magnitude and direction of the interaction force during human–machine collaboration. The IPC-610H industrial control computer (Shanghai Senke Electronic Technology Co., Ltd.) was used to compile the program, collect the force sensor information, and control the servo motor (robot) when the program was applied. PCI-1245E control card (Beijing Konrad Technology Co., Ltd.) was used as the information collection system. 60HBM0130CM servo motor was also utilized with a rated power of 400W and a rated torque of 1.27 nm.

For the human–machine cooperative control system, the rotation angle α was taken as the generalized coordinate. If the loss between the shaft couplings was neglected, the kinetic energy in the system energy was expressed as Equation (8).

$$\begin{aligned} K &= J_s \times \tfrac{\alpha^2}{2} + J_b \times \tfrac{\alpha^2}{2} + \tfrac{mv^2}{2} \\ &= J_s \times \tfrac{\alpha^2}{2} + J_b \times \tfrac{\alpha^2}{2} + \tfrac{m\left(\frac{L\alpha}{2\pi}\right)^2}{2} \end{aligned} \tag{8}$$

In Equation (8), K stood for the kinetic energy, and J_s and J_b were the magnetic flux of each pole of the servo motor and the ball motor, respectively. m was the mass of the object, and L was the lead of the screw.

$$V = \left(F - mg\right) \times z = \left(F_2 - mg\right) \times \frac{L\alpha}{2\pi} \tag{9}$$

In Equation (9), V represented potential energy, and F_2 was the force of the operator.

$$D = \frac{Cz^2}{2} = \frac{C\left(\frac{L\alpha}{2\pi}\right)^2}{2} \tag{10}$$

In Equation (10), D meant the energy consumption, and C was the system damping.

Generalized dynamic function for rotation angle α was expressed as Equation (11).

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \alpha}\right) - \frac{\partial L}{\partial \alpha} + \frac{\partial D}{\partial \alpha} = K \tag{11}$$

After the final handling, Equation (12) was obtained.

$$\begin{aligned} K &= \left\lfloor J_s + J_b + m \times \left(\tfrac{L}{2\pi}\right)^2 \right\rfloor \times \alpha + \\ &\left(-F_2 + mg\right) \times \tfrac{L}{2\pi} + C\left(\tfrac{L}{2\pi}\right)^2 \times \alpha \end{aligned} \tag{12}$$

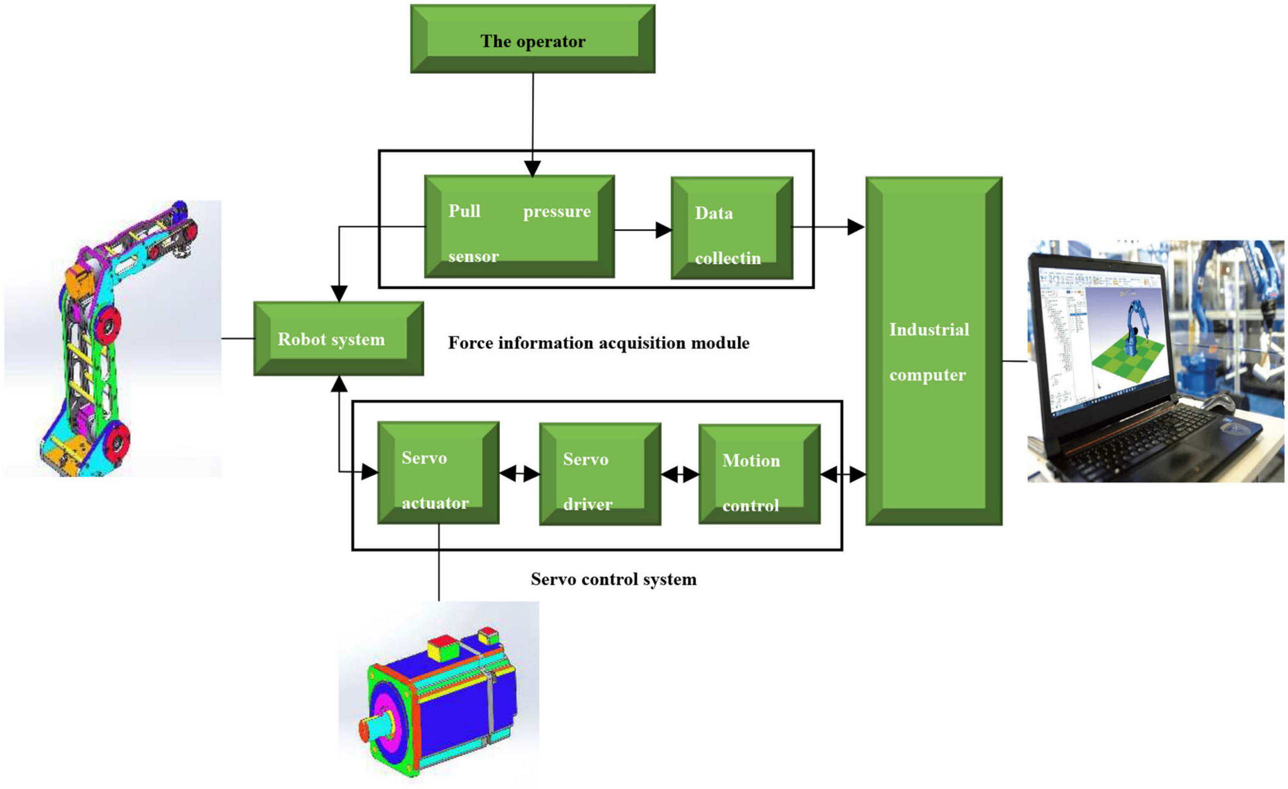


FIGURE 6 | Hardware structure of the human–machine coordinated motion experiment system.

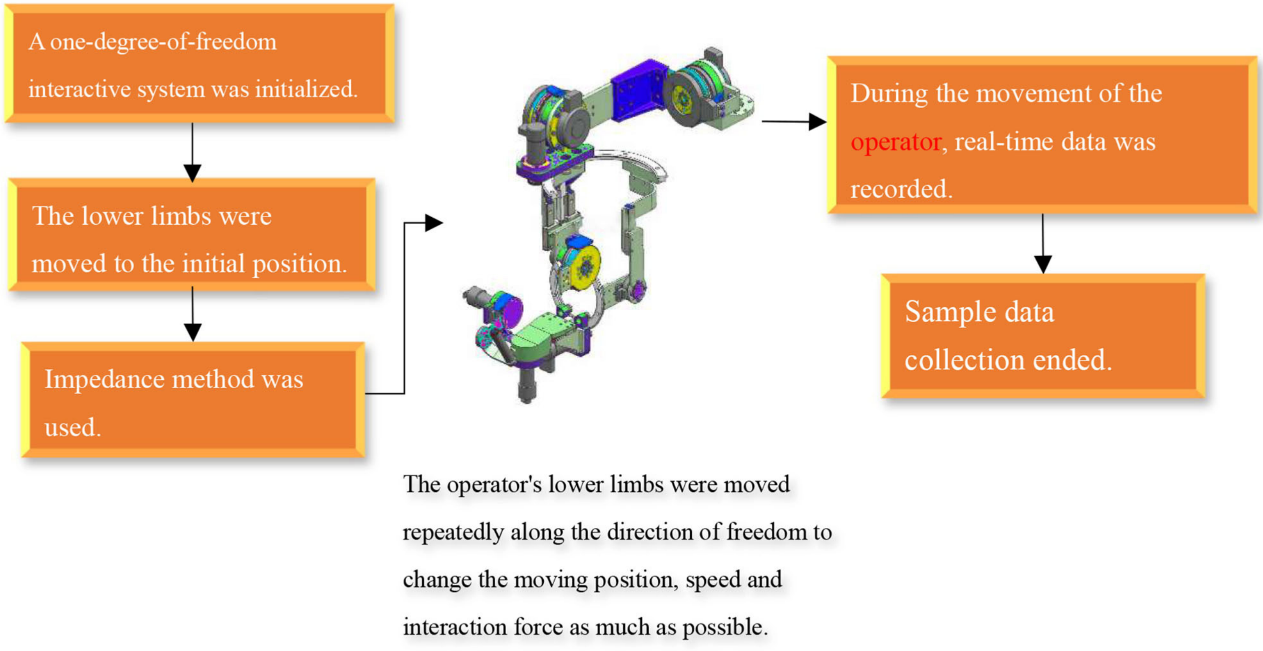

A one-degree-of-freedom interactive system was initialized.
The lower limbs were moved to the initial position.
Impedance method was used.
During the movement of the operator, real-time data was recorded.
Sample data collection ended.
The operator's lower limbs were moved repeatedly along the direction of freedom to change the moving position, speed and interaction force as much as possible.

FIGURE 7 | Flowchart of data collection.

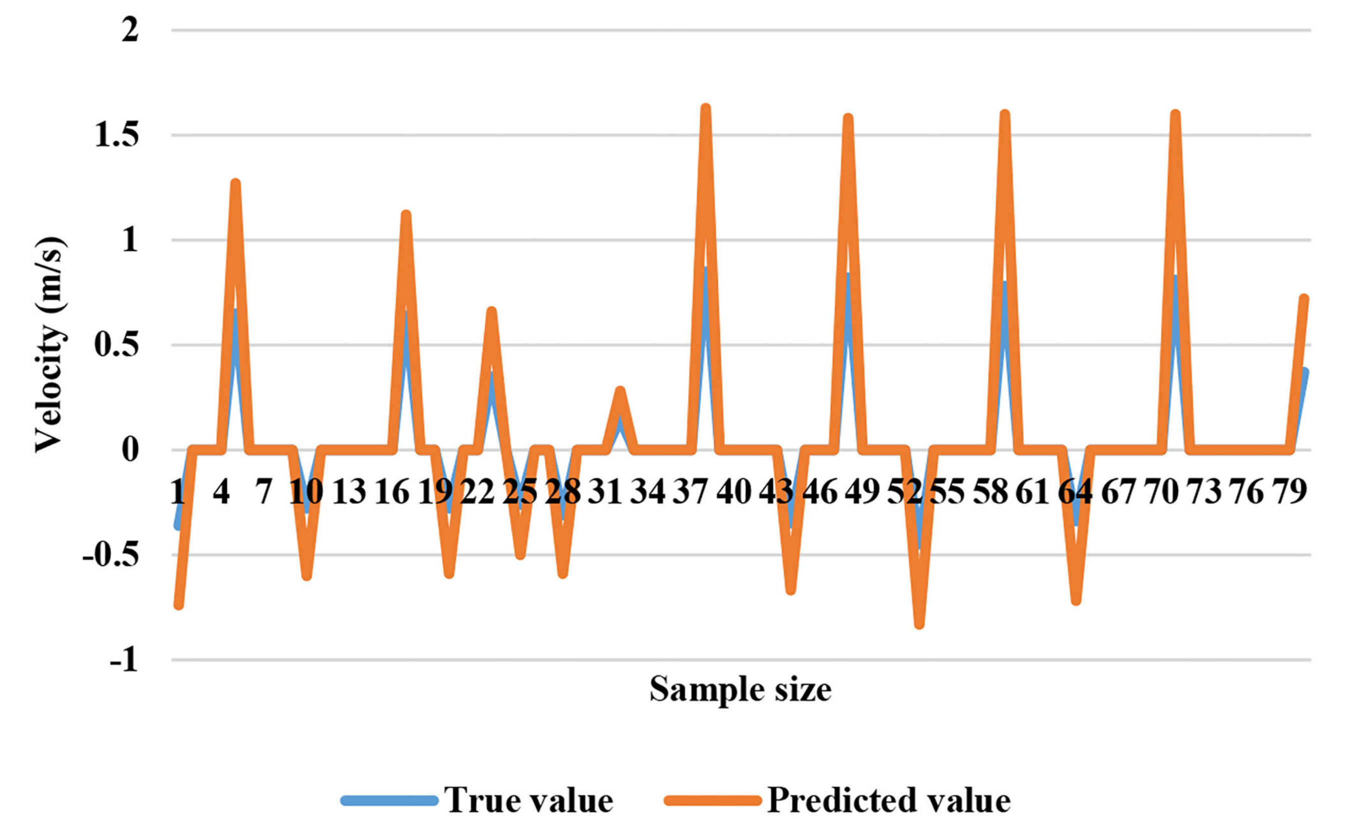


FIGURE 8 | Predicted value and true value of the operator's intention recognition in the human–machine collaboration experiment system under BPNN.

Statistical Methods

The data were processed and analyzed *via* SPSS19.0. The measurement data were expressed as the mean ± standard deviation (± *s*), and the enumeration data were expressed as the percentage (%). Pairwise comparison was performed through one-way analysis of variance. The difference was statistically significant at $p < 0.05$.

RESULTS

Verification of Experiment Results

It was compared with the RBNN model, to test the experimental effect of identifying and predicting the operator's intention under the BPNN model constructed above. The test samples collected by impedance control were detected. With 80 sets of test data, the predicted value and true value of the operator's intention under the BPNN and RBNN are shown in **Figures 8**, **9**, respectively. There was less difference between the two networks in prediction effect, and the predicted results were both acceptable. After training, the mean square error (MSE) of the BPNN model was 0.9975, and that of the RBNN model was 0.9642. The MSE was closer to the real number 1, the better the prediction and training effect. It was indicated that the prediction effect of the BPNN was better than that of RBNN.

Amplitude of Driving Torque

The simulation time was set as the standard gait cycle of human walking of 1.3 s, which can be obtained by motion analysis and calculation. The maximum driving torque required by the single-leg movement of the robot was 14 Nm, and the maximum driving torque required by the whole lower limb rehabilitation robot was 28 Nm. At this time, the crank speed was 4.67 md/s, and the maximum power required by the system was 125.7 W. The driving torque amplitude curve is shown in **Figure 10**.

Comparison of Baseline Data of Patients in the Two Groups

Among the 80 patients included, those in the experimental group received training of the intelligent lower limb rehabilitation robot. At the end, all 80 patients completed the entire training. There were no significant differences between the two groups in gender, age, course of disease, stroke type, hemiplegic location, past history of atrial fibrillation, history of coronary heart disease, history of diabetes, history of smoking, and history of drinking ($p > 0.05$). The details are shown in **Table 1**.

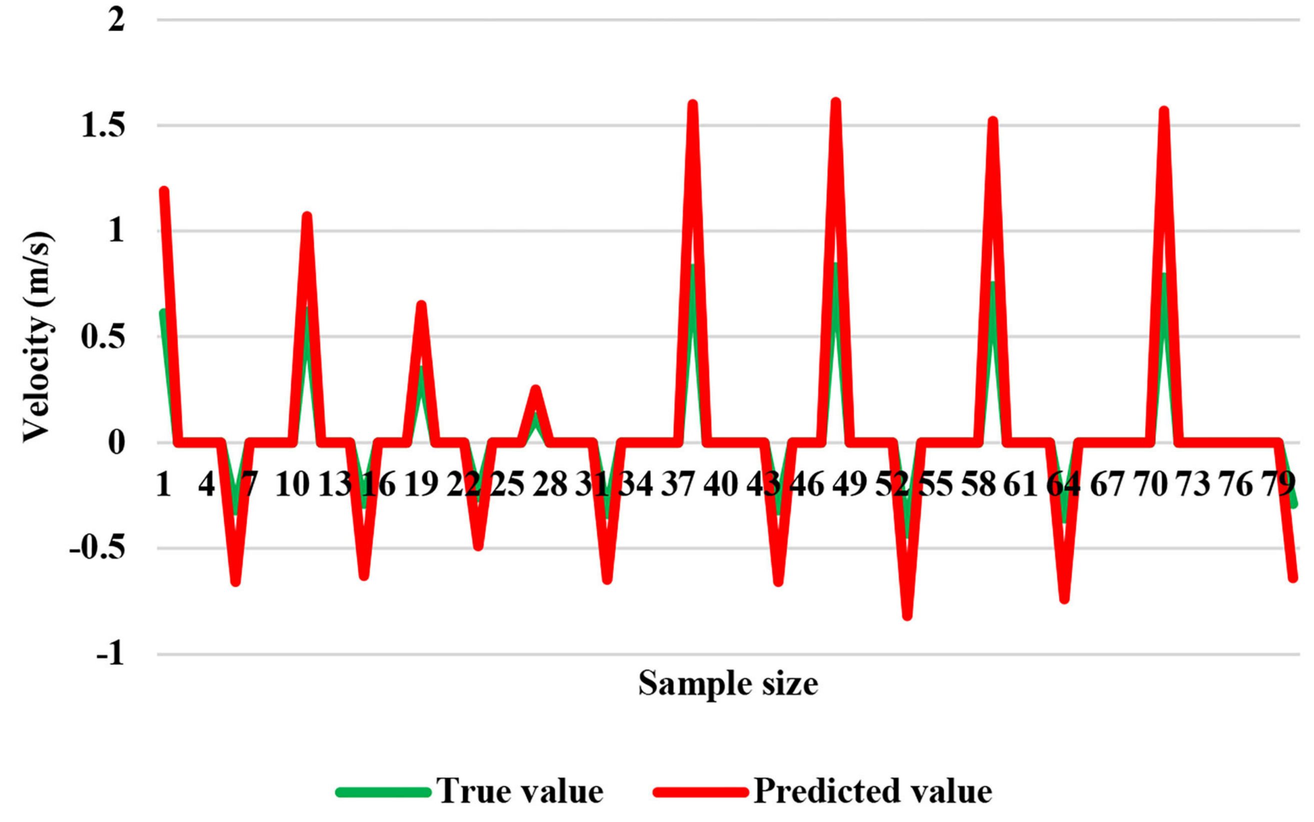


FIGURE 9 | Predicted value and true value of the operator's intention recognition in the human–machine collaboration experiment system under RBNN.

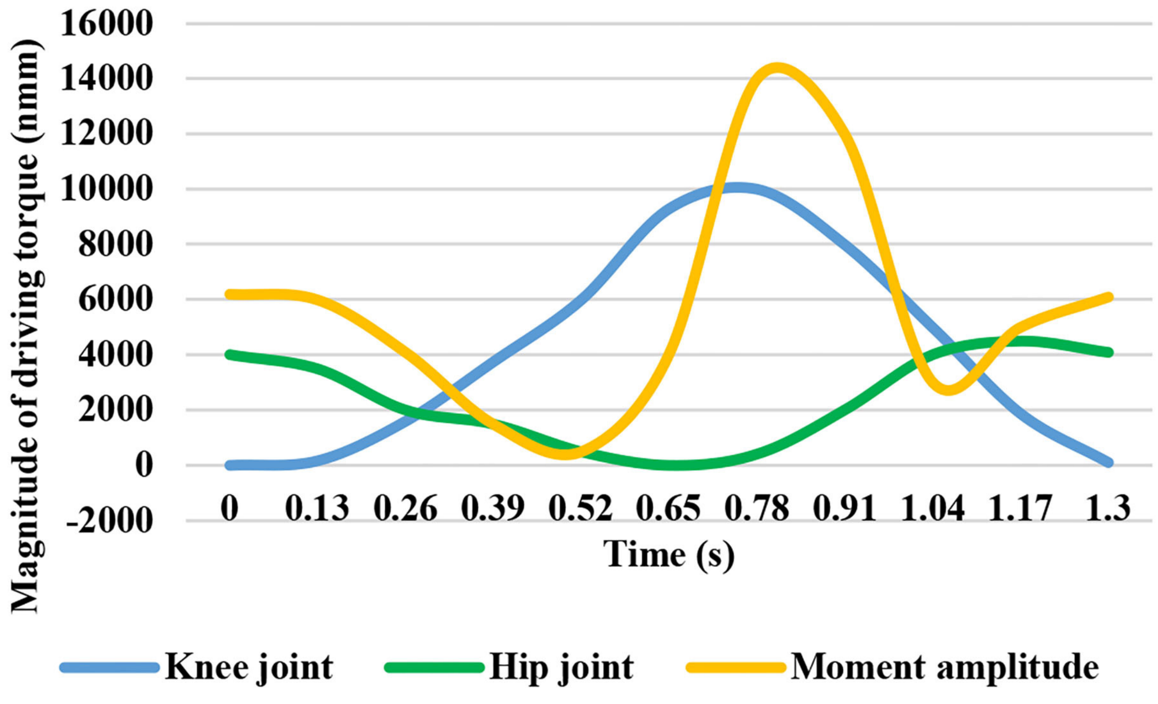


FIGURE 10 | Driving torque amplitude curve.

TABLE 1 | Comparison of patients' baseline data in the two groups.

Indicators		**Control group ($n = 40$)**	**Experimental group ($n = 40$)**	***p***
Gender	Male	17	19	0.256
	Female	23	21	
Age (years old)		48.68 ± 11.21	49.68 ± 12.16	0.312
Course of disease (days)		43.56 ± 19.34	45.27 ± 20.58	0.283
Stroke types	Ischemic	22	22	1.000
	Hemorrhagic	18	18	
Hemiplegic location	Left side	20	21	1.000
	Right side	20	19	
History of atrial fibrillation	Yes	2	0	1.000
	No	38	40	
History of coronary heart disease	Yes	3	3	1.000
	No	37	37	
History of diabetes	Yes	11	12	1.000
	No	29	28	
History of smoking	Yes	20	19	1.000
	No	20	21	
History of drinking alcohol	Yes	19	18	1.000
	No	21	22	

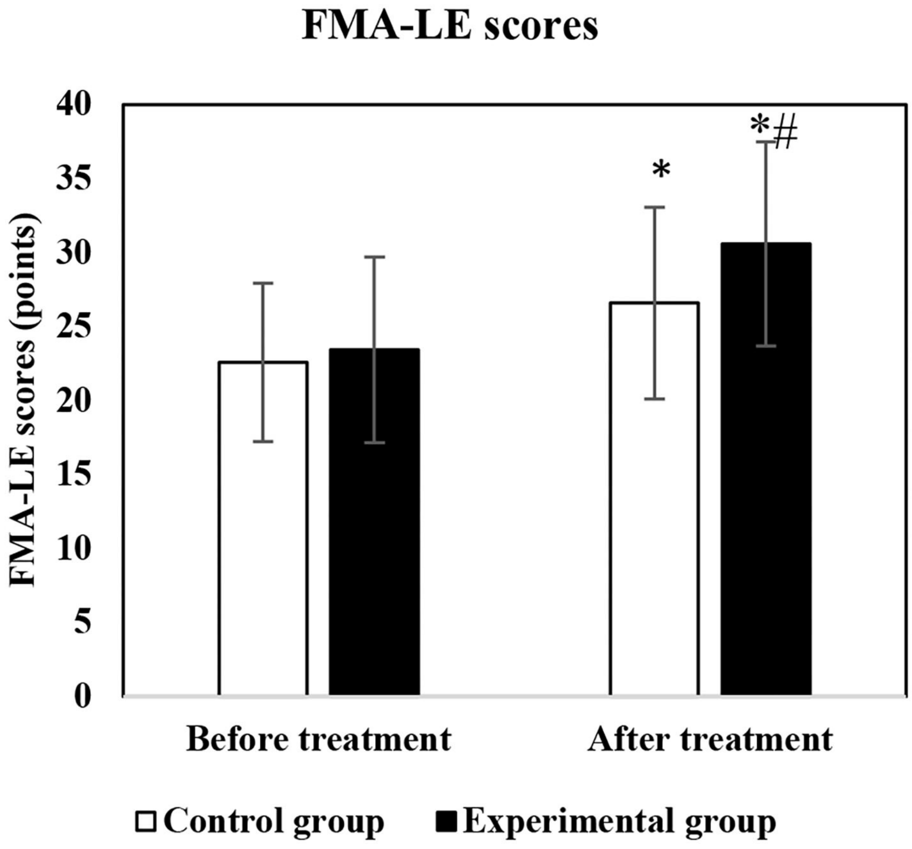


FIGURE 11 | Comparison of FMA-LE scores before and after treatment. * and # indicated that the difference was statistically significant compared with the data before treatment and that in the control group, respectively ($p < 0.05$).

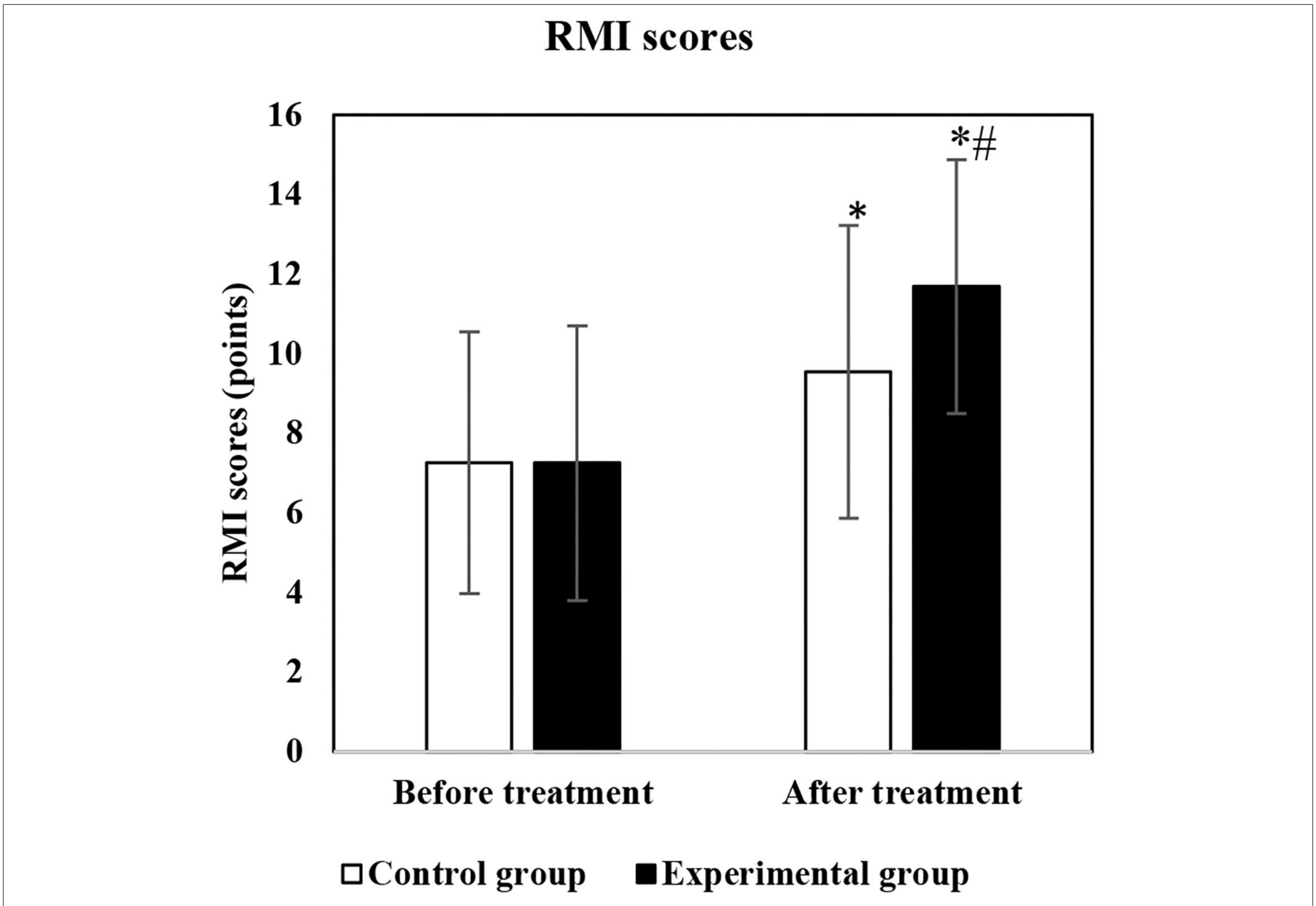


FIGURE 12 | Comparison of RMI scores before and after treatment. * and # indicated the statistically significant difference compared with the score before treatment and that of the control group, respectively ($p < 0.05$).

Comparison of FMA-LE Scores of Lower Limb Motor Function Between the Two Groups

Before treatment, there was no significant difference in FMA-LE scores between the two groups ($p > 0.05$). After 2 weeks of treatment, the FMA-LE scores of both the experimental group and the control group were higher than those before treatment, and the difference was statistically significant ($p < 0.05$). In addition, as shown in **Figure 11**, the FMA-LE score of the experimental group (10.58 ± 6.89) was significantly higher than that of the control group (26.57 ± 6.26), and the differences were statistically significant ($p < 0.05$).

Comparison of the Patients' Mobility Scores and Total Recovery Time Between Two Groups

Before treatment, there was no significant difference between the two groups of RMI scores ($p > 0.05$). After 2 weeks of treatment, the RMI scores of both groups were higher than those before treatment ($p < 0.05$). Moreover, the score of the experimental group was significantly higher than that of the control group ($p < 0.05$), as more details are shown in **Figure 12**.

Before treatment, no significant difference was found in the MBI scores between the two groups ($p > 0.05$). After 2 weeks of treatment, the MBI scores of both the experimental group and the control group were increased than those before treatment ($p < 0.05$). It is also shown in **Figure 13** that the BMI score of the experimental group was significantly higher than the score of the control group ($p < 0.05$).

The total rehabilitation time of the experimental group was significantly shorter than that of the control group (4.57 ± 1.21 vs. 6.23 ± 2.13) weeks ($p < 0.05$).

DISCUSSION

Stroke has gradually become the leading cause of death and disability in middle-aged and elderly people (Cipolla et al., 2018). Therefore, how to achieve the recovery of lower limb dysfunction quickly has become the primary goal of rehabilitation for stroke patients with hemiplegia. The bed-type rehabilitation robot was designed for intensive motor rehabilitation training, to help

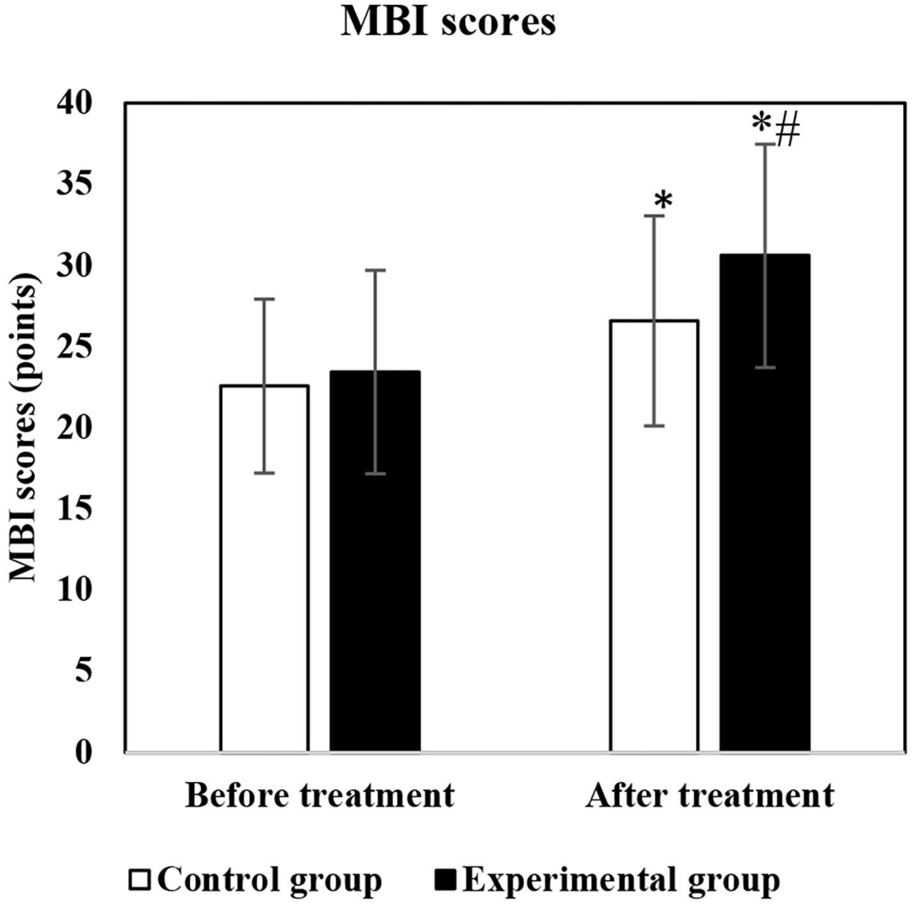


FIGURE 13 | Comparison of MBI scores before and after treatment. * and # indicated that statistically significant differences as the MBI score were compared with that before treatment and that of the control group, respectively ($p < 0.05$).

stroke patients with hemiplegia who lack the motor ability of lower limbs. First, the human–machine collaboration experiment system was constructed, and the software and hardware of the control system were designed. Then, the experimental platform for the lower limb rehabilitation training robot was set up, and the specific rehabilitation training methods for stroke patients with hemiplegia were determined by the contact force evaluation experiment.

The training control experiment was completed under BPNN and RBNN, to verify the feasibility of the robot control algorithm. The results showed that the prediction effect of BPNN was better than that of RBNN, which was consistent with what was obtained by Espinoza Bernal et al. (2021). Although the mechanical structure design, control system construction, and control method design of the bed-type rehabilitation robot were basically completed, the robot still needed to be further improved with the issues found in the experiments and the future development trend of robots. For the mechanical structure, the bed-type rehabilitation robot basically conformed to the trajectory of human walking joints merely, but cannot complete a real-time change. It was still necessary to further optimize the structure design later to work out a mechanical structure that was more in line with the walking of human body. In addition, due to the cantilever beam structure, the mechanical strength did not meet the expected requirements, and further improvements were needed in the future. For the robot control system, the single motor was mainly used to drive both lower limbs for rehabilitation training synchronously. It was temporarily unable to complete the single-leg training. In the subsequent control system design, drive motors should be added for the needs for separate rehabilitation training for the two lower limbs. Besides, the passive control training of this control system was only for the speed control of the servo motor, and the control accuracy was insufficient. In the follow-up, the joint information at the end of the rehabilitation training robot is needed to be fed back to complete the closed-loop control, so as to improve the control accuracy.

After stroke hemiplegia, the upper motor neuron of patients will be damaged, and the motor reflex of the lower center will be released, resulting in motor dysfunction. The main clinical manifestations are weakened muscle strength, increased muscle tone, and tendon hyperreflexia (Zhang et al., 2018; Kimura et al., 2019; Ratanapinunchai et al., 2019). The FMA-LE motor function scale was used to evaluate the lower extremity motor function of stroke patients with hemiplegia. Before treatment, no significant difference was in FMA-LE score between the two groups ($p > 0.05$). After 2 weeks of treatment, the FMA-LE scores were higher than those before treatment in both groups ($p < 0.05$), and the score of the experimental group was significantly higher than that of the control group ($p < 0.05$), which were consistent with the research results of Huang et al. (2019). This suggested that a machine learning-based bed-type rehabilitation robot combined with intensive motor training could significantly improve the lower limb motor function of stroke patients with hemiplegia, and the effect was better than that of single intensive motor training.

Motor dysfunction after stroke often leads to a decline in the mobility of patients with hemiplegia, thereby reducing the life quality of patients (Sethy and Sahoo, 2018). The RMI score and the MBI score were adopted to evaluate the mobility of stroke patients with hemiplegia. Before treatment, no significant difference was discovered between the two groups in RMI score as well as BMI score ($p > 0.05$). After 2 weeks of treatment, the RMI and BMI scores of both groups were higher than those before treatment ($p < 0.05$). Both RMI and BMI scores of the experimental group were significantly higher than those of the control group ($p < 0.05$), which were consistent with the results of Yao et al. (2020). It proved that a machine learning-based bed-type rehabilitation robot combined with intensive motor training improved the mobility of stroke patients with hemiplegia significantly, with a better effect than that under intensive motor training alone.

In recent years, as the incidence of stroke has gradually shown a younger trend, patients' expectations for rehabilitation are also increasing. The efficacy evaluation of the rehabilitation not only focuses on weakening muscle strength, increasing muscle tone, tendon hyperreflexia, etc., but also paid more attention to the application of limbs in real life (Yu et al., 2020). Therefore, in addition to treating the damaged structure and function of the patients, the ultimate goal is to make patients restore motor and social activities. Based on the conventional intensive motor training, the bed-type rehabilitation robot under machine learning could help to improve the motor function and walking function of the lower limbs of stroke patients with hemiplegia, further improving the ability to transfer, go up and downstairs, walk, and do other daily living activities. It aimed to make patients return to their family and society to the greatest extent, thus reducing the family and social burdens.

CONCLUSION

The human–machine collaboration experiment system was built with the software and hardware designs of the control system. The experimental platform for lower limb rehabilitation training robots was also established to determine the rehabilitation training methods for stroke patients with hemiplegia through the contact force evaluation experiment. It was aimed to discuss the effect of a machine learning-based bed-type rehabilitation robot combined with intensive motor training on the lower limb motor function of stroke patients with hemiplegia. The bed-type rehabilitation robot under machine learning combined with intensive motor training had the effect of improving the motor function and mobility of the lower limbs of stroke patients with hemiplegia. There were certain shortcomings shown. For the limitation of the study time, there were no long-term follow-ups. Thus, the patients needed to be followed up for a long time in the later period to verify the long-term efficacy. The included sample size was also too small to represent the training effect on all patients with stroke hemiplegia. It was necessary to increase the sample size for further clinical research in the future. It was believed that some ideas and experimental support were offered for the diagnosis and treatment of motor dysfunction in stroke patients with hemiplegia.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

REFERENCES

Alawieh, A., Zhao, J., and Feng, W. (2018). Factors affecting post-stroke motor recovery: Implications on neurotherapy after brain injury. *Behav. Brain Res.* 340, 94–101. doi: 10.1016/j.bbr.2016.08.029

Calabrò, R. S., Cacciola, A., Bertè, F., Manuli, A., Leo, A., Bramanti, A., et al. (2016). Robotic gait rehabilitation and substitution devices in neurological disorders: where are we now? *Neurol. Sci.* 37, 503–514. doi: 10.1007/s10072-016-2474-4

Cha, B., Lee, K. H., and Ryu, J. (2021). Deep-learning-based emergency stop prediction for robotic lower-limb rehabilitation training systems. *IEEE Trans. Neural. Syst. Rehabil. Eng.* 29, 1120–1128. doi: 10.1109/TNSRE.2021.3087725

Cipolla, M. J., Liebeskind, D. S., and Chan, S. L. (2018). The importance of comorbidities in ischemic stroke: Impact of hypertension on the cerebral circulation. *J. Cereb. Blood Flow Metab.* 38, 2129–2149. doi: 10.1177/0271678X18800589

D'Ancona, G., Ketterer, U., Kische, S., Murero, M., Feickert, S., Ortak, J., et al. (2020). Percutaneous left atrial appendage closure for cerebrovascular accident prevention: rationale, indications, technical aspects, clinical results and future perspective. *Fut. Cardiol.* 16, 237–250. doi: 10.2217/fca-2019-0086

Espinoza Bernal, V. C., Hiremath, S. V., Wolf, B., Riley, B., Mendonca, R. J., and Johnson, M. J. (2021). Classifying and tracking rehabilitation interventions through machine-learning algorithms in individuals with stroke. *J. Rehabil. Assist. Technol. Eng.* 8, 20556683211044640. doi: 10.1177/20556683211044640

Hellmich, B., Agueda, A., Monti, S., Buttgereit, F., Boysson, d. e., Brouwer, H., et al. (2020). 2018 Update of the EULAR recommendations for the management of large vessel vasculitis. *Ann. Rheum. Dis*. 79, 19–30. doi: 10.1136/annrheumdis-2019-215672

Huang, Y. C., Chen, P. C., Tso, H. H., Yang, Y. C., Ho, T. L., and Leong, C. P. (2019). Effects of kinesio taping on hemiplegic hand in patients with upper limb post-stroke spasticity: a randomized controlled pilot study. *Eur. J. Phys. Rehabil. Med*. 55, 551–557. doi: 10.23736/S1973-9087.19.05684-3

Kimura, D., Fukuda, I., Tsushima, T., Sakai, T., Umetsu, S., Ogasawara, Y., et al. (2019). Management of acute ischemic stroke after pulmonary resection: incidence and efficacy of endovascular thrombus aspiration. *Gen. Thorac. Cardiovasc. Surg*. 67, 306–311. doi: 10.1007/s11748-018-1024-9

Lim, J. Y., An, S. H., and Park, D. S. (2019). Walking velocity and modified rivermead mobility index as discriminatory measures for functional ambulation classification of chronic stroke patients. *Hong Kong Physiother. J*. 39, 125–132. doi: 10.1142/S1013702519500112

Luney, M. S., Lindsay, W., McKeever, T. M., and Moppett, I. K. (2020). Cerebrovascular accident and acute coronary syndrome and perioperative outcomes (CAPO) study protocol: a 10-year database linkage between Hospital Episode Statistics Admitted Patient Care, Myocardial Infarction National Audit Project and Office for National Statistics registries for time-dependent risk analysis of perioperative outcomes in English NHS hospitals. *BMJ Open*. 10, e037904. doi: 10.1136/bmjopen-2020-037904

Madhoun, H. Y., Tan, B., Feng, Y., Zhou, Y., Zhou, C., and Yu, L. (2020). Task-based mirror therapy enhances the upper limb motor function in subacute stroke patients: a randomized control trial. *Eur. J. Phys. Rehabil. Med*. 56, 265–271. doi: 10.23736/S1973-9087.20.06070-0

Pan, X. L. (2018). Efficacy of early rehabilitation therapy on movement ability of hemiplegic lower extremity in patients with acute cerebrovascular accident. *Medicine*. 97, e9544. doi: 10.1097/MD.0000000000009544

Park, J. S., Lee, S. H., Yoo, W. G., and Chang, M. Y. (2021). Immediate effect of a wearable foot drop stimulator to prevent foot drop on the gait ability of patients with hemiplegia after stroke. *Assist. Technol*. 33, 313–317. doi: 10.1080/10400435.2019.1634658

Ratanapinunchai, J., Mathiyakom, W., and Sungkarat, S. (2019). Scapular upward rotation during passive humeral abduction in individuals with hemiplegia post-stroke. *Ann. Rehabil. Med*. 43, 178–186. doi: 10.5535/arm.2019.43.2.178

Sarabadani Tafreshi, A., Riener, R., and Klamroth-Marganska, V. (2016). Distinctive steady-state heart rate and blood pressure responses to passive robotic leg exercise and functional electrical stimulation during head-up tilt. *Front. Physiol*. 7, 612. doi: 10.3389/fphys.2016.00612

Sethy, D., and Sahoo, S. (2018). Kinesiophobia after complex regional pain syndrome type one in a case of stroke hemiplegia and effect of cognitive behavior therapy. *Indian J. Psychiatry*. 60, 152–154. doi: 10.4103/psychiatry.IndianJPsychiatry_201_17

Suri, R., Rodriguez-Porcel, F., Donohue, K., Jesse, E., Lovera, L., Dwivedi, A. K., et al. (2018). Post-stroke movement disorders: the clinical, neuroanatomic, and demographic portrait of 284 published cases. *J. Stroke Cerebrovasc. Dis*. 27, 2388–2397. doi: 10.1016/j.jstrokecerebrovasdis.2018.04.028

Taghizadeh, G., Martinez-Martin, P., Meimandi, M., Habibi, S. A. H., Jamali, S., Dehmiyani, A., et al. (2020). Barthel Index and modified Rankin Scale: Psychometric properties during medication phases in idiopathic Parkinson disease. *Ann. Phys. Rehabil. Med*. 63, 500–504. doi: 10.1016/j.rehab.2019.08.006

Tang, Y., Wang, L., He, J., Xu, Y., Huang, S., and Fang, Y. (2021). Optimal method of electrical stimulation for the treatment of upper limb dysfunction after stroke: a systematic review and bayesian network meta-analysis of randomized controlled trials. *Neuropsychiatr. Dis. Treat*. 17, 2937–2954. doi: 10.2147/NDT.S332967

Tomida, K., Sonoda, S., Hirano, S., Suzuki, A., Tanino, G., Kawakami, K., et al. (2019). Randomized controlled trial of gait training using gait exercise assist robot (GEAR) in stroke patients with hemiplegia. *J. Stroke Cerebrovasc. Dis*. 28, 2421–2428. doi: 10.1016/j.jstrokecerebrovasdis.2019.06.030

Tsuchimoto, S., Shindo, K., Hotta, F., Hanakawa, T., Liu, M., and Ushiba, J. (2019). Sensorimotor connectivity after motor exercise with neurofeedback in post-stroke patients with hemiplegia. *Neuroscience*. 416, 109–125. doi: 10.1016/j.neuroscience.2019.07.037

Uwatoko, H., Nakamori, M., Imamura, E., Imura, T., Okada, K., Matsumae, Y., et al. (2020). Prediction of independent gait in acute stroke patients with hemiplegia using the ability for basic movement scale II score. *Eur. Neurol*. 83, 49–55. doi: 10.1159/000506421

Xia, P., Yang, T., Wang, X., and Li, X. (2021). Combination of pregabalin and transcutaneous electrical nerve stimulation for neuropathic pain in a stroke patient after contralateral C7 nerve transfer: a case report. *Int. J. Neurosci*. 131, 1248–1253. doi: 10.1080/00207454.2020.1786687

Yang, F., Chen, L., Wang, H., Zhang, J., Shen, Y., Qiu, Y., et al. (2021). Combined contralateral C7 to C7 and L5 to S1 cross nerve transfer for treating limb hemiplegia after stroke. *Br. J. Neurosurg*. 10, 1–4. doi: 10.1080/02688697.2021.1910764

Yao, D., Stukenborg-Colsman, C., Ettinger, S., Claassen, L., Plaass, C., Martinelli, N., et al. (2020). Subjective outcome following neurostimulator implantation as drop foot therapy due to lesions in the central nervous system-midterm results. *Musculoskelet Surg*. 104, 93–99. doi: 10.1007/s12306-019-00604-3

Yu, X. M., Jin, X. M., Lu, Y., Gao, Y., Xu, H. C., Xue, X., et al. (2020). Effects of body weight support-tai chi footwork training on balance control and walking function in stroke survivors with hemiplegia: a pilot randomized controlled trial. *Evid. Based Compl. Alternat. Med*. 2020, 9218078. doi: 10.1155/2020/9218078

Zhang, C., Huang, M. Z., Kehs, G. J., Braun, R. G., Cole, J. W., and Zhang, L. Q. (2021). Intensive in-bed sensorimotor rehabilitation of early subacute stroke survivors with severe hemiplegia using a wearable robot. *IEEE Trans. Neural. Syst. Rehabil. Eng*. 29, 2252–2259. doi: 10.1109/TNSRE.2021.3121204

Zhang, S., Chen, M., Gao, L., and Liu, Y. (2018). Investigating muscle function after stroke rehabilitation with 31P-MRS: a preliminary study. *Med. Sci. Monit*. 24, 2841–2848. doi: 10.12659/MSM.907372

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

frontiers | Frontiers in Neurorobotics

ORIGINAL RESEARCH
published: 13 June 2022
doi: 10.3389/fnbot.2022.905313

Neuromorphic Signal Filter for Robot Sensoring

Luis M. García-Sebastián*, Victor H. Ponce-Ponce*, Humberto Sossa, Elsa Rubio-Espino and José A. Martínez-Navarro

Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico City, México

Noise management associated with input signals in sensor devices arises as one of the main problems limiting robot control performance. This article introduces a novel neuromorphic filter model based on a leaky integrate and fire (LIF) neural model cell, which encodes the primary information from a noisy input signal and delivers an output signal with a significant noise reduction in practically real-time with energy-efficient consumption. A new approach for neural decoding based on the neuron-cell spiking frequency is introduced to recover the primary signal information. The simulations conducted on the neuromorphic filter demonstrate an outstanding performance of white noise rejecting while preserving the original noiseless signal with a low information loss. The proposed filter model is compatible with the CMOS technology design methodologies for implementing low consumption smart sensors with applications in various fields such as robotics and the automotive industry demanded by Industry 4.0.

Keywords: neuromorphic, filter, CMOS, low-frequency, sensoring

OPEN ACCESS

Edited by:
Florian Röhrbein,
Technische Universität
Chemnitz, Germany

Reviewed by:
Dante Mujica-Vargas,
Centro Nacional de Investigación y
Desarrollo Tecnológico, Mexico
Enrique Garcia-Trinidad,
Tecnológico de Estudios Superiores
de Huixquilucan, Mexico
Alejandro Zacarías,
Instituto Politécnico Nacional
(IPN), Mexico

****Correspondence:***
Luis M. García-Sebastián
lgarcias2020@cic.ipn.mx
Victor H. Ponce-Ponce
vponce@cic.ipn.mx

Received: *26 March 2022*
Accepted: *09 May 2022*
Published: *13 June 2022*

Citation:
García-Sebastián LM,
Ponce-Ponce VH, Sossa H,
Rubio-Espino E and
Martínez-Navarro JA (2022)
Neuromorphic Signal Filter for Robot
Sensoring.
Front. Neurorobot. 16:905313.
doi: 10.3389/fnbot.2022.905313

1. INTRODUCTION

The term neuromorphic, coined by Mead (1990), refers to Very Large Scale of Integration (VLSI) systems aiming to reproduce biological neuron behaviors. Neuromorphic computing platforms are relatively simple regarding the number of active elements (transistors) compared to complex traditional digital units (microprocessors) to replicate brain-like responses. Today, the convergence of electronics, computing science, and neuroscience offers bountiful inspiration to explore novel hardware structures, algorithms, and innovative ways to process information more efficiently, maintaining low levels of energy waste and material use (Schuman et al., 2017). One of the most remarkable contributions of this inter-discipline convergence is the conception of spiking neurons (SN), also called the third generation of artificial neurons (Maass, 1997). The main difference concerning previous generations is the inclusion of temporal information in the computing process, and this feature offers the possibility to process signals efficiently with variations across time. Unfortunately, the large-scale modeling of SN units is limited due to the high computational cost involved in solving numerically the whole set of differential equations representing each SN unit. Therefore, the design and implementation of these units are more convenient at the silicon plane and in the analog domain to overpass this vast amount of numerical computation effort.

Traditional analog filters are designed based on scaling specific frequency domain signal components and attenuating the rest. This approach has been proven effective with noise that is primarily out of signal frequency range. However, linear filters cannot clear noisy signals when disturbance affectation is in the same frequency range as primary signal. In this context, digital filters, especially average filter techniques, take precedence at the cost of resources expense. For this reason, some filter proposals based on the use of several SNs have been made

(Orchard et al., 2021; Sharifshazileh et al., 2021). Generally, integrated circuits hosting neuromorphic implementations possess an inherent capacity to extract primary features of given entries since integrals tied to the SN model can be interpreted as the average value operator on a time window. Thus, neuromorphic systems allow average filtering while retaining the benefits of analog circuits.

2. METHODS

2.1. Neural Circuit

There are several proposals reported on analog implementations of neural model circuits (Abbott, 1999; Wijekoon and Dudek, 2008; Zamarreño-Ramos et al., 2011; Wu et al., 2015; Zare et al., 2021). Throughout the development of this study, the leaky integrate and fire neuron (IFN) circuit, proposed in Wu et al. (2015) was used, as seen in **Figure 1**. However, the proposed methodology could be easily adapted to work with other neural circuit models. This proposal is divided into two main parts, a leaky, current integrator circuit (LI) to emulate the behavior of a neuron during the period of depolarization and a reset engine that returns the output voltage of the operational amplifier (V_{mem}) to a reference voltage level (V_{Ref}). It also assumes the generation of a convenient spike shape, compliant with memristors technology to allow weight adjustment during the learning phase.

Wu's neural circuit shown in **Figure 1** operates in the integration and firing mode. At the integration mode, the OPAMP works as a leaky integrator, over the current, I_{in}, flowing at its negative input. At the integration mode, the voltage level at the output of the OPAMP decreases until it reaches a determined threshold voltage, V_{thr}. The comparator circuit compares the membrane voltage, V_{mem}, with v_{thr}, to generate a signal activating the Phase Control block when the descending V_{ref} reaches V_{thr}. At this moment, the Phase Control block commands the Spike Generator block to initiate a spike event with a predefined waveform and it changes the control signals, α_{fire} to ON state, while α_{int} to OFF state. These control signals are complementary. The neural circuit is reconfigured by the current states of α_{fire} and α_{int}. If α_{int} is ON, the OPAMP works in the integration mode, if α_{fire} is ON the neuron is in fire mode. During integration mode the neuron output V_{out} is set in V_{ref}, at the same time, the spike generator block must hold a V_{ref} at the positive OPAMP input, which is buffered at the negative OPAMP input. In the firing mode, V_{out}, is connected to V_{mem}, generating a spike event, feedbacking, V_{mem} to the negative OPAMP input. At the end of the firing mode, C_{mem} is reset to a V_{ref}, potential.

The equivalent model of the LI section is presented in Equation (1).

$$\frac{dV_{mem}}{dt} = \frac{V_p - V_{mem}(t)}{R_{leak}C_{mem}} - \frac{I_{in}}{C_{mem}} \tag{1}$$

Where V_p is the voltage objective, while the circuit is in integration mode V_p corresponds to V_{thr}.

Wu's circuit functioning could still be simplified to implement the proposed methodology, performing a noise signal filtering process. The simplification consists of establishing constant delays before switching states and restarting the integration phase. It imposes a period of neuron inactivity corresponding to the refractory period, seen in biological brains. This behavior is modeled as shown below:

$$\begin{gathered}\textbf{if } V_{mem} \approx V_{thr} \\ \textbf{then } V_p \leftarrow V_{ref},\ R_{leak} \leftarrow 1,\ I_{in} \leftarrow 0\end{gathered}$$

Once the neuron is in the refractory period, it maintains its state for a predefined period, after which it returns to the previous (integration) state.

2.2. Tuning Curves

Since spike trains convey information through their timing and any spike-wave produced by neural circuits models are supposed to be identical (Gerstner et al., 2016), the membrane's potential in neuromorphic circuits can be characterized simply by a list of events: $t^0, t^1, ..., t^n$, where $0 \leq t^i \leq T$, with $i = 0, 1, 2, ..., n$ is the *i-th* spike time in an observed period T (Dayan and Abbot, 2001). **Figure 2** shows a representation of this list.

A simple way to characterize the response of a neuromorphic circuit is by counting the number of peak voltages fired during the presentation of a stimulus (input current). By repeating this operation for a certain number of different stimuli, it is possible to estimate a function, f, that describes the relationship between an input current, I_{in}, and a frequency of spikes fr (Dayan and Abbot, 2001; Elliasmith, 2013). In this study, an alternative way to estimate the neuron frequency is proposed. Since neuromorphic circuits have no stochastic behavior, it is possible to prove that the same neuron frequency response will always be obtained for a given I_{in}. Therefore, by measuring the time elapsed between the event of two spikes, $p = t^n - t^{n-1}$, the frequency is obtained by using $\frac{1}{p}$.

2.3. Mean Value Theorem for Integrals

The time between spikes in the circuit presented in **Figure 1** corresponds to the mean value of the input current.

Rearranging elements from Equation (1), we find the next expression.

$$I_{in} = \frac{V_p - V_{mem}(t)}{R_{leak}} - C_{mem}\frac{dV_{mem}}{dt}, \tag{2}$$

Equation (2) corresponds to Current Kirchhoff's Law, producing a summation of all the currents at input node, $I_{in} = I_{R_{Leak}} + I_{C_{mem}}$, where $I_{R_{Leak}}(t)$ is the current across R_{Leak}, which behavior is unknown in advance, thus:

$$I_{in} = I_{R_{Leak}}(t) - C_{mem}\frac{dV_{mem}}{dt}. \tag{3}$$

Now, integrating both sides of Equation (3) with the defined time intervals limits between neural events (spike occurrences) results in Equation (4). Internal values of the neuromorphic units are reset at the end of each neural event,

$$\int_{t^{n-1}}^{t^n} I_{in}dt = \int_{t^{n-1}}^{t^n} I_{R_{Leak}}(t)dt - \int_{V_{ref}}^{V_{thr}} C_{mem}dV_{mem} \tag{4}$$

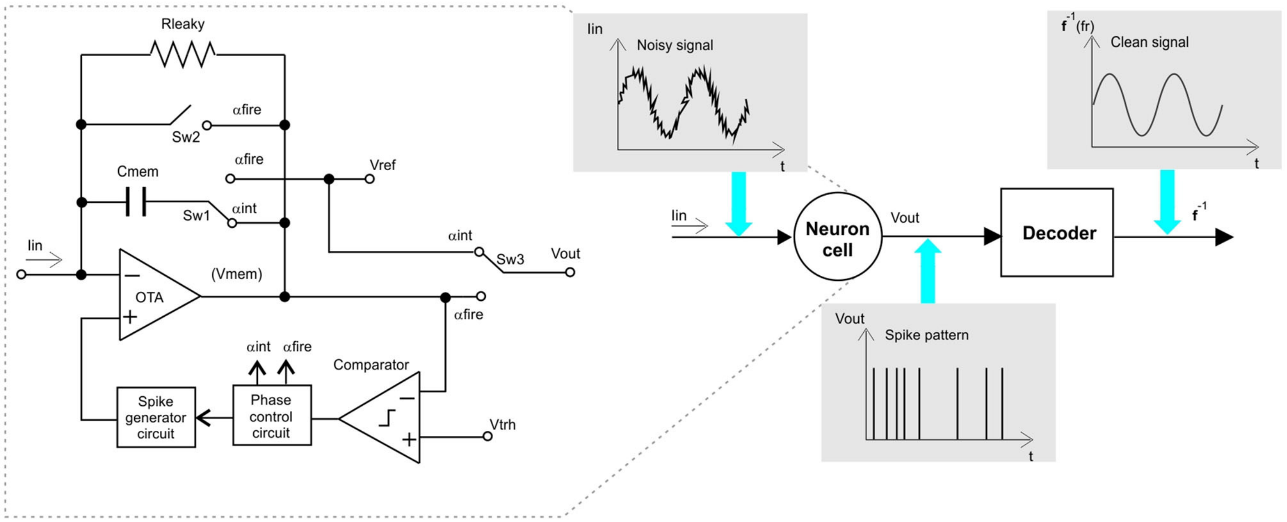


FIGURE 1 | Neuromorphic filter architecture, showing a spiking neuron CMOS circuit implementation, see Wu et al. (2015).

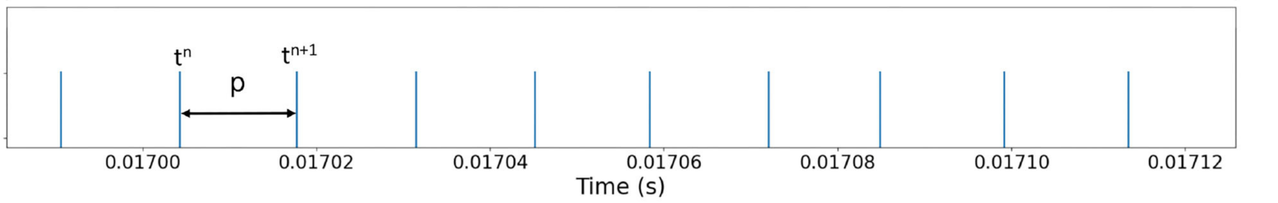


FIGURE 2 | Neuron circuit spike response. Since spike waveform is not needed for the filtering process, spike events are registered as a list of times when the neuron circuit reach threshold voltage. Time elapsed between t^n and t^{n+1} is denoted as p.

Solving integrals on both sides:

$$I_{in}(t^n - t^{n-1}) = \int_{t^{n-1}}^{t^n} I_{R_{Leak}}(t)dt - C_{mem}V_{mem} \tag{5}$$

Where the value $V_{mem} = V_{thr} - V_{ref}$ results at the end of the integration period. Equation (5) corresponds to the Mean Value Theorem for integrals (Stewart, 2018). Thus, a constant value of I_{in} exists such that applied for the time interval, $t^n - t^{n-1}$, equals the value of the current $I_{R_{Leak}}(t)$ on the same period. Particularly, I_{in} can be seen as the mean value of current on the period $t^n - t^{n-1}$ plus a constant value ($C_{mem}V_{mem}$).

3. PROPOSED METHODOLOGY FOR A NEURAL FILTER DESIGN

Our proposal consists of using the tuning curve function of the neuromorphic circuit to estimate the I_{in} value on Equation (5). The methodology proposed to use a neural circuit as a signal filter is depicted in **Figure 3**.

The tuning curve for the circuit introduced in **Figure 4**, is obtained by sweeping the current I_{in} of Equation (1) between a current interval $\in [0, 300]\mu A$, considering the following electrical and timing parameters: $C_{mem} = 1\mu F$, $R_{leak} = 10k\Omega$, and a refractory period of $10\mu s$, we proceed to measure the time elapsed between potential membrane spikes. The below equation is proposed as a prototype to estimate function f.

$$f(I_{in}) = ln(I_{in} + a)b - c \tag{6}$$

Parameters $a = 1724.8761$, $b = 21.6051$, $c = -161.1285$, are determined using nonlinear least squares curve fitting (Virtanen et al., 2020).

Because Equation (6) is invertible, we can take two produced spikes and calculate the current in the elapsed period between spikes. That is to say, f^{-1} computes the equivalent input current value in the system (I_{in}).

$$f^{-1}(fr) = I_{in}(fr) = \beta\left[e^{\frac{fr-c}{b}} - a\right] \tag{7}$$

Where:

$$fr = \frac{1}{\alpha p}$$

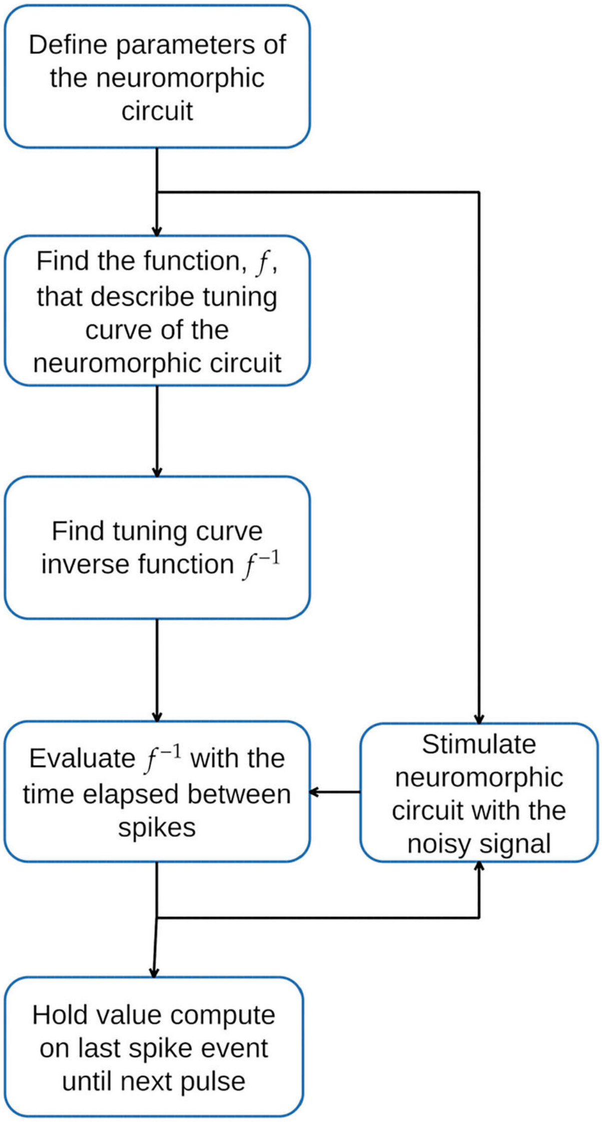

Define parameters of the neuromorphic circuit
Find the function, f, that describe tuning curve of the neuromorphic circuit
Find tuning curve inverse function f^{-1}
Evaluate f^{-1} with the time elapsed between spikes
Stimulate neuromorphic circuit with the noisy signal
Hold value compute on last spike event until next pulse

FIGURE 3 | Proposed Methodology. First, define the parameters of the neural circuit. Second, characterize the response to obtain the circuit tuning curve function, f, and its inverse function. Finally, stimulate the neural circuit with the noisy input signal and use the time elapsed since the last spike to evaluate the inverse function f^{-1}. The value computed using f^{-1} is held until the next event occurs.

with: $p = t^n - t^{n-1}$. t^n is the time of the n-th spike. In order to maintain values on a more convenient time-scale $\alpha = 1 \times 10^4$ and $\beta = 1 \times 10^{-6}$ are added as scale factors. Equation (7) is evaluated at each spike and the value is held until the next spike occurs, refer to **Figure 5B**.

Observe that Equation (7) is a decaying exponential function; therefore, it is possible to define a circuit that reproduces this behavior by using the capacitor discharging dynamic in a commuted capacitor scheme working as follows. At each spike event, a low impedance branch quickly charges a capacitor during the refractory period of the SN unit. Once the refractory period concludes, the charging branch for the capacitor is open, and discharge becomes through a branch with fixed impedance such that the current on the capacitor has a behavior similar to Equation (7), refer to **Figure 5A**. Once a new spike event occurs, the current value of the capacitor is registered and held until the next spike event.

This scheme based on frequency shows a better performance than other strategies previously introduced (Dupeyroux et al., 2021; Guo et al., 2021) since it demonstrates good noise mitigation capacity employing only one neuron.

4. EXPERIMENTS AND RESULTS

To demonstrate the performance of our proposal, the following experiment was conducted. First, synthetic white noise is simulated to ensure a critical noise condition affectation over the clean signal with a uniform frequency distribution (Grinsted, 2022) (**Figure 6**). Second, the white noise is added to an arbitrary signal, refer to **Figure 7**. Finally, the noisy signal is used as the input for Equation (1), and the equivalent current output is computed using Equation (7). The results are shown in **Figure 8**.

It is possible to appreciate a significant noise reduction after the rebuilding operation. **Figure 9** shows the Power Spectral Density of both original and noisy signals, and preservation of fundamental frequency is observed, thus we could conclude that the recovered signal is a good approximation of the original one. Notice that fundamental frequencies of the original signal are within the frequency range of noise.

In order to compare this proposal with other approaches, the same noisy signal was filtered using linear (Chebyshev, Butterworth Butterworth, 1930, Elliptic) and digital (median Tukey, 1977) filters (Virtanen et al., 2020). **Figure 10** shows the output responses obtained from these standard filters. **Figure 11** shows the error measure of each filter, computed as the difference between the original and the output of the corresponding filter. Additional experiments were conducted using Gaussian Multiplicative Noise and Impulsive Random Noise, results are shown in **Figures 12**, **13**.

To define a figure of merit for the proposed filter, we measure the distance between each filter response and the original signal (noiseless signal). As each filter introduces a different amount of time delay caused by the filtering process, Euclidean distance is not an appropriate choice since filtering techniques with a minimum delay will tend to render better results. Therefore, Fast Dynamic Time Wrapping (Salvador and Chan, 2007) (FastDTW) was used as a performance evaluation criterion. Comparisons of error results applying both Euclidean distance and FastDTW are shown in **Table 1**.

5. CONCLUSION

This study introduced the capacity and performance of simulated spiking neural network circuits to recognize primary signal

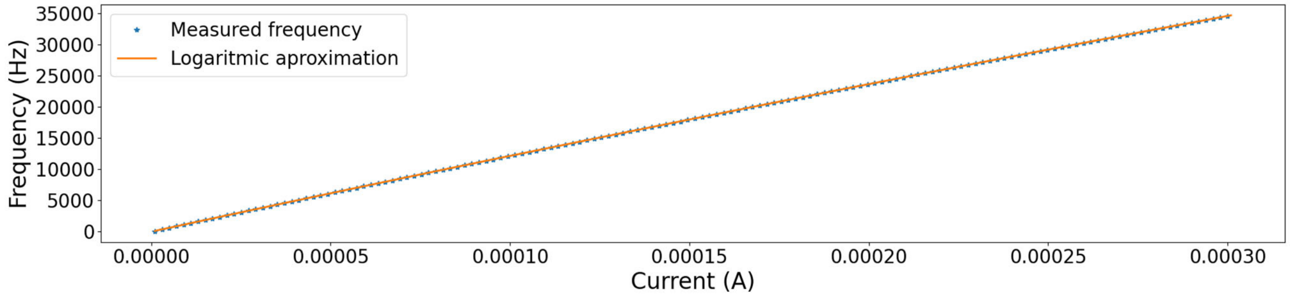


FIGURE 4 | Tuning curve of the used circuit. Marks shows measurements frequencies obtained by sweeping the current I_{in} of Equation (1) in interval $0 - 300\mu A$, with electrical and timing parameters: $C_{mem} = 1\mu F$, $R_{leak} = 10k\Omega$, and a refractory period of $10\mu s$. The orange line shows approximation by 7. * refers to the measured frequency of spikes. - refers to measurments approximation made by Equation (6).

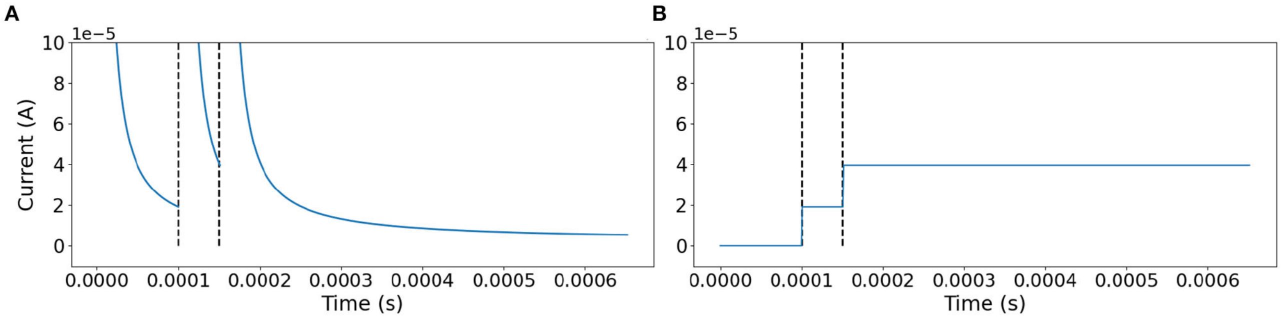


FIGURE 5 | Signal rebuild scheme. **(A)** Behavior of Equation (7) evaluated at each step of the simulation. **(B)** At each spike Equation (7) is evaluated with *p* equal to the time elapsed from the last spike, and the value is preserved until the next event is reached. The dotted line marks the time of two spikes.

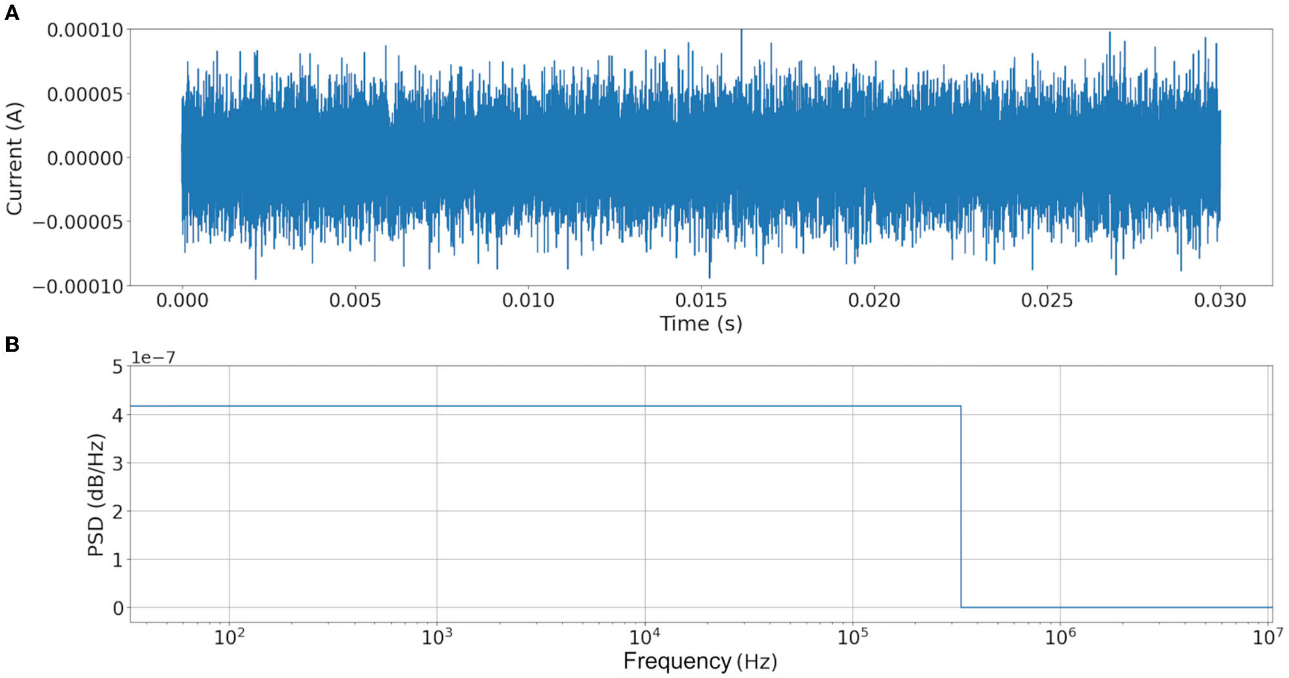


FIGURE 6 | Synthetic white noise generate for this study. **(A)** Noise signal added. **(B)** Fast Fourier Transform of noise. It is possible to observe that the noise has a frequency uniform distribution between 1Hz and 31.6228×10^4 Hz.

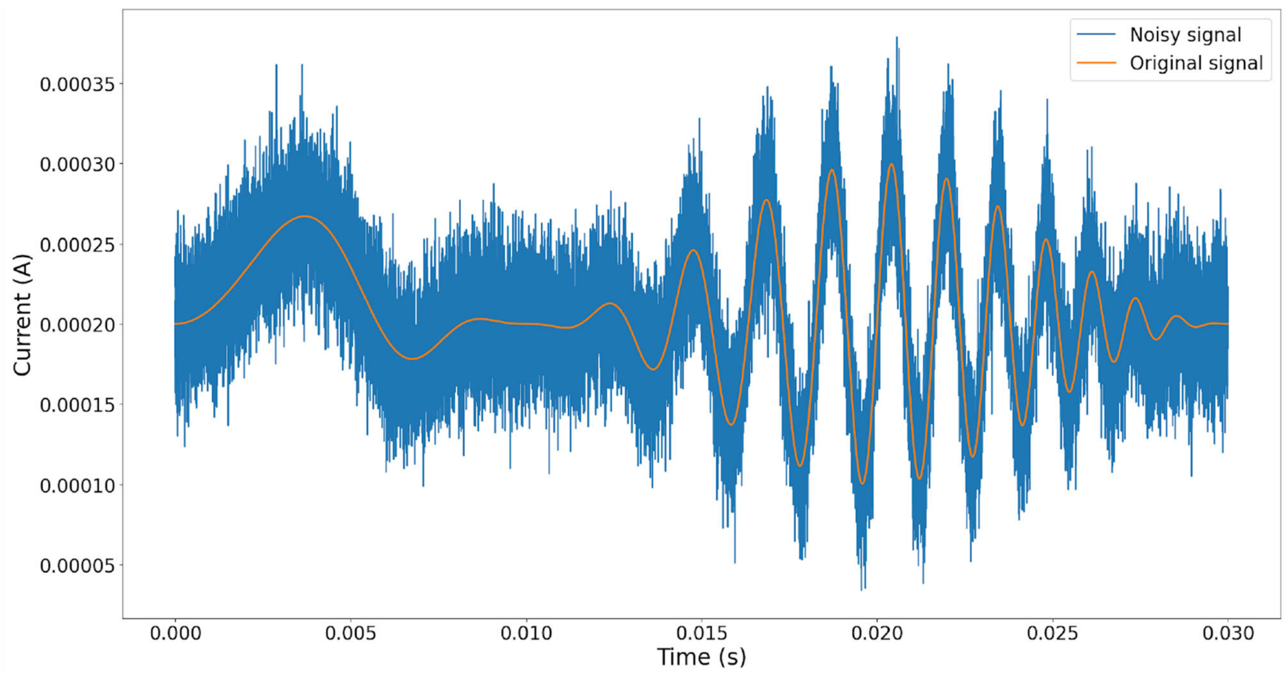


FIGURE 7 | Comparison between the original signal with the noisy signal. For this experiment, the signal was computed evaluating the function $\frac{sin(1500t^2)cos(50t)}{50\times10^6}$ $+200 \times 10^{-6}$. It is important to observe that the signal must be positive at any moment.

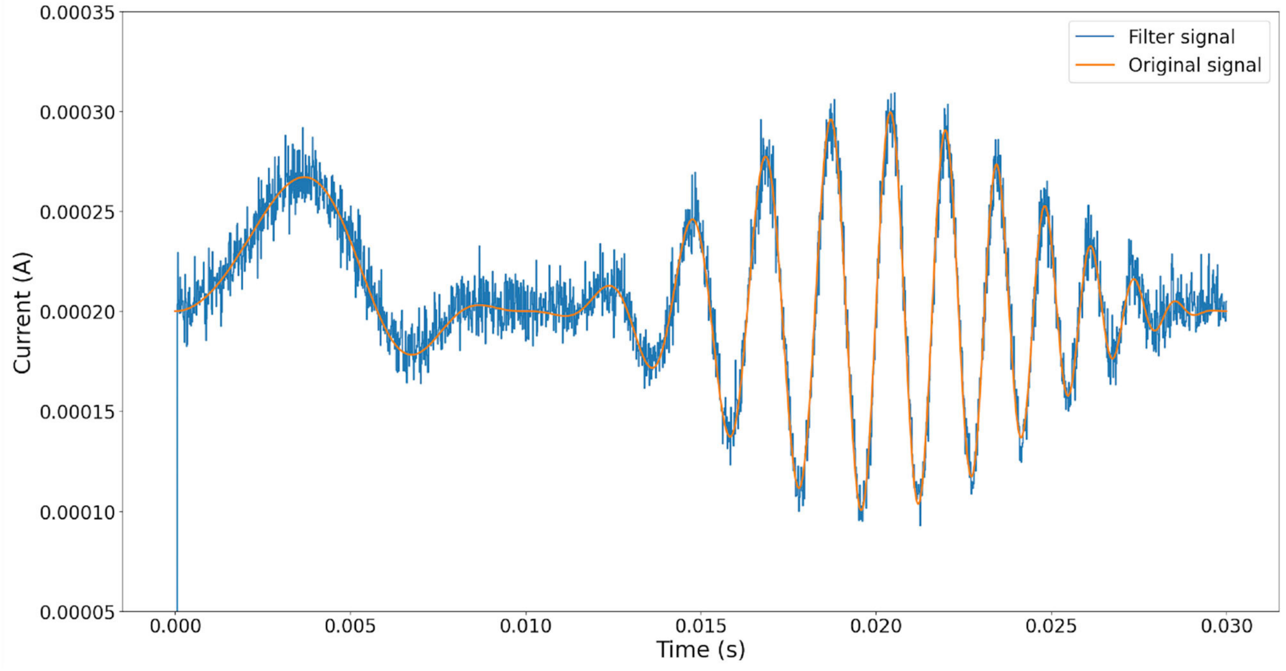


FIGURE 8 | Comparison between signal rebuild from spike frequency on neuron output (blue line) and original signal (orange line).

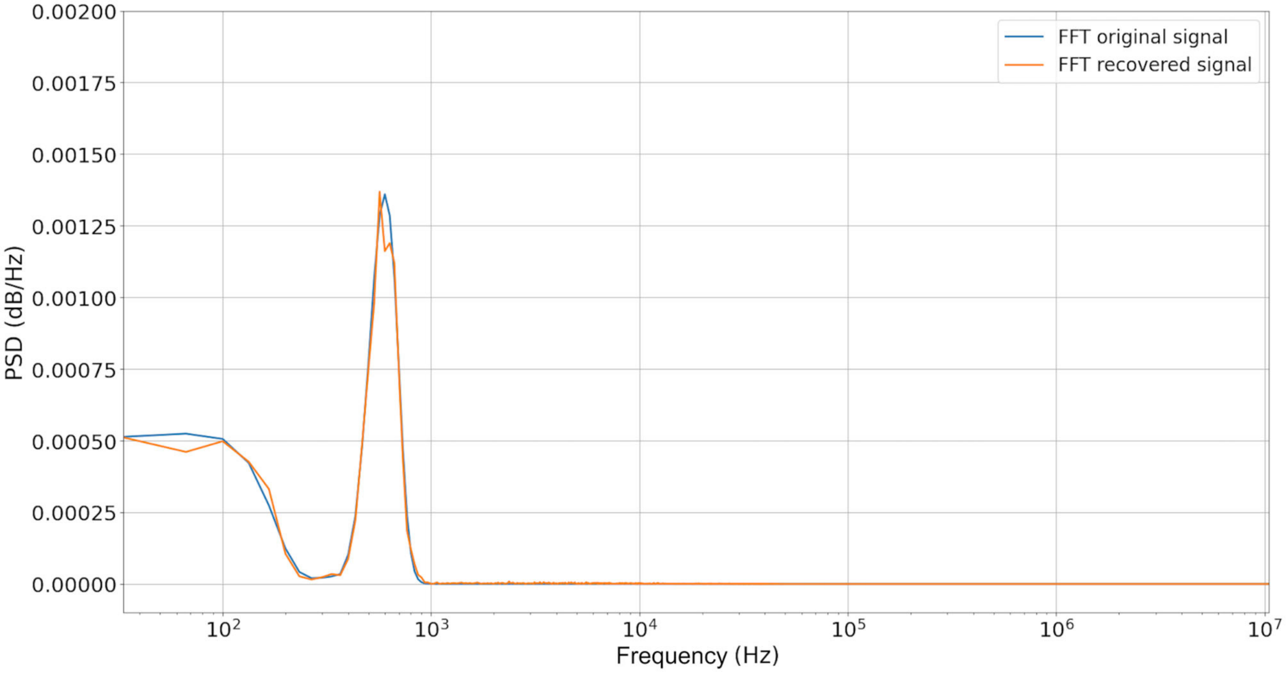


FIGURE 9 | Power Spectral Density graph of the original signal and filter signal by our proposal. We can observe than primary frequencies are presented with low degradation.

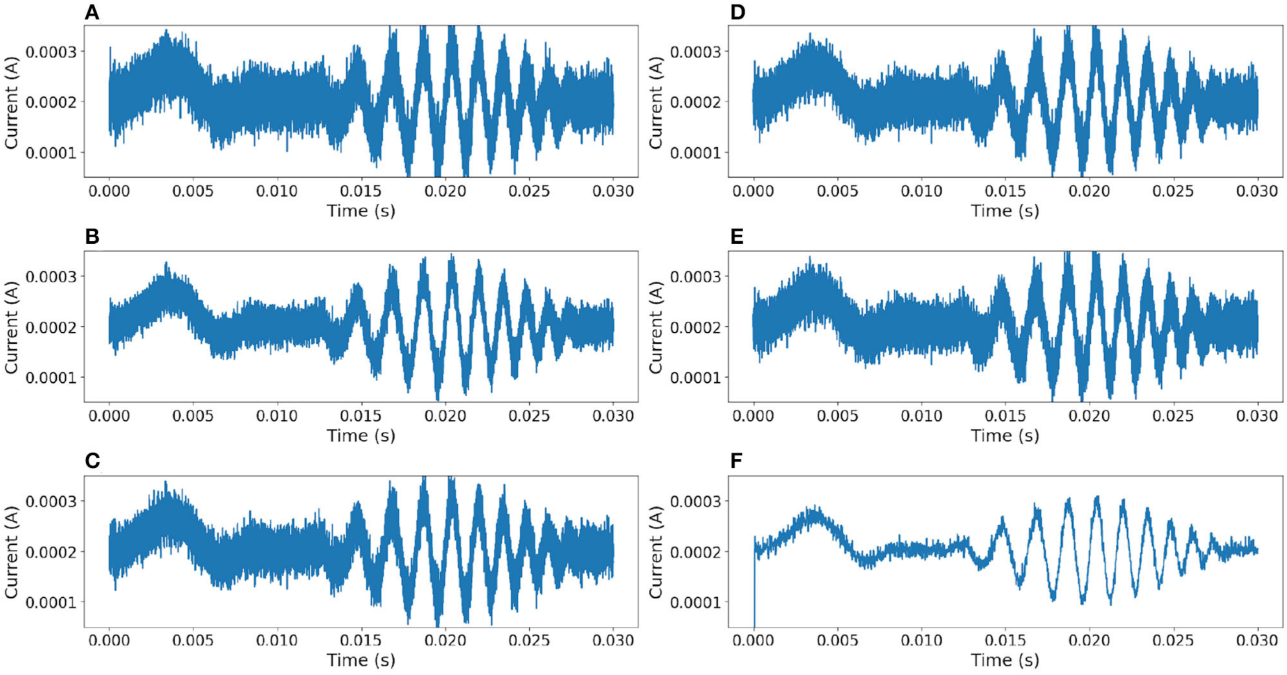


FIGURE 10 | Comparison of results between different filters. **(A)** Median filter. **(B)** 3rd-order Chebyshev type 1 filter with cutoff frequency at 55*kHz*. **(C)** 3rd-order Chebyshev type 2 filter with cutoff frequency at 55*kHz*. **(D)** 3rd-order Butterworth filter with cutoff frequency at 55*kHz*. **(E)** 3rd-order Elliptic filter with cutoff frequency at 55*kHz*. **(F)** Neural filter proposed.

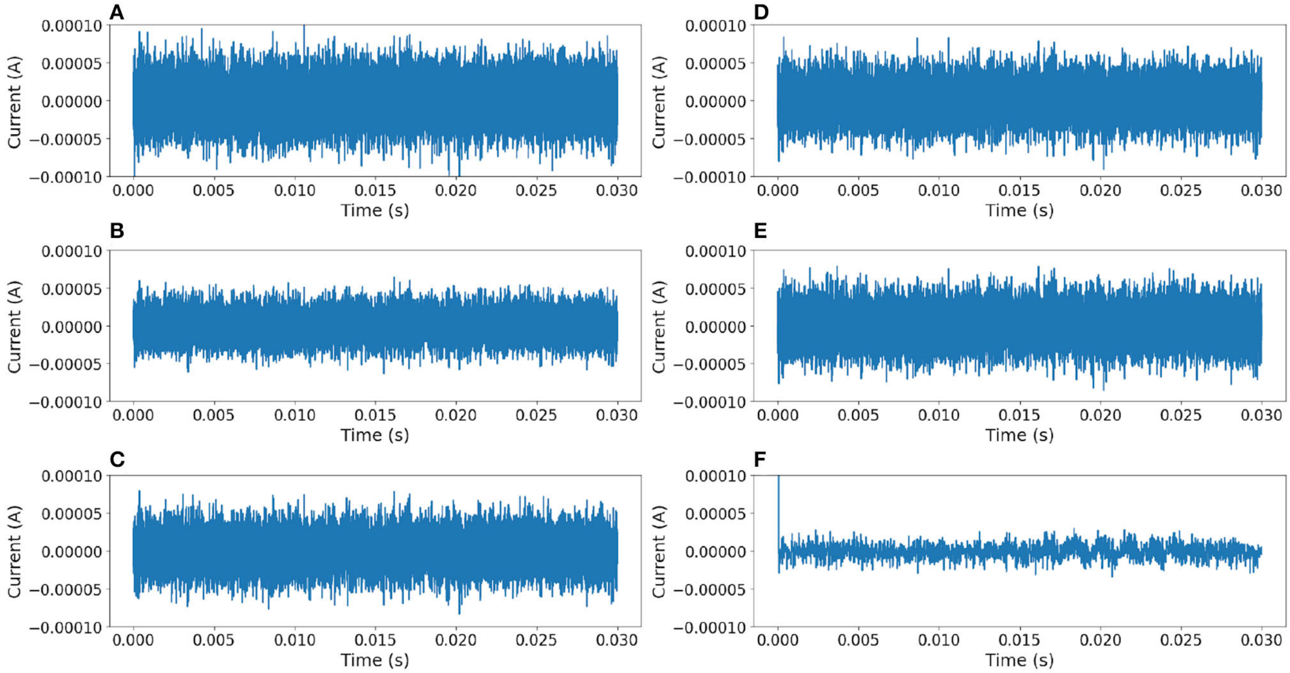


FIGURE 11 | Comparison of error between outputs of different filters (additive white noise) and the original signal. The ideal error signal must be 0 at any time. **(A)** Median filter. **(B)** 3rd-order Chebyshev type 1 filter. **(C)** 3rd-order Chebyshev type 2 filter. **(D)** 3rd-order Butterworth filter. **(E)** 3rd-order Elliptic filter. **(F)** Neural filter proposed.

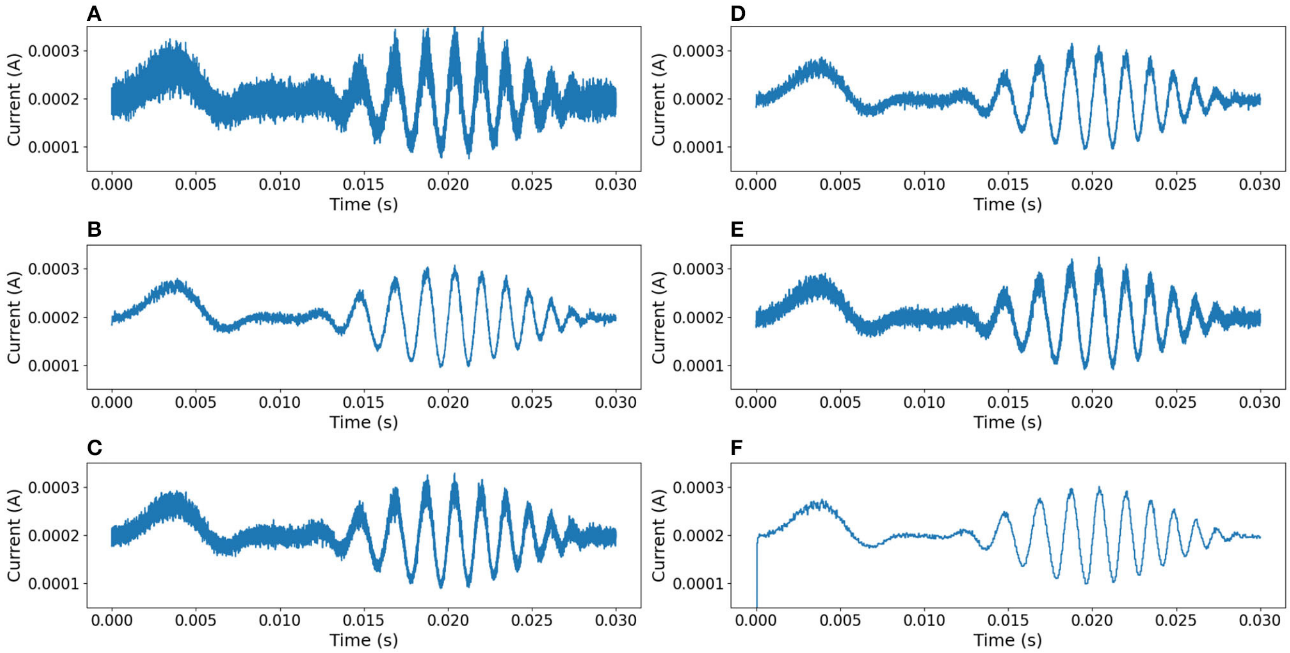


FIGURE 12 | Comparison of results between different filters, using a signal affected by Gaussian Multiplicative Noise of 30% . **(A)** Median filter. **(B)** 3rd-order Chebyshev type 1 filter with cutoff frequency at 55*kHz*. **(C)** 3rd-order Chebyshev type 2 filter with cutoff frequency at 55*kHz*. **(D)** 3rd-order Butterworth filter with cutoff frequency at 55*kHz*. **(E)** 3rd-order Elliptic filter with cutoff frequency at 55*kHz*. **(F)** Neural filter proposed.

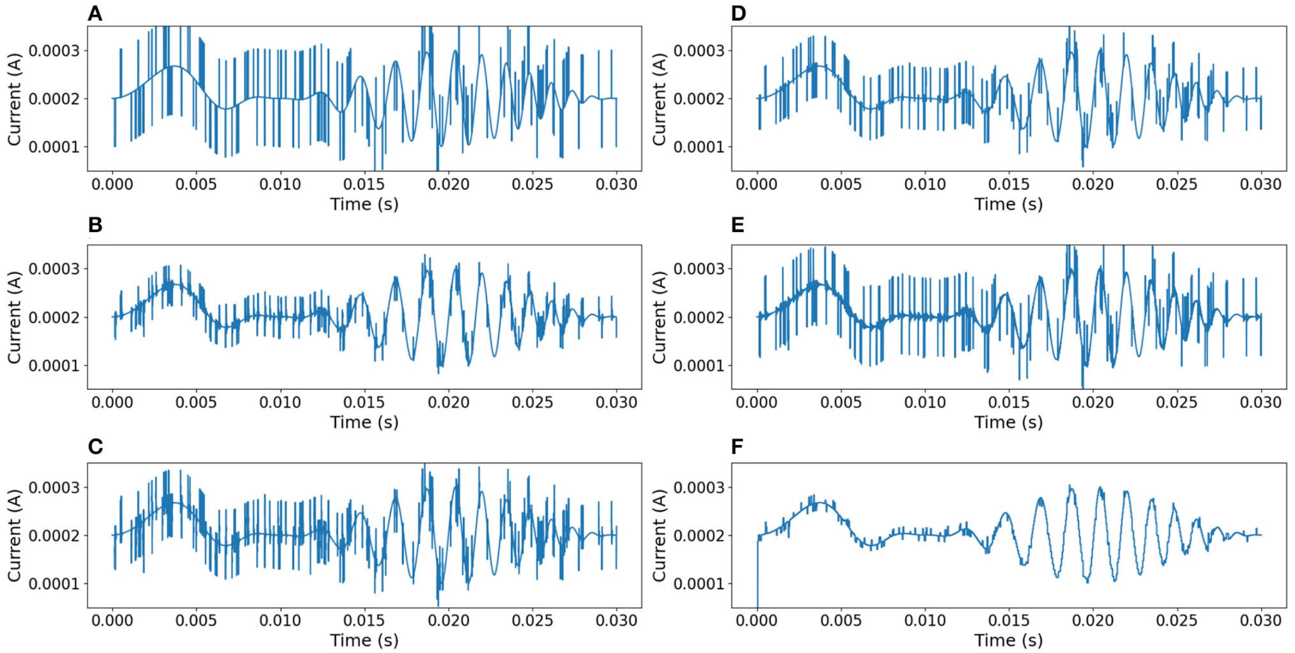


FIGURE 13 | Comparison of results between different filters. The signal is affected by impulsive random noise (random pulses of 5μs and 0.1*mA* were added). **(A)** Median filter. **(B)** 3rd-order Chebyshev type 1 filter with cutoff frequency at 55*kHz*. **(C)** 3rd-order Chebyshev type 2 filter with cutoff frequency at 55*kHz*. **(D)** 3rd-order Butterworth filter with cutoff frequency at 55*kHz*. **(E)** 3rd-order Elliptic filter with cutoff frequency at 55*kHz*. **(F)** Neural filter proposed.

TABLE 1 | Comparison between filter performance with Additive White Noise, Multiplicative Gaussian Noise, and Impulsive Random Noise.

Filter	Euclidean	FDTW	MSE	PSNR*
Additive white noise				
Median filter	0.257062	2.66536	4.40539×10^{-09}	15.6014
3^{rd}-order Chebyshev type-1 filter	0.0300501	0.246951	6.02007×10^{-11}	34.2452
3^{rd}-order Chebyshev type-2 filter	0.0575698	0.576117	2.20952×10^{-10}	28.5982
3^{rd}-order Butterworth filter	0.0378712	0.342871	9.56154×10^{-11}	32.2359
Elliptic filter	0.047365	0.463987	1.49563×10^{-10}	30.293
Proposed Neural filter	0.043605	0.187784	1.26760×10^{-10}	31.0114
Multiplicative gaussian noise				
Median filter	0.0574893	0.543259	2.20335×10^{-10}	28.6104
3^{rd}-order Chebyshev type-1 filter	0.0278647	0.153496	5.17626×10^{-11}	34.901
3^{rd}-order Chebyshev type-2 filter	0.0365451	0.299232	8.90364×10^{-11}	32.5455
3^{rd}-order Butterworth filter	0.0306838	0.206129	6.27664×10^{-11}	34.0639
Elliptic filter	0.0338538	0.260648	7.64055×10^{-11}	33.21
Proposed neural filter	0.0453503	0.170839	1.37110×10^{-10}	30.6705
Impulsive Random Noise				
Median filter	0.0653834	0.142627	2.85000×10^{-10}	27.4928
3^{rd}-order Chebyshev type-1 filter	0.0364574	0.172277	8.86096×10^{-11}	32.5664
3^{rd}-order Chebyshev type-2 filter	0.0488858	0.20908	1.59321×10^{-10}	30.0185
3^{rd}-order Butterworth filter	0.0468088	0.160894	1.46071×10^{-10}	30.3956
Elliptic filter	0.0486396	0.226337	1.57720×10^{-10}	30.0623
Proposed Neural filter	0.0438345	0.173065	1.28098×10^{-10}	30.9658

*Highlighted values represent the best performance according to the used criterion. A lower value implies better performance (*Higher value is better). The results shown in this table are the average values obtained from 15 conducted simulations.*

information from signals corrupted deliberately with noise. Our proposal works as the analog mobile mean filter (refer to Mean Value Theorem for Integrals section) minimizing digital electronics, thus reducing the required number of transistors. Our frequency base decoding scheme has proven to have a good noise rejection, specially added white noise, but maintaining good performance with other types of noise, bringing artificial intelligence closer to circuit technology to deliver innovative solutions to filter white noise with the same frequency domain as the original signal, with minimal latency and low information loss. It is also a promising approach to, i.e., the conception of future innovative lab-on-chip implementations. Increasing the signal-to-noise ratio rejection ratio, cost efficiency, and sensitivity, is essential in these devices.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

LG-S proposed, developed, programmed the neural filter code, conducted the simulation runs, and wrote the first draft of the manuscript. VP-P and HS proposed modifications to the encoding and decoding strategy architectures. ER-E reviewed the test bench for the filter experiments. JM-N helped with the neural modeling in Python. All authors contributed to the conception and design of the study, manuscript revision, read, and approved the submitted version.

FUNDING

The authors would like to thank the economic support of the projects SIP 20210124, 20221780, 20211657, 20220268, 20212044, 20221089, 20210788, 20220226, and COFAA and CONACYT FORDECYT-PRONACES 6005.

ACKNOWLEDGMENTS

The authors would like to thank the support provided by Instituto Politécnico Nacional, Secretaría de Investigación y Posgrado, Comisión de Operación y Fomento de Actividades Académicas and CONACYT-México for the support to carry out this research.

REFERENCES

Abbott, L. (1999). Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain Res. Bull*. 50, 303–304. doi: 10.1016/S0361-9230(99)00161-6

Butterworth, S. (1930). On the theory of filter amplifier. *Exp. Wireless Wireless Eng*. 7, 536–541.

Dayan, P., and Abbot, L. K. (2001). *Theoretical Neuroscience, Computational and Mathematical Modeling of Neural Systems*. Cambridge, MA: MIT Press. p. 14–16.

Dupeyroux, J., Stroobants, S., and de Croon, G. (2021). A toolbox for neuromorphic sensing in robotics.

Elliasmith, C. (2013). *How to Build a Brain: A Neural Architecture for Cognition, 1st Edn*. Oxfort: Oxfort University Press.

Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. (2016). *Neuronal Dynamics*. Cambridge: Cambridge University Press.

Grinsted, A. (2022). fftnoise - generate noise with a specified power spectrum.

Guo, W., Fouda, M. E., Eltawil, A. M., and Salama, K. N. (2021). Neural coding in spiking neural networks: a comparative study for robust neuromorphic systems. *Front. Neurosci*. 15, 638474. doi: 10.3389/fnins.2021.638474

Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural Netw*. 10, 1659–1671. doi: 10.1016/S0893-6080(97)00011-7

Mead, C. (1990). Neuromorphic electronic systems. *Proc. IEEE* 78, 1629–1636. doi: 10.1109/5.58356

Orchard, G., Frady, E. P., Dayan Rubin, D. B., Sanborn, S., Shrestha, S. B., Sommer, F. T., et al. (2021). "Efficient neuromorphic signal processing with loihi 2," in *2021 IEEE Workshop on Signal Processing Systems (SiPS)* (Coimbra: IEEE).

Salvador, S., and Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intell. Data Anal*. 11, 561–580. doi: 10.3233/IDA-2007-11508

Schuman, C. D., Potok, T. E., Patton, R. M., Birdwell, J. D., Dean, M. E., Rose, G. S., et al. (2017). A survey of neuromorphic computing and neural networks in hardware. *CoRR, abs/1705.06963*. doi: 10.48550/arXiv.1705.06963

Sharifshazileh, M., Burelo, K., Sarnthein, J., and Indiveri, G. (2021). An electronic neuromorphic system for real-time detection of high frequency oscillations (hfo) in intracranial EEG. *Nat. Commun*. 12, 3095. doi: 10.1038/s41467-021-23342-2

Stewart, J. (2018). *Calculus Early Transcendentals, 8th Edn*. Belmont, CA: Cengage. p. 461–462.

Tukey, J. W. (1977). *Exploratory Data Analysis*. Princeton, NJ: Addison-Wesley Publishing Company.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in python. *Nat. Methods* 17, 261–272. doi: 10.1038/s41592-020-0772-5

Wijekoon, J. H., and Dudek, P. (2008). Compact silicon neuron circuit with spiking and bursting behaviour. *Neural Netw*. 21, 524–534. doi: 10.1016/j.neunet.2007.12.037

Wu, X., Saxena, V., Zhu, K., and Balagopal, S. (2015). A cmos spiking neuron for brain-inspired neural networks with resistive synapses and in-situ learning. *IEEE Trans. Circ. Syst. II*. 62, 1088–1092. doi: 10.1109/TCSII.2015.2456372

Zamarreño-Ramos, C., Camuñas-Mesa, L. A., Pérez-Carrasco, J. A., Masquelier, T., Serrano-Gotarredona, T., and Linares-Barranco, B. (2011). On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex. *Front. Neurosci*. 5, 26. doi: 10.3389/fnins.2011.00026

Zare, M., Zafarkhah, E., and Anzabi-Nezhad, N. S. (2021). An area and energy efficient lif neuron model with spike frequency adaptation mechanism. *Neurocomputing* 465, 350–358. doi: 10.1016/j.neucom.2021.09.004

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The reviewer AZ declared a shared affiliation with the authors to the handling editor at the time of review.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 27 June 2022
doi: 10.3389/fnbot.2022.819784

Artificial Intelligence Technology in Basketball Training Action Recognition

Yao Cheng [1], *Xiaojun Liang* [2,3]*, *Yi Xu* [4] *and Xin Kuang* [5]

[1] Shaoxing University Yuanpei College, Shaoxing, China, [2] College of Humanities, Zhaoqing Medical College, Zhaoqing, China, [3] Graduate School, University of Perpetual Help System DALTA, Manila, Philippines, [4] Ministry of Basic Education, Guangdong Eco-Engineering Polytechnic, Guangzhou, China, [5] School of Management, Guang Dong AIB Polytechnic, Guangzhou, China

The primary research purpose lies in studying the intelligent detection of movements in basketball training through artificial intelligence (AI) technology. Primarily, the theory of somatosensory gesture recognition is analyzed, which lays a theoretical foundation for research. Then, the collected signal is denoised and normalized to ensure that the obtained signal data will not be distorted. Finally, the four algorithms, decision tree (DT), naive Bayes (NB), support vector machine (SVM), and artificial neural network (ANN), are used to detect the data of athletes' different limb movements and recall. The accuracy of the data is compared and analyzed. Experiments show that the back propagation (BP) ANN algorithm has the best action recognition effect among the four algorithms. In basketball training athletes' upper limb movement detection, the average accuracy rate is close to 93.3%, and the average recall is also immediate to 93.3%. In basketball training athletes' lower limb movement detection, the average accuracy rate is close to 99.4%, and the average recall is immediate to 99.4%. In the detection of movements of upper and lower limbs: the recognition method can efficiently recognize the basketball actions of catching, passing, dribbling, and shooting, the recognition rate is over 95%, and the average accuracy of the four training actions of catching, passing, dribbling, and shooting is close to 98.95%. The intelligent basketball training system studied will help basketball coaches grasp the skilled movements of athletes better to make more efficient training programs and help athletes improve their skill level.

Keywords: somatosensory motion gesture recognition, recall, accuracy, artificial neural network, basketball training

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
Jia-Lang Xu,
National Chung Hsing University, Taiwan
Bunthit Watanapa,
King Mongkut's University of Technology Thonburi, Thailand

Correspondence:
Xiaojun Liang
liangxiaojun@zqmc.edu.cn

Received: *22 November 2021*
Accepted: *07 March 2022*
Published: *27 June 2022*

Citation:
Cheng Y, Liang X, Xu Y and Kuang X (2022) Artificial Intelligence Technology in Basketball Training Action Recognition.
Front. Neurorobot. 16:819784.
doi: 10.3389/fnbot.2022.819784

INTRODUCTION

The rapid development of artificial intelligence (AI) technology in computer vision, mobile internet, big data analysis and other fields, deep learning, cross-border integration, and other technologies along with new features have gradually become the new focus of international competition. Additionally, in computer vision, AI technology combines cameras and computers to replace human eyes to segment, classify, and identify targets. These functions are used in virtual reality and human–computer interaction, especially in sports video analysis, which has become a research hotspot in industries and academia. Basketball is a collective sport. Compared with other sports, basketball is very technical. In basketball, the basketball level of players has a significant impact on the team (Gonzalo-Skok et al., 2019). Currently, basketball is very popular on college campuses. The development of basketball teaching is popularized. Many schools have created the conditions for education in basketball lessons. College students are also very keen to participate in basketball, and playing basketball has become the first choice for most students to enrich their extracurricular activities. However, the public basketball teaching and training in colleges and universities emphasize more on offense technique while ignoring defensive practice. If students can have a tactical training robot, it will significantly improve their basketball ability and technical and tactical level (Hagiwara et al., 2020). The analysis of sports technology can compare and evaluate athletes' training and game videos. After the technique is analyzed, the athlete's movement standardization, physical fitness, and other aspects are targeted for training. Additionally, in the team competition, the movement and position of the athletes are detected, tracked, and analyzed, which can promote the improvement of the technical level (Garcia et al., 2020). Basic movements in basketball games include dribbling, shooting, and layup. Dribbling is the most basic action in basketball, and shooting is the key to scoring the whole game. The accuracy of the basic movements has a greater impact on the game's score. With the development of basketball competitions, the human pose estimation algorithm is integrated with the action recognition algorithm. These algorithms play a vital role in assisting to improve the scoring rate. Human pose estimation detects and estimates the position, orientation, and scale information of each target human body part from the image. This information needs to be converted into a digital form that the computer can interpret and output the current human pose. However, action recognition is to judge whether a person's actions are normative and how to improve the normativeness according to pose estimation as the input object (Sobko et al., 2021).

This study explores the intelligent detection of movements in basketball training through AI technology. Firstly, relevant literature is researched and learned. The relevant theory of somatosensory gesture recognition is mastered. Secondly, the collected motion data are denoised and normalized. Finally, the superiority of the four algorithms is analyzed in terms of the recall rate and accuracy rate. The most suitable motion recognition algorithm is selected for the basketball training intelligent system. The experimental results may be applied to the research of basketball assistant robots. By basketball teaching assistant robots, basketball coaches can better grasp the technical movements of athletes, formulate more efficient training programs, and help athletes improve their technical level. The innovation point is to achieve more efficient detection of basketball players' technical movements during training through the four action recognition algorithms. By analyzing the recall rate and accuracy rate of the four algorithms, the optimal algorithm is selected to detect technical movements. These several rounds can serve as a good reference for developing basketball assistant robots in the future.

LITERATURE REVIEW

Related Research Analysis

With the development of AI technology pair, the technology is gradually applied in various fields. Several research teams have begun to study how to improve the quality of basketball training through AI technology.

Xu and Tang (2021) applied machine learning algorithms to the path planning of intelligent robots. Firstly, the movements in basketball training are identified and analyzed in combination with the basketball movement trajectory model. Secondly, a mathematical model of the trajectory of the basketball shot is established. An improved machine-learning based Q-learning algorithm is proposed. The algorithm realizes the path planning of the mobile robot and effectively completes the behavior of avoiding obstacles. The results show that the fuzzy controller applied to the basketball robot can effectively avoid the obstacles during the robot's motion. Therefore, the proposed machine learning algorithm has a good obstacle avoidance effect when applied to path planning in basketball training. The application of this algorithm can effectively prevent sports injuries in basketball (Xu and Tang, 2021). Zhi and Jiang (2020) proposed a path planning algorithm based on behavioral module control, aiming at problems, such as the strong dependence of traditional algorithms on environmental information, the path planning of basketball robots in unknown environments, and the improvement of autonomous navigation safety. They applied the fuzzy control theory to behavior control structures and combined these two path planning algorithms to solve the path planning problem of basketball robots in unknown environments. The results show that the basketball robot can overcome the uncertainty in the environment and effectively achieve good path planning, which verifies the feasibility of the fuzzy control algorithm and the validity and correctness of the path planning strategy (Zhi and Jiang, 2020). Cox et al. (2021) built a generic controller for regulating the motion of an inertia-driven jumping robot. The robot can specify the desired speed and jump height. The controller can ensure that the desired value is achieved. The controller can achieve the maximum response of the basketball robot the maximum jump height of 0.25 m (Cox et al., 2021). Carnevale et al. (2021) used the emerging distributed constrained aggregation optimization framework for application to basketball robots.

They proposed a constant-step distributed algorithm for solving online optimization problems. In the static case (i.e., with constant costs and constraints), they show that the knowledge estimate converges to the optimal solution at a linear rate. Finally, numerical values show that the method is effective in robotic basketball games and robotic surveillance scenarios (Carnevale et al., 2021). Yang (2020) expounded the development trend of AI technology, analyzed the development status of AI, and expounded the status of AI in sports. Finally, he focused on the AI training strategy for basketball players. Yang analyzed and studied the application in other aspects, aiming to provide theoretical support and guidance for further development of modern basketball. The basketball player training system should be based on AI, fully understand and evaluate the physical condition and competitive ability of the players, and measure the sports skills of the basketball players in time. The training system should propose training strategies according to the sports conditions of basketball players and optimize the training plan of the players (Yang, 2020).

When the action recognition technology based on AI technology is applied to the action recognition of basketball players, the action point data of the hand posture during the shooting process and the lower limb state during the movement process are collected. Basketball movements are complex movements completed by the upper and lower limbs. The identification of basic basketball movements plays an important role in improving the skills of basketball players. Therefore, the basic movements of the upper and lower limbs in basketball have been studied, and the identification of the basic movements of basketball has been preliminarily realized.

Somatosensory Motion Gesture Recognition

As a branch of pattern recognition, human attitude recognition has been widely studied and developed in recent years. Human body recognition based on inertial sensors has gradually become a research hotspot. Based on pattern recognition, many scientists have applied imaging technology to the human body recognition of handheld devices (Zhang and Shi, 2021). **Figure 1** shows inertial sensors' human body recognition process, comprising specific steps of data acquisition, data processing, data segmentation, feature extraction, and classification training. Especially, in the data acquisition stage, it mainly includes physical or physiological signals, such as acceleration and angular velocity, heart rate, and body temperature. These are primarily covered by detection devices. The data processing can purify and standardize the data to meet the system's requirements. In data processing, data are extracted and analyzed in time and frequency range, separately. In the stage of function extraction, the analysis of element function is basically completed, and the extraction of associated attributes is taken as the sample data of calculation. Selected samples are formed according to the different classification principles, which will lead to the separation of unknown samples (Sannino et al., 2019; Xu and Yi, 2021).

The recognition technology based on image analysis also has many defects, such as the high precision requirement for machines, large size, and unsuitability for wearing. It is difficult to observe in some areas, and the monitoring coverage is limited. Although a large amount of collected data may lead to insufficient storage, real-time monitoring cannot be realized (Soferman, 2019). The development of science and technology based on inertial sensors has promoted the progress of sensor technology, such as smaller size, higher precision, better flexibility, high environmental protection requirements, high sensitivity, low energy consumption, and good real-time performance. It has become the best way to understand the human condition and has been widely used in sports and other fields. Many inertial sensor devices form a spatial network and are widely used (Liu, 2020).

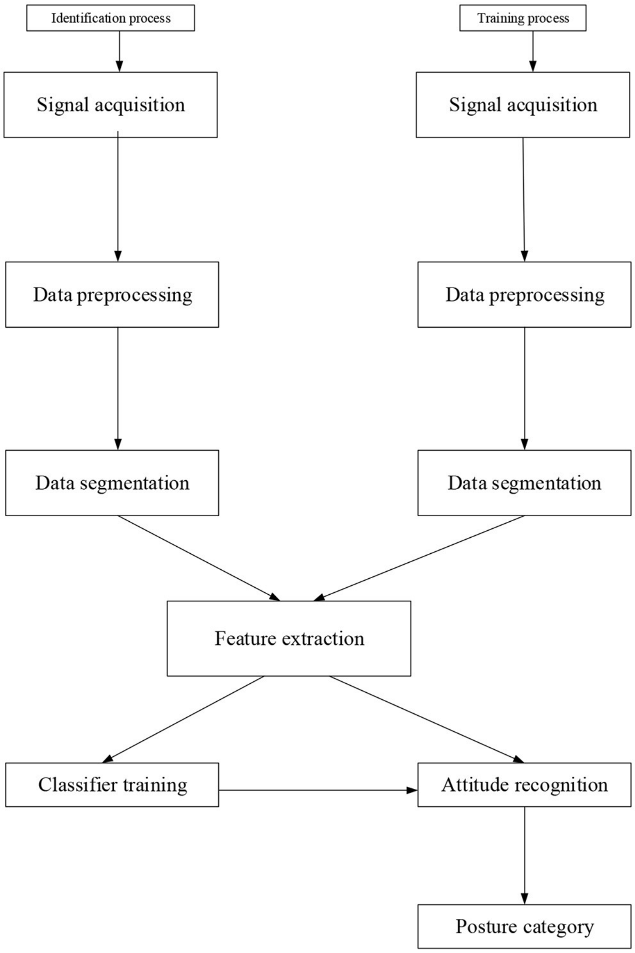


FIGURE 1 | Method flow of human posture recognition.

Data Preprocessing

In the data collection stage, the data signal collected by the sensor device is usually interfered with by the outside world or itself. Interferences include: (1) the jitter generated by the body or periodic signals generated by the surrounding environment during human movements. (2) There are measurement errors in the signal acquisition equipment itself. (3) During the movement, the signal caused by the position deviation of the node is inaccurate. The collected raw data cannot be directly used for analysis and calculation in practical applications. The signal needs to be preprocessed to obtain a more accurate signal after it is ordered. Standard preprocessing methods include denoising and normalization. The two processes of preprocessing will be introduced.

Denoising

The original data signals collected by the detection equipment often contain noise signals from an external environment, which is inevitably inaccurate. The signal denoising method in software design is generally called digital filtering, mainly composed of two types of filters: classical filter and current filter (Taborri et al., 2021). When the useful signal and the noise signal are in different frequency bands, adding another noise signal into the linear system can eliminate the noise signal. Commonly used high- and low-pass filters are established according to the principle of the distribution of signals in different frequency bands (Zhao et al., 2020). Still, the traditional filters have some defects and are no longer suitable for signal noise and frequency bands. Modern filters decompose proper signals and sound into random movements. The corresponding autocorrelation function is used to determine the beneficial signal or noise, such as the statistics of the autocorrelation spectrum. Commonly used filtering algorithms are Kalman and Wiener (Estevez et al., 2019).

Standardization

Standardization also belongs to normalization, which is the basic step of data mining and plays an essential role in simplifying the calculation. The normalization method is also adopted in human posture recognition. For data processing (Graham et al., 2019), due to the different positions of nodes in the network system of the human body domain, the nodes will shake with the movement of the human body. Data standardization can eliminate the dimensional influence between different data and solve the problem of data comparability. Specifically, their expressions are converted into different data for a comprehensive comparison and evaluation (Hussein et al., 2020). Two data standardization methods are described in detail as follows.

Linear Function Transformation

The linear function transformation maps the initial parameters into the interval [0,1] through a linear transformation. Its calculation method are presented as follows:

$$X_{norm} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{1}$$

In Equation 1, X_{norm} represents the result of linear function after conversion operation, X refers to the initial parameter, $X_{\max}$ stands for the largest data in the selected sample, and $X_{\min}$ means the smallest data in the selected sample.

Zero Mean Standardization

The zero mean standardization method processes the original data into a normal distribution set with a mean of 0 and a variance of 1. The specific calculation method is defined as:

$$y = \frac{x - \mu}{\delta}, \tag{2}$$

where x represents the data set, y refers to the result after calculation, μ stands for the initial parameter mean, and δ denotes the variance of the initial parameter.

Here, in calculating the node attitude, the three data of angular velocity, acceleration, and magnetic field strength are fused to achieve a more accurate attitude calculation and reduce the noise interference of the sensor. Quaternion means space attitude. Extended Kalman filtering is selected to implement data fusion. The accuracy of the posture solution result is improved. Quaternion and the principle of extended Kalman filtering are introduced in detail.

In 1843, William Rowan of Hamilton created the mathematical concept of Quaternion. It is a simple, super complex number to describe the rotation of a rigid body. Quaternion combines an actual number and three imaginary number units, as shown in Equation 3:

$$Q = w + xi + yj + zk, \tag{3}$$

where w, x, y, and z are real numbers, i, j, and k are the three imaginary units. A quaternion can also be represented by (w, x, y, z). Q satisfies Equation 4:

$$w^2 + x^2 + y^2 + z^2 = 1, \tag{4}$$

where Q is called the unit Quaternion. The unit Quaternion (1, 0, 0, 0) describes the attitude of the rigid body at rest. Only, the unitized Quaternion can describe the rotation of a rigid body, therefore the Quaternion is normalized. Its normalized form is shown in Equation 5:

$$\mathrm{Q}_{\mathrm{nwn}} = \frac{Q}{\sqrt{w^2 + x^2 + y^2 + z^2}}, \tag{5}$$

where Q_{norm} means Quaternion after being normalized.

The unit Quaternion can describe the rotation of a rigid body in a three-dimensional (3D) space, as shown in Equation 6. Equation 6 is also Quaternion's differential equation:

$$\dot{Q} = 0.5 \cdot Q \cdot p, \tag{6}$$

where p stands for Quaternion. It consists of the angular velocity detected by the gyroscope. The actual part is 0. $\dot{Q}$ is the derivative of Quaternion concerning time, as shown in Equation 7:

$$p = 0 + \omega_x i + \omega_y j + \omega_z k, \tag{7}$$

$$\dot{Q} = a + bi + cj + dk. \tag{8}$$

Applying Equations 3, 7 are brought into Equation 6 for calculation according to the complex number arithmetic, as shown in Equation 9:

$$\begin{aligned} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} &= 0.5 \begin{bmatrix} -x\omega_x - y\omega_y - z\omega_z \\ w\omega_x - z\omega_y + y\omega_z \\ z\omega_x + w\omega_y - x\omega_z \\ -y\omega_x + x\omega_y + w\omega_z \end{bmatrix} \\ &= 0.5 \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix} \begin{bmatrix} w \\ x \\ y \\ z \end{bmatrix} \end{aligned} \tag{9}$$

Equation 9 reflects the relationship between the angular velocity of the rigid body carrier and the derivative of Quaternion concerning time. According to the original state Quaternion, the Quaternion in the new state can be obtained. The normalization process can bring the unit Quaternion used to describe the transformation of a rigid body from one posture to the next. The updated equation is shown in Equation 10:

$$Q_{k+1} = Q_k + \Delta t \cdot \dot{Q}_k, \tag{10}$$

where k is a nonnegative integer, representing the time of the system, Q_k, and Q_{k+1} represent the unit Quaternion of the rigid body posture at the kth and $k + 1$th moments, respectively, represents the derivative of Quaternion for time at the kth moment, and Δt represents the time interval between the two samples. As the value of Δt is small, it is assumed that the rigid body rotates at a constant speed in it.

$\dot{Q}_k$ represents the derivative of Quaternion concerning time at the kth moment. According to Equation 6, Equation 11 can be obtained as

$$\dot{Q}_k = 0.5 \cdot Q_k \cdot p_k \tag{11}$$

Equation (11) is brought into Equation 10, and Equation 12 is obtained:

$$Q_{k+1} = Q_k + 0.5 \cdot Q_k \cdot p \cdot \Delta t \tag{12}$$

When the initial state of the system and the rotational angular velocity of the rigid body are known, the Quaternion of the system's state is obtained from Equation 12 to determine the current posture of the rigid body.

Expand Kalman filtering is converted from the nonlinearly changing scene into a linearly changing state for the solution. A non-linearly changing differential equation set is defined to apply the extended Kalman filtering technology to the actual location of nonlinear changes, as shown in Equation 13:

$$x = f(x) + w, \tag{13}$$

where x is the state vector of the system. The function $f(x)$ can calculate the predicted value of the system state at the next moment through the current system state. w represents the process noise vector expected to be 0.

The process noise matrix C is composed of the process noise vector w, as shown in Equation 14:

$$C = E\left(ww^T\right) \tag{14}$$

The measurement equation is a nonlinear state equation, as shown in Equation 15:

$$z = h(x) + v \tag{15}$$

The function $f(x)$ is the observation equation of the system. v is the measurement error vector expected to be 0, represented by the measurement noise matrix R, as shown in Equation 16:

$$R = E\left(vv^T\right) \tag{16}$$

As the measurement process is a nonlinear changing equation, k is used to represent the time. Equations 13, 15 are transformed into Equations 17, 18, respectively:

$$z_k = h(x_k) + V_k \tag{17}$$

$$x_{k+1} = f(x_k) + W_k \tag{18}$$

Assuming that the current optimal estimation state of the system is, the prior error e_k^- of the system can be obtained according to the expected state value of the system, as shown in Equation 19:

$$e_k^- = x_k - \hat{x}_k^- \tag{19}$$

The posterior error of the system is shown in Equation 20:

$$e_k = x_k - \hat{x}_k \tag{20}$$

Knowing that the prior error of the system is the expectation of the covariance of the system error, the covariance of the preceding error is shown in Equation 21:

$$P_k^- = E\left(e_k^- e_k^{-T}\right) \tag{21}$$

The covariance of the posterior error is shown in Equation 22:

$$P_k = E\left(e_k e_k^T\right) \tag{22}$$

According to the principle of the extended Kalman filtering algorithm, the expected value of the next moment of the system can be obtained by the optimal estimation of the current system. The desired correction of the system state is completed by mapping the system's measurement error to the system's state domain, thereby obtaining the optimal estimation of the system at the next moment. The state of the system is estimated to be continuously updated. The best estimate of each moment of the system is obtained. The Kalman filtering method of the nonlinear system is shown in Equations 23–27:

$$\hat{x}_{k+1}^- = f\left(\hat{x}_k, u_k\right), \tag{23}$$

$$P_{k+1}^- = F_k P_k^- F_k^T + C, \tag{24}$$

$$K_{k+1} = P_{k+1}^- H_{k+1}^T \left(H_{k+1} P_{k+1}^- H_{k+1}^T + R\right)^{-1}, \tag{25}$$

$$\hat{x}_{k+1} = \hat{x}_{k+1}^- + K_{k+1}\left(z_{k+1} - h\left(\hat{x}_{k+1}^-\right)\right), \tag{26}$$

$$P_{k+1} = P_{k+1}^- - K_{k+1} H_{k+1} P_{k+1}^- \tag{27}$$

Equation 23 represents the desired state at the next moment. It can be obtained from the well-estimated state at the current moment. In Equation 24, F is the Jacobi matrix of the function $f(\cdot)$. This equation expresses that the current error covariance estimate is obtained through the posterior error covariance matrix now. Equation 25 gives the method of solving Kalman gain K_k. H_k is the Jacobi matrix of the function. Equation 26 combines the expected value of the state at the next moment and the observed value. It obtains the Kalman gain, thereby obtaining the optimal estimation value of the form at the current moment. Kalman filtering is an iterative cycle process. Therefore, the updated test error covariance is used for future iterations. Equation 27 uses the prior error covariance at the next moment to update. It is brought into Equation 24 for a new round of iterations.

The expansion of Kalman filtering mainly includes two processes: time iteration and measurement iteration. Time iteration includes Equations 23, 24, which completes the prediction of system state and error covariance primarily. Measurement iteration uses observations to correct expectations, to obtain more accurate estimates, and to obtain optimal forecast, as shown in Equations 25–27. In Equation 26, Kalman gain is mainly used to weigh the state expectation and the proportion of the system measurement value in calculating the optimal estimation. The greater the Kalman gain, the closer the optimal estimate is to the observed value.

In actual situations, the process noise w_k and the measurement noise v_k conform to the white noise of the normal distribution. The two types of noise are not correlated with each other. Therefore, the covariance matrix C of the process noise and the covariance matrix R of the measurement noise are constants. At the initial moment of the system, the posterior error covariance matrix P_0 of the system can be set to any diagonal matrix as the initial value. The initial values of other states of the system can be set arbitrarily. In the iterative process of the system, these values will automatically converge and get closer to the actual situation.

Quaternions Method

In 1843, the British mathematician W.R. Hamilton introduced Quaternions in mathematics. However, until the late 1960's, this method had not been put into practical use. With the development of space technology and strapdown inertial navigation system (SINS) technology, Quaternions have attracted people's attention. Solving the Quaternions differential equations involves solving four differential equations. Although this is one more equation than solving Euler's differential equations, it has the advantages of less computation, high precision, and the avoidance of singularities. This method is one of the focuses of current research. Due to the direction cosine method, skew, scale, and drift errors will be generated when solving the attitude dynamics of the carrier. However, it is essential in SINS to estimate these errors when doing attitude solutions. The advantage of the Quaternions method compared to the directional cosine method is that the skew error is equal to 0. The derivation of the scale error leads to an analytical expression that facilitates further analysis. The directional cosine method can only analyze and detect scale errors in exceptional cases without drawing general conclusions. A comparison of the Euler angle

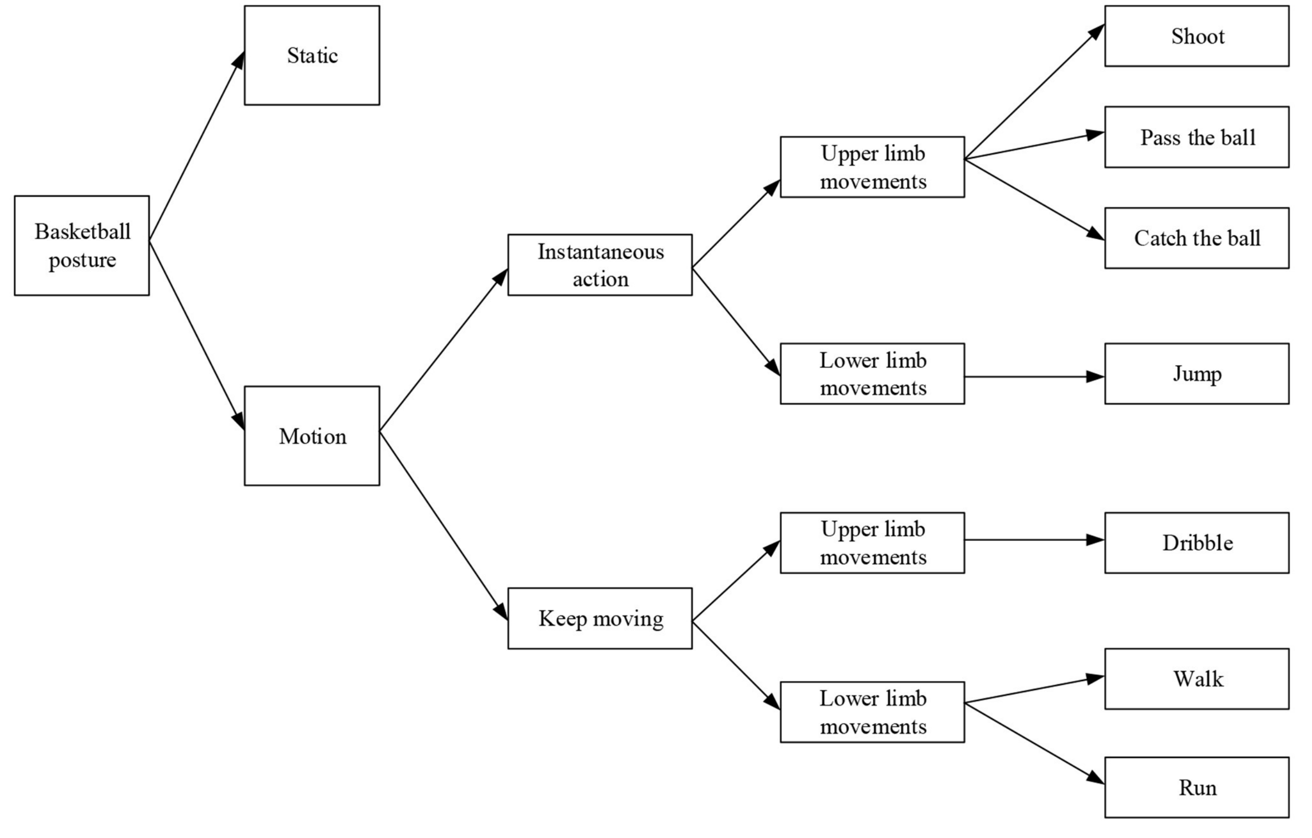


FIGURE 2 | Composition of basketball training action.

method, directional cosine method, and Quaternions method from different angles shows that Quaternions method performs the best.

Kalman Filtering

Kalman filtering does not require the assumption that both signal and noise are stationary processes. For the system disturbance and observation error (i.e., noise) at each moment, if some appropriate deductions are made about their statistical properties, and by processing the observation signal containing noise, the minimum error can be obtained in the average sense. Therefore, the Kalman filtering theory is applied in communication systems, power systems, aerospace, environmental pollution control, industrial control, radar signal processing, and other industries and has achieved many successful application results. For example, in image processing, Kalman filtering restores the images blurred by some noise. After making some statistical assumptions about the noise, the Kalman filtering algorithm can be used to obtain the actual image with the minor mean square error from the blurred image in a recursive way, so that the blurred vision can be restored.

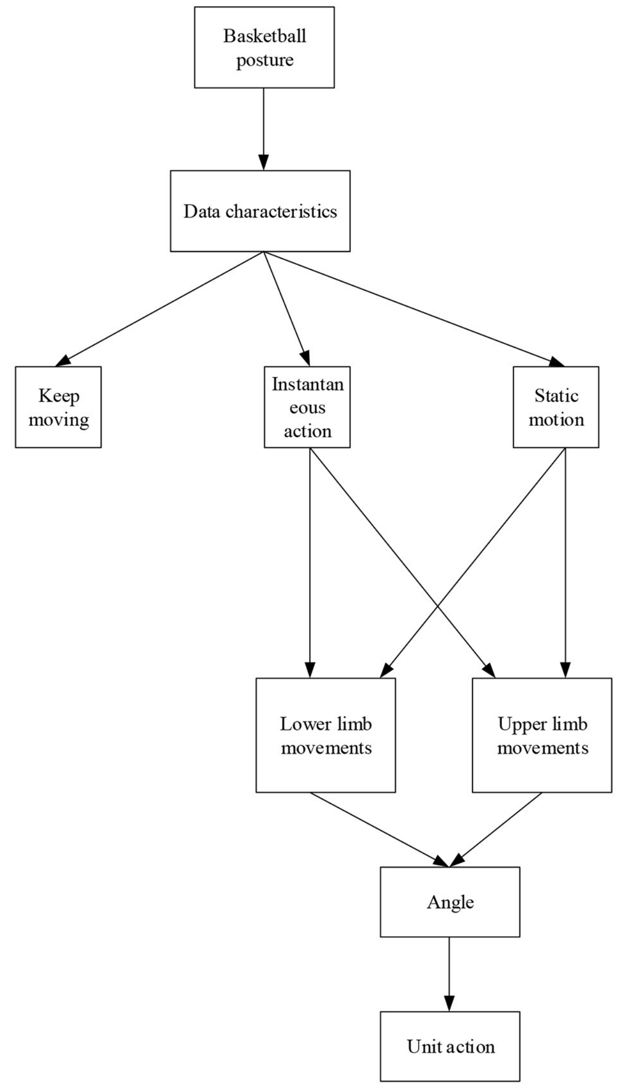


FIGURE 3 | Data division method of basketball posture.

Classification Algorithm

Decision Tree

Decision tree (DT) is a monitoring learning method, which is often used for data classification and regression. It is the essence of DT to simplify complex problems into a hierarchical structure to solve these problems, therefore it is also a multilevel decision model. DT can be regarded as a tree structure composed of nodes and sharp edges, a theoretical statistical model (Rampersad, 2020). The node types include leaf and internal nodes. The internal nodes represent the recognition of specific attributes of multiple samples, the extended branches stand for the recognition results, and the leaf nodes denote the specific classification results. The construction of DT is complex. Feature selection and partition is the primary step of constructing DT, in which feature selection contains detailed information, which is mainly based on some indicators related to characteristics. The most common hands are gain rate and information gain and inversion (Senhaji et al., 2021). Feature segmentation can also be regarded as a re-segmentation method. Below them, different categories can be filtered so that every data pointing to an edge has the same type as possible. Standard DT construction algorithms are C4.5, Iterative Dichotomiser 3 (ID3), Classification and Regression Trees (CART), etc. The principle of DT is relatively simple, the construction process is not complicated, and the construction time is relatively short (Kotter and Ranschaert, 2021). However, this method is not suitable for the problem of missing data, and it is prone to overfitting.

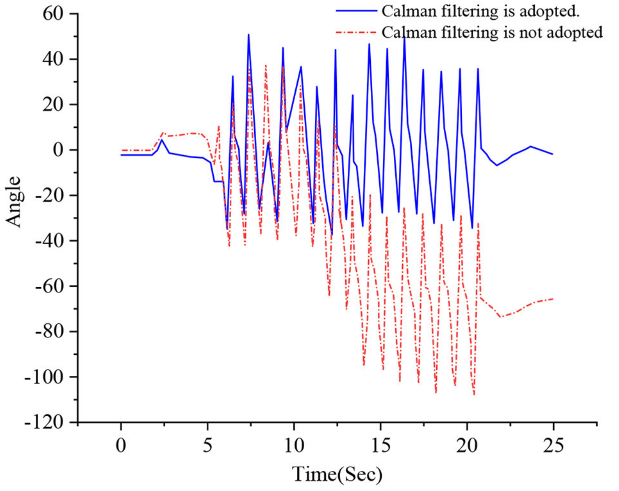


FIGURE 4 | A comparison of Kalman filtering effects of changing the walking angle.

Naive Bayes

Naive Bayes (NB) method is a simple classification method. Based on the Bayesian theory, the classification probability of each type of sample is calculated, and the most probable category is selected as the classification result. Simple Bayesian algorithm has a strong theoretical foundation and is a relatively stable classification method based on classical mathematical theory. Because this method is simple and insensitive to lost data, there are some limitations in practical applications, which need to be improved to make the models independent of each other. These problems are usually encountered (Радутний, 2019). In the application of human identification markers, the occurrence of events is uncertain, therefore it is difficult to obtain the prior probabilities of different classification behaviors, hence the Bayesian algorithm is not applicable.

Support Vector Machine

Support vector machine (SVM) refers to a supervised classification method, which is widely used in machine learning at present. It was first used in 1960 and applied to the optimal classification hyperplane. The basic principle of SVM is its minimal structure, therefore it has strong generalization ability. Initially, this method was mainly used to deal with the classification of two simple categories. It can automatically find out the vector machine to achieve the optimal classification, which is linear. SVM uses the kernel technique to make it applicable to nonlinear classification. Multiple SVMs are combined to meet the requirements (Ha et al., 2019). SVMs are constantly being optimized and improved. The SVM algorithm is widely used in text classification, image classification, handwriting font recognition, and other fields.

Artificial Neural Network

It is a machine learning algorithm that imitates the biological neural network model, which has a complex network structure. Each neuron has a simple structure and unique function. This is an adaptive nonlinear information processing system. Artificial neural network (ANN) can be summarized as a three-layer mathematical model, including an input layer, an output layer, and a hidden layer (Luo et al., 2019). The input layer consists of many neurons for receiving information. The hidden layer is also called as concealed layer. The neural network can be divided into input and output layers. It is widely used in image processing and language recognition. A neural network has strong self-learning and nonlinear adaptive ability and has relatively little interference to noise data.

RESEARCH MODEL AND THE METHODOLOGY

Recognition of Basketball Players' Actions During Training

The Definition of Basketball Posture

When basketball players train, they usually perform complex skill actions. **Figure 2** presents an analysis of the composition of skill actions in the training process of basketball players. According to the different states of different limbs at a certain moment, the athletes' action states in basketball training can be simply divided into two types: motion and static. Under the static state, the movement of limbs of athletes belongs to a stationary state, and the relative movement state refers to the state when the limbs perform related exercises. For example, when the

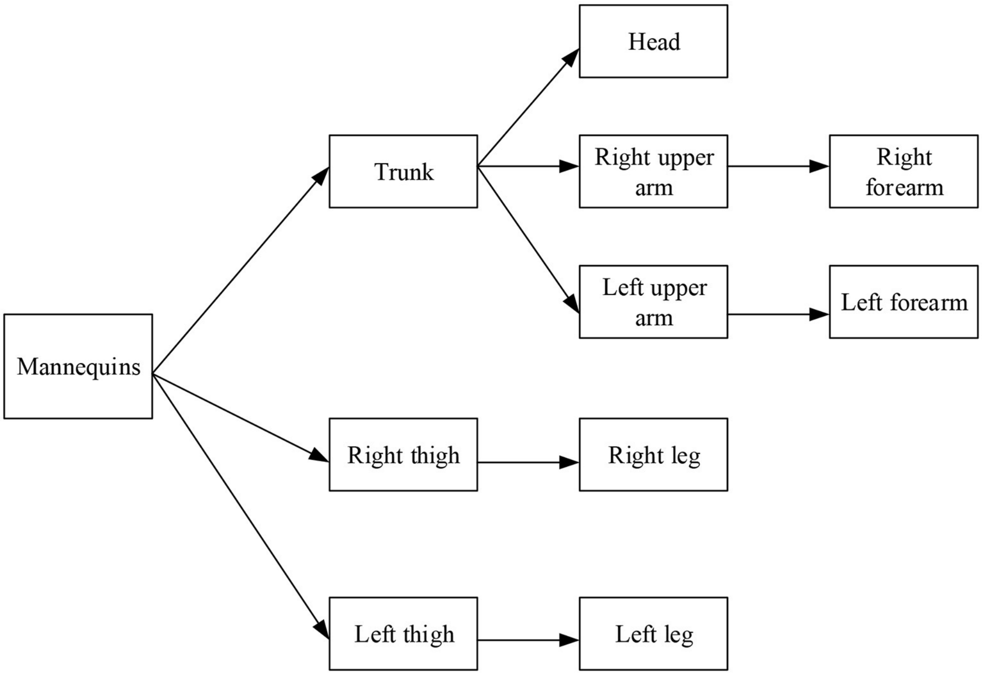


FIGURE 5 | The model of the human body.

athletes perform the ball-catching action, the legs of professional athletes will not change, therefore the legs at this time belong to a static state, while the upper limbs participate in the ball-catching action. Therefore, the arm is in motion (He, 2021). The shooting, catching, passing, and dribbling of upper limbs and jumping, walking, and running of lower limbs are defined as unit movements. In the motion state, the exercises can be divided into rapid movements and continuous movements according to whether they are periodic or not. Generally, instantaneous action does not have periodicity. It only contains one-unit action, such as catching, dribbling, and shooting. In this way, the continuous basketball training action is periodic. There will be many small unit actions when completing a set of training actions, such as constant walking, dribbling, and running dribbling. Therefore, in recognizing athletes' movements in basketball training, it is imperative to distinguish the activities of upper and lower limbs. Therefore, a division method based on unit action extraction is adopted here.

Data Segmentation

Data segmentation is generally divided into two stages, and **Figure 3** shows the specific data partitioning process. In the data division of the first stage, according to the discreteness characteristics of action data in basketball training of athletes in two different states, what is extracted are the athletes' action data and the sustained and instantaneous actions of athletes in a series of activities. Because there are many athletes' persistent movements and much need of unit movements, in the second stage of data division, according to the changes in the limbs angles of basketball players during the training process, it is finally realized of the data extraction of athletes' continuous movements (Dudnik et al., 2021).

Division of Exercise Status

The degree of dispersion indicates the degree of difference between the values of the observed variable. The difference between the sensor signal sample values is taken as the dispersion.

FIGURE 6 | The model of 3dMax basketball player.

Taking angular velocity as an example, ω_n^x represents the x-axis angular velocity data at the nth time. ω_{n-1}^x represents the x-axis angular velocity data at the n − 1th time. d_n^x represents the difference between the angular velocity of the x-axis of the sensor at the nth moment and the angular velocity at the last moment. The dispersion d_n^x can be obtained by Equation 28:

$$d_n^x = \left|\omega_n^x - \omega_{n-1}^x\right| \tag{28}$$

The motion data includes angular velocity data and acceleration data. The data characteristics of each sensor need to be considered comprehensively to achieve the accurate division of actions. D_n^x represents the dispersion of the acceleration sensor data at the nth time. d_n^g represents the dispersion of the angular velocity sensor data at the nth time. represent the dispersion of the acceleration and angular velocity of each axis, and the expressions of D_n^a and D_n^g are shown in Equations 29, 30:

$$D_n^a = d_n^{a_s} + d_n^{a_y} + d_n^{a_z}, \tag{29}$$

$$D_n^g = d_n^{g_s} + d_n^{g_y} + d_n^{g_z} \tag{30}$$

In the stationary state, the acceleration and angular velocity dispersion are kept below the threshold values λ_a and λ_g, respectively. In the state of exercise, the sensor data will change rapidly with the athlete's movements. Dispersion can reflect the degree of difference in the sensor data. Therefore, the dispersion feature can realize the athlete's limb state division. γ_n represents the state of the athlete's limb at the nth moment. When γ_n is 0, it indicates a static state. When γ_n is 1, it represents the state of motion, as shown in Equation 31:

$$\gamma_n = \begin{cases} 0, \mathrm{D}_n^a < \lambda_a \text{ and } \mathrm{D}_n^g < \lambda_g \\ 1, \mathrm{D}_n^a \geq \lambda_a \text{ or } \mathrm{D}_n^g \geq \lambda_g \end{cases} \tag{31}$$

The data dispersion degree of each sensor is calculated, and the threshold value can identify each movement state.

Division of Unit Actions

By dividing the action state, instantaneous action and continuous action are obtained. The division of unit actions is the further processing of constant measures. In constant movement, the movements of the legs and arms are in constant periodic changes, and the periodicity of the continuous changes is more pronounced. Therefore, it is feasible to realize the division of unit actions based on the movement data of the arms and legs. Through data comparison, the angular velocity data can intuitively describe the angular change during the movement of the rigid body. Angular velocity is used as a reference, and the data are divided as a reference.

The degree of dispersion represents the degree of difference between the values of the observed variable. The difference between the sample values of the sensor signal is used as the dispersion. Taking the angular velocity as an example, ω_n^x represents the x-axis angular velocity data at the nth time, ω_{n-1}^x represents the x-axis angular velocity data at the n − 1th time, and d_n^x represents the difference between the angular velocity of the sensor x-axis at the nth moment and the angular velocity of the last moment. The degree of dispersion d_n^x is shown in Equation 32:

$$d_n^x = \left|\omega_x^n - \omega_{n-1}^x\right| \tag{32}$$

The motion data includes angular velocity data and acceleration data. Each sensor data feature is comprehensively considered to achieve an accurate division of actions. d_n^a represents the dispersion of the acceleration sensor data at the nth time, and d_n^g represents the dispersion of the angular velocity sensor data at the nth time. D_n^a and D_n^g are shown in Equations 33, 34:

$$D_n^a = d_n^{a_x} + d_n^{a_y} + d_n^{a_z} \tag{33}$$

$$D_n^g = d_n^{g_x} + d_n^{g_y} + d_n^{g_z} \tag{34}$$

At rest, the acceleration and angular velocity dispersions remain below the thresholds λ_a and λ_g, respectively. In the state of motion, the sensor's data will change rapidly with the athlete's movement, and the degree of dispersion can reflect the degree of difference of the sensor data. Therefore, the discrete feature can realize the athlete's limb state division. γ_n represents the state of the athlete's limb at the nth moment, as shown in Equation 35:

$$\gamma_n = \begin{cases} 0, D_n^a < \lambda_a \ and \ D_n^g < \lambda_g \\ 1, D_n^a \geq \lambda_a \ and \ D_n^g \geq \lambda_g \end{cases} \tag{35}$$

In basketball, the sensor signal is easily affected by the human body and the external environment. When calculating the angle during limb movement, the Kalman filtering algorithm is used to fuse acceleration, magnetic field strength, and angular velocity data to reduce the influence of external noise. **Figure 4** shows the change curve of the calf angle during walking. The abscissa represents time, and the ordinate represents the calf angle. The dotted line is the angle curve obtained without the Kalman filtering algorithm, periodically changing. After a period, the angle value shifted significantly. The solid line is the angle curve obtained by the Kalman filtering algorithm, which fluctuates at the same amplitude on both sides of 0°. Kalman filtering algorithm processing data can reduce the interference of noise signals.

Figure 4 uses the Kalman filtering algorithm to process data whose value remains at [−40, 60]. The data values that are not processed by the Kalman filtering algorithm remain at [−110, 40], which is less stable than the former data because the Kalman filtering algorithm can effectively remove the interference data and ensure the accuracy of the obtained data.

Posture Feature Extraction of Basketball Players

Data division obtains the unit action data composed of acceleration and angular velocity. a_n^x, a_n^y, and a_n^z represent the accelerations of the three axes of the nth sampling point, and, g_n^y, and indicates the angular velocity of three axes at the nth sampling point. Parameters a_n and g_n express the acceleration vector sum and angular velocity vector sum, respectively, as shown in Equations 36, 37:

$$a_n = \sqrt{(a_n^x)^2 + (a_n^y)^2 + (a_n^z)^2}, \tag{36}$$

$$g_n = \sqrt{(g_n^x)^2 + (g_n^y)^2 + (g_n^z)^2} \tag{37}$$

The eight-dimensional vector is mainly composed of combined acceleration, triaxial acceleration, combined angular velocity, and triaxial angular velocity. And, n is the number of points that unit basketball training movements can collect, each dimension in the vector contains the sampling data of N basketball training actions. And, each basketball training action is taken as a sample. Every sample will become an N*8-dimensional matrix. Simultaneously, every basketball training action should be calculated in all dimensions, therefore it is necessary to calculate the signal, which mainly has two signal features: frequency and time. There are two main features in the time domain: variance and mean. Parameters μ_a and δ^2 represent the mean and variance of acceleration components in basketball training actions, which are obtained by Equations 38, 39 (Shivale et al., 2020):

$$\mu_a = E(a) = \frac{1}{N}\sum\nolimits_{i=1}^{N} a_i \tag{38}$$

$$\delta^2 = \frac{1}{N}\sum\nolimits_{i=1}^{N} (a_i - \mu_a)^2 \tag{39}$$

The frequency-domain features mainly include two types: frequency and signal peak value after the discrete Fourier transform. According to the discrete Fourier transform, the obtained action signal is transformed from the time domain to the frequency domain. $S_{\mathrm{DFT}}(n)$ represents the Fourier transform result of the nth sampling point, and j represents the imaginary unit, and its calculation process is as shown in Equation 40:

$$S_{DFT}(n) = \sum\nolimits_{i=0}^{N-1} a_i e^{-j\frac{2\pi}{N}in} \tag{40}$$

The peak value $S_{\mathrm{DFT}}(k)$ of the signal is obtained by Fourier transform, where K represents the sampling point corresponding to each peak during Fourier transform. The frequency F in the Fourier transform is calculated by Equation 41, and the sampling frequency is represented by f_x:

$$f = K \times \frac{f_x}{N} \tag{41}$$

Through feature calculation, it can figure out the features of collected action signals in the frequency domain and time domain.

Basketball Posture Recognition

The essence of basketball gesture recognition is constructing a classification model that satisfies the classification of basketball action data. For each specific basketball action, after data collection, data preprocessing, data division, and feature extraction, the attribute set of the specified basketball action can be obtained, that is, the feature vector set. These feature vector sets are abstract data sets of basketball actions. The corresponding classification can be obtained in the calculation of the classifier model. The attributes contained in the feature vector are complex. The feature vector feature is selected to eliminate irrelevant and redundant attribute values in the feature vector. The priority search algorithm and principal component analysis method are adopted in attribute screening. Feature selection realizes the dimensionality reduction of feature vectors. This reduces the complexity of the classification calculation process and improves the working efficiency of the system. In the experiment, the sensor nodes are fixed on the calf and forearm of the subject to detect the action and behavior information of different limbs. According to the different placement positions of the nodes, the data set of each kind of movement is divided into the upper and the lower limb movement data set. Different sample sets are separately constructed classifiers to realize the specific division of upper and lower limb unit actions. The upper and lower limb movements are combined, and the basketball movement posture performed by the current examinee is obtained. In constructing the classification model, four commonly used classification algorithms are used: C4.5 DT, SVM, Bayesian network, and backpropagation ANN. The output results of these four classification methods are compared and analyzed. The best classification method following the experimental environment is obtained.

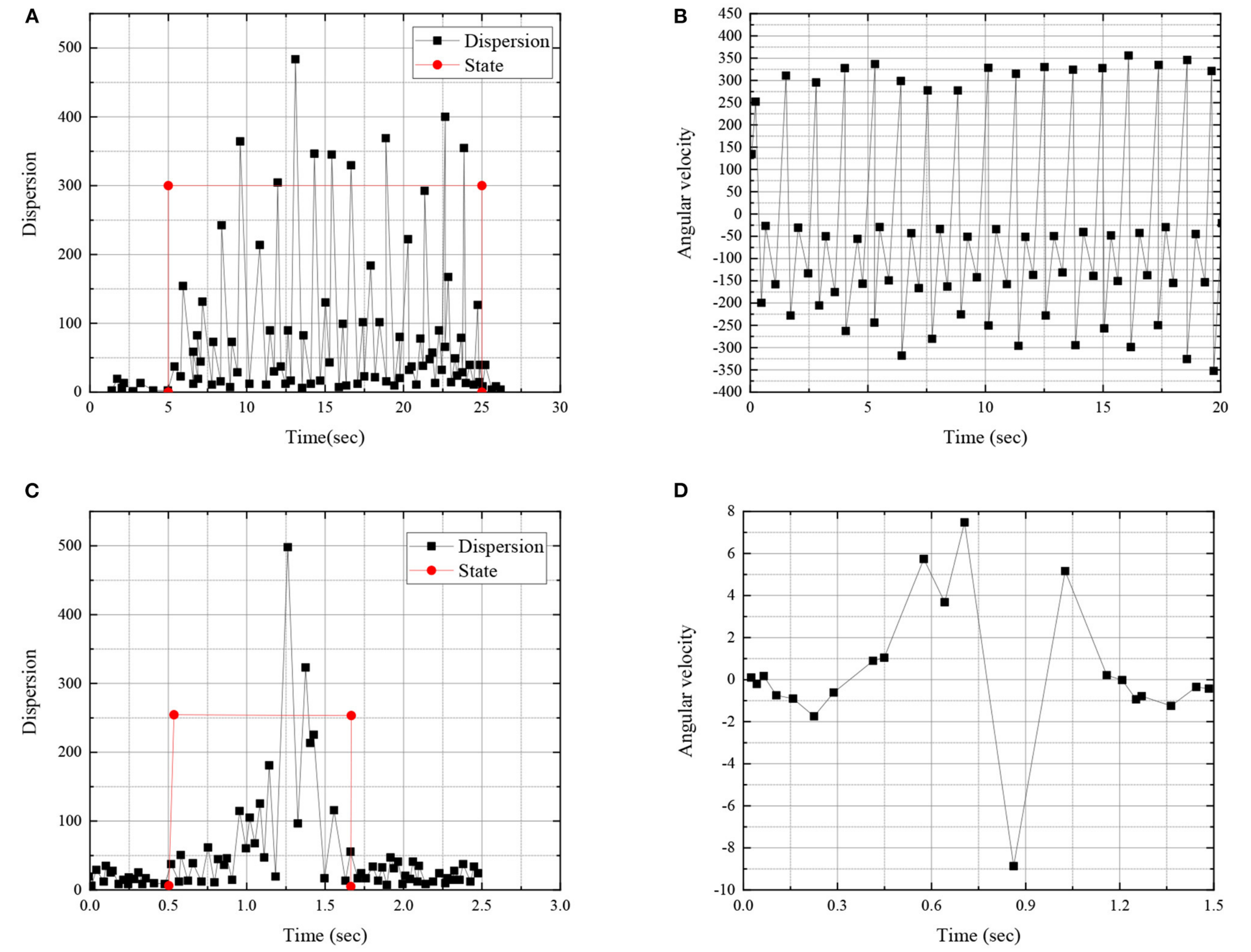


FIGURE 7 | Principle of the data partition method based on discreteness **(A)** angular velocity discreteness and motion state during walking; **(B)** angular velocity of legs during walking; **(C)** angular velocity dispersion and motion state during passing; and **(D)** arm angular velocity during passing.

Intelligent System Design

Target Machine Software

It realizes signal transmission by the position calculation module and runs on the nodes of the sensor. The function of the base station is to receive the nodes and collect the parameters of the obtained nodes. Therefore, a serial port is used to transmit the data to the upper computer, and the upper computer uses the serial port to collect the action data of basketball players and perform classified calculations and analysis of actions. The data are passed to the upper computer to shape the athlete's body posture (Echevarría et al., 2019). Data initialization mainly completes Quaternion initialization, acceleration initialization, and magnetic field intensity initialization. The received data mainly includes the position information of each sensor, including acceleration, angular velocity, and magnetic field strength. According to the height calculation method, one-quarter of the data is recursive. In the data fusion process of Kalman filter design, the acceleration, angular velocity, and magnetic field strength determine whether the acceleration data can be used for position calculation when the object position changes (How and Wei, 2019). If the position of the object changes and the acceleration data cannot be used for filtering calculation, the position data are directly sent to a computer through a serial port and converted into a cosine matrix. And then, it is converted into unified coordinates for the upper computer to display.

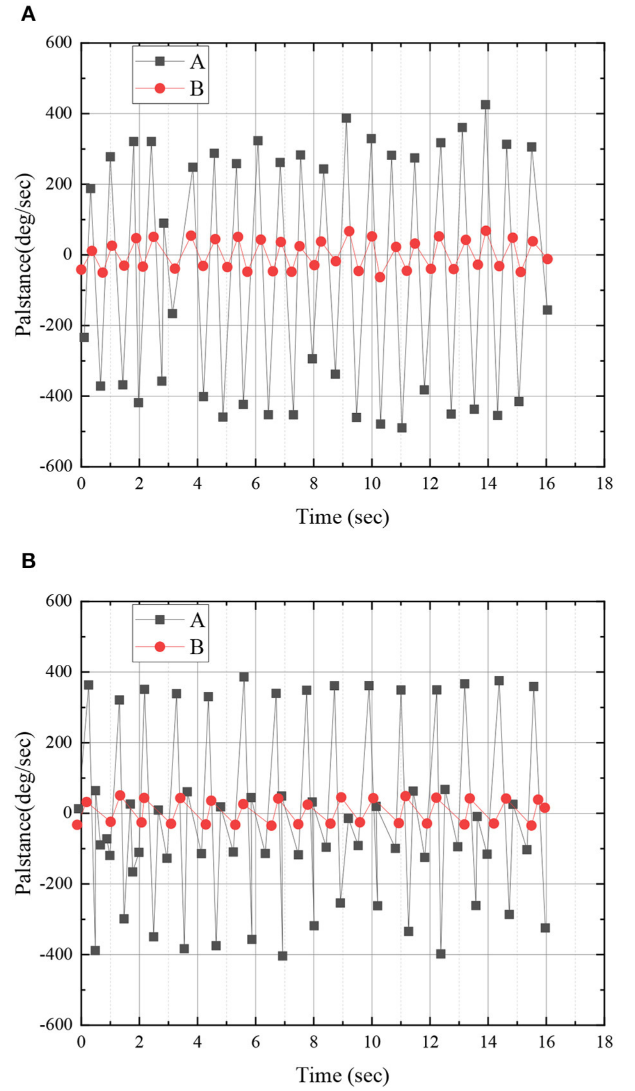


FIGURE 8 | Comparison of the angles and angular velocities of basketball players forearm dribbling and calf walking: **(A)** forearm and **(B)** calf. (A: forearm/calf angular momentum while dribbling and B: forearm/calf angle while dribbling).

Upper Computer Software

The function of the upper computer is to realize the consignment of athletes' skill movement data and the display of athletes' postures. The upper computer reads the relevant attitude information data sent from the attitude node to the base station through a serial port. And, the upper computer part constructs a 3D model of the human body through visualization toolkit (VTK) development tools, and through processing and calculating the collected posture information, thereby intuitively displaying the posture of the human body.

The upper computer module only receives data of the quaternion data type. According to the OpenGL theory, OpenGL cannot directly utilize and draw quaternions. Therefore, before applying the OpenGL theory, it is necessary to process the obtained data. Therefore, a data processing module is added between the received data and the human body display module (Joshua and Niloufer, 2020). By changing Quaternion into directional cosine and then converting directional cosine into four-dimensional homogeneous coordinates, the combination with OpenGL can be realized after such a transformation.

The 3D model of basketball elements can be divided into 10 structural parts. In these 10 parts, the bones will produce associated animation effects after rotation and translation due to the interaction of various structures. **Figure 5** presents the specific structural relationship. Each node in **Figure 5** can be regarded as a rigid body. In the spatial structure, rigid bodies are usually represented by the two points. The rotation centers of these two nodes are the origin, and the other node is called the endpoint. The relationship between two nodes is the parent–child relationship by default. When the parent node rotates, it will drive the child node to translate.

In **Figure 5**, the method of skeletal animation is referred to make that the established basketball player model have animation effects. Geometric vertices are bound to bones, and spirits are generated by controlling the translation and rotation of these bones. The specific performance is that each bone has a weight factor for all vertices, and the size is 0 to 1, where 0 means that the bone is not related to the vertex. Multiple bones may drive a vertex. However, the weights of the bones connecting the vertex should add up to 1. The obj file does not contain bone information, and it is necessary to use 3dMax to export the set weight data in text form. Afterward, 3dMax tools are used to draw the concrete human model and bind the bones.

The experiment rotates continuously from 0° to 1,080°, with data sampling every 90°. The investigation is divided into two groups. To compare the compensation effect of Kalman filtering on the angle calculation, the first group of the investigation does not use any compensation method. It directly obtains the rotation angle of the sensor through the angular velocity integration. The second group of experiments uses the extended Kalman filtering method to compensate for the angular output of the sensor nodes. **Figure 6** shows the model constructed in 3dMax.

The related methods of skeletal animation to connect the geometric vertices of the model characters are combined with the bones' positions to generate animation by using the rotation and translation of the bones of the characters in the model. Specifically, each bone has a weight for all vertices, ranging from 0 to 1, where 0 indicates that the bone is irrelevant to the vertex. Multiple bones may drive a vertex, but the weights of the bones connecting the vertex should add up to 1.

Design of Basketball Assistant Robot

The overall shape of the robot is designed as a humanoid omnidirectional wheeled mobile robot with a relatively simple structure. According to the characteristics and functional requirements of the training land, the overall design plan is: the mechanical system includes three parts: the swinging robotic arm, the robot torso, and the mobile base. A motor drives the mechanical arm to swing up and down at a certain angle, which can expand the defensive area and simply block the pass and pitch. The robot's torso is generally a "bell-shaped" cavity structure with a narrow upper and a lower width to facilitate the installation of the middle and outer protective layers. A gyroscope is installed in the middle and lower part of the torso to keep the robot balanced when it is bumped. The mobile base consists of a driving gear train, a base frame, and base armor. The driving wheel train adopts the vertical layout of four omnidirectional wheels. Four servo motors independently drive the four omnidirectional wheels to meet the requirements of all-round steering of the robot. As the robot needs to sense and defend the position of the ball and players, vision sensors are required. A gyroscope device is installed in the robot body to prevent the robot from being knocked down during training.

TABLE 1 | The number of samples collected by each tester's thinking data.

Behavior	Number of actions
Walk	50
Running	50
Jump	50
Stationary dribble	50
Walking dribble	50
Running dribble	50
Shot	50
Pass	50
Catch the ball	50

TABLE 2 | The number of different movements of the upper and lower limbs.

Upper and lower limbs	Behavior	Number of actions
Arm movements	Stationary dribble	400
	Walking dribble	400
	Running dribble	400
	Shot	400
	Pass	400
	Catch the ball	400
Leg movements	Walk	400
	Running	400
	Jump	400
	Shot	400
	Running dribble	550
	Walk dribble	450

During the data collection process of basketball training movements, the basketball teaching assistant robot tested the walking, jumping, running, and dribbling in place without the ball, running dribbling, walking dribbling, passing with the ball, and eight male testers, respectively. Nine kinds of basketball training actions, shooting, and catching are collected for related data. A total of 5,000 samples are collected this time. When dribbling the ball, the upper body movements of athletes mainly include dribbling in place, running dribbling, walking dribbling, passing, shooting, and catching, totaling 2,400. There are 2,600 data records of lower body movements during athletes' training, including walking, jumping, running, dribbling in place, running dribbling, and shooting while dribbling the ball. During the data collection process, each athlete participating in the test must complete all technical movements, and the basketball assistant robot will record the number of basketball technical movements.

EXPERIMENTAL DESIGN AND PERFORMANCE EVALUATION

Evaluation of the Athlete's Attitude Estimation of the Intelligent System

The evaluation methods of joint human points include percentage of correct keypoints (PCK) and object keypoint similarity (OKS). Here, Euclidean distance is used to compute the similarity between the two poses and the score between 0 and 1 (Wetzel et al., 2020). Euclidean distance is one of the most common distance measures. It measures the absolute distance between all points in a multidimensional space. The Euclidean distance of two n-dimensional vectors $X = (x_1, x_2, x_3, ..., x_n)$ and $Y = (y_1, y_2, y_3, ..., y_n)$ is shown in Equation 42:

$$D(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{42}$$

When the OKS value is more significant than 0.5, the athlete's posture can be correctly detected. Athlete detection and pose estimation are both detection tasks. The evaluation method of the inspection task is selected as the commonly used evaluation index, namely, the maximum recall rate. For experimental results, there are four possible situations between the predicted and the actual values.

TP, FP, FN, TN, and TP indicate that positive samples are predicted to be true. FP means that negative samples are expected to be true. FN means that positive samples are predicted to be false. TN indicates that the pessimistic sample prediction is incorrect. *P* stands for precision, *R* stands for recall, and *S* represents specificity, as shown in Equations 43–45:

$$P = \frac{TP}{TP + FP}, \tag{43}$$

$$R = \frac{TP}{TP + FN}, \tag{44}$$

$$S = \frac{TN}{FP + TN} \tag{45}$$

Different detection confidence thresholds are set. Different test results will be produced. The detection result of each threshold is calculated as *P* and *R*. The recall obtained is the maximum recall when the confidence threshold is set to the minimum value.

Data Partition Principle Based on Discreteness

Through the principle of data division of discreteness, the skill movements of athletes during basketball training are detected, and **Figure 7** displays the collected data.

Figure 7 shows that, by dividing the action states of basketball players in the training process, it is possible to extract the action data of basketball players in the training process. **Figure 8** shows the relative angular velocity and angle analysis between the forearm and calf during walking dribbling.

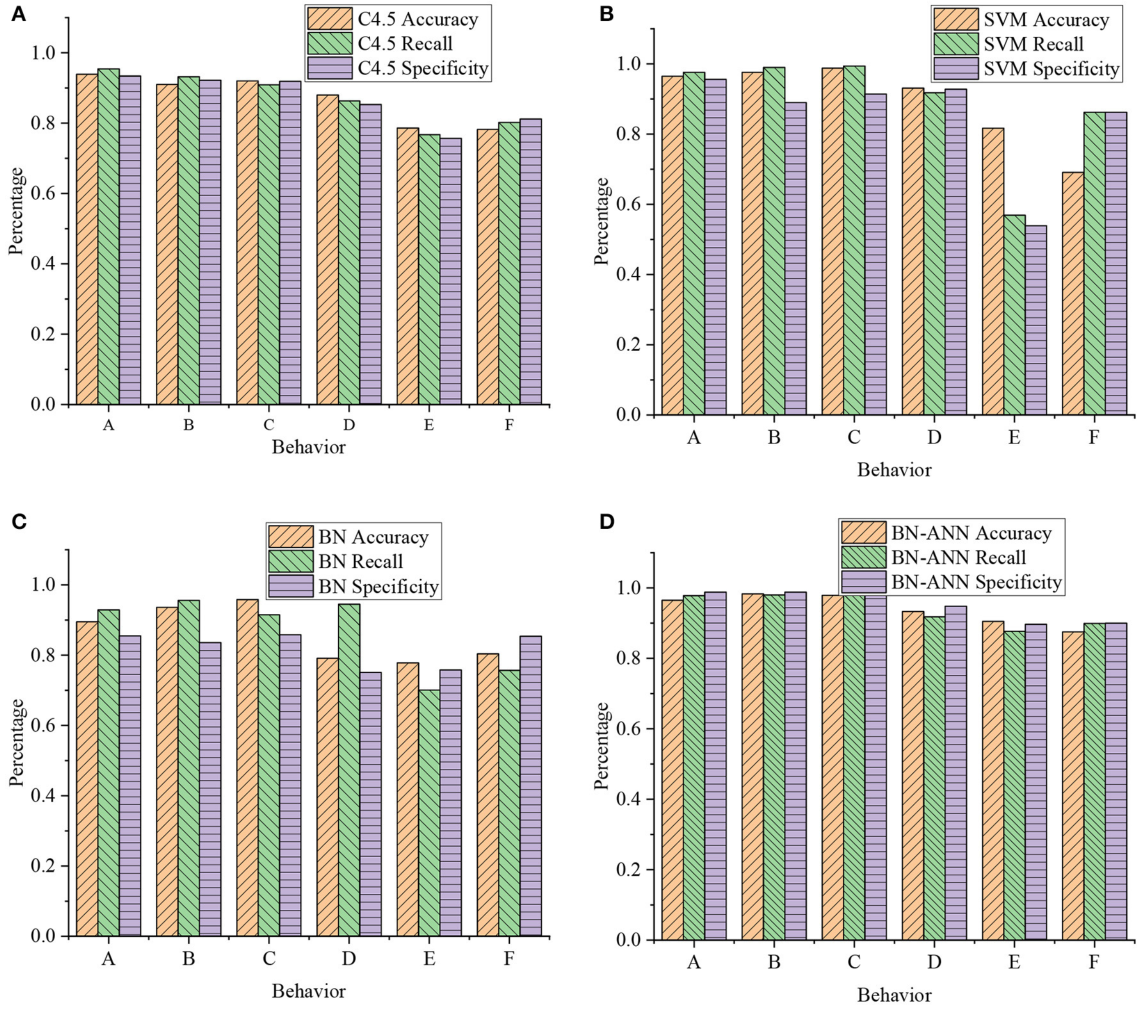


FIGURE 9 | Classification results of upper limb movements by different classification algorithms (A: catching; B: passing; C: shooting; D: dribbling on foot; E: dripping while standing; F: running dribbling). **(A)** accuracy and recall of C4.5; **(B)** accuracy and recall of SVM; **(C)** accuracy and recall of BN; and **(D)** accuracy and recall of back propagation–artificial neural network (BP-ANN)].

According to the experimental data in **Figure 8**, the results indicate that the angular velocity signals detected by basketball players when dribbling on the forearm and walking on the calf have significant noise signals, which lead to the unsmooth signal data curve. However, the angle signal curves detected by basketball players are relatively smooth when dribbling on their arms and walking on their calves. Therefore, the conclusion suggests that the angle can be taken as the division of unit actions, reducing the complexity.

Basketball Posture Recognition Test

The test of basketball gesture recognition is to verify the effect of basketball gesture recognition. The scenes are set of various basketball actions. The physical information data are obtained from the different detectors in varying basketball actions. Experiment, a large amount of data are collected. The corresponding experimental scene is also tested. The collected data are processed according to the described data processing and data division methods. The feature vector set describing the specific action is obtained. Thus, the sample set is constructed. Finally, the sample set is sent to the classifier. Here, the realization of the classifier uses the existing Weka platform to compare and analyze the performance of several classifiers.

The experiment will organize the walking, running, and jumping motion data of eight male testers without the ball to collect basketball motion data. When holding the ball, the motion data of stationary dribble, walking dribbling, running dribble, shooting, passing, and catching are collected separately. Each action is repeated for 50 times. There are 5,000 samples in total. When holding the ball, the upper body movements included stationary dribble, walking dribble, running dribble, shooting, passing, and catching, a total of 2,400 times. There are 2,600 lower

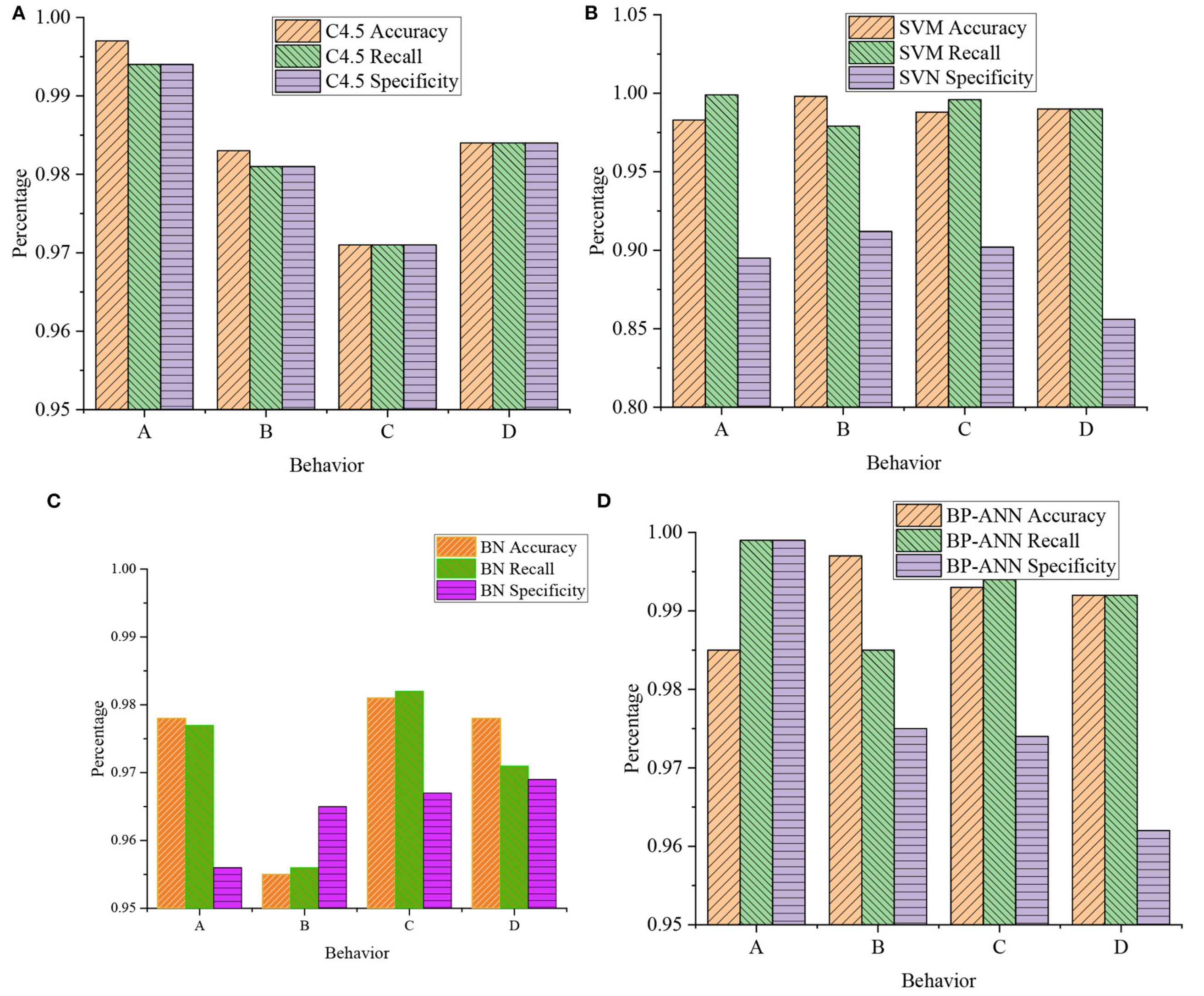


FIGURE 10 | Classification results of lower limb movements in basketball training by different classification algorithms (A: jumping; B: running; C: on foot; D: average level). **(A)** Accuracy and recall of C4.5; **(B)** accuracy and recall of SVM; **(C)** accuracy and recall of BN; and **(D)** accuracy and recall of BP-ANN.

body movements, including walking without the ball, running, jumping, walking, dribbling running, dribbling, and shooting. Each tester completed the required actions as required during the sampling process and monitored the recorded amount of activity. **Table 1** presents the statistical results of the samples collected by each tester during the data collection process. **Table 2** shows the statistics of different movements of the upper and lower extremities.

Basketball action is mainly an overall movement completed by the coordinated movement of the upper and lower limbs of an athlete. Therefore, when basketball moves are recognized, the upper and lower limb movements are discussed separately. In the data collection process, sensor nodes are placed in different positions of the body, and the data on upper and lower limb movements are collected and discussed, respectively. For upper and lower limb movements, classifiers are constructed separately for recognition. The moves made by the athletes are determined through the combination of upper and lower limb movements. The classification characteristics of different classifiers are analyzed. The classification performance of different classifiers for basketball gesture recognition is compared. A corresponding classification algorithm is constructed for training the motion data of additional limbs. In **Figures 9**, **10**, the whole experimental process is implemented on

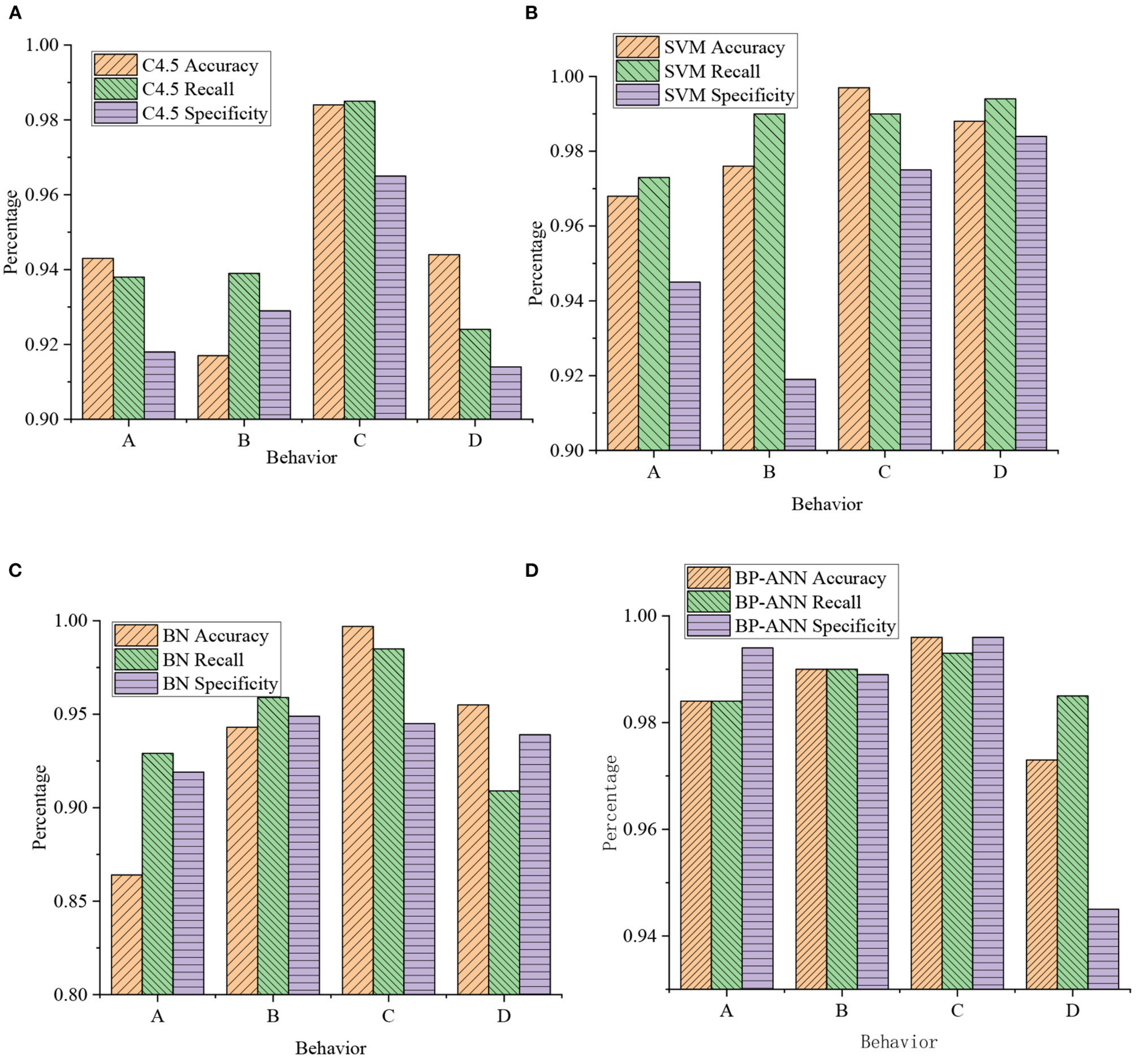


FIGURE 11 | Classification of upper limb movements by the four classification algorithms: **(A)** Accuracy and recall of C4.5; **(B)** accuracy and recall of SVM; **(C)** accuracy and recall of BN; and **(D)** accuracy and recall of BP-ANN (A: catching; B: passing; C: dribbling; D: shooting).

the Weka platform, and the recognition effect is analyzed from the two aspects of precision and recall. A 10-fold cross-validation method is used.

Figures 9, **10** are obtained through the data analysis of the body movements, as presented in **Tables 1**, **2**. The recognition effect of the back propagation (BP) ANN is better for the action classification of different limbs. Among them, the average accuracy rate of upper limb movements reached 93.2%. The average recall rate reached 93.2%. The average accuracy of lower body movements reached 99.2%. The average recall rate reached 99.2%. For the four recognition algorithms, the average accuracy of lower extremity movements (jumping, running, and walking) ranged from 97 to 99.2%. The average accuracy of upper body movements ranged from 84.9 to 93.2%. The recognition accuracy of upper body movements (catching, passing, and shooting) is relatively low. This is because the upper limb movement states of *in situ* dribbling, walking dribbling, and running dribbling are all dribbling states. The three dribbling characteristics are similar and difficult to distinguish. Meanwhile, the upper body movements of dribbling in place, walking, and running are considered as a movement state. The average recognition rate is up to 99%, and the average recall rate is up to 99%.

Analysis of Experimental Results of Different Classification Algorithms

Figure 11 shows the classification results of athletes' upper limb movements in basketball training after the four classification algorithms are merged.

Figure 11 shows the classifier established by the BP ANN algorithm, which can efficiently identify the basketball movements of the ball, passing, dribbling, and shooting. In detecting upper and lower limb movements, the recognition rate exceeds 95%. The average accuracy of the four actions is close to 98.95%. Meanwhile, among the four algorithms, the BP ANN algorithm can more accurately identify the technical movements of the upper and lower limbs.

CONCLUSION

The angular velocity signal detected by the basketball player's forearm dribbling and calf walking has a significant noise signal. The drawn signal data curve is not smooth enough. The angle signal curve detected by the basketball player's forearm dribbling and calf walking is relatively smooth. Therefore, the division of angles as unit actions can reduce the complexity. Meanwhile, the BP ANN algorithm showed the best action recognition effect. In the detection of upper limb movements of athletes in basketball training, the average accuracy is close to 93.3%. The average recall is immediate to 93.3%. In detecting lower limb movements of basketball-trained athletes, the average accuracy rate is close to 99.4%. The average recall is close to 99.4%. In detecting upper and lower limb movements, the BP ANN algorithm is used to build a classifier: the recognition method can efficiently identify basketball movements of ball, passing, dribbling, and shooting with a recognition rate of over 95%. The average accuracy of the four actions is close to 98.95%. This can accurately collect the sports parameters of the athletes in real-time, analyze and identify the sports postures of the athletes, and build a training effect evaluation model. The coaches make reasonable adjustments to the training program and scientifically evaluate the training quality. This is of great significance to improving athletes' competitive ability and coaches' decision-making ability.

However, this study also has certain limitations. The technical movements of basketball training are not rich enough, resulting in preliminary results. In the future, the basketball teaching assistant robot should combine the trainee's different action designs and tactical combinations by meeting the trainee's needs efficiently and purposefully. The robot will be equipped with various peripheral sensors, infrared (IR) sensors, visual sensors, etc., to monitor the target position in real-time and make corresponding defenses and counterattacks. These problems will be further improved to ensure the routine implementation of various assumptions and meet the needs of college physical education teaching basketball tactics.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

REFERENCES

Carnevale, G., Camisa, A., and Notarstefano, G. (2021). Distributed online aggregative optimization for dynamic multi-robot coordination. *arXiv. 2104.09847* 2, 12–14. doi: 10.48550/arXiv.2104.09847

Cox, A., Razzaghi, P., and Hurmuzlu, Y. (2021). Feedback linearization of inertially actuated jumping robots. *Actuators* 10:114. doi: 10.3390/act10060114

Dudnik, O., Marina, V., Kuznetsov, N., Marina, P., Irina, N., Larisa, V., et al. (2021). Trends, impacts, and prospects for implementing artificial intelligence technologies in the energy industry: the implication of open innovation. *J. Open Innov. Technol. Market Complex*. 7:155. doi: 10.3390/joitmc7020155

Echevarría, Y., Blanco, C., and Sánchez, L. (2019). Engineering Applications of Artificial Intelligence Learning human-understandable models for the health assessment of Li-ion batteries *via* Multi-Objective Genetic Programming. *Eng. Appl. Artif. Intelligene*. 86, 1–10. doi: 10.1016/j.engappai.2019.08.013

Estevez, J., Garate, G., and Grana, M. (2019). Gentle introduction to artificial intelligence for high-school students using scratch. *IEEE Access* 1, 1–1. doi: 10.1109/ACCESS.2019.2956136

Garcia, F., Vázquez-Guerrero, J., and Castellano, J. (2020). Physical demands between game quarters and playing positions on professional basketball players during official competition. *J. Sports Sci. Med*. 19, 256–263. doi: 10.1371/journal.pone.0221818

Gonzalo-Skok, O., Sánchez-Sabaté, J., and Izquierdo-Lupón, L. (2019). Influence of force-vector and force application plyometric training in young elite basketball players. *Eur. J. Sport Sci*. 19, 305–314 doi: 10.1080/17461391.2018.1502357

Graham, S., Depp, C., and Lee, E. E. (2019). Artificial intelligence for mental health and mental illnesses: an overview. *Curr. Psychiatry Rep*. 21:116. doi: 10.1007/s11920-019-1094-0

Ha, T., Pan, J., and Wang, H. (2019). Clinical value of lesion detection by using artificial intelligence system and reports for digital mammography. *Chinese J. Med. Imag. Technol*. 35, 1789–1793. doi: 10.1148/ryai.2021200299

Hagiwara, Y., Yabe, Y., and Sekiguchi, T. (2020). Upper extremity pain is associated with lower back pain among young basketball players: a cross-sectional study. *Tohoku J. Exp. Med*. 250, 79–85. doi: 10.1620/tjem.250.79

He, W. (2021). Research on basketball technical training and tactics training of college students based on big data. *J. Phys. Conf. Ser*. 1744:042004. doi: 10.1088/1742-6596/1744/4/042004

How, M. L., and Wei, L. (2019). Educational stakeholders' independent evaluation of an artificial intelligence-enabled adaptive learning system using bayesian network predictive simulations. *Educ. Sci*. 9:110. doi: 10.3390/educsci9020110

Hussein, M., Brozik, J., and Hopewell, H. (2020). Artificial intelligence: a potential prioritisation tool for chest radiographs with suspected thoracic malignancy. *Lung Cancer* 139:25. doi: 10.1016/S0169-5002 20)30086-6

Joshua, G., and Niloufer, S. (2020). A statutory right to explanation for decisions generated using artificial intelligence. *Int. J. Law Inform. Technol*. 2020:3. doi: 10.1093/ijlit/eaaa016

Kotter, E., and Ranschaert, E. (2021). Challenges and solutions for introducing artificial intelligence (AI) in daily clinical workflow. *Eur. Radiol*. 31, 5–7. doi: 10.1007/s00330-020-07148-2

Liu, Z. (2020). Application of artificial intelligence technology in basketball games. *IOP Conf. Ser. Mater. Sci. Eng*. 750:12093. doi: 10.1088/1757-899X/750/1/012093

Luo, Z., Bui, X. N., and Nguyen, H. (2019). A novel artificial intelligence technique for analyzing slope stability using PSO-CA model. *Eng. Comput*. 1, 1–12. doi: 10.1007/s00366-019-00839-5

Радутний, O. E. (2019). Moral and law for artificial intelligence and digital human: robotics laws and the "trolley problem". *Inform. Law* 3, 78–95. doi: 10.37750/2616-6798.2019.3(30).194801

Rampersad, G. (2020). Robot will take your job: innovation for an era of artificial intelligence - ScienceDirect. *J. Bus. Res*. 116, 68–74. doi: 10.1016/j.jbusres.2020.05.019

Sannino, G., Bouguila, N., Pietro De, G., and Celesti, A. (2019). Artificial intelligence for mobile health data analysis and processing. *Mobile Inform. Syst*. 12, 1–2. doi: 10.1155/2019/2673463

Senhaji, S., Faquir, S., and Jamil, M. O. T. (2021). robotics and artificial intelligence for the prevention of Covid 19 pandemic. *E3S Web Conf*. 229:01035. doi: 10.1051/e3sconf/202122901035

Shivale, N. M., Virkar, G., and Bhosale, T. L. A. (2020). System for classification of COVID-19 suspicious person using support vector machine classifier. *Int. J. Comput. Appl*. 176, 16–19. doi: 10.5120/ijca2020920470

Sobko, I. M., Chucha, Y., and Podmaryova, I. A. (2021). Application of the video-tutorial "challenge for referees" in sports training of young basketball referees for the game season. *Health Sport Rehabil*. 7, 42–53. doi: 10.34142/HSR.2021.07.01.04

Soferman, R. (2019). The transformative impact of artificial intelligence on healthcare outcomes. *J. Clin. Eng*. 44, 1–3. doi: 10.1097/JCE.0000000000000345

Taborri, J., Molinaro, L., and Santospagnuolo, A. (2021). A machine-learning approach to measure the anterior cruciate ligament injury risk in female basketball players. *Sensors* 21:3141. doi: 10.3390/s21093141

Wetzel, S. J., Melko, R. G., and Scott, J. (2020). Discovering symmetry invariants and conserved quantities by interpreting siamese neural networks. *Phys. Rev. Res*. 2, 14–19. doi: 10.1103/PhysRevResearch.2.033499

Xu, J., and Yi, C. (2021). The scoring mechanism of players after game based on cluster regression analysis model. *Math. Probl. Eng*. 2021, 1–7. doi: 10.1155/2021/5524076

Xu, T., and Tang, L. (2021). Adoption of machine learning algorithm-based intelligent basketball training robot in athlete injury prevention. *Front. Neurorobot*. 14, 1–2. doi: 10.3389/fnbot.2020.620378

Yang, Z. (2020). Research on basketball players' training strategy based on artificial intelligence technology. *J. Phys. Conf. Ser*. 1648:42057. doi: 10.1088/1742-6596/1648/4/042057

Zhang, J., and Shi, X. (2021). Design of execution system based on artificial intelligence technology. *J. Phys. Conf. Ser*. 1852:22033. doi: 10.1088/1742-6596/1852/2/022033

Zhao, Z., Liu, X., and She, X. (2020). Artificial intelligence based tracking model for functional sports training goals in competitive sports. *J. Intelligent Fuzzy Syst*. 40, 1–13. doi: 10.3233/JIFS-189374

Zhi, Y., and Jiang, Y. (2020). Design of basketball robot based on behavior-based fuzzy control. *Int. J. Adv. Robot. Syst*. 17:172988142090996. doi: 10.1177/1729881420909965

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

BRIEF RESEARCH REPORT
published: 29 June 2022
doi: 10.3389/fnbot.2022.867028

Application of Deep Learning Technology in Strength Training of Football Players and Field Line Detection of Football Robots

Daliang Zhou[1], *Gang Chen*[1] and *Fei Xu*[2*]

[1] School of PE, Nanjing Xiaozhuang University, Nanjing, China, [2] School of Physical Education, Hangzhou Normal University, Hangzhou, China

The purpose of the study is to improve the performance of intelligent football training. Based on deep learning (DL), the training of football players and detection of football robots are analyzed. First, the research status of the training of football players and football robots is introduced, and the basic structure of the neuron model and convolutional neural network (CNN) and the mainstream framework of DL are mainly expounded. Second, combined with the spatial stream network, a CNN-based action recognition system is constructed in the context of artificial intelligence (AI). Finally, by the football robot, a field line detection model based on a fully convolutional network (FCN) is proposed, and the effective applicability of the system is evaluated. The results demonstrate that the recognition effect of the dual-stream network is the best, reaching 92.8%. The recognition rate of the timestream network is lower than that of the dual-stream network, and the maximum recognition rate is 88%. The spatial stream network has the lowest recognition rate of 86.5%. The processing power of the four different algorithms on the dataset is stronger than that of the ordinary video set. The recognition rate of the time-segmented dual-stream fusion network is the highest, which is second only to the designed network. The recognition rate of the basic dual-stream network is 88.6%, and the recognition rate of the 3D CNN is the lowest, which is 86.2%. Under the intelligent training system, the recognition accuracy rates of jumping, kicking, grabbing, and starting actions range to 97.6, 94.5, 92.5, and 89.8% respectively, which are slightly lower than other actions. The recognition accuracy rate of passing action is 91.3%, and the maximum upgrade rate of intelligent training is 25.7%. The pixel accuracy of the improved field line detection of the model and the mean intersection over union (MIoU) are both improved by 5%. Intelligent training systems and the field line detection of football robots are more feasible. The research provides a reference for the development of AI in the field of sports training.

Keywords: deep learning, human action recognition, dual-stream network, functional strength training, recognition accuracy

OPEN ACCESS

Edited by:
Mu-Yen Chen,
National Cheng Kung University, Taiwan

Reviewed by:
José Luis Felipe,
European University of Madrid, Spain
Wei-Che Chien,
National Dong Hwa University, Taiwan

****Correspondence:***
Fei Xu
feixu@hznu.edu.cn

Received: *31 January 2022*
Accepted: *19 April 2022*
Published: *29 June 2022*

Citation:
Zhou D, Chen G and Xu F (2022) Application of Deep Learning Technology in Strength Training of Football Players and Field Line Detection of Football Robots. Front. Neurorobot. 16:867028. doi: 10.3389/fnbot.2022.867028

INTRODUCTION

Football has become the most popular sport in the world today due to its strong antagonism and enjoyment, and it is known as "the largest sport in the world." As the top football league, the Fédération Internationale de Football Association (FIFA) World Cup is held every 4 years. As of the 21st World Cup in 2018, the global TV audience has reached more than 3.5 billion (Pons et al., 2021). Football requires high physical fitness of athletes, especially strong jumping ability, which is an indispensable skill for shooting. It not only requires athletes' leg strength, but also other parts of the body (Bueno et al., 2021). The functional training of football refers to the realization of specific sports functions of the body for football (Teixeira et al., 2021). In addition to jumping ability, functional strength training in football sports includes training for kicking strength of their legs, the stability of athletes' upper and lower limbs during football competition, and training for athletes' acceleration and starting strength during competitions, etc. (Vella et al., 2021). Currently, the functional strength training of football players requires the coaches to adjust training items, supervise the athletes to complete strength training, and to analyze the training situation (Guan and Wang, 2021). Due to the limitation of the number of coaches and the interference factors of man-made supervision, the speed and strength of the athletes are improved to a small extent within a certain period. Therefore, artificial intelligence (AI) is introduced to study the functional training of football. Since the requirements for AI in the functional training of football players are in semantic extraction and action recognition, based on deep learning (DL) and human action recognition related knowledge, the speed and strength improvement of athletes in functional training is studied. Currently, human action recognition is one of the research hotspots in the field of computer vision, and it is gradually being applied in many fields, such as virtual reality, video surveillance, the interaction of AI, medical assistance, etc. (Materne et al., 2020). The research content of human action recognition mainly includes the extraction and analysis of human action and the detection of moving targets (Parim et al., 2021). The research often encounters discrepancies in different environments, so many methods of expressing features have been proposed, such as DL algorithms. DL algorithms are different from traditional artificial feature extraction, which saves a lot of time and manpower (Aslam et al., 2021).

Deep learning algorithms belong to the field of machine learning and are mainly used in the learning of complex semantics by machines. The DL uses hierarchical extraction of features and feature learning methods, and currently, it is widely used in audio recognition, processing of natural language, and in other fields (Khan et al., 2021). As one of the common models of DL, the convolutional neural network (CNN) shows great advantages in visual recognition tasks. Human action recognition will be studied by combining CNN with the DL algorithm (Khan et al., 2020). Human action recognition under DL mainly extracts the global and local information of the video image in the computer, and it performs feature extracting and analysis on this part of the information, and then conducts the recognition of the target behavior (Ullah et al., 2020).

As an important experimental platform for the research of AI application technology and multi-agent system (MAS), the football robot is a vital way to break through the research in the field of AI technology. Meanwhile, it is also extremely challenging for scientific researchers. Its related technologies include robotics, AI, intelligent control technology, sensor technology, communication, computer and image processing, and other fields. A football robot is a typical MAS in the form of competition. The research on its structure theory and control method has promoted the development of methods and theories of robotics and AI technology (Setyawan et al., 2022). Intelligent robots have been widely used in various fields of social life. As one of the intelligent robots, football robot games have often been held around the world in recent years. It integrates technology, viewing and fun, effectively stimulates the enthusiasm of young people for research, and provides a stage for cultivating all aspects of ability. Students closely combine scientific knowledge with practice, link theory with practice, and exert their intelligence through practical operation, teamwork, innovation, creation, etc. (Suwarno et al., 2022). The actual technological level and control skills can be evaluated in the fairest, most reasonable, and objective way through the robot football game. Through the competition, its own deficiencies are found, to improve and innovate technology, and promote the development of related technologies (Wang, 2022). The Robo Cup Humanoid Competition is the highest-level competition in the field of humanoid football robots. The robot captures the field image through the camera, performs field line detection on the image, extracts the feature information in the field line that can be used for robot positioning, combines with the robot's movement information, performs robot self-positioning, and makes decisions and path planning based on the detection and positioning information to complete match (Watanabe et al., 2022). Since the current football robot positioning detection is unstable, it will be combined with the fully convolutional network (FCN) for research.

The traditional functional strength training of football players has the problems of insufficient coaches and slow speed in improving the speed and strength of football players. In the era of AI, the training problem of football players can be effectively solved. Based on the current situation of football player training, the AI is introduced to study the players' functional training, aiming to improve the detection technology of football robots. First, the research background of football training and football robots in the era of AI is introduced. Second, combined with video recognition technology, a functional strength training system by AI and CNN is established, combining time flow and spatial flow network to build a field line positioning system for football robots based on FCN. Finally, the feasibility of the two systems is evaluated. The innovation lies in the application of the human action recognition system under DL to solve the problems existing in the ability training of football players. The main difference between the designed algorithm and other algorithms is to propose an improved model based on FCN and residual network for field line detection of football robots. The used data set is an image data set containing 4,800 videos. To sum up, the feasibility of the provided intelligent training system and the high accuracy of the football detection of football robots are obtained. The research results provide a new way for the development of strength training of football players and the detection technology of football robots.

DESIGN OF AN INTELLIGENT SYSTEM FOR FUNCTIONAL STRENGTH TRAINING AND RESEARCH ON FIELD LINE DETECTION OF FOOTBALL ROBOT

Research Status of Human Action Recognition

Human action recognition algorithms before the emergence of DL algorithms were mainly studied through shallow learning and spatiotemporal features. For example, sampling the pixel points in the video for dense trajectory acquisition, and extracting feature information as the result of the recognition action; acquiring the research target trajectory in the video by the information of the optical flow field, extracting feature information, and then performing feature encoding to obtain recognition results; researching the key point of optical flow information in the video through the sequence of video; combining Gaussian distribution to perform feature extraction on the information in the video, etc. (Fan et al., 2020).

Since the system of human action recognition involves a lot of domains, including feature selection, machine vision, pattern recognition, etc., it is also a very challenging processing method for the computer (Pareek and Thakkar, 2020), so it has important significance in the research of human action recognition. The current research on human action recognition has made great progress, but there are still some shortcomings, such as insensitivity to the differences among different targets, difficulty in processing under the influence of complex video environment background, and differences between video databases and real data, etc. (Ozcan and Basturk, 2020). The emergence of DL algorithms has gradually replaced artificial representation methods, and the above problems have been studied. Methods of action recognition based on DL include CNN and dual-stream algorithms (Abdelbaky and Aly, 2021).

The main research direction in DL is the dual-stream algorithm, which aims at using the two dimensions of space and time to study video, and then combines the optical flow image and the video frame image to train the DL model, and to finally obtain the recognition result (Lagemann et al., 2021). The CNN has a simple network structure and a large-scale processing data set, so it is widely used. Among them, the 3D CNN extracts and recognizes video images through the three-dimensional convolution kernel, which is more comprehensive than the two-dimensional algorithm (Fan et al., 2020).

The research on human action recognition has been carried out earlier, but it has only been applied in sports in recent years. Research on human movement in the field of sports involves sports, such as basketball, badminton, etc. Football research is mainly focused on tracking the trajectory of football players in the game, and there is less research on strength training (Stoeve et al., 2021). Thakkar and Shah (2021) proposed the use of wearable devices for action recognition (Thakkar and Shah, 2021). In addition, the research methods for human action recognition include computer vision research (Newman et al., 2021). Football is a highly antagonistic sport and is not suitable for wearing equipment. Therefore, the visual aspects of human action recognition are analyzed.

Research Status of Football Robots

In the 1992 International AI Conference, Professor Alan Mackworth of Canada first proposed the idea of robot football in his paper, “On Seeing Robot.” The Federation of International Robot Soccer Association (FIRA) competition was first proposed by Professor Kim Jong-hwan of South Korea in 1995. The first and second international competitions were held in Korea in 1996 and 1997, respectively (Chen and Gao, 2020). In June 1997, the FIRA was announced during the second micro-robot football competition. Since then, FIRA holds an international competition every year and an academic conference (FIRA Congress) to exchange experience and technology in the robot football research. In 2011, the 16th FIRA Robo World Cup and Congress were held in Taiwan for the first time, with more than 10 national teams including South Korea, Canada, Mexico, the United Kingdom, Singapore, Malaysia, Thailand, Argentina, Slovakia, and China participated in the competition, and 53 teams competed on the same stage. This is the FIRA game, one of the international robot soccer game series organized by FIRA so far (Chen, 2020).

With the development of the FIRA competition, another major series of football robot competitions, the Robo Cup, has also been developed by leaps and bounds during the same period. The Robo Cup J League was planned in June 1993 by Minoru Asada, Hiroaki Kitano, and Yasuo Kuniyoshi. However, within a month, it received a positive response from scientific researchers around the world, and therefore changed its name to Robo Cup, that is, the Robot Football World Cup. In 1997, the first official competition and meeting of the Robo Cup was held in Nagoya, Japan, with 38 teams from 11 countries participating in the competition (Hong et al., 2021). The competition was a huge success. Before this competition, many scientific researchers did a lot of pre-competition preparations, such as drafting rules, organizing simulation group competitions, etc., to ensure the success of this competition. Since then, the Robo Cup competition is held around the world every year, and then the exchange meeting on its scientific research progress is held, which provides a good communication platform for scientific researchers and promotes the rapid development of their related scientific and technological fields (Houtman et al., 2021).

In 1997, Chinese researchers began to study robot football, starting relatively late. In 1998, the Chinese branch of the FIRA was established. In the Robo Cup held in Brazil in August 1999, the New Neu team of Northeastern University won the first place in standard action and the fifth place in Mriosot, realizing the breakthrough of the Chinese team’s 0 gold medal in the Robo Cup (Antonioni et al., 2021). In 2000, the Robo Cup was held in Beijing for the first time. In the Robo Cup in August

2001, 8 championships among 9 competitions were won by the Chinese team, which fully proves that China has achieved a very high development level in the field of robot football research. After more than 10 years of unremitting efforts, with the strong support of FIRA and the Chinese Society for AI, the Chinese robot football team has spread all over the country. So far, there are more than 200 teams belonging to more than 80 universities. More and more scientific research institutes and universities have begun to organize scientific research forces to join the team of football robots, which has greatly promoted the development of football robots and the related fields of science and technology.

The rules of the game before 2013 stipulated that the game ball is orange, the goal is yellow or blue, and the field line is white, so the target detection algorithms are all algorithms based on color segmentation. As the rules of the game continue to change, the ball becomes a mixture of black and white, the goal and field lines are both white, and the algorithm based on color segmentation is no longer effective. Based on the odometry-based positioning method, Park et al. (2022) calculated the current robot's pose relative to the initial moment by accumulating the measurement results (Park et al., 2022). To achieve higher accuracy, filtering methods, such as Kalman filtering can be used for coordination. The advantage of this method is that it can quickly provide the robot pose, but the disadvantage is that it has accumulated errors. As the movement time or distance increases, the error of its pose estimation will also increase. Due to the high confrontation of robot football games, the robot collisions between them occur frequently, so this method is only suitable for short-term and short-distance self-positioning. Jeong et al. (2022) proposed a positioning method based on sensors, such as laser rangefinders and sonars, using a rotating mirror mechanism to emit laser beams outwards, and to detect the laser beams reflected by objects, and to obtain external environmental information. This positioning method has high positioning accuracy, good anti-interference, no cumulative error, and a short positioning period, but the equipment is expensive, and the cost is high (Jeong et al., 2022). Colombini et al. (2022) proposed a vision-based positioning method by obtaining images of surrounding scenes through visual sensors, using some natural or artificial features in the scenery, and obtaining models of the surrounding environment through image processing methods to achieve positioning (Colombini et al., 2022). According to the different positioning markers, this method can be divided into the positioning method based on road signs and based on white lines. Due to the limited observation range of general omnidirectional vision, it is difficult to observe enough positioning marks in most areas of the site. The CNN has developed rapidly in recent years. Its excellent performance in image processing has led more and more researchers to abandon algorithms based on color segmentation and use CNN to detect balls and goals but still uses a color segmentation-based algorithm to detect yield lines. The field line detection algorithm based on color segmentation has a small amount of calculation and a fast operation speed, but it is only suitable for the situation of stable lighting conditions, and the detection effect becomes worse when the lighting conditions change. The technical committee of the Robo Cup has issued a notice that the competition venue will be illuminated with natural light. Therefore, the field line detection effect based on the color segmentation algorithm will become unstable, and new methods need to be studied to cope with changes in lighting conditions and improve the detection stability.

Therefore, in the field detection part of the football robot, aiming at improving the detection stability, based on the FCN model, an improved model is proposed for the field line detection of the football robot. The design idea of the residual network model is adopted, and the number of network convolutional layers is increased. The residual structure is introduced to enhance the ability of the network to extract image features, and more low-level field line information is incorporated in the process of upsampling to improve the accuracy of the field site line.

Theory Basis of DL

The Neuron Model of Neural Network

The principle of DL is to improve the computer performance from both new data and historical experience after being input into the computer (Sarker, 2021). The research of DL algorithms originated from the development of artificial neural networks, so DL also has the characteristics of neural network hierarchical structure (Chen et al., 2021). The neural network model is a model similar to the neural mechanism of the human brain designed by humans in order to allow computers to recognize behaviors and understand images and other complex behaviors. The neural model contains a hierarchical structure composed of neuron nodes. Neurons are its basic components. The specific expression of neurons is shown in Equation (1) (Kuwana et al., 2020).

$$\sigma_{w,b}(X) = \sigma\left(W^T x\right) = \sigma\left(\sum_{i=0}^{n} w_i x_i + b\right) \tag{1}$$

In Equation (1), X is the output value; W is the weight; b means the offset value; n is the number of inputs, i represents the input sequence number; σ is the output value.

Neurons between different levels are connected through input and output, and the relationship between output and input is described by an activation function. In general action recognition research, choosing a non-linear function as the activation function is of great significance in solving complex problems in real life. The more commonly used activation functions include the hyperbolic tangent function, the step function, the S-shaped growth curve, modified linear unit, etc. The specific expressions of these four functions are shown in Equations (2)–(5).

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2}$$

Equation (2) is the expression of the hyperbolic tangent function, and the output value ranges from −1 to 1.

$$sgn(x) = \begin{cases} 1, \ x \geq 0 \\ 0, \ x < 0 \end{cases} \tag{3}$$

Equation (3) is the expression of the step function. It can be seen from Equation (3) that when the input value is less than 0, the output of the step function is 0, which means that the neuron has not received excitation; when the input value is greater than or equal to 0, the output of step function is 1. It can be seen that the step function is an idealized expression, and it is used to solve complex practical problems.

$$sigmoid(x) = \frac{1}{1+e^{-x}} \tag{4}$$

Equation (4) is the expression of the S-shaped growth curve function, and the output value range is between 0 and1. When the input value of the S-shaped growth curve function has an extreme value, the gradient of the function approaches 0, which will affect the back propagation, so it is not suitable for DL.

$$Relu(x) = max(0,\ x) \tag{5}$$

Equation (5) is the expression of the modified linear element function. The calculation of the modified linear element is relatively simple, and it has the characteristics of fast derivative convergence, so it is widely used to study the problem of gradient disappearance.

The feedforward neural network composed of a certain combination of multiple neurons is an important part of the study. The function of the feedforward neural network is to spread samples. The network structure of the feedforward neural network has no closed loops and no feedback information in the forward direction. The specific structure is shown in **Figure 1**.

Figure 1 shows the three components of the feedforward neural network, including the output layer, input layer, and hidden layer. In the figure, a_i represents the output of the *i*th neuron, and the specific equation expressions are shown in Equations (6)–(9) (Haldorai and Ramu, 2020).

$$a_1^{(2)} = f\left(H_{11}^{(1)} m_1 + H_{12}^{(1)} m_2 + H_{13}^{(1)} m_3 + c_1^{(1)}\right) \tag{6}$$

$$a_2^{(2)} = f\left(H_{21}^{(1)} m_1 + H_{22}^{(1)} m_2 + H_{23}^{(1)} m_3 + c_2^{(1)}\right) \tag{7}$$

$$a_3^{(2)} = f\left(H_{31}^{(1)} m_1 + H_{32}^{(1)} m_2 + H_{33}^{(1)} m_3 + c_3^{(1)}\right) \tag{8}$$

$$q_{H,c}(m) = a_1^{(3)} = f\left(H_{11}^{(2)} a_1^{(2)} + H_{12}^{(2)} a_2^{(2)} + H_{13}^{(2)} a_3^{(2)} + c_1^{(2)}\right) \tag{9}$$

In Equations (6)–(9), *m* is the input information; *H* is the weight parameter; *c* is the bias parameter; *q* is the final output result of the hidden layer, and $a_i^{(1)}$ represents the output of *i*th neuron node in the *l*th layer. $H_{ij}^{(l)}$ represents the weight of the *i*th node of the *l*th layer connected to the *i*th node of the first layer, and $c_i^{(l)}$ represents the bias parameter of the *i*th neuron of layer *l*, and *f* is the activation function.

The input weighted sum is introduced to represent the *i*th node of the *l*th layer, and the above equations are simplified to obtain Equations (10) and (11).

$$v^l = H^{(l-1)} f(v^{(1-l)}) + c^{(l-1)} \tag{10}$$

$$a^l = f(v^{(l)}) \tag{11}$$

f is the excitation function in Equations (10) and (11).

Next, data update is needed to complete network training. The update of parameter is performed by the back propagation method. The loss function expression of the network output layer is shown in Equation (12).

$$P(H, c;\ m, y) = \frac{1}{2}\left\| q_{H,c}(m) - y \right\|^2 \tag{12}$$

In Equation (12), *y* is the output result obtained, and *P* is the loss function.

The residuals of different nodes are expressed by Equation (13).

$$\beta_i^{(l)} = \left(\sum_{j=0}^{s_{l+1}} H_{ji}^{(l)} \beta_i^{(l+1)}\right) f'(v_i^{(l)}) \tag{13}$$

In Equation (13), *s* represents the number of nodes in all layers.

The relationship between the loss function and the input value is expressed as Equations (14) and (15) using residuals.

$$\frac{\partial}{\partial H_{ji}^{(l)}} P(H, c;\ m, y) = a_j^{(l)} \beta_i^{(l+1)} \tag{14}$$

$$\frac{\partial}{\partial b_j^{(l)}} P(H, c;\ m, y) = \beta_i^{(l+1)} \tag{15}$$

Equations (14) and (15) express the rate of change of the loss function to the input.

Finally, the gradient descent method is used to update the parameters *H* and *c*, and equations (16) and (17) are obtained.

$$H_{ji}^{(l)} = H_{ji}^{(l)} - \alpha \frac{\partial}{\partial H_{ji}^{(l)}} P(H, c) \tag{16}$$

$$c_j^{(l)} = c_j^{(l)} - \alpha \frac{\partial}{\partial c_j^{(l)}} P(H, c) \tag{17}$$

α represents the learning rate in Equations (16) and (17).

Theoretical Basis for CNN and DL

Convolutional neural network is mainly used to extract feature information. It can effectively combine the training process and feature extraction, and has good results in image processing. **Figure 2** shows the structure of CNN.

Figure 2 implies that CNN is mainly composed of an output layer, a fully connected layer, a pooling layer, a convolutional layer, and an input layer.

The convolutional layer is the core component of CNN and contains a certain number of convolution kernels. Convolution is similar to a filtering operation and is mainly used to extract features. The CNN is an inter-layer structure, which uses neurons between layers to connect. After the original image is input, it performs certain operations with the convolution kernel to output certain characteristic information. The main function of the pooling layer is to improve the generalization ability of the neural network model, while preventing the occurrence of over-fitting. The fully connected layer is usually set at the

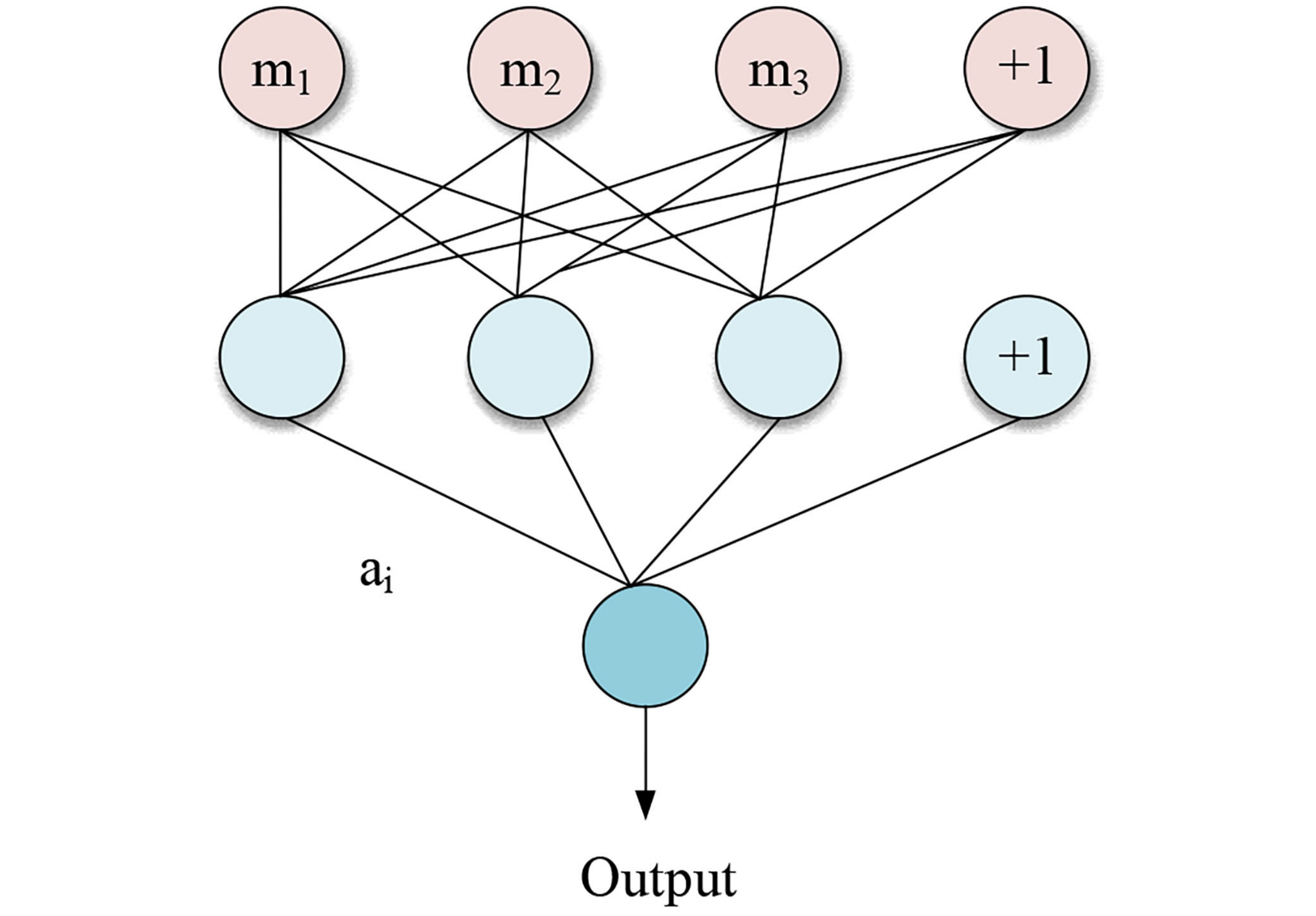


FIGURE 1 | Structure of feedforward neural network.

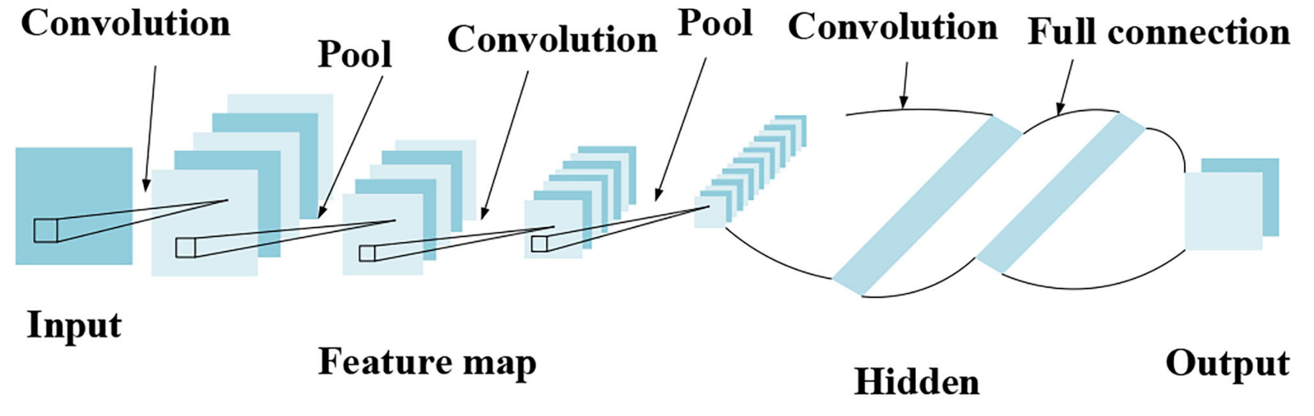


FIGURE 2 | Structure of CNN.

end of the network structure, and the convolutional layer and the pooling layer are connected to synthesize the extracted two-dimensional feature vectors and form a one-dimensional feature vector. Each neuron in the fully connected layer is fully connected to all neurons in the previous layer to integrate local information in the network, so it occupies most of the parameters in the network. In actual use, the fully connected operation can still be implemented by a convolution operation. The main difference between the fully connected layer and the convolutional layer is that the neurons of the convolutional layer and the input data are locally connected, and the neurons share parameters.

TABLE 1 | The specific content of the design system.

Design system module	Specific content
Service processing module	Training information storage query, identity information authentication and data processing
Video image processing module	Face recognition, motion recognition system, scoring system

There are many application frameworks for DL. Among them, the Caffe framework (Somu et al., 2021) can be well combined with CNN, which is widely used in solving image problems. The Caffe framework stores data and builds training and network core units, and finally calculate the loss to get the function of a given task.

Design of the System of Functional Strength Training Based on AI

Adopting a client/server architecture, football players' action information is collected and transmitted to convert the information into optical flow diagrams for action recognition, and finally the analysis results are obtained (Zhang et al., 2020). Camera equipment is mainly used to capture the action videos of the players to achieve the purpose of intelligent functional training for football players. The coach only needs to analyze the results in the server to get the training effect of each football player.

The design system includes two modules, the business processing module, and the video image processing module. The specific content is shown in **Table 1**.

Table 1 suggests that the two modules cooperate with each other to finally complete the intelligent functional training for football players.

The overall framework of the design system is shown in **Figure 3**.

Figure 3 reflects that the program of server-side recognizes and synchronizes the video data of the football players' training actions, and finally obtains the action recognition and feature extraction of the video data. The client program collects video data, then accepts the results of data recognition, and finally displays the training results of each athlete.

The specific workflow chart of the whole system is shown in **Figure 4**.

Figure 4 shows that the client has instructions for the training start. Once the training starts, the client performs camera shooting and synchronizes the video data to the server. The server starts the action recognition at the same time. During the training process, the client can accept user instructions throughout the entire process to stop training. The server finally feeds back the results. After receiving feedback, the client compares the recognition result of the action in the video with the training mode in the database, displays the completion of training on the client interface, and saves the data (Pareek and Thakkar, 2020).

The main content of the client software workflow is shown in **Figure 5**.

Figure 5 presents that the client is composed of five parts. The client starts to work after receiving the start instruction. After the training video is obtained, it is synchronized to the server port immediately. After the server feeds back the recognition result, it performs real-time show on the client section interface.

The server-side software workflow is shown in **Figure 6**.

Figure 6 shows that the server-side workflow consists of six parts. After receiving the video data, the server-side starts preprocessing the data and extracting feature, then performs network fusion, and finally feeds back the recognition results to the client terminal for interface displaying.

Since the time of data transmission of Socket is short and the amount of data is small, the communication protocol among the modules in the design system adopts Socket (Elharrouss et al., 2020).

Based on the above content, the overall communication connection of the system of functional strength training is summarized, and the results are shown in **Figure 7**.

Figure 7 indicates that the overall process of communication connection of the system of functional strength training strengthens the connection between the client and the server segment, which makes the video information processing faster, and at the same time, the server continues to establish new threads to ensure a complex client service demand resolution.

Video Action Recognition

In recent years, most research has been inspired by the dual-stream CNN, which combines the spatiotemporal information extracted from the Red Green Blue (RGB) image and the optical flow image of the video, and extracts and recognizes two types of features from separate CNNs. Finally, video information generated from the prediction result contains two parts: spatial information and time information. Compared with static images, the time series components of video provide additional time-representing motion information for action recognition. The spatial information in the video is the position on each frame of the image, which represents the spatial information, such as the target and scene in the video; the temporal information refers to the change between video frames and carries the movement information of the object between video frames, including the movement of the camera or movement information of the target object, etc. The idea of recognizing video action mainly includes two categories, namely, the method of extracting video spatiotemporal features for video recognition, and the method of retraining by using human skeleton node information as network input data. The system uses the ordinary camera to collect motion video for recognition, so the dual-stream CNN is mainly used to extract the temporal and spatial features of the video for action recognition and analysis. The streaming CNN is composed of a convolutional network that expresses two-dimensional information of spatial flow and time-domain flow. It is used to process the spatiotemporal information of video data. The design architecture of dual-stream networks comes from the dual-stream hypothesis in the research of biological vision systems. Both streams are CNNs with the same

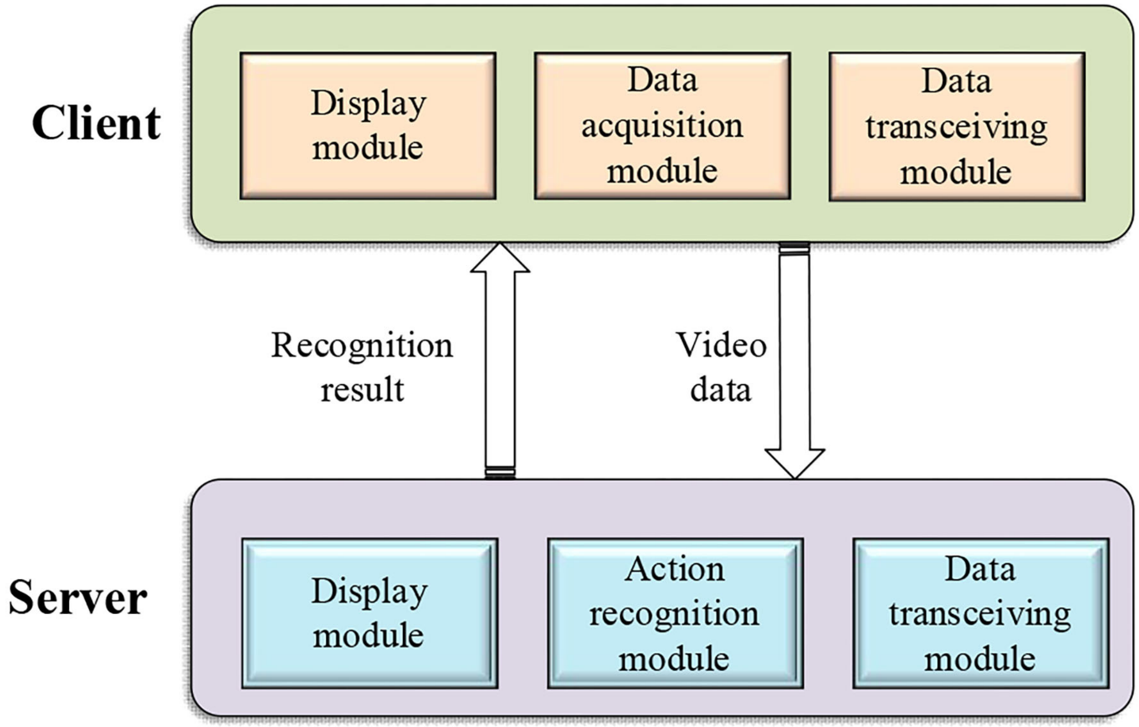


FIGURE 3 | The overall framework of the design system.

structure. The spatial stream network takes a single-frame RGB image of the video as input. The decoupling of the spatial stream and the temporal stream network also enables the use of image data on large image data sets. The pretraining of the spatial stream network is used to identify the surface features related to the action and describe the feature in the spatial domain in the video. The spatial flow network is the same as the common static image recognition network, while the network of time flow inputs multi-frame stacked flow images into the network for training, and is used to learn the time features contained in the action, such as the movement and deformation of the target. It also describes the characteristics of video actions in the time domain. Using the method of multi-task training to provide two output layers of softmax for fusion, the output of the softmax layer is the probability of identifying the action category, and providing two softmax outputs is equivalent to the process of regularization. There are two main fusion methods: averaging and using the softmax layer to retrain a support vector machine (SVM) classifier for recognition.

This section gives a detailed description of the core of the research, namely video action recognition. Based on the previous content, a dual-stream CNN is used to study the recognition of remote mobilization training actions. Optical flow is the velocity vector of human motion in the video. The optical flow image is extracted from the video for data processing and motion feature extraction.

The most famous methods for calculating optical flow are the Hom—Schunck algorithm and the Lucas—Kanade algorithm. The Hom—Schunck algorithm is based on the fact that the gray level of the object image remains unchanged in a short time interval, assuming that the velocity vector field changes slowly in a given neighborhood, and then the calculation of optical flow under global smoothness constraint of the optical flow field is proposed. However, due to the smoothness assumption that the algorithm is based on, the vector estimation of optical flow may have a large error in the edge area of the image or the presence of occluded areas. The Lucas—Kanade algorithm uses local smoothness constraints, that is, assuming that all pixels in a small neighborhood have similar motions, and then realize the estimation of optical flow. Compared with the previous algorithm, the Lucas—Kanade algorithm is simple to implement, and it has lower computational complexity. After comparison, the Lucas—Kanade algorithm is selected to calculate the optical flow (Pratama, 2021). The specific calculation equation of optical flow is shown in Equation (18).

$$D^T D\vec{w} = D^T(-c) \tag{18}$$

In Equation (18), T refers to time; c is a constant term; and D is a coefficient matrix, $w{=}dy/dt$, where y is the ordinate position of the action. Solving Equations (18) and (19) (Zhao et al., 2021) can be obtained.

$$\begin{bmatrix} u \\ w \end{bmatrix} = \begin{bmatrix} \sum_i F_x(X_i)^2 \& \sum_i F_x(X_i)F_y(X_i) \\ \sum_i F_x(X_i)F_y(X_i) \& \sum_i F_x(X_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i F_x(X_i)F(X_i) \\ -\sum_i F_y(X_i)F_t(X_i) \end{bmatrix} \tag{19}$$

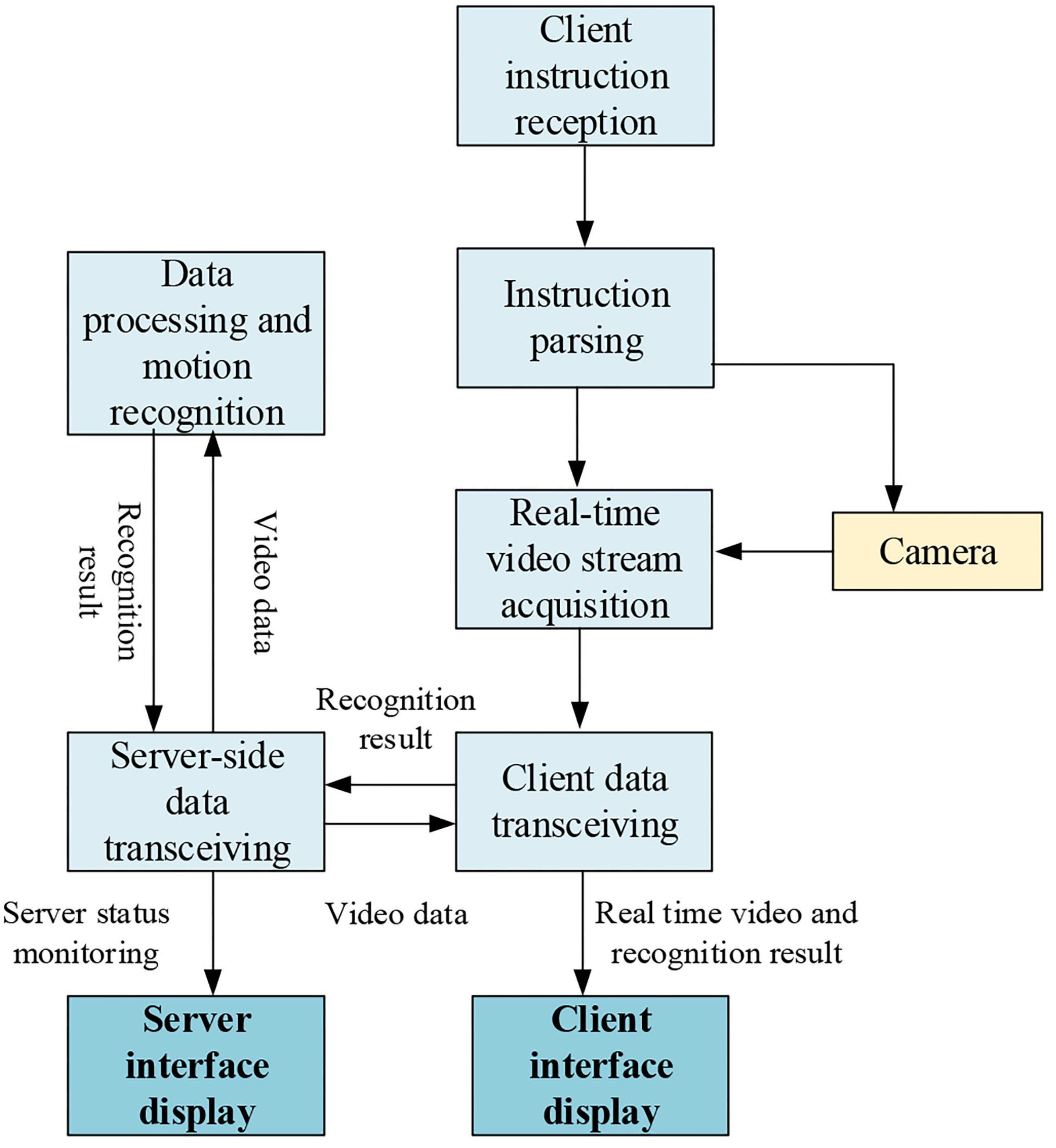


FIGURE 4 | The specific workflow chart.

In Equation (19), u=dx/dt, F is the brightness, and x is the abscissa of the position of the action.

Data collection uses Haikang camera equipment to obtain the video, and then uses the kit for software development of equipment network to convert the information flow into a three-color mode, and finally uses a sparse sampling strategy to process the video information. The collection process of real-time video stream first needs to initialize the software development kit (SDK). The device network SDK is developed by Haikang based on the communication protocol of private networks. It is a supporting module for network hard disk recorders, video servers, network cameras, and other products. It can be used for

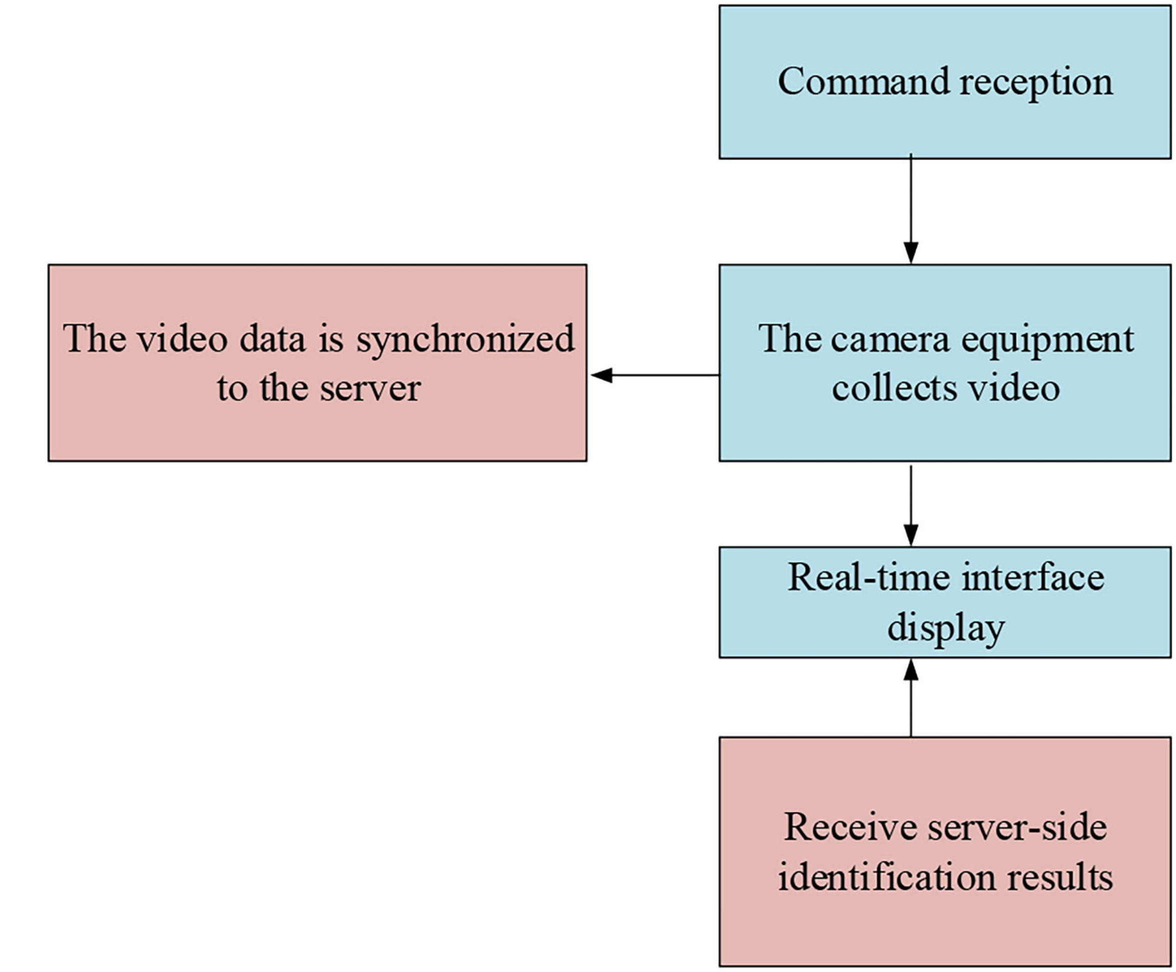

Command reception
The video data is synchronized to the server
The camera equipment collects video
Real-time interface display
Receive server-side identification results

FIGURE 5 | The client software workflow.

remote visit equipment and conduct secondary development. It could verify and login by setting the Internet Protocol (IP), port number, and other information of the webcam; then real-time video preview could be realized by setting the playback channel, video stream types, connection methods, and other information. It could be realized that the real-time video stream acquisition is done through the playback callback function; in addition, setting the connection timeout, reconnection time, and abnormal state callback functions need to be done; finally, the SDK is called to convert the obtained real-time video stream into Red, Green, Blue, Alpha (RGBA) data, and the entire video stream collection is completed. After using the Haikang camera to obtain the real-time video stream, the data transmission module on the client end will synchronize the video stream data frame by frame to the server-side program for processing, and save it as a video file in the client hard disk at the same time.

Based on the previous content, the image data set the size of the input training of the network design as 224 × 224. Through the first layer of convolution conv1, the size of the convolution kernel is set to 7 × 7, and the step size is set to 2, with a total of 96 convolution kernels, outputting 96 feature maps of 112 × 112 size. After the pooling operation, 96 feature maps with a size of 56 × 56 are output. Then through the second layer of convolution, conv2, the size of the convolution kernel is set to 5 × 5, and the step size is set to 2, and a total of 25 convolution kernels and 256 feature maps with a size of 28 × 28 are output. After the pooling operation, 256 14 × 14 feature maps are output. Then through the third layer of convolution, conv3, the fourth layer of convolution, conv4, and the fifth layer of conv5 are output. If the size of the convolution kernels of the three layers are all 3 × 3, the step size is also 1, and the number is also 512, so that after the last three layers of convolution, 512 14 × 14 feature maps are output. Then, after the pooling operation, 512 7 × 7 feature maps are output. Finally, through the two fully connected layers, full 6, full 7 (global average pooling avg pool), and the final Softmax, the output is obtained as a 1 × 1 × 101 or 1 × 1 × 51 vector. The main content of the functional training database for football players involved in this study is shown in **Table 2** (Cournoyer et al., 2021).

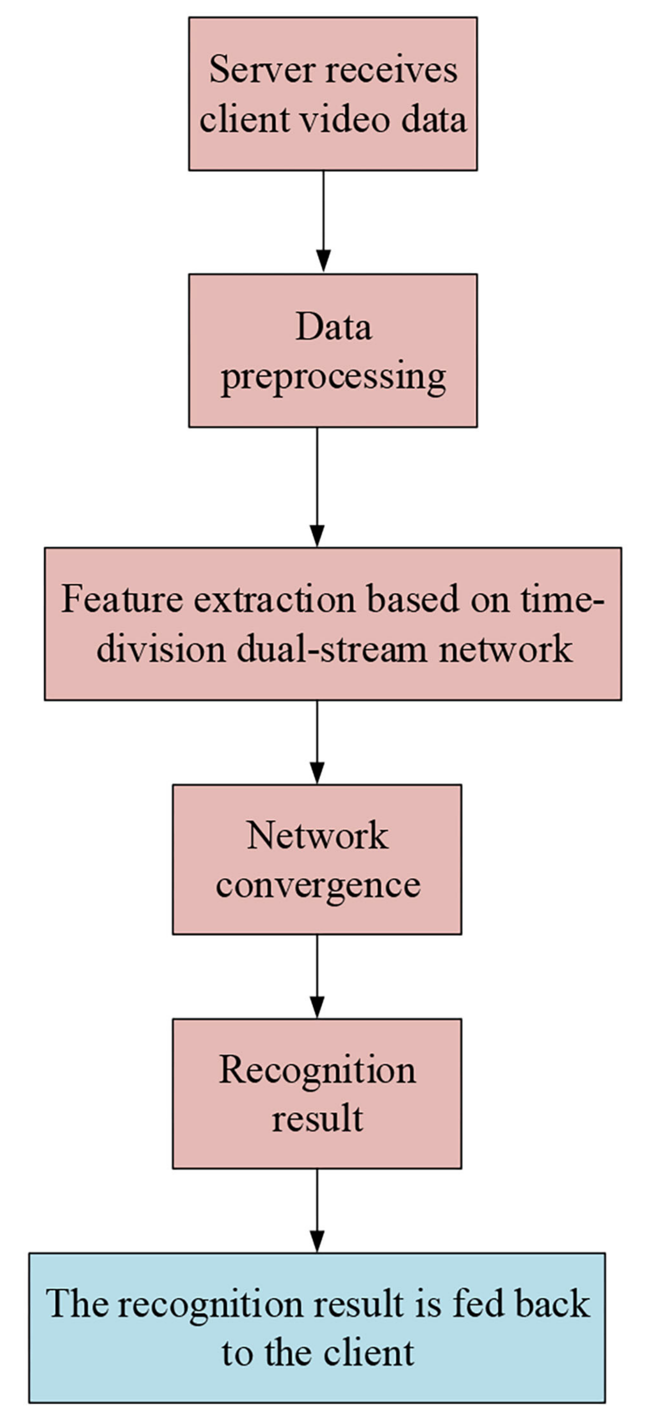

Server receives client video data
Data preprocessing
Feature extraction based on time-division dual-stream network
Network convergence
Recognition result
The recognition result is fed back to the client

FIGURE 6 | The server-side software workflow.

Table 2 presents that the functional strength training of football players is mainly carried out from the aspects of jumping strength, kicking strength, and competing strength. The training video in the database of this design system is obtained by referring to the current football player training video. After obtaining the real-time video, the computer is used to compare the training actions in the video with the actions in the database to obtain the quantified scores of speed and strength.

Based on the above content, the overall architecture of the network model of action recognition is given here, as shown in **Figure 8**.

As **Figure 8** suggests, a time-segmented CNN is used based on a sparse sampling strategy. The time-segmented CNN is used to segment the entire video and sparsely sample short segments from it as the network input. The temporal characteristics of optical flow images and the spatial characteristics of RGB images are extracted to perform action recognition. The time segmentation CNN first divides a video containing an action into several equal parts, and then randomly extracts a short sequence from it, so that the short segment generated by its sampling can effectively express the motion information in the entire video. For each sampled segment, feature extraction is performed through a dual-stream CNN. The temporal stream network captures the temporal structure information of the video and the spatial appearance information of the image captured by the spatial stream network, and it generates a corresponding dual-stream network prediction for each short segment. Finally, an aggregation function is used to fuse the characteristics of time flow and spatial flow network as the recognition result of the whole video. This method can effectively extract the long-term information of the entire video, which is more accurate and effective than the recognition method by densely sampling the entire video segment and does not increase the calculation cost. In the learning process, the loss value of the entire video prediction is optimized through the iterative update parameter calculation to realize the end-to-end network training process.

The application of the research system to the speed and strength of football players will also be reflected by a scoring system. The scoring system is based on video recognition. After comparing and analyzing the standard data in the video recognition system, the scoring system will score the athletes' movements according to the differences in their movements. Since the improvement of the speed and strength of football players will be reflected in the performance during the game, the scoring system mainly deducts in accordance with the foul action of the players. The system will give the detailed foul or unqualified actions of the football player for reference and promotion.

Under the premise of using the action recognition system to identify the result of foul action, Harris3D operator is used to establishing the potential function of the foul action sequence, which provides support for foul action feature extraction. By Harris3D operator, the foul action is mined; the spatiotemporal interest points of each different foul action are extracted in the image; the features of gradient histogram and streamer direction histogram are obtained, which are divided into 72-dimensional and 90-dimensional images, respectively. Fusion gradient histogram and the streamer direction obtain a 162-dimensional feature vector based on the histogram, which constitutes the underlying feature of the foul action and selects the key skeleton points of the football player. According to the characteristics of human body structure, the football player's body is divided into seven local reference points: shoulders, left and right arms, left and right legs, and left and right feet. The

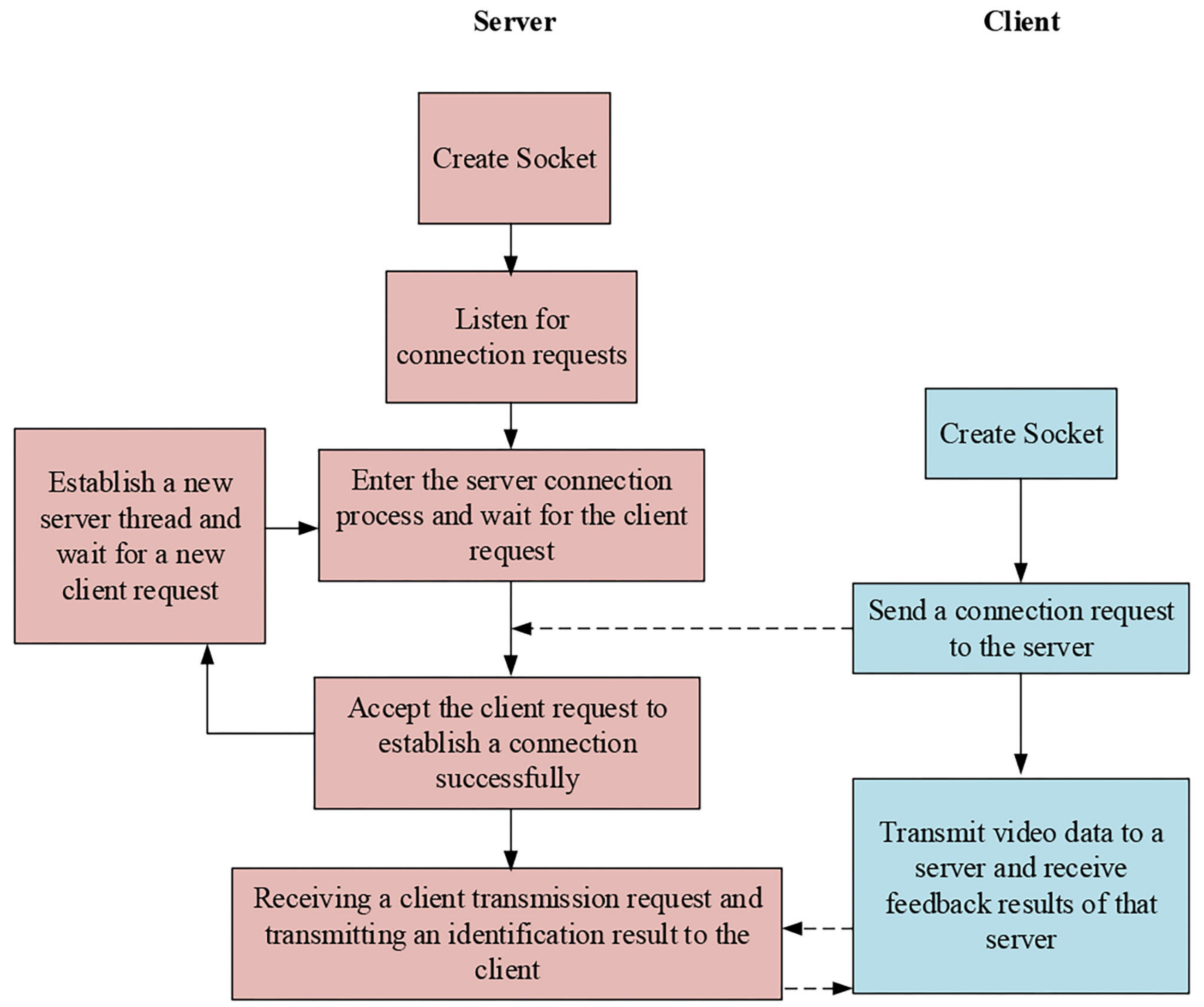

Server
Client
Create Socket
Listen for connection requests
Establish a new server thread and wait for a new client request
Enter the server connection process and wait for the client request
Accept the client request to establish a connection successfully
Receiving a client transmission request and transmitting an identification result to the client
Create Socket
Send a connection request to the server
Transmit video data to a server and receive feedback results of that server

FIGURE 7 | The overall communication connection of the functional strength training system.

shortest Euclidean distance between each point of interest in time and space of the foul action and the local reference center point is shown in Equation (20).

$$n = \frac{\text{argmin}\sqrt{\left(a_i - x_j\right)^2 + \left(b_i - y_i\right)^2}}{\left(x_{\dot{n}}, y_n\right)} \tag{20}$$

In Equation (20), n represents the local reference center human body range mark, which is called the spatiotemporal interest point range. It is divided into 7 regions according to the local reference point; *(xj, yj)* represents each spatiotemporal interest point.

The foul action set time and space interest points are divided into three levels, which are the characteristics of all foul actions of football players; the corresponding foul action features of the shoulder, left and right arms, and left and right legs; the foul action characteristics of the shoulder, left and right arms, left and right legs, and left and right feet. Using K-means to cluster the bottom foul action features, the number of cluster centers is K, and $K \times 162$-dimensional foul action images are obtained. At the same time, three levels of foul action images in seven areas are generated, and the T frame of the foul action in each area is defined as a foul action spatiotemporal action module, which represents a certain part of the foul action feature of a football player. The number of overlapping frames of two foul action image units is $T/2$, and all action units in each foul action image are organically synthesized to obtain a spatiotemporal foul action unit sequence with a length. When comparing the sequence data extracted from standard actions and actions of football players, since everyone's action sequence time is not necessarily the same, the direct use of conventional distance comparison methods, such as Euclidean distance will result in poor scoring accuracy,

TABLE 2 | Actions of functional training database.

Functional strength training method	Specific requirements
Jumping power	In groups of three, tie sandbags on your legs to fight for the top ball
Kicking strength	After finishing the forward roll, jumping the low hurdle and drilling the high hurdle, sprint and shoot
Power in competition	Two football players cooperated, one dribbled the ball quickly, and the other made a tackle. After the tackle was completed, he quickly got up and chased the ball
Starting force in running	In groups of 10 people, after hearing the whistle, they began to sprint 100 meters
Leg strength	In groups of two, one person throws a solid ball to the other party, and the other party passes it back with his feet after receiving the ball

so this method of sequence comparison is adopted to match the dynamic time warping (DTW) algorithm.

Simulation Experiment of Intelligent System of Functional Strength Training

It can be seen from **Figure 8** that after the video segment is input to the algorithm, the tricolor mode image and the optical flow image are first extracted (Fan et al., 2020), and then combined with the commonality of the segment to classify the video action, and finally the video recognition result is obtained.

The feature fusion method is convolution fusion, and the fusion is achieved through the combination of two CNNs after feature extraction. The dual-stream fusion adopts pooling and three-dimensional convolution.

Shen et al. (2019) proposed a method for mining the text of two-layer concept link analysis. Based on this, a data set is selected. The data set contains 4,800 videos with a resolution of 320× 240, and the duration of each video action is 3–9 s, 25 frames per second. The content of the videos are presented in **Table 2**. First, the grid training of the gradient descent algorithm is performed, the parameters are set to a momentum of 1.0, and the batch size is 256 (Wang et al., 2019). The parameters of the dual-stream network set as the initial weight of time flow is 0.01, and the space flow is 1. Three methods are used for experimental verification, namely, time flow, and space flow network zeroing training, the CNN of only pretraining the space flow, and initializing the time flow network using the tricolor mode to compare the recognition effect of the dual-stream CNN. On this basis, the commonly used quantity set is introduced, and the recognition effect between the algorithm designed herein and other common algorithms under the two kinds of quantity sets is compared.

Finally, the practicability and reliability of the entire system are verified. About 20 football players are selected to collect and verify 5 training actions. Finally, the best result of each recognized action is selected. The recognition accuracy greater than 90% is considered an excellent one. If the recognition accuracy is between 85 and 90%, it is good (Gong et al., 2019).

Due to the addition of a scoring system in the experiment, the performance of football players will be reflected in scores. The scores under the traditional training mode are counted by the manual score of the football team coach, and the training system with AI is scored by the computer system. The training improvement is reflected by the football action score improvement rate before and after the use of the two training modes.

The operating system of the experimental environment adopts the operating system (CentOS) Linux release 7.3.1611, the central processing unit (CPU) adopts Intel Xeon E5-2680 v2. It has the memory capacity of 128 Gigabytes (GB), double data rate (DDR) of 2 2,400/3,200MHz, and hard disk capacity of 512 GB+4 Terabyte (TB). The graphics processing unit (GPU) used is GeForce Giga-Texel (GTX) Titan V.

The Algorithm Design Effect Is Compared With Traditional Training

Since the purpose is to compare with traditional strength training to improve the speed and strength of football players, a comparative experiment is carried out here. About 20 athletes are selected and divided into two groups. They are given a traditional functional strength training for a week, and based on that, an algorithm has been designed herein. Finally, the speed and strength results of the two groups before and after training are tested. The intelligently trained athletes are scored by statistics, and the ability of athletes under traditional training are scored by coaches. Five kinds of action are set up to improve the speed and strength of athletes.

The learning of network parameters is realized by a small batch of gradient descent algorithms, with batch size set to 256 and momentum set to 0.9. Gradient descent in small batches can be accelerated by the calculation of matrices and vectors, and the variance of updated parameters are reduced to obtain more stable convergence. Using a batch each time can reduce the number of iterations of convergence, and at the same time make the result of convergence closer to the effect of gradient descent. For the traditional descendent gradient algorithm, if the functional plane of the actual objective is a partially concave surface, then a negative gradient will make it point to a steeper position. This situation near the local optimal value of the objective function will slow down the convergence speed. At this time, it is necessary to give the gradient a momentum, so that it can jump out of the local optimum and continue to optimize in the direction of the gradient descent, so that the network model can more easily converge to the global optimum. For the time segmentation dual-stream network used in this system, the initial weight of the spatial stream convolutional neural network is set to 1, and the initial weight of the time stream convolutional network is set to 1.5. The learning rate for the network training is set smaller: the initial value of the spatial flow of CNN is set to 0.01, and is adjusted to one tenth for every 2,000 iterations; the initial value of the time flow of CNN is set to 0.005, and is adjusted to one-tenth after 12,000 and 18,000 iterations of

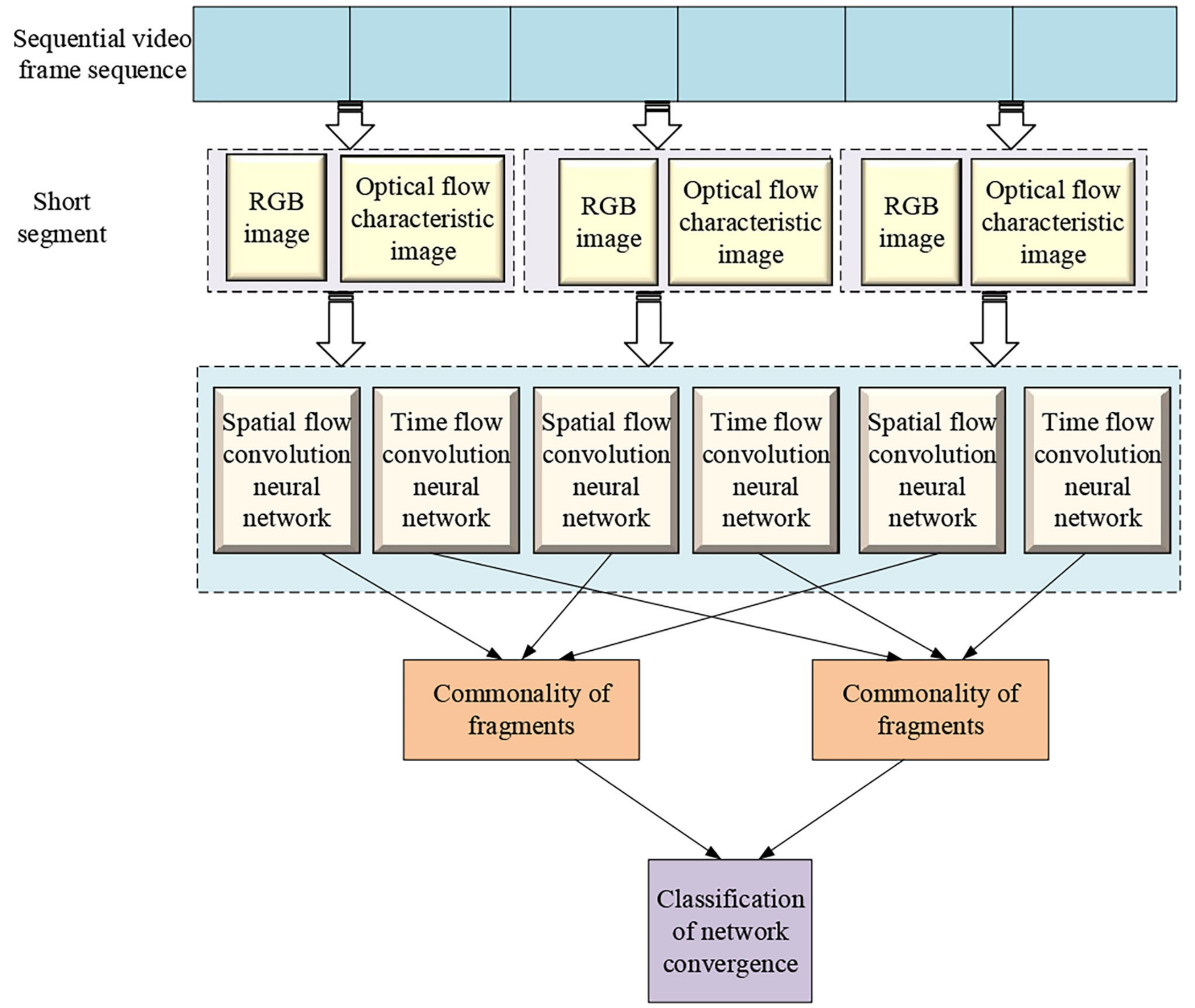

Sequential video frame sequence
Short segment
RGB image
Optical flow characteristic image
RGB image
Optical flow characteristic image
RGB image
Optical flow characteristic image
Spatial flow convolution neural network
Time flow convolution neural network
Spatial flow convolution neural network
Time flow convolution neural network
Spatial flow convolution neural network
Time flow convolution neural network
Commonality of fragments
Commonality of fragments
Classification of network convergence

FIGURE 8 | The overall architecture of the network model of action recognition.

network parameters. In addition, the total time-consuming data training are as follows: the spatial streaming network requires about 2 h; the time streaming network requires about 11 h.

There may be a risk of overfitting when deep CNN is used for network training. In order to further alleviate this problem, three network training strategies are used here to compare the ability to alleviate the risk of overfitting. The first method is to train the spatial stream and time stream network directly from scratch by using the Gaussian distribution to initialize the CNN parameters, without any other means of pretraining processing; the second method is to perform the pretraining processing for convolving the spatial stream CNN. Since the spatial stream convolutional network only uses RGB images as the network input data, the convolutional network can be pretrained through the Image Net image database, and the pre-rained network parameters are used as the initial parameters of spatial stream network; the third method is the pretraining processing method of the cross-input model that initializes the time flow network with the RGB model, while still initializing the Image Net data set as the pretraining input data of the spatial flow network. First, the image pixels are deep CNN discretized at the interval of [0, 255] through linear changes of optical flow characteristics, so that the range of the optical flow image is the same as the RGB; then the mapped image is input to the spatial flow convolutional network for training, and then the preprocessing is performed. The weights of the first convolutional layer of the trained spatial stream convolutional network model are averaged, and the average value is copied according to the number of input channels of the time stream convolutional network. The time-splitting dual-stream network divides the input video sequence into three segments for processing. Therefore, the average value is used as the initial weight of the three channels of the time flow network for training.

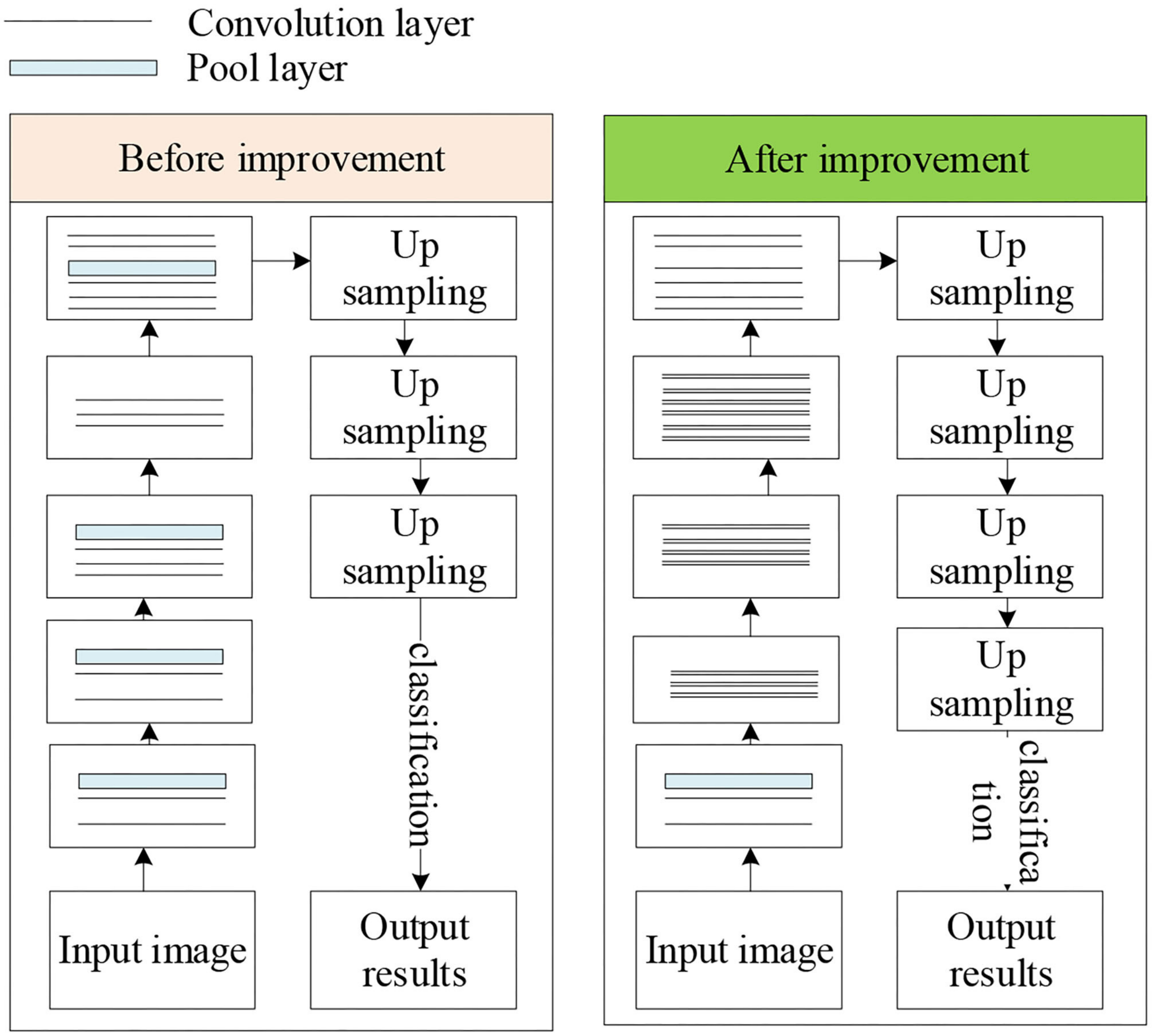


FIGURE 9 | The model before and after improvement.

Field Line Detection for Football Robots

Based on the previous content, combined with CNN for yield line detection, but the extraction ability of traditional FCN model is weak, so it needs to be improved. By increasing the number of convolutional layers, introducing residual structure, constructing residual blocks, and enhancing the model feature extraction ability, the accuracy of field line detection is improved finally. After the residual structure is introduced, the input feature of each convolutional layer is the sum of the input and output features of the previous convolutional layer, so each convolutional layer can extract more features, which is more conducive to improving the detection accuracy. Residual block refers to a residual structure containing multiple consecutive convolutional layers, that is, a direct correlation channel is introduced between the input of the first convolutional layer and the output of the last convolutional layer, so that the input features of the residual block are directly added to the output features as input features for the next convolutional layer or residual block. The model before and after improvement is shown in **Figure 9**.

In **Figure 9**, compared with the FCN model, the convolutional layers of the improved model are increased from 15 to 34, and the convolutional layers are constructed into 4 residual blocks, which improves the ability of feature extraction. In the residual block, a convolutional layer with a stride of 2 is used to replace the pooling layer for downsampling. In the process of upsampling,

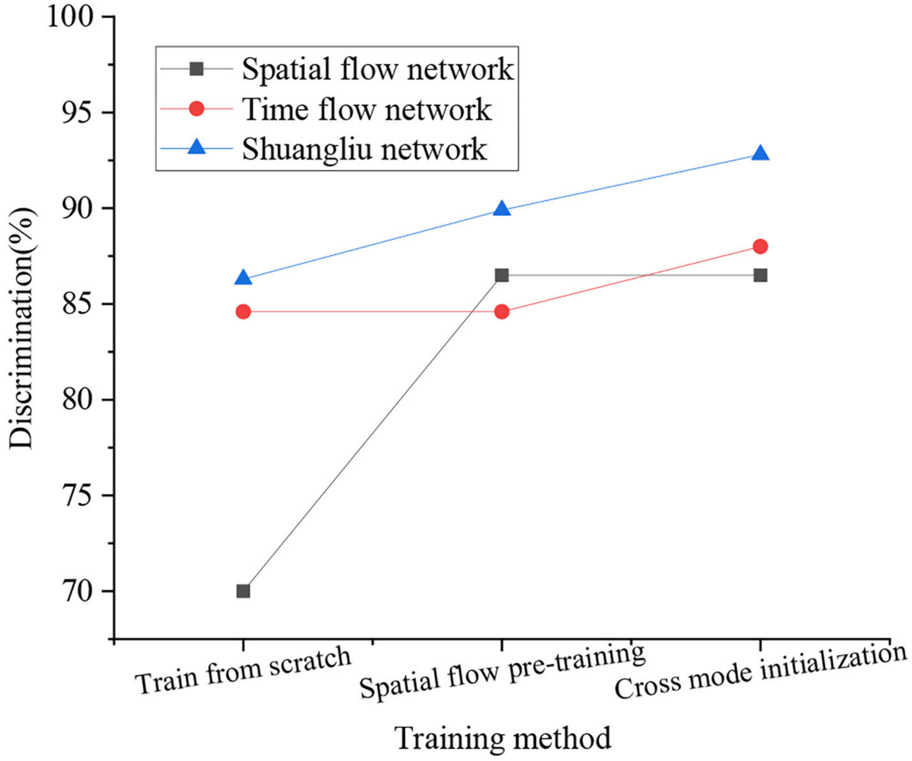


FIGURE 10 | Comparison of recognition effects of CNN under different training methods.

a cross-layer connection structure is used to fuse the output feature maps of all residual blocks to make the final detection result more refined, and the field line detection can be performed more accurately.

The evaluation index of the field line adopts the average pixel precision *P* and the mean intersection over Union (MIoU) *Q*, and the specific calculation is shown in Equations (21) and (22).

$$P = \frac{1}{k+1}\sum_{i=0}^{k}\frac{\sum_{i=0}^{k} P_{ii}}{\sum_{j=0}^{k} P_{ij}} \tag{21}$$

$$Q = \frac{1}{k+1}\sum_{i=0}^{k}\frac{P_{ii}}{\sum_{j=0}^{k} P_{ij} + \sum_{j=0}^{k} P_{ji} - P_{ii}} \tag{22}$$

In Equations (21) and (22), $k+1$ is the number of categories (including 1 background class); P_{ii} indicates the point that is correctly predicted; P_{ij} means that the pixel marked as class i is predicted to be the pixel of class j; P_{ji} denotes that the pixel marked as class j is predicted to be pixels of class i.

In the experimental field, three lighting conditions are set, namely, lighting, sufficient natural light, and insufficient natural light. The average pixel precision and MIoU of field line detection between the color segmentation model and the improved model are compared under different lighting environments, and the average pixel precision and MIoU of the FCN model and the improved model are compared. Finally, the continuously running time of three models is compared, and the model uses 100 images of the research area.

ANALYSIS OF INTELLIGENT DETECTION RESULTS OF FUNCTIONAL TRAINING INTELLIGENT SYSTEM AND FOOTBALL ROBOT

Analysis of Functional Training Results

Based on the previous content, a comparison of the recognition effect of CNN under different network training methods is obtained, as shown in **Figure 10**.

Figure 10 displays that under different training methods, the recognition effect of CNN is somewhat diverse. Among them, the dual-stream network has the best effect, with a recognition rate of 92.8%. On the whole, the recognition rate of the time-stream network is lower than that of the Shuangliu network, and its highest rate is 88%; the recognition rate of the spatial stream network is the lowest among the three comparisons, which is 86.5%. The overall recognition rate of the network under the three training methods is sorted from large to small, cross mode initialization, spatial flow pretraining, and training from the scratch. Analysis of the reason for this phenomenon is that the feature recognition dimension of the dual-stream network is higher, and the coverage is more comprehensive, and the single

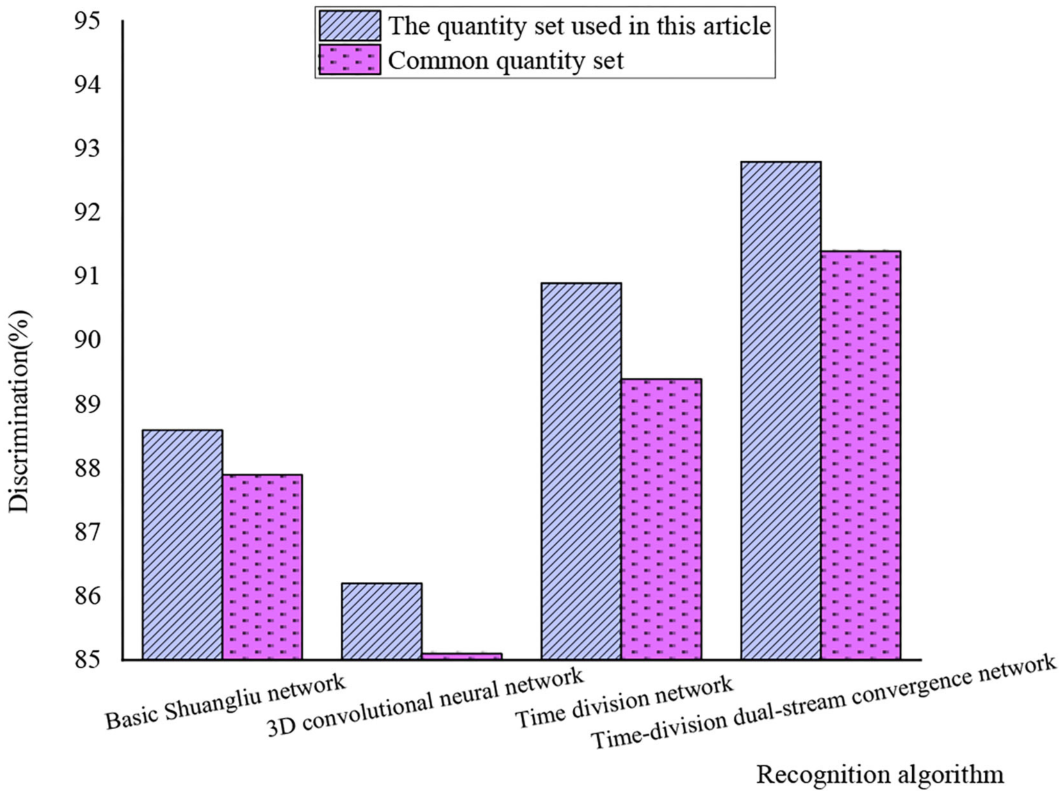


FIGURE 11 | Comparison of recognition effects among different algorithms.

network recognition often has only one dimension. The method of training from scratch is prone to overfitting when the data set is small, so the recognition rate is the lowest among the three training methods, and the spatial flow pretraining method can reduce the risk of overfitting to a certain extent, and the training method of crossover mode could reduce risks to the largest extent. It can be seen that the design algorithm is more feasible.

After the introduction of common quantity sets, the processing results of common algorithms for the two kinds of quantity sets are shown in **Figure 11**.

Figure 11 shows that in the processing of the data set, the four algorithms have different effects on the video set selected from herein and the common video set. The processing power of the data set herein is stronger than that of the common video set. The recognition rate of the time-divided dual-stream fusion network is the highest, reaching 92.8%. The recognition rate of the time segmentation network is second only to the network designed herein, which is 90.9%. The recognition rate of the basic dual-stream network is 88.6%, and the recognition rate of the 3D convolutional neural network is the lowest, just 86.2%. When analyzing the reasons, it can be seen that the commonly used data set has a large amount of data, so the processing effect is poor. The time-divided dual-stream fusion network can sample efficiently, and other algorithms are prone to excessive information due to a long time for sampling. The algorithm designed herein is highly reliable.

On the basis of **Table 2**, the recognition accuracy of the five actions of this design system is compared, and the results are shown in **Figure 12**.

Figure 12 demonstrates that under the intelligent training system, the recognition accuracy of different actions is high, and the highest is that of jumping actions, reaching 97.6%, and the recognition accuracy of kicking actions is 94.5%. Grabbing action is 92.5%, starting action is slightly lower, reaching 89.8%, and that of passing action is 91.3%. It can be seen that the recognition accuracy is beyond the good level. Except for the recognition accuracy of the starting movement, others are all at an excellent level. The analysis shows that when the 10-person group set up starts, the overlap rate of the athletes in the video is relatively high, so the system identification is more difficult. The intelligent training system can meet the design requirements.

Finally, the results of the functional strength training of the athletes under the AI system are counted, and it is compared with the scores under the traditional training methods. The results are shown in **Figure 13**.

Figure 13 shows that the upgrade rate of speed and strength under intelligent training is higher than that under traditional training. The maximum upgrade rate of intelligent training is 25.7%, which is the improvement of passing ability, and the

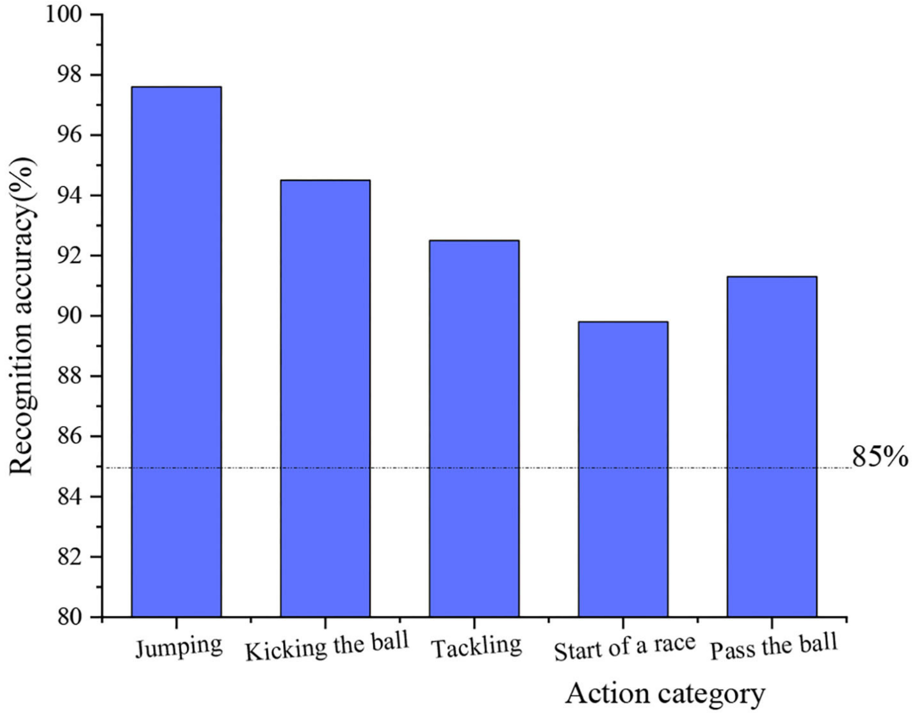


FIGURE 12 | The recognition accuracy of different actions.

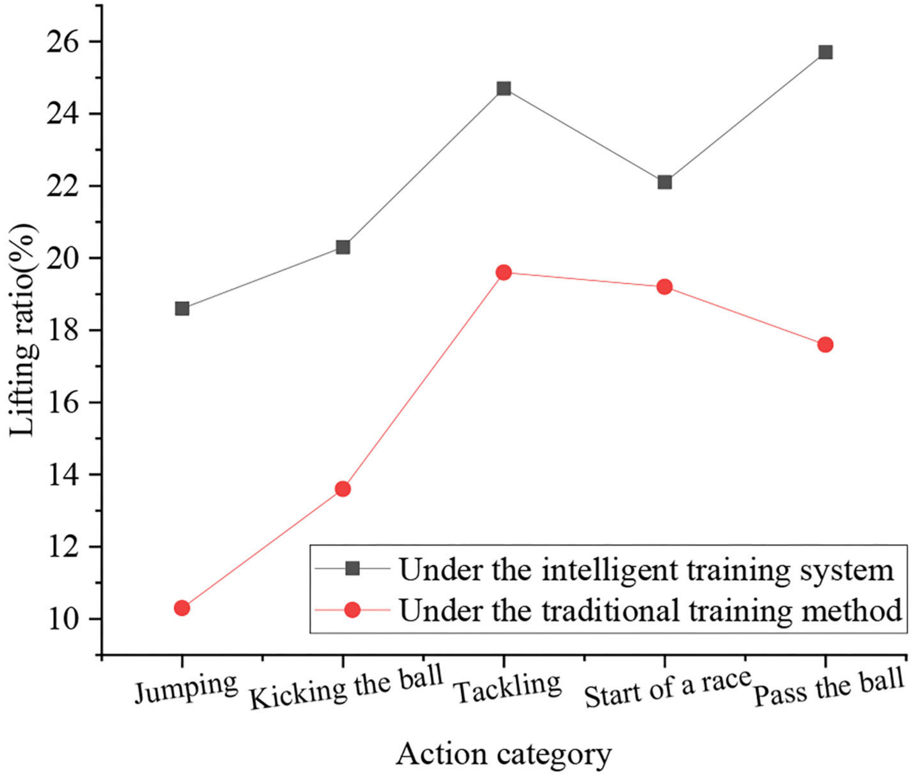


FIGURE 13 | Comparison of training results.

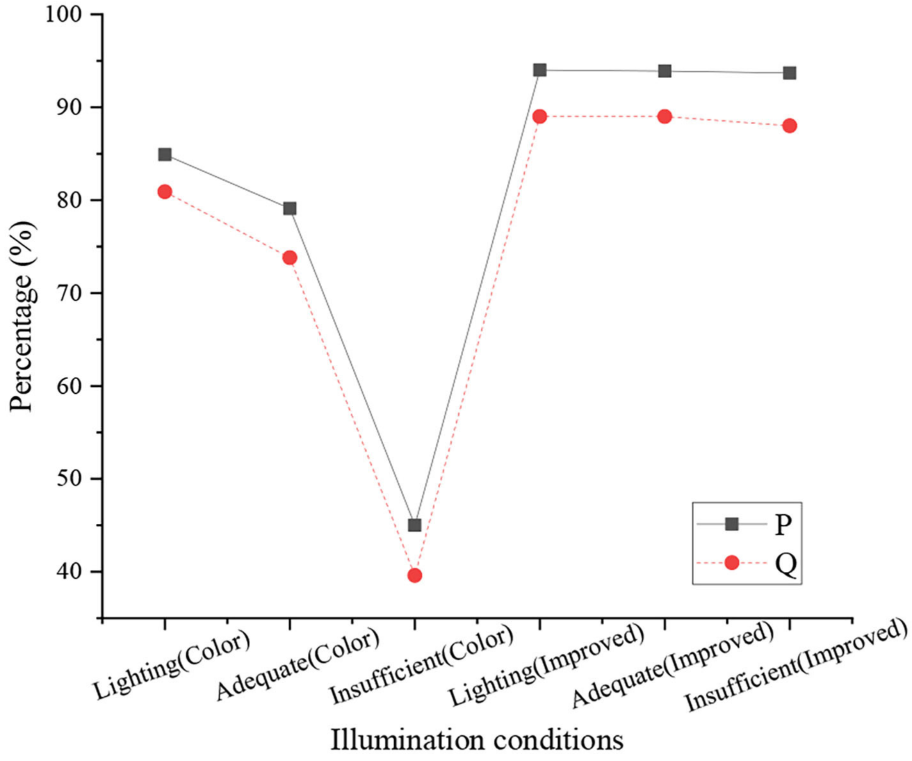


FIGURE 14 | Performance comparison of color segmentation model and improved model.

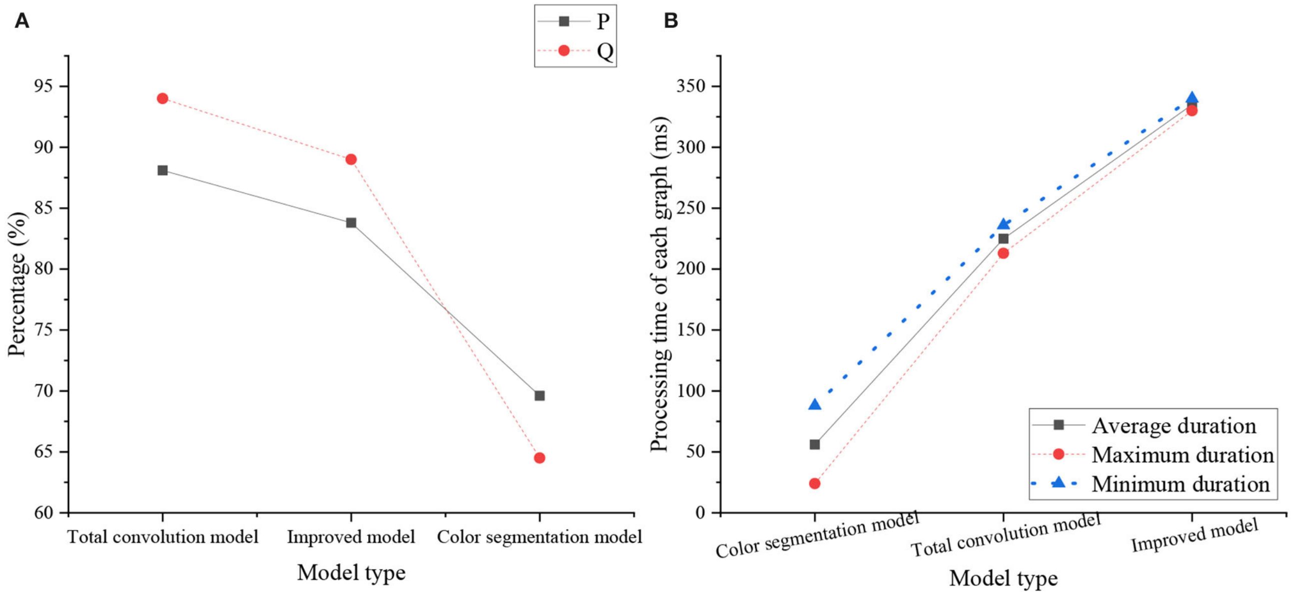


FIGURE 15 | Performance comparison of the models [**(A)** The comparison between the FCN model and the improved model, **(B)** The comparison of running time of the three models].

minimum is 18.6%. For the improvement of jumping ability, the highest upgrade rate of traditional training is the ability to compete for the ball, which is 19.6%, and the lowest is the improvement of jumping ability, which is 10.3%. Since intelligent training strengthens the autonomy of football players, the upgrade rate increases rapidly. The experiment is set for 1 week, but it takes a long time for the training of jumping ability to be effective. Therefore, the upgrade rate of jumping ability in the two training groups is low. In summary, the intelligent training system has a strong applicability.

Analysis of Field Detection Results of Football Robots

The average pixel precious P and MIoU Q of the color segmentation model and the improved model under different lighting are shown in **Figure 14**.

Figure 14 indicates that the P of the improved model exceeds 90%, especially under insufficient natural light, the P of the improved model is 93.7%, while the average pixel accuracy of the color segmentation model under insufficient natural light is 45%. The Q of the improved model is about 89% in the three lighting modes, and the Q of the color segmentation model is larger in different lighting modes. It refers that the improved model algorithm has better stability.

The performance comparison of the color segmentation model, the FCN model, the improved model, and the comparison of the running time of the three models are shown in **Figure 15**.

Figure 15A shows that the P and Q of the improved model are higher than those of the FCN model. Due to the weak feature extraction ability of the FCN model, it is easy to be interfered by other nonsite line objects in the field line recognition, and the detection results are not accurate enough. In contrast, the improved model can complete the yield line detection more accurately due to its strong feature extraction ability. **Figure 15B** shows that the improved model takes the longest time due to the larger amount of computation, but the detection effect is good, the accuracy is high, and the detection speed basically meets the requirements of the competition.

CONCLUSION

To improve the effectiveness of functional strength training for football players and the field detection stability of football robots, first the DL and human action recognition are combined under the background of AI to study intelligent training methods suitable for football players. Second, a functional strength training system is established by combining time flow and spatial flow networks. Based on CNN, an improved yield detection method is designed. Finally, the application ability of the system is evaluated. The results manifest that the intelligent training system is feasible, and the designed football robot has high field detection accuracy. The research provides a reference for the intelligent training of football players and the development of football robots. The training system will be mainly used for football training in schools and clubs to realize the autonomy and intelligence of football training and provide a certain reference for improving the overall level of football players in China. Field detection of football robots can be used in football robot games. But there are some drawbacks. There are few statistics about different actions in football in the dataset, and the main limitations are not to study the positioning of football robots. In the follow-up research, these two parts will be deeply analyzed to make the research content more comprehensive. The practical application value of research is that after further improvement, it will be expected to promote the level of football training in colleges and universities, and improve the level of football capabilities of the young players.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

FUNDING

This work was supported by a grant from the Foundation of Social Science and Humanity, China Ministry of Education (Grant No. 19YJC890050), the Natural Science Foundation of Zhejiang Province of China (Grant No. LY18C110002), the Major Planning Program of Hangzhou Philosophy and Social Science Key Research Base (2021; Grant No. 2021JD19), and by the Foundation of Educational Commission of Zhejiang Province (Grant No. Y202147781).

REFERENCES

Abdelbaky, A., and Aly, S. (2021). Human action recognition using three orthogonal planes with unsupervised deep convolutional neural network. *Multimed. Tools Appl*. 80, 20019–20043. doi: 10.1007/s11042-021-10636-2

Antonioni, E., Suriani, V., Riccio, F., and Nardi, D. (2021). Game strategies for physical robot soccer players: a survey. *IEEE Trans. Games* 13, 342–357. doi: 10.1109/TG.2021.3075065

Aslam, S., Herodotou, H., Mohsin, S. M., Javaid, N., Ashraf, N., and Aslam, S. (2021). A survey on deep learning methods for power load and renewable energy forecasting in smart microgrids. *Renew. Sust. Energ. Rev*. 144, 110992. doi: 10.1016/j.rser.2021.110992

Bueno, C. A., de Araujo Ribeiro-Alvares, J. B., dos Santos Oliveira, G., Grazioli, R., Veeck, F., and Pinto, R. S. (2021). Post-match recovery of eccentric knee flexor strength in male professional football players. *Phys. Ther. Sport*. 47, 140–146. doi: 10.1016/j.ptsp.2020.11.032

Chen, D. (2020). Fuzzy obstacle avoidance optimization of soccer robot based on an improved genetic algorithm. *J. Ambient Intell. Human. Comput*. 11, 6187–6198. doi: 10.1007/s12652-019-01636-0

Chen, D., Wawrzynski, P., and Lv, V. (2021). Cyber security in smart cities: a review of deep learning-based applications and case studies. *Sustain. Cities Soc*. 66, 102655. doi: 10.1016/j.scs.2020.102655

Chen, X., and Gao, P. (2020). Path planning and control of soccer robot based on genetic algorithm. *J. Ambient Intell. Human. Comput*. 11, 6177–6186. doi: 10.1007/s12652-019-01635-1

Colombini, E. L., Drews-Jr, P. L., and Gonçalves, L. M. (2022). Editorial notes for topical collection on robotica 2019. *J. Intell. Robot. Syst*. 104, 1–3. doi: 10.1007/s10846-021-01557-1

Cournoyer, J., Karton, C., Koncan, D., Gilchrist, M. D., Cantu, R. C., and Hoshizaki, T. B. (2021). Brain trauma exposure for American tackle football players 5 to 9 and 9 to 14 years of age. *J. Biomech*. 127, 110689. doi: 10.1016/j.jbiomech.2021.110689

Elharrouss, O., Almaadeed, N., Al-Maadeed, S., Bouridane, A., and Beghdadi, A. (2020). A combined multiple action recognition and summarization for surveillance video sequences. *Appl. Intell*. 51, 690–712. doi: 10.1007/s10489-020-01823-z

Fan, L., Zhang, T., and Du, W. (2020). Optical-flow-based framework to boost video object detection performance with object enhancement. *Expert. Syst. Appl*. 170, 114544. doi: 10.1016/j.eswa.2020.114544

Gong, J., Li, R., Yao, H., Kang, X., and Li, S. (2019). Recognizing human daily activity using social media sensors and deep learning. *Int. J. Env. Res. Public Health* 16, 3955. doi: 10.3390/ijerph16203955

Guan, S., and Wang, X. (2021). Optimization analysis of football match prediction model based on neural network. *Neural Comput. Appl*. 41, 1–17. doi: 10.1007/s00521-021-05930-x

Haldorai, A., and Ramu, A. (2020). Canonical correlation analysis based hyper basis feedforward neural network classification for urban sustainability. *Neural Process. Lett*. 53, 2385–2401. doi: 10.1007/s11063-020-10327-3

Hong, C., Jeong, I., Vecchietti, L. F., and Kim, J. (2021). AI world cup: robot-soccer-based competitions. *IEEE Trans. Games*. 13, 330–341. doi: 10.1109/TG.2021.3065410

Houtman, W., Martinez, C. L., Wang, S., Ketels, A., and Bruyninckx, H. P., van de Molengraft, M. J. G. (2021). Dynamic control of steerable wheeled mobile platforms applied to an eight-wheeled RoboCup Middle Size League soccer robot. *Mechatronics* 80, 102693. doi: 10.1016/j.mechatronics.2021.102693

Jeong, J., Yang, J., and Baltes, J. (2022). Robot magic show as testbed for humanoid robot interaction. *Entertain. Comput*. 40, 100456. doi: 10.1016/j.entcom.2021.100456

Khan, M. A., Zhang, Y. D., Khan, S. A., Attique, M., Rehman, A., and Seo, S. (2020). A resource conscious human action recognition framework using 26-layered deep convolutional neural network. *Multimed. Tools Appl*. 80, 8. 35827–35849. doi: 10.1007/s11042-020-09408-1

Khan, S., Khan, M. A., Alhaisoni, M., Tariq, U., Yong, H. S., Armghan, A., et al. (2021). Human action recognition: a paradigm of best deep learning features selection and serial based extended fusion. *Sensors* 21, 37941. doi: 10.3390/s21237941

Kuwana, R., Ariji, Y., Fukuda, M., Kise, Y., Nozawa, M., Kuwada, C., et al. (2020). Performance of deep learning object detection technology in the detection and diagnosis of maxillary sinus lesions on panoramic radiographs. *Dentomaxillofac. Rad*. 50, 20200171. doi: 10.1259/dmfr.20200171

Lagemann, C., Lagemann, K., Mukherjee, S., and Schröder, W. (2021). Deep recurrent optical flow learning for particle image velocimetry data. *Nat. Mach. Intell*. 3, 641–651. doi: 10.1038/s42256-021-00369-0

Materne, O., Chamari, K., Farooq, A., Weir, A., Hölmich, P., and Bahr, R. (2020). Injury incidence and burden in a youth elite football academy: a four-season prospective study of 551 players aged from under 9 to under 19 years. *Br. J. Sport. Med*. 55, 493–500. doi: 10.1136/bjsports-2020-102859

Newman, J., Lin, J. W., Lee, D. J., and Liu, J. J. (2021). Automatic annotation of American Football Video footage for game strategy analysis. *J Electron. Imag.* 21, 303. doi: 10.2352/ISSN.2470-1173.2021.6.IRIACV-303

Ozcan, T., and Basturk, A. (2020). Performance improvement of pre-trained convolutional neural networks for action recognition. *Comput. J*. 64, 1. 1715–1730. doi: 10.1093/comjnl/bxaa029

Pareek, P., and Thakkar, A. (2020). A survey on video-based human action recognition: recent updates, datasets, challenges, and applications. *Artif. Intell. Rev*. 54, 2259–2322. doi: 10.1007/s10462-020-09904-8

Parim, C., Güneş, M. S., Büyüklü, A. H., and Yildiz, D. (2021). Prediction of match outcomes with multivariate statistical methods for the group stage in the UEFA Champions League. *J. Hum. Kinet*. 79, 197–209. doi: 10.2478/hukin-2021-0072

Park, C., Kim, B., Kim, Y., Eum, Y., Song, H., and Yoon, D. (2022). Carved turn control with gate vision recognition of a humanoid robot for giant slalom skiing on ski slopes. *Sensors* 22, 816. doi: 10.3390/s22030816

Pons, E., Ponce-Bordón, J. C., Díaz-García, J., López del Campo, R., Resta, R., Peirau, X., et al. (2021). A longitudinal exploration of match running performance during a football match in the Spanish La Liga: a four-season study. *Int. J. Env. Res. Public Health* 18, 1133. doi: 10.3390/ijerph18031133

Pratama, A. T. (2021). Augmented reality trasnportasi darat menggunakan FAST Corner Detection dan Lucas Kanade. *JATISI* 8, 1663–1671. doi: 10.35957/jatisi.v8i3.1076

Sarker, H. (2021). Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Comput. Sci*. 2, 1–20. doi: 10.1007/s42979-021-00815-1

Setyawan, N., Mardiyah, N. A., Zulfatman, Z., and Fajar, D. N. (2022). Navigasi Robot Sepak Bola Beroda Menggunakan particle filter localization. *Cyclotron* 5, 116. doi: 10.30651/cl.v5i1.9419

Shen, C., Luong, T., Ho, J., and Djailani, I. (2019). Social media marketing of IT service companies: analysis using a concept-linking mining approach. *Ind. Market. Manag*. 90, 593–604. doi: 10.1016/j.indmarman.2019.11.014

Somu, N., Gauthama Rama, M. R., and Ramamirtham, K. (2021). A deep learning framework for building energy consumption forecast. *Renew. Sust. Energ. Rev*. 137, 110591. doi: 10.1016/j.rser.2020.110591

Stoeve, M., Schuldhaus, D., Gamp, A., and Eskofier, B. M. (2021). From the laboratory to the field: IMU-based shot and pass detection in football training and game scenarios using deep learning. *Sensors* 21, 3071. doi: 10.3390/s21093071

Suwarno, I., Ma'arif, A., Raharja, N. M., and Shomad, M. A. (2022). Using a combination of PID control and Kalman filter to design of IoT-based telepresence self-balancing robots during COVID-19 pandemic. *Emerg. Sci. J.* 4, 241–261. doi: 10.28991/esj-2021-SP1-016

Teixeira, J. E., Forte, P., Ferraz, R., Leal, M., Ribeiro, J., and Silva, A. J. (2021). Monitoring accumulated training and match load in football: a systematic review. *Int. J. Env. Res. Pub. He*. 18, 3906. doi: 10.3390/ijerph18083906

Thakkar, P., and Shah, M. (2021). An assessment of football through the lens of data science. *Ann. Data Sci*. 71, 1–14. doi: 10.1007/s40745-021-00323-2

Ullah, A., Muhammad, K., Hussain, T., and Baik, S. W. (2020). Conflux LSTMs network: a novel approach for multi-view action recognition. *Neurocomputing* 435, 321–329. doi: 10.1016/j.neucom.2019.12.151

Vella, A., Clarke, A. C., Kempton, T., Ryan, S., Holden, J., and Coutts, A. J. (2021). Possession chain factors influence movement demands in elite Australian football match-play. *Sci. Med. Football* 5, 72–78. doi: 10.1080/24733938.2020.1795235

Wang, T. (2022). Exploring intelligent image recognition technology of football robot using omnidirectional vision of internet of things. *J Supercomput*. 15, 1–20. doi: 10.1007/s11227-022-04314-9

Wang, W. G., Shen, J. B., Xie, J. W., Cheng, M. M., Ling, H., and Borji, A. (2019). Revisiting video saliency prediction in the deep learning era. *IEEE T. Pattern Anal*. 43, 220–237. doi: 10.1109/TPAMI.2019.2924417

Watanabe, K., Ma, Y., Kono, H., and Suzuki, H. (2022). A self-localization method using a genetic algorithm considered kidnapped problem. *J. Adv. Comput. Intell. Intell. Inform*. 26, 32–41. doi: 10.20965/jaciii.2022.p0032

Zhang, D., Zhan, J., Tan, L., and Gao, Y., Župan, R. (2020). Comparison of two deep learning methods for ship target recognition with optical remotely sensed data. *Neural Comput. Appl*. 33, 4639–4649. doi: 10.1007/s00521-020-05307-6

Zhao, D., Wu, Y., Wang, C., Shen, C., Tang, J., and Liu, J. (2021). Gray consistency optical flow algorithm based on mask-R-CNN and a spatial filter for velocity calculation. *Appl. Optics* 60, 10600–10609. doi: 10.1364/AO.441233

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

TYPE Original Research
PUBLISHED 28 July 2022
DOI 10.3389/fnbot.2022.934109

Check for updates

OPEN ACCESS

EDITED BY
Jose De Jesus Rubio,
Instituto Politécnico Nacional (IPN),
Mexico

REVIEWED BY
Wellington Pinheiro dos Santos,
Federal University of Pernambuco,
Brazil
Francisco Gonzalez-Salazar,
Mexican Social Security Institute
(IMSS), Mexico

*CORRESPONDENCE
Hiram Calvo
hcalvo@cic.ipn.mx

RECEIVED 02 May 2022
ACCEPTED 24 June 2022
PUBLISHED 28 July 2022

CITATION
Angeles-García Y, Calvo H, Sossa H and Anzueto-Ríos Á (2022) Dynamic balance of a bipedal robot using neural network training with simulated annealing.
Front. Neurorobot. 16:934109.
doi: 10.3389/fnbot.2022.934109

Dynamic balance of a bipedal robot using neural network training with simulated annealing

Yoqsan Angeles-García[1,2], Hiram Calvo[1]*, Humberto Sossa[2] and Álvaro Anzueto-Ríos[3]

[1]Computational Cognitive Sciences Laboratory, Center for Computing Research, Instituto Politécnico Nacional, Mexico City, Mexico, [2]Robotics Laboratory, Center for Computing Research, Instituto Politécnico Nacional, Mexico City, Mexico, [3]Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas, Instituto Politécnico Nacional, Mexico City, Mexico

This work proposes using an evolutionary optimization method known as simulated annealing to train artificial neural networks. These neural networks are used to control posture stabilization of a humanoid robot in a simulation. A total of eight multilayer perceptron neural networks are used. Although the control is used mainly for posture stabilization and not displacement, we propose a posture set to achieve this, including right leg lift in sagittal plane and right leg lift in frontal plane. At the beginning, tests are carried out only considering gravitational force and reaction force between the floor and the humanoid; then tests are carried out with two disturbances: tilted ground and adding a mass to the humanoid. We found that using simulated annealing the robot maintains its stability at all times, decreasing the number of epochs needed to converge, and also, showing flexibility and adaptability to disturbances. The way neural networks learn is analyzed; videos of the movements made, and the model for further experimentation are provided.

KEYWORDS

simulated annealing, bipedal robot, neural network control, neurorobotics, machine learning

1. Introduction

Since ancient times, machines have been created that attempt to replicate the human form (Boden, 2006). With the development of robotics, this search for the development of machines with human characteristics has continued. This search is of vital importance as we aim for robots that can carry out tasks that at the moment are achievable only by human beings. A very useful feature that human morphology possesses is the ability to locomote and this feature is the focus of this work. Research has been done that shows that human morphology is the best option when using legged robots due to its energy efficiency (Kuo A., 2007).

To program the locomotion of a biped, classical control strategies were previously used, making use of complex equations to model the dynamics of the robot, however, with the development of some areas in artificial intelligence, and especially with the rise of artificial neural networks, significant progress has been made in simplifying the locomotion control process of bipedal robots. In Jha et al. (2005), a method combining fuzzy control and genetic algorithms was proposed to control a stair-climbing biped in a simulation. Another work in which fuzzy logic was also used is in Murakami et al. (1995) where a fuzzy controller was used for each leg in a biped. In Miller (1994) and Kim et al. (2005, 2012) the authors use trained neural networks with supervised learning to control the balance of a biped. In Lin et al. (2006) and Wu et al. (2007) unsupervised learning is used to control the biped, however these proposals have the disadvantage of additionally needing a controller to compensate the torque with a PID controller. In other applications, Sun et al. (2021) uses a neural network-based adaptive control approach to stabilize the airgap of the nonlinear maglev vehicle.

In this work, what is sought is to use a posture stability control method for a biped in a simulated environment. This control is based on artificial neural networks and evolutionary optimization, but unlike other works, the use of transfer equations and other classical control methods is ruled out. It is sought that the control system has a simpler implementation allowing the algorithm to be easily understood and reproducible while its performance meets the assigned task. As a contribution to stability control strategies in bipeds, this paper proposes a method in which only artificial neural networks trained by evolutionary optimization methods are used to achieve stabilization in the posture of a simulated biped in a computer-generated environment (MATLAB's Simulink). The novelty of this work is that it uses an optimization method in which prior knowledge is not required, since its operation is similar to reinforcement learning, being an important difference that learning is done in real time, so that fewer iterations are required.

This work focuses on the goal to achieve a stable standing position in each moment. A sequence of these positions creates movement. With the knowledge of this positions, it is possible to produce a faster motion using reinforcement learning (Gil et al., 2019), among other techniques; not only for walking on a straight line but also to make another movements including movement with disturbances. The prediction of movement can be applied changing the architecture to a recurrent neural network, this could also help to achieve a faster motion.

The benefits of having this technology are that there is no need of large datasets to train the neural network, it can be trained online while the robot is moving. It can also be implemented in other morphologies. So it can be applied in robots with different shapes and the performance should be the same. Nevertheless, more research is necessary.

The rest of the document is organized as follows: In Section 2, the state of the art is presented, works related to the area of bipeds are described. The works are presented starting with classical approaches, then works with more modern control methods are shown. Section 3 describes the methodology followed for the development of this work. Section 4 details the experiments carried out and the results obtained. Finally, in Section 5 we draw our conclusions, and some proposals are made to continue developing the project proposed here.

2. Related work

Bipedal locomotion is an area that has been developed for several decades and for locomotion, balance control is an indispensable requirement. Bipedal robots are high dimensional systems, the dimension varies depending on the configuration of the robot but even those systems that only have the legs and waist, have many dimensions. A common way to solve the dimensionality problem is to represent the biped as a low-dimensional inverted pendulum (Kajita et al., 2001; Kuo A. D., 2007; Pratt and Drakunov, 2007). The robot is thus controlled so that its center of mass follows a specific target. However, this approach has some challenges: finding the stable solution in the complete model, deciding how to associate the states of the pendulum with the complete high-dimensional system, even realizing the correct model of the inverted pendulum is not an easy task (Da and Grizzle, 2019). These are the reasons why a different non-linear control option is proposed. The ability of neural networks to approximate functions makes them a valuable tool for the design of nonlinear controllers (Plumer, 1996; Zhang et al., 2003; Geng et al., 2006).

Movement trajectories can be generated depending on the application. For example, that a bipedal robot moves in a straight line. However, these trajectories usually do not consider disturbances that may exist in the environment. Therefore, the adaptability of the bipedal control system to environmental disturbances is an important aspect to take into account. To solve this problem, classical control options have been considered. In Cho and Kim (2018), the authors create a dynamic model of a biped based on an inverted pendulum with a spring and damper. Later they calculated the transfer equations to make a closed control loop.

The next sections present a brief summary of some works related to the control of bipedal robots. These works focus on stabilization of bipedal posture and not on the gait cycle.

2.1. Dynamic posture stabilization of a biped robot SUBO-1 on slope-changing grounds

This work (Cho and Kim, 2018) describes a dynamic posture stabilization model for a bipedal robot on a tilted floor. The work uses the Zero Moment Point method to stabilize posture. It also makes use of an altering spotter that was designed to counteract alteration due to ground tilt. This work focuses on the control of *blind walk* on globally sloped terrain. Without using a vision system or any mechanism in the foot. It is assumed that the slope of the terrain changes continuously and that the floor is flat and without local slopes. Combining Zero Moment Point control and disturbance observer, a stabilization strategy using force sensors in the feet and an inertial sensor in the pelvis is proposed.

The control frequency used in that work is 200 Hz, for which the central controller receives the sensory data and sends the positions in a period of 5 ms. The structure of the walking algorithm consists of a walking pattern generation and a posture stabilizer with feedback. The pattern generator is a feedforward control and the posture stabilizer is a feedback control. Finally, the desired angle is calculated by solving the inverse kinematics for the two control inputs.

The robot was modeled as an inverted pendulum with a flexible joint consisting of a spring and a damper. Transfer equations were calculated to apply the control loop. At the end of the work it was concluded that adding an observer was very useful to deal with sloping floors. In addition to stabilization with ZMP control. It is hoped that in the future a control system including vision can be built for use in rough terrain.

2.2. Nearly optimal neural network stabilization of bipedal standing using genetic algorithm

In this work (Ghorbani et al., 2007), the stability control for a biped was studied. The model of the biped was simplified as an inverted pendulum with one joint. The controller consists of a general regression neural network with feedback that stabilizes the biped in a vertical position, and a PID control with feedback that maintains the pendulum in a vertical position. The neural network is also designed to minimize energy cost.

For that work, a General Regression Neural Network (GRNN) is used, which has the advantage that it is not necessary to define the number of hidden layers or the number of neurons per layer. When generating the trajectory, it is assumed that the biped moves in a sagittal plane and is simplified as an inverted pendulum with a rigid joint that is the foot.

As a first step, a closed-loop control with a GRNN was designed to move the pendulum in a region around the vertical position while minimizing the energy related to the cost function (torque). To increase stabilization, a PID control tuned by trial and error is activated to keep the biped upright. Three restrictions were considered: there is no lifting of the foot, there is no sliding, and the center of pressure is always maintained in the region of contact between the ground and the foot.

It has been reported that when standing subjects are exposed to small disturbances, they typically respond by moving in the sagittal plane and tend to keep their knees, neck, and hips straight, moving primarily at the ankle (Kuo, 1995). In conclusion, in the work it was possible to minimize energy consumption by comparing the proposed system with a previously proposed one (Yang and Wu, 2006). By comparison, the new system managed to cut energy consumption in half

We have discussed several works that use modern control methods such as artificial neural networks or bioinspired algorithms; however, they still rely on classical techniques to ensure correct control of the biped. In Ghorbani et al. (2007), the authors use a general regression neural network with feedback for vertical stabilization of a biped, and a PID control with feedback to maintain the pendulum in a vertical position. Again, modeling the biped as an inverted pendulum, and using Lyapunov exponents to analyze the stability control. In the next section, we present our proposal, where we seek to use only neural networks for the purpose of moving a biped humanoid without losing its balance.

3. Methodology

Our proposed method that allows a biped humanoid to move without losing its balance consists of six stages.

1. Calculation of direct kinematics equations of the biped
2. Encoding of training algorithm
3. Modeling of the biped
4. Sense data filtering
5. Neural network training
6. Testing.

The next sections with provide more details on our proposed model.

3.1. Kinematic model of the biped humanoid

For this work, a kinematic model of the biped was conceived in order to calculate the center of mass and thus perform a stability analysis. Link lengths were modeled on average adult male limb measurements. The model has 20 degrees of freedom, however some of these will remain rigid during the simulation. Of the 20 degrees of freedom, 16 have a biological counterpart and four are necessary to make the complete model. The joints with a biological counterpart are: ankle, knee, hip, shoulder and elbow. Of these joints, the knee and elbow have one degree of freedom, while the ankles, hips, and shoulders have two degrees of freedom each.

In the model, both legs are modeled as a single powertrain. This indicates that the foot that is resting on the ground is the base of the robot. Therefore, the point $X_0Y_0Z_0$ corresponds to the center of the foot of the supporting leg. In case both legs serve as support, $X_0Y_0Z_0$ is taken from the center of the foot of the left leg.

With the kinematic model, the position equations were calculated by making the necessary multiplications of the homogeneous transformation matrices. The position equations of each joint in the X, Y, and Z axes are taken from the last column of the matrix resulting from said matrix multiplications. Once the position of each joint is known and assuming the body has a uniform mass distribution, the center of mass can be calculated with respect to the first link (which will default to the left foot unless otherwise stated). In this model there are 13 different elements, which are:

- Left foot with mass $m_1 = 578$ g
- Right foot with mass $m_2 = 578$ g
- Left tibia with mass $m_3 = 4{,}000$ g
- Right tibia with mass $m_4 = 4000$ g
- Left femur with mass $m_5 = 5{,}000$ g
- Right femur with mass $m_6 = 5{,}000$ g
- Hip with mass $m_7 = 6{,}000$ g
- Torso with mass $m_8 = 18{,}000$g
- Left humerus with mass $m_9 = 3{,}000$ g
- Right humerus with mass $m_{10} = 3{,}000$ g
- Left radius with mass $m_{11} = 2\ 000$ g
- Right radius with mass $m_{12} = 2{,}000$ g
- Head with mass $m_{13} = 4{,}189$ g

Considering that each cm^3 equals 1 g. The formula to calculate the center of mass on the X axis is:

$$X = \sum_{i=1}^{13} X_i * m_i \tag{1}$$

And the same procedure is followed for the Y and Z axes.

3.2. Using simulated annealing for training artificial neural networks

The problem we are trying to solve at this point is that we have a movement to perform (for example, raising the right leg laterally) and we have the data on the angles of each joint to achieve this movement, but the available data does not consider any stability criteria. So, if such data is entered into the simulation, it is likely that the biped will fall as it is not in a stable position. Therefore, the control system must find the value of the angle for each joint that allows the biped to remain stable. As there are no environmental disturbances, the stability criterion only considers that the center of mass of the biped is within the support polygon, which is delimited by the position of the feet of the humanoid.

The control system chosen to solve this task is a multilayer perceptron type neural network. This neural network requires unsupervised training, since, although the data at the input (the desired position) is known, there is no set of training data at the output.

For the training of neural networks in an unsupervised way, simulated annealing was used. As seen in the theoretical framework, this algorithm has shown great performance in optimization. Simulated annealing has been used in a wide variety of problems. However, depending on the problem, it can be encoded in different ways.

The first variable to define is an error function. For convenience we will call this function $E[f(s)]$, where **s** is the output of the network and **f(s)** is a function that represents the environment in which the element is evaluated. output performance. If the training is not online, at each iteration the input data set can be evaluated and fed into the function f(s). Then calculate the error of each of the outputs and, as is common in backpropagation training, calculate the mean square error. With the value of the error it is evaluated if the change is accepted or not. To calculate the mean square error, Algorithm 1 is followed.

To use simulated annealing in neural network training, each weight and each bias are considered as a dimension of the problem. The calculation of the neighbor of a given point will be done for each of the variables of the neural network. Ideally, in each iteration, the neighbor of each one of the variables would have to be calculated and see if with the neighbor there is a better performance, however, due to the fact that in the perceptron the network is completely connected, there is a large number of variables that increase with each neuron added to each layer. This is why the algorithm is

Algorithm 1 Calculation of the mean error (emean).

```
m: number of training examples
n: data number of the output vector of the neural network
medium = 0
for $i$ = $1{:}m$ do
    for $j$ = $1{:}n$ do
        emean = emean + $(y(i,j) - output(i,j))^2$
emean = $\frac{\sqrt{emean}}{m*n}$
```

coded so that it only calculates the neighbor of K variables for each iteration. Each modified variable is chosen randomly. The training algorithm for the neural network is shown in Algorithm 2.

Figure 1 shows an example of the error by epoch without (left) and with (right) simulated annealing for a sample of joint points. Without simulated annealing, the number of epochs is greater, and an unstable progression can be observed.

3.3. Modeling the biped

The biped model consists of a series of rectangular blocks joined by rotational joints modeled in Simulink. Simulink was chosen because, being part of MATLAB, it does not need to connect to external programs and all the code is written in MATLAB. Simulink also allows to obtain useful information since it has blocks of pressure sensors or inertial sensors. Additionally, the position of each block in the three axes can be known, but this last feature will not be used. For the distance of each block, the average measurements of the extremities of an adult man are taken into account. The humanoid model consists of two legs, made up of three blocks each, a torso, hips, two arms, made up of two blocks each, and a head. Figure 2 shows the total connection of the humanoid, however, the blocks of the arms and legs are kept as a subsystem to make the diagram more understandable.

In order to illustrate the posture control system applied to movement, in the following section we present details for right leg raise in frontal plane.

Algorithm 2 Neural network training method.

```
choose a large variable T
choose a number $k \in [1, length(X)]$
for each variable $x_i \in X$
$x_i = rand(-1, 1)$
for epoch = 1:Epochs do
    for k = 1:K do
        evaluate the neural network with the weights X for all inputs
        calculate mean error
        r = rand(1,length(X))
        $x_i = X(r)$
        calculate neighbor $x_i'$ of $x_i$
        substitute $x_i$ for $x_i'$ in X
        evaluate the neural network with the new weight for all inputs
        calculate newmean error
        if $newmeanerror < meanerror \,||\, rand(0,1) < \frac{1}{e^{\frac{f(x_i)-f(x_i')}{T}}}$ then
            $x_i = x_i'$
    Decrease the temperature T
```

4. Experiments and results

4.1. First neural network training for biped balance

As a first approximation for balance control of a biped, we continue with the procedure presented in the previous section. A MATLAB simulation of the biped was performed. Figure 3A shows this biped and a small movement in Figure 3B. This simulation is a plot of the position of each joint of the biped calculated in Section 3.1. In this simulation, the only information obtained to calculate the control is the center of mass. But thanks to the calculation of the Zero Moment Point, it is possible to calculate the biped's error and determine whether or not it is in a balanced position.

The purpose of this simulation is to train a neural network whose input is a series of positions of the biped to perform a movement, these positions do not consider any stability criteria, so the movement may or may not be in a stable region. In the case presented in this section, the movement consists of raising the right leg in the frontal plane. The movement consists of 50 positions in which the lateral hip joint opens from an angle 0 to $\pi/2rad$.[1]

The neural network proposed to solve this task is a fully connected multilayer perceptron-type neural network. This network has 20 neurons in the input layer and 20 neurons in the output layer. Figure 4 shows the correspondence of each joint with each neuron in the input layer. Sixteen blue lines and four red lines are shown in the image. The blue lines correspond to mobile joints that the biped has, the red lines are rigid joints that are necessary to show the complete model. The joints in red could have been omitted in the training of the neural network, but keeping them in the simulations did not affect the model. It is important to clarify that during training a limit is not considered in the angle of each joint, a limit that would exist in the case of testing with a physical robot.

Apart from the joints, there are 15 points of interest in the humanoid that are useful to calculate the similarity in the input and output trajectories of the neural network. These points coincide with the joints of the humanoid except for the points on the hands and on the head.

The problem to be solved in this neural network is the minimization of the error. The calculation of the error is made considering two metrics: the proximity of the center of mass with the 0 point of the XZ plane and the average error between the input and output joints of the network. The calculation of the mean error is done by subtracting the total of the training data set from the output data set and squaring the result. For more details on the calculation of the mean error, refer to Algorithm 1.

[1] The movement can be seen on https://youtu.be/DKR9xuieMFk. The path marked in red is the center of mass and how it changes position according to the movement made.

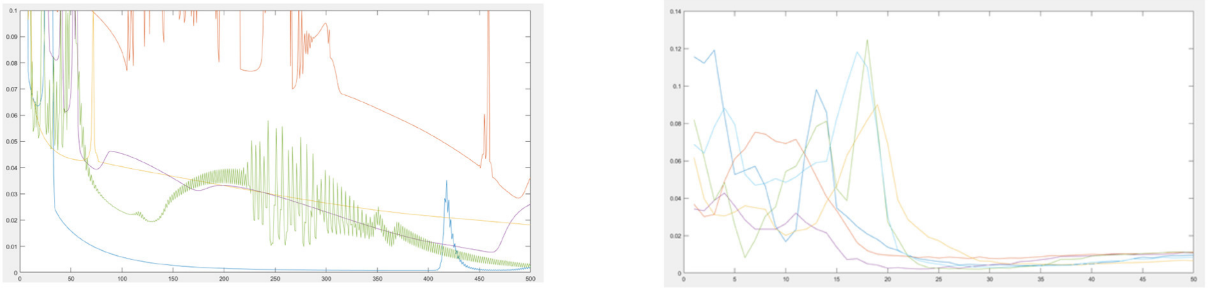


FIGURE 1
Error by epoch when training without **(left)** and with **(right)** simulated annealing.

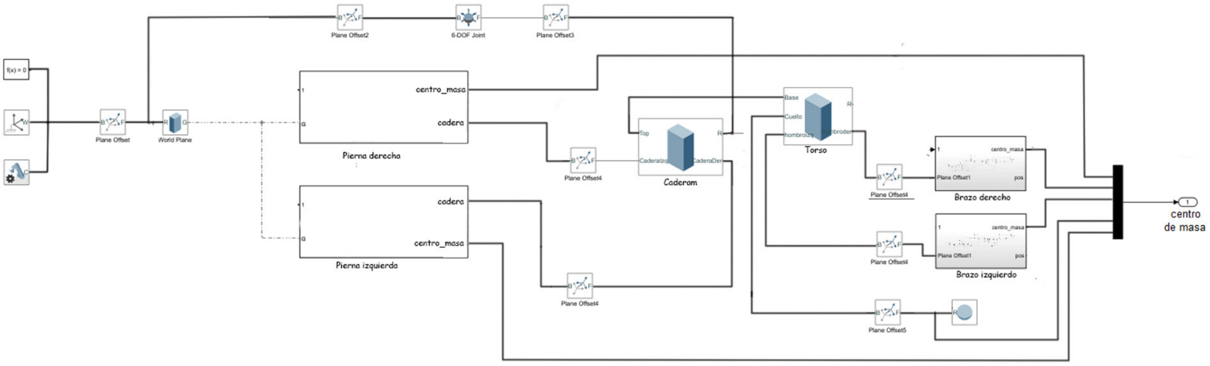


FIGURE 2
Block model of the complete system.

To calculate the center of mass error, the direct kinematics function is used, which gives the position of the center of mass in the three axes X,Y,Z. Since the humanoid stands on the XZ plane, it is on this plane that the position of the center of mass must be minimized. For this, we calculate the distance from 0 to the point of the center of mass with the equation $cm = \sqrt{x^2 + z^2}$.

Both the mean error of the joints and the position of the center of mass are multiplied by a constant to give greater or lesser importance to each metric. The error related to the difference between coordinates, called $\boldsymbol{error_{equal}}$ is multiplied by a constant α. The error related to the position of the center of mass, called $\boldsymbol{error_{cm}}$ is multiplied by a constant β. The total error is the sum of both errors, as shown in Equation (3).

$$T_e(ep, n, m, p) = ep(4mnp + 2m)error \;=\; \alpha * error_{equal} + \beta * error_{cm} \qquad (2)$$

The parameters related to simulated annealing are: **T** s the initial temperature, **TempVar** indicates every how many epochs the temperature decreases, and **DeltaTemp** indicates the percentage of temperature that will remain after **TempVar** epochs. The variables that must be defined are:

1. T Initial temperature
2. DeltaT Temperature Change
3. TempVar, or the number of epochs before the temperature drops
4. DeltaEpoch or the number of epochs until the temperature is increased again
5. Epochs
6. LimitsW or the range in which the weights and bias are initialized
7. LimiteV or the maximum value of neighbor value for each weight
8. K or the number of weights and bias that are modified for each epoch
9. alpha and beta or the importance value given to each error parameter.

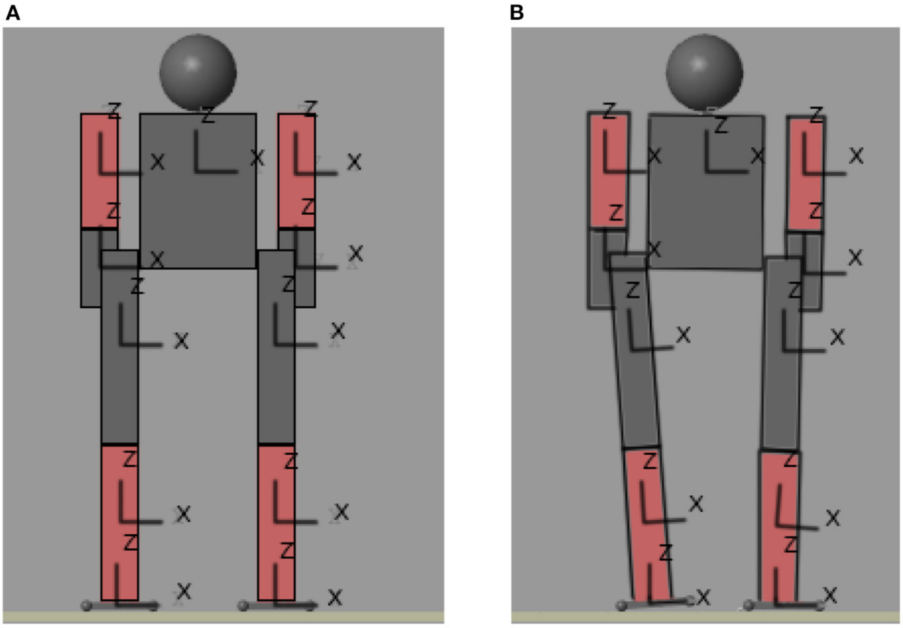


FIGURE 3
Movement of the biped. **(A)** Initial position. **(B)** First stable position.

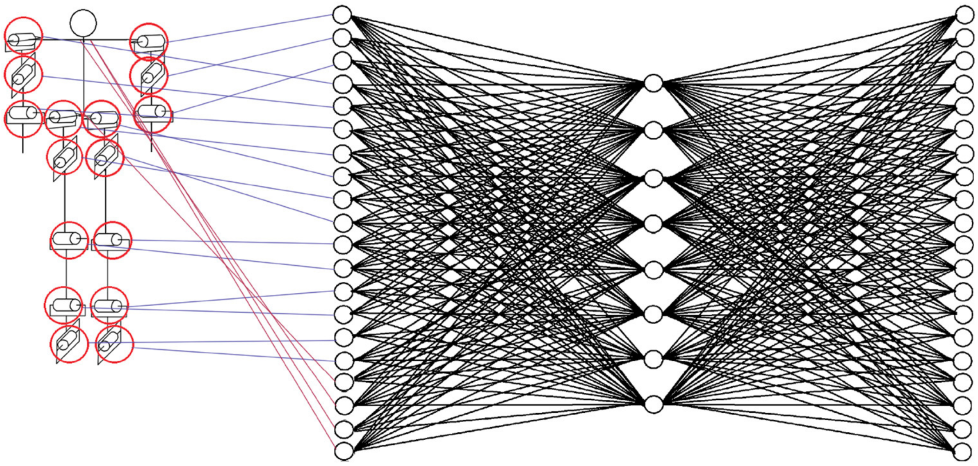


FIGURE 4
Correspondence of each joint with each neuron in the input layer.

The parameters of the algorithm used in this work are shown below.

1. $T = 1{,}000$
2. $TempVar = 5$
3. $DeltaT = 0.85$
4. $DeltaEpoch = 100$
5. $LimitsW = [-2, 2]$
6. $LimitsV = [-0.5, 0.5]$
7. $K = 100$
8. $\alpha = 0.75$
9. $\beta = 0.25$

To know if the center of mass is within the support polygon, it must not exceed 0.085 both in the x axis and in the z axis. This due to the architecture of the humanoid that is taken as a reference.[2]

[2] The video of the complete movement can be seen at https://youtu.be/ViPiFOuDbDA

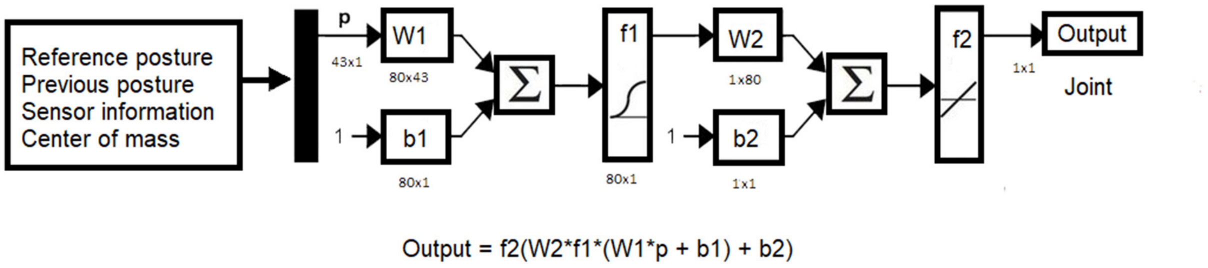


FIGURE 5
Architecture of the neural networks used.

4.2. First standing training (front plane right leg raise)

The biped must always start in position 0 (in which all joints are at 0), then take it to some initial position other than 0 (if required) and from there start doing the experiments. This is because if the robot is initialized in a different position, the pressure sensors can appear inside the ground and mark very high values that decrease over time until the biped "goes up" and is positioned on the ground. It is possible to initialize the biped to a position other than 0, however there are sometimes errors, so it is preferable to follow the instructions above. Figure 3A shows the biped in position 0, which is when all its joints are at 0 position. For this simulation, we chose to perform the movement which is to raise the right leg 90° in the frontal plane. To accomplish this, the biped was started at position 0. Once in this position, instructions are given for the biped to move each of its joints to a range of $[-0.2, 0.2]$ radians, moving one joint at a time. This to know what is the position in which the center of mass is placed on the left leg, which will be the one that remains on the ground[3]. To know the position of the center of mass, the Inertial Sensor block of the Simscape Multibody Body Elements library was used. This block gives information on the position of the center of mass of each body. Therefore, calculating the average of the center of mass of each body, gives the total center of mass of the humanoid. This position is the absolute position of the center of mass with respect to point 0 of the world in which we are working. To calculate the position relative to the supporting leg it is necessary to subtract the total center of mass minus the center of mass of the foot of the supporting leg.

Once the degree configuration of the joints necessary to place the center of mass on the supporting leg is known, the neural networks are trained to reach said position. A total of eight neural networks were used: two for right hip (one for each joint) and other two for left hip, one for each knee and one for each arm.

For the control of the ankle joints, a different algorithm was used, which is explained later. The architecture is the same for all neural networks and is shown in Figure 5. The inputs of the neural network are:

1. Reference path consisting of 16 data.
2. Previous position consisting of 18 data.
3. Left and right sensors consisting of 8 data.
4. Relative position of the center of mass of the body with respect to the position of the foot of the supporting leg in the x,y plane.

Hyperbolic tangent was used as activation function. The neural networks were trained so that, while at the input the position is position 0, each one delivers the necessary value to place the center of mass of the biped on the left leg, we will call this position the first position of stability, which is shown in Figure 6. Once the neural network is trained, it runs until it reaches the first stability position. Figure 7 shows the trajectory of the center of mass in the X axis of the body relative to the support leg. A red line is shown, which is the limit at which a stable position is considered if only the left leg is supported.

In this problem of optimization, we are trying to minimize the error, for this we are taking into account two parameters: the center of mass of the body and the difference between angle joint input and output. We know the center of mass of the body using the function block described called Inertia Sensor, which gives the position of the center of mass in the three axes X,Y,Z. Since the humanoid stands on the XY plane, it is on this plane that the position of the center of mass must be minimized. For this, we calculate the distance from 0 to the point of the center of mass with the equation $cm = \sqrt{x^2 + y^2}$.

$$error = \alpha * error_{equal} + \beta * error_{cm} \tag{3}$$

[3] The video showing these movements can be found at https://youtu.be/AHMae7Ki1Go

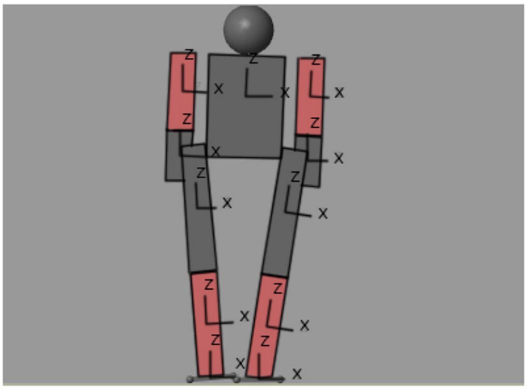


FIGURE 6
First stable position of the humanoid.

Algorithm 3 Learning algorithm in Simulink.

```
c = 1;
The vector of positions called Vector
is received **while** $simulation == true$ **do**
    $target = Vector[c]$
    Calculate the output of the neural network
    $error = \alpha(target - output) + \beta cm$
    **while** $error > \Delta e$ **do**
        The neural network is modified according to the simulated
          annealing algorithm
        Calculate the output of the neural network
        $error = \alpha(target - output) + \beta cm$
    **if** $c < length(Vector)$ **then**
        $c = c + 1;$
```

Both the error of the joint and the position of the center of mass are multiplied by a constant to give greater or lesser importance to each metric. The error related to the difference between angle joint, called $\boldsymbol{error_{equal}}$ is multiplied by a constant α. The error related to the position of the center of mass, called $\boldsymbol{error_{cm}}$ is multiplied by a constant β. The total error is the sum of both errors, as shown in Equation (3).

Once the biped is in the first position of stability, the first value of the position vector, which is specific for each joint, is entered into each neural network as a target. Once the difference between the network output and the target is small enough (remembering that the output may not be the same since the neural network considers the center of mass error). The target is updated to the next value in the array of positions. A brief description of the learning algorithm is shown in Algorithm 3. This ensures that it will not advance to the next position until the previous position has been learned first, and will not advance beyond the last indicated input.

Algorithm 4 Storage method.

```
The output is calculated by doing the matrix multiplication.
The information is sent to the model and the error is evaluated.
If the error is less than or equal to a given value e, the input and output
  are saved in the respective vector and it is repeated from Step 1 with a
  new position.
If the error is still not small enough, another modification is made to the
  neural network and steps 2–5 are repeated.
```

Figure 8 shows how one of the neural networks learns. In this case it is the neural network that controls the lateral right hip joint. The blue signal is the target, the red signal is the output delivered by the neural network. The staggered shape of the target is due to the fact that it will not change to the next position until the error between the two signals is not small enough, that is why when the red signal approaches the blue one, the target signal jumps.

Since the neural networks ensure a stable movement for the current position, a series of inputs and their corresponding output are known to move in a stable way. So the information is stored in case you want to train a subsequent neural network in a supervised way. To store such information, a vector of inputs and outputs is initialized to zero. As the epochs progress, the inputs collected from the sensors are saved. Only those outputs that ensure a sufficiently small error are saved and not all the outputs delivered by the neural network. Algorithm 4 shows the steps to follow to collect the data that can be used later to train a neural network in a supervised way.

It is important to clarify that the output delivered by the neural network is not directly input to the algorithm. This is due to the sensitivity of the simulator to sudden changes in the joints.[4] Therefore, once the neural network delivers an output with a sufficiently small error, the joints are instructed to reach this value by changing their value by Δc each cycle. For this work, $\Delta c = 0.00007rad$ was selected. This value may seem small but it is necessary so that the change in position of the joint is not made abruptly.

It should be sought that the four sensors of each foot are in contact with the ground and that the force is evenly distributed in each sensor. This is why the algorithm is relatively simple. For the four joints, an algorithm similar to the one shown in Algorithm 5 is followed.

Figure 9 shows the initial and final position of the biped performing the movement of raising a the right leg in frontal plane[5]. The trajectory of the center of mass in the X axis is

[4] Video at https://youtu.be/HU5L8lLqmrA shows a simulation in which the ankle joint was changed from position 0 to 0.1.

[5] The full video of this move is at https://youtu.be/2SNglSgEwzw

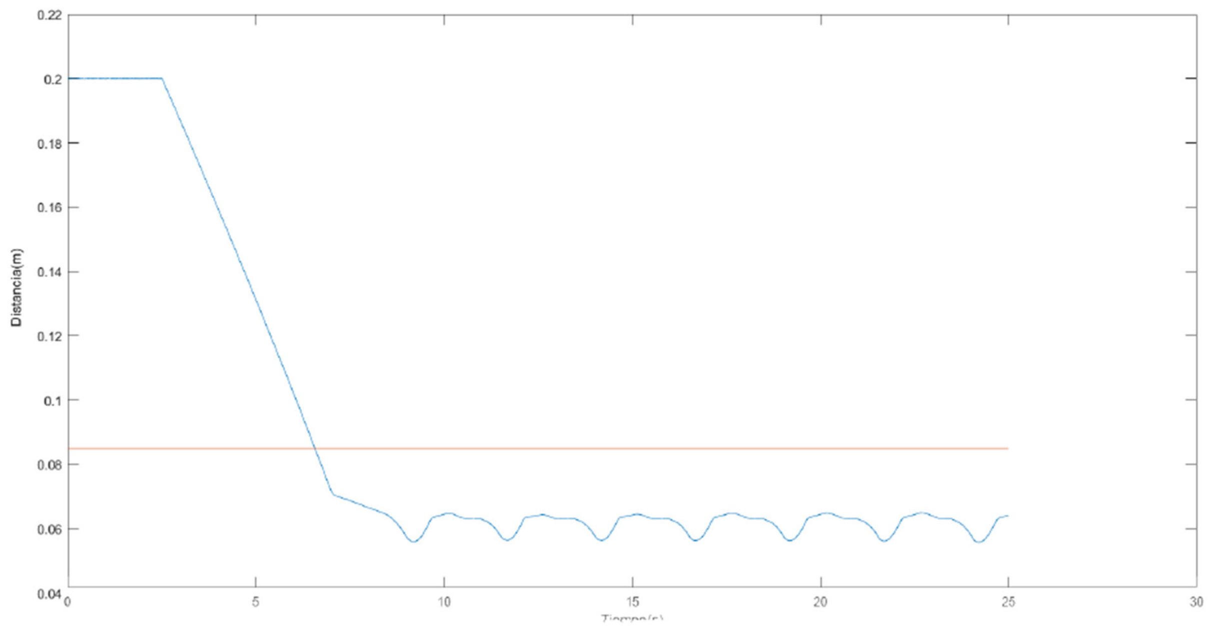


FIGURE 7
Trajectory of the center of mass on the X axis.

Algorithm 5 Ankle control algorithm.

```
if $s1 < s2|s4 < s3$ then
  | joint = joint + num;
if $s1 > s2|s4 > s3$ then
  | joint = joint - num;
```

observed in Figure 10A. Recall that the biped is standing on the X,Y plane, the X axis is the frontal plane of the biped and the Y axis is the sagittal plane. Therefore, for this movement the axis that has more importance is the X axis. Figures 10B,C also shows the trajectory of the center of mass in the Y and Z axes[6].

Figure 11 shows the output of the neural network in blue as well as the trajectory of the joint that moves the right hip laterally in red. There is a clear time lag between the output of the neural network and the articulation. This displacement is due to what was previously mentioned, by having to slowly modify the angle of the articulations, the neural network converges faster than the joint reaches the desired position.

4.3. Training the biped with added perturbation (ground tilt)

As explained in previous sections, the training used for neural networks is a learning style that does not require full knowledge of the desired output. This **flexibility** in learning allows the neural network to learn in an environment with certain disturbances. As an added disturbance, the platform on which the humanoid is standing tilted. The tilt was 1° and was done on the X axis.

Figure 12 shows the initial and final position of the biped seen from the front (left) and the same positions seen from the side for a better visualization of the slope of the ground (right).[7] The learning trajectory and the actual motion path, compared with regard to the desired one, can be seen in Figure 13. This figure shows the output trajectory of the neural network in blue, this is to visualize how the trajectory of the neural network is modified according to the desired target. Figure 13B shows the same target, this time in blue but with the actual output of the joint. In this figure, the disturbance of inclined ground is considered.

[6] For a better visualization the video of the trajectory of the center of mass in the X and Y axes while the movement is executed can be found at https://youtu.be/xxBiI43VpeM

[7] The full motion video can be found at https://youtu.be/7ktSoxzQo_0

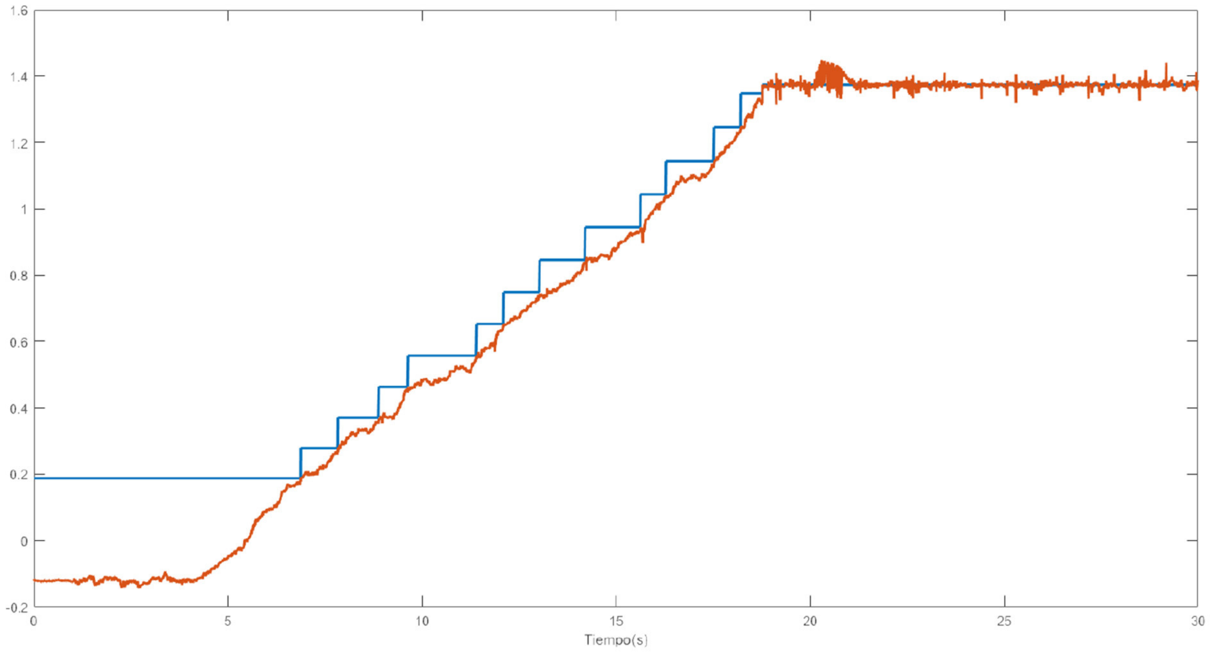


FIGURE 8
Output trajectory and target of the neural network.

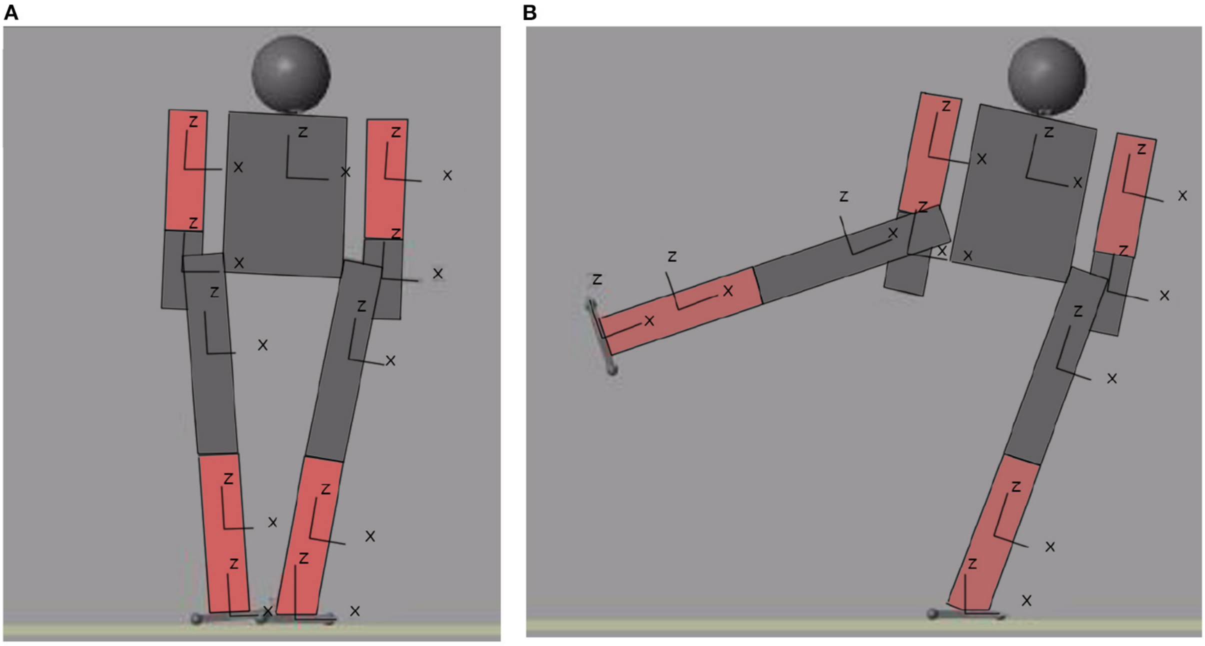


FIGURE 9
Initial **(A)** and final **(B)** position of the movement performed.

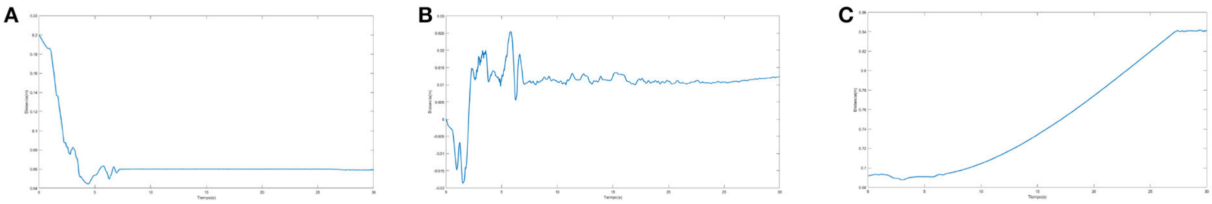


FIGURE 10
(A) Trajectory of the center of mass in X axis, **(B)** Y axis, and **(C)** Z axes of movement 1.

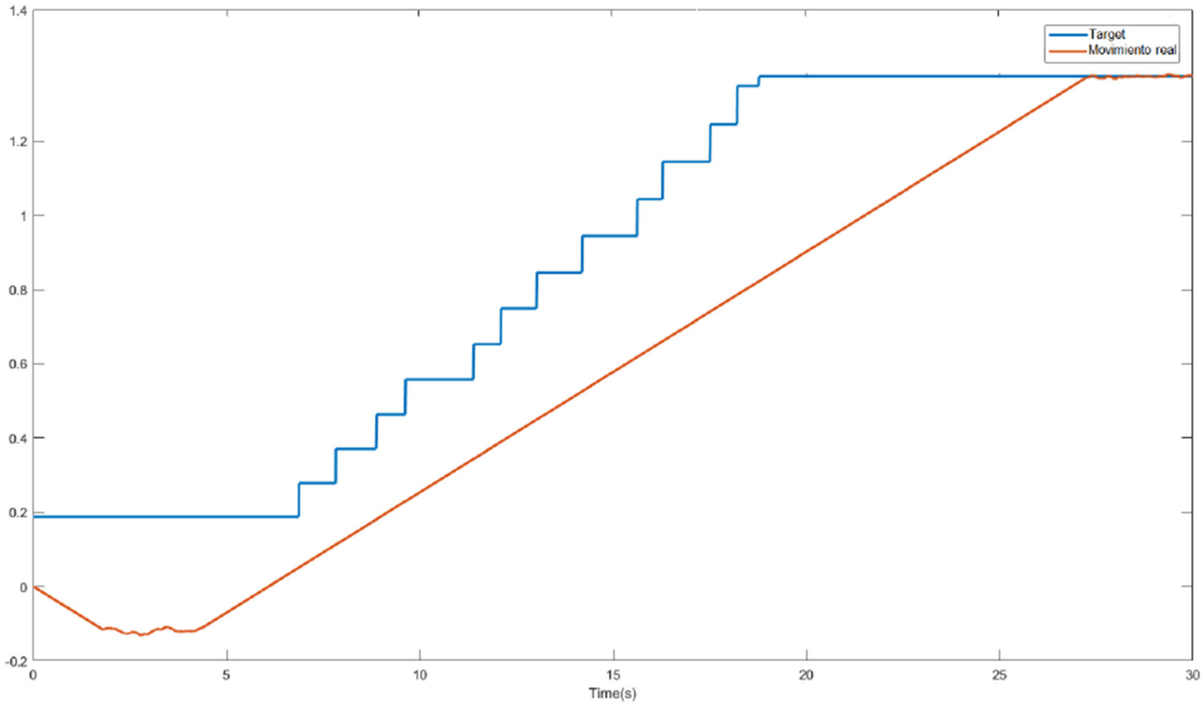


FIGURE 11
Comparison of desired move (blue) with performed movement (red).

4.4. Training the biped with added mass perturbation

As an added disturbance a 0.1 m sphere was added to the biped's right arm. This represents 7.5% of the humanoid's weight. Figure 14 shows the initial (Figure 14A) and final (Figure 14B) position of the biped's movement. The movement he performs is the lifting of the right leg in the frontal plane. The learning trajectory and the trajectory of the desired (blue) and actual (red) movement are shown in Figure 15. This figure allows to visualize how the trajectory of the neural network is modified according to the desired target. Figure 15B shows the same target in blue, with the actual output of the joint when a mass is added to the biped.

5. Conclusions

This work has presented the use of neural network control for robot stabilization and displacement. Although there are already several works that use neural networks in control, they usually require the help of external control methods or complicated training of many iterations to solve the assigned task. In this case the control was implemented in a simple way with little prior knowledge. The method we have presented allows having both flexibility and adaptability, as it has been presented in the last two sections above: the neural network adapted its weights despite having changing conditions such as a floor inclination, or adding a mass. The presented method is

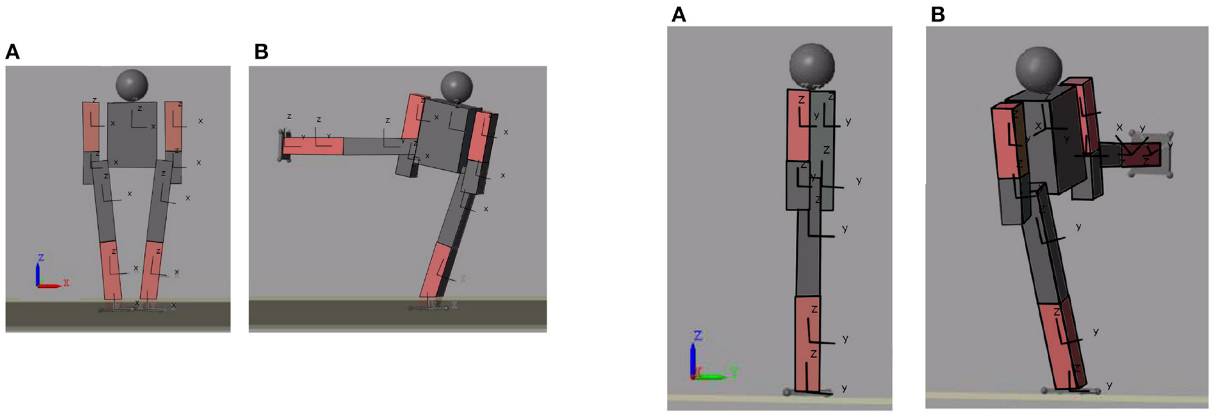


FIGURE 12
Initial and final position of the biped seen in the XZ (left–**A,B**) and YZ (right–**A,B**) planes for ground tilt.

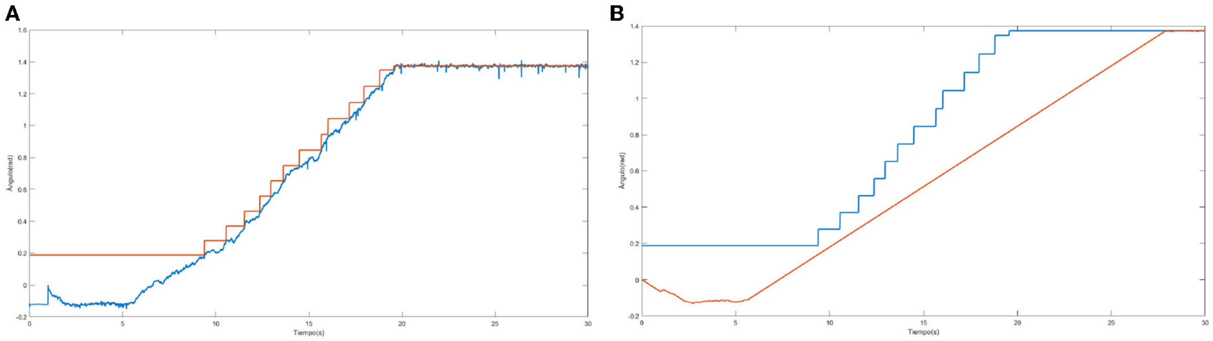


FIGURE 13
(A,B) Learning path for ground tilt and desired and actual motion path for ground tilt.

a mixture of supervised and reinforcement learning, combined in a novel way, allowing to implement several neural network applications easily and with little prior knowledge. Simulated annealing was used as a means of training the neural networks; although this optimization method has been used previously, it is not common to apply this method for training this kind of application. We have found that using this method in the application of robot control can help to reach a goal position while keeping the biped humanoid stable. Although the presented solution is slower at some steps than other better known methods, by using simulated annealing, in overall there is a noticeable reduction in the epochs needed to converge, making this method feasible for many applications. A difficulty to consider is that the learning time is variable because, being a heuristic method, the random component in learning means that the learning time is not always optimal.

One of the applications of postural stability is that, by knowing or calculating the stable positions, it is possible to implement passive locomotion. Although this type of locomotion is more inefficient than active locomotion, it allows movement and has a robust response to disturbances. Although there is software specialized in robot simulation, sometimes it is difficult to know how to run or control these simulators. If it is not necessary to have many sensors, and the purpose is to test robots whose constitution is relatively simple, MATLAB is a good option, since within this program the necessary actuators can be controlled without the need for an external program. There is a drawback, however, in that the simulation time can be long. In this work, a humanoid was developed that can be used for future projects. The link to download the humanoid can be found at Yoqsan (2022).

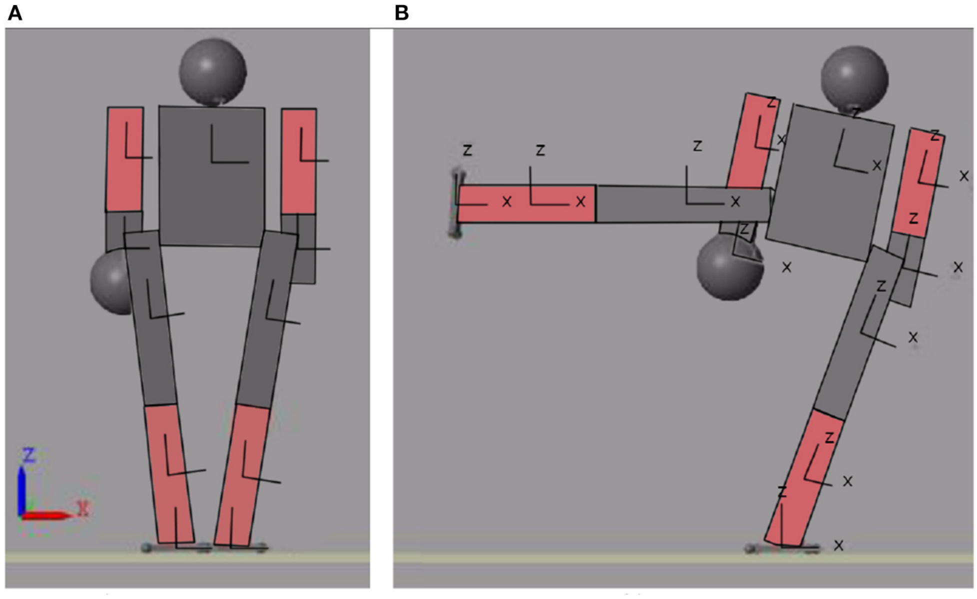


FIGURE 14
Initial and final position of the movement seen from the XZ plane. **(A)** Initial position of motion. **(B)** Final position of motion.

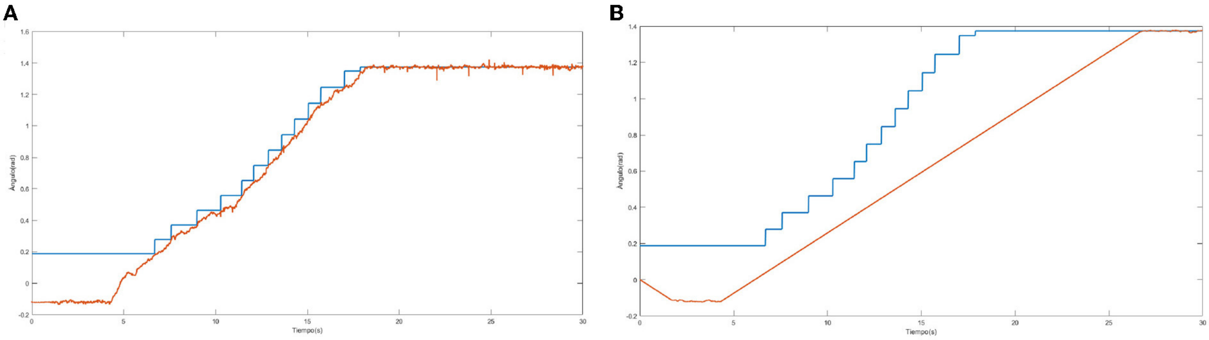


FIGURE 15
Learning Path for Added Mass and Trajectory of the desired (blue) and actual (red) movement for Added Mass **(B)**. **(A)** Desired trajectory (blue) and output network (red).

5.1. Future work

This work presented a robot control method that had not been presented before, so it is still in its infancy and can be optimized in many areas. The simulated annealing method can be mixed with other optimization methods to find more optimal neighbors than looking for close neighbors, which is what was done in this work. Other types of architectures can also be tested, in this work a multilayer perceptron was used, however, there are other neural architectures that may help improve the response.

The objective of this work was to achieve a balance in the posture of a humanoid and, although the objective was achieved, slow movements were required to achieve it. If, in addition to the anterior position of the joint, information on the speed and acceleration of the joints is included, it would be possible to increase the speed at which the biped moves. If the error also considers a criterion of energy expenditure, it would also be possible to make the optimal movements energetically, just as human beings do.

Furthermore, having this type of locomotion, it is possible to optimize it to achieve dynamic locomotion. Experiments of this latter point are left as future work. According to the data obtained from the experiments, it is known that this method allows a robot to function in an environment where there are unknown disturbances. Motion prediction is not done yet, however, it is possible to implement another neural network or modify existing ones by adding memory and recursion to predict motion.

As seen in the experiments presented, neural networks behaved correctly despite adding disturbances, so it is possible that the same control system works to move robots with another configuration. Although this was not tested, in future works the control system could be implemented with different configurations of robots and obtain information on the difference between their movements.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, specifically Yoqsan (2022). Humanoid in Simulink. Further inquiries can be directed to the corresponding author/s.

Author contributions

HC and HS: original concept and supervision. YA-G: proposal, implementation, and original draft. HC: final writing and funding. ÁA-R: guidance, supervision, and general revision. All authors contributed to the article and approved the submitted version.

Funding

This work was partially supported by CONACyT, Mexico, and by Secretaría de Investigación y Posgrado of the Instituto Politécnico Nacional, Mexico, under Grant 20220553.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2022.934109/full#supplementary-material

References

Boden, M. (2006). *Mind as Machine: A History of Cognitive Science, 1st Edn.* Oxford: University Press Inc.

Cho, B.-K. and Kim, J.-Y. (2018). Dynamic posture stabilization of a biped robot subo-1 on slope-changing grounds. *Int. J. Precis. Eng. Manuf.* 19, 1003–1009. doi: 10.1007/s12541-018-0118-8

Da, X., and Grizzle, J. (2019). Combining trajectory optimization, supervised machine learning, and model structure for mitigating the curse of dimensionality in the control of bipedal robots. *Int. J. Robot. Res.* 38, 1063–1097. doi: 10.1177/0278364919859425

Geng, T., Porr, B., and Wörgötter, F. (2006). A reflexive neural network for dynamic biped walking control. *Neural Comput.* 18, 1156–1196. doi: 10.1162/neco.2006.18.5.1156

Ghorbani, R., Wu, Q., and Wang, G. G. (2007). Nearly optimal neural network stabilization of bipedal standing using genetic algorithm. *Eng. Appl. Artif. Intell.* 20, 473–480. doi: 10.1016/j.engappai.2006.09.007

Gil, C. R., Calvo, H., and Sossa, H. (2019). Learning an efficient gait cycle of a biped robot based on reinforcement learning and artificial neural networks. *Appl. Sci.* 9, 502. doi: 10.3390/app9030502

Jha, R. K., Singh, B., and Pratihar, D. K. (2005). On-line stable gait generation of a two-legged robot using a genetic-fuzzy system. *Robot. Auton. Syst.* 53, 15–35. doi: 10.1016/j.robot.2005.06.006

Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., and Hirukawa, H. (2001). "The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the Next Millennium (Cat. No. 01CH37180), Vol. 1*, 239–246. doi: 10.1109/IROS.2001.973365

Kim, D., Seo, S.-J., and Park, G.-T. (2005). Zero-moment point trajectory modelling of a biped walking robot using an adaptive neuro-fuzzy system. *IEE Proc. Control Theory Appl.* 152, 411–426. doi: 10.1049/ip-cta:20045007

Kim, D. W., Kim, N.-H., and Park, G.-T. (2012). ZMP based neural network inspired humanoid robot control. *Nonlin. Dyn.* 67, 793–806. doi: 10.1007/s11071-011-0027-1

Kuo, A. (2007). Choosing your steps carefully. *IEEE Robot. Autom. Mag.* 14, 18–29. doi: 10.1109/MRA.2007.380653

Kuo, A. D. (1995). An optimal control model for analyzing human postural balance. *IEEE Trans. Biomed. Eng.* 42, 87–101. doi: 10.1109/10.362914

Kuo, A. D. (2007). The six determinants of gait and the inverted pendulum analogy: a dynamic walking perspective. *Hum. Mov. Sci.* 26, 617–656. doi: 10.1016/j.humov.2007.04.003

Lin, C.-M., Fan, W.-C., Chen, C.-H., and Hou, Y.-L. (2006). "Robust control for biped robot using cerebellar model articulation controller," in *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, 2485–2490.

Miller, W. T. (1994). Real-time neural network control of a biped walking robot. *IEEE Control Syst. Mag.* 14, 41–48. doi: 10.1109/37.257893

Murakami, S., Yamamoto, E., and Fujimoto, K. (1995). "Fuzzy control of dynamic biped walking robot," in *Proceedings of 1995 IEEE International Conference on Fuzzy Systems, Vol. 1* (Yokohama: IEEE), 77–82. doi: 10.1109/FUZZY.1995.409663

Plumer, E. (1996). Optimal control of terminal processes using neural networks. *IEEE Trans. Neural Netw*. 7, 408–418. doi: 10.1109/72.485676

Pratt, J. E., and Drakunov, S. V. (2007). "Derivation and application of a conserved orbital energy for the inverted pendulum bipedal walking model," in *Proceedings 2007 IEEE International Conference on Robotics and Automation* (Rome: IEEE), 4653–4660. doi: 10.1109/ROBOT.2007.364196

Sun, Y., Xu, J., Lin, G., and Sun, N. (2021). Adaptive neural network control for maglev vehicle systems with time-varying mass and external disturbance. *Neural Comput. Appl*. 1–12. doi: 10.1007/s00521-021-05874-2

Wu, Y., Song, Q., and Yang, X. (2007). Robust recurrent neural network control of biped robot. *J. Intell. Robot. Syst*. 49, 151–169. doi: 10.1007/s10846-007-9133-1

Yang, C., and Wu, Q. (2006). On stabilization of bipedal robots during disturbed standing using the concept of lyapunov exponents. *Robotica* 24, 621–624. doi: 10.1017/S0263574706002700

Yoqsan (2022). *Humanoid in Simulink*. Available online at: https://drive.google.com/file/d/1XysLsFHL6VGZ2xKypekPi6QhedwzPCzX

Zhang, Y., Wang, J., and Xia, Y. (2003). A dual neural network for redundancy resolution of kinematically redundant manipulators subject to joint limits and joint velocity limits. *IEEE Trans. Neural Netw*. 14, 658–667. doi: 10.1109/TNN.2003.810607

frontiers | Frontiers in Neurorobotics

TYPE Brief Research Report
PUBLISHED 04 August 2022
DOI 10.3389/fnbot.2022.875058

Check for updates

OPEN ACCESS

EDITED BY
Mu-Yen Chen,
National Cheng Kung
University, Taiwan

REVIEWED BY
Nuryono S. Widodo,
Ahmad Dahlan University, Indonesia
Pei-Hsuan Lin,
National Chung Hsing
University, Taiwan

*CORRESPONDENCE
Yue Chu
chuyue@dlu.edu.cn

RECEIVED 13 February 2022
ACCEPTED 07 July 2022
PUBLISHED 04 August 2022

CITATION
Chu Y (2022) Recognition of musical beat and style and applications in interactive humanoid robot.
Front. Neurorobot. 16:875058.
doi: 10.3389/fnbot.2022.875058

Recognition of musical beat and style and applications in interactive humanoid robot

Yue Chu*

Music College, Dalian University, Dalian, China

The musical beat and style recognition have high application value in music information retrieval. However, the traditional methods mostly use a convolutional neural network (CNN) as the backbone and have poor performance. Accordingly, the present work chooses a recurrent neural network (RNN) in deep learning (DL) to identify musical beats and styles. The proposed model is applied to an interactive humanoid robot. First, DL-based musical beat and style recognition technologies are studied. On this basis, a note beat recognition method combining attention mechanism (AM) and independent RNN (IndRNN) [AM-IndRNN] is proposed. The AM-IndRNN can effectively avoid gradient vanishing and gradient exploding. Second, the audio music files are divided into multiple styles using the music signal's temporal features. A human dancing robot using a multimodal drive is constructed. Finally, the proposed method is tested. The results show that the proposed AM-IndRNN outperforms multiple parallel long short-term memory (LSTM) models and IndRNN in recognition accuracy (88.9%) and loss rate (0.0748). Therefore, the AM-optimized LSTM model has gained a higher recognition accuracy. The research results provide specific ideas for applying DL technology in musical beat and style recognition.

KEYWORDS

multi-modal features, humanoid robot, recurrent neural network, recognition technologies, musical beat and recognition of style source

Introduction

Music is an indispensable part of modern life, which can assist the expression of emotion in different situations. Musical factors are complex. A composer considers some basic elements in his/her music composition: rhythm, melody, harmony, and timbre. Thus, one must gain a professional understanding of the basic musical elements to understand musical contents or themes. Non-professionals can also empathize with music through musical styles and beats. Of these, style is the overall grasp of music and people's intuitive feeling of a piece of music. Most music-playing software recommends music through users' historical musical style selections. Music recommendation accuracy has become the key metric for users in choosing music recommendation applications (APPs). The beat in music is generally understood as the combination law between strong and weak rhythms, reflected in a stressed note in a piece of a song. Following the musical beat, people will swing their bodies unconsciously or perform other activities. The recognition of style and beat is of great significance to robot performance (Abbaspour et al., 2020; Zhou et al., 2020). People can obtain massive amounts of audio, image, and video information through the internet. Internet music has a large user base, and the internet music library is enriched with the diversification of user needs. Given the massive amount of music information, users urgently demand a personalized information retrieval approach. However, music recommendation is extremely challenging. As a form of artistic expression, music is endowed with a certain emotion. It involves trivial elements, such as melody, rhythm, harmony, and form, thus forming different musical styles. Usually, different music styles emphasize different music elements. These features in music styles can be used in music classification and retrieval through the content information. Currently, the music style classification is most commonly studied and has seen successful commercialization by music dealers to organize and describe music. On the other hand, with the increased capacity of the Internet music library, style-based music retrieval has become the mainstream method of music information retrieval. Classifying music by style can meet the users' personalized music retrieval and facilitate users to retrieve and efficiently manage their preference music styles timely. At the same time, it is convenient for music dealers to manage and label music styles and recommend music styles of interest to users. Automatic and accurate classification and recognition of music styles can effectively reduce labor costs. Therefore, improving the accuracy of music style classification and recognition can promote the intelligent development of music platforms. It provides better services for music listeners, improves user experience, and expands their choices, which have great research and economic value.

As artificial intelligence (AI) research becomes mature, its application gets closer to public life. For example, intelligent robots are seeing various applicational scenarios, such as service robots and unmanned aerial vehicles (UAVs). Meanwhile, robotic technologies are oriented toward entertainment from practical works. Research on service-oriented robots is abundant both in and outside China, while there is relatively little research on dancing robots. Dancing to the beat might seem natural to a human, but getting robots to respond to beats requires tons of work and design.

Chronologically, Robots' applications can be segmented into several phases, from industrial robots to service robots and household robots. From the economic sector's perspective, robot applications are experienced practical->industrial-entertainment->domestic development stage. Researchers have also done many works in robotics, deep learning (DL), and music interaction in robotics. Wen (2020) designed an intelligent background music system based on DL, the internet of things (IoT), and the support vector machine (SVM). They used a recurrent neural network (RNN) structure to extract image features. Nam et al. (2019) developed an automatic string plucking system for guitar robots to generate music without machine noise. The soft robot technology was used for a new silent actuator: a soft elastic cone as a buffer to prevent impact noise. As a result, an elastic cone design method based on nonlinear finite element analysis (FEA) was proposed. The silent characteristics of the silent actuator were confirmed by the noise test that compares the silent actuator with the traditional actuator. Rajesh and Nalini (2020) represented that music was an effective medium to convey emotions. Emotional recognition in music was the process of recognizing emotions from music fragments. They proposed an instrument-like emotional recognition method in view of DL technology. The music data set was collected from strung, percussion, woodwind, and brass instruments corresponding to four emotions, namely, happiness, sadness, neutrality, and fear. From the instrumental data set, the features of Mel frequency cepstral coefficient (MFCC), normalization statistics of chroma energy, short-term Fourier Transform (FT) of chroma, spectral characteristic, spectral centroid, bandwidth, attenuation, and time characteristics were extracted. Based on the extracted features, the RNN was trained for emotional recognition. Then, the performance of RNN and baseline machine learning (ML) classification algorithm was compared. The results showed that deep RNN had an excellent effect on instrument emotional recognition. Instrument classes played an important role in music-induced emotions. Briot and Pachet (2020) indicated that in addition to traditional tasks, such as prediction, classification, and translation, DL was receiving increasing interest as a music generation method. The latest research groups, such as Google's Magenta and Spotify's Creator Technology Research Lab (CTRL), were evidenced. The motivation was to automatically use DL architecture to learn music style from any music corpus and then generate samples from the estimated distribution. Then, DL-based music generation reached certain limitations, such as feedforward in circular architecture, because they tended to imitate the learned corpus without the incentive of creativity. Besides, the DL architecture did not provide a direct method to control music generation. DL architecture automatically generated music without human-computer interaction (HCI). However, given its generated content, it still could not help musicians create and refine music. They focused on the issues of control, creativity, and interaction analysis. Then the limitations of applying DL to music generation were listed, and possible solutions were outlooked. Martin-Gutierrez et al. (2020) pointed out that the application of multimedia promoted the services provided by platforms, such as Spotify, Lastfm, or Billboard. However, the innovative methods of retrieving specific information from a large amount of music-related data have become a potential challenge in music information retrieval. They studied the creation of SpotGenTrack popular data sets. They proposed an innovative multi-mode end-to-end DL architecture HitMusicNet to predict the popularity of music recording. Experiments showed that the architecture proposed was better than the existing technology.

The innovation of this work is to propose a lightweight multi-task cascaded convolutional neural network (MTCNN). With the help of the proposed MTCNN, the notes are located and extracted for normalization operation. The innovative combination of independent recurrent neural network (RNN) and attention mechanism (AM) is used for music style recognition, and the data are transferred to the multi-attention CNN and long-short term memory (LSTM) network for feature extraction and recognition. This method does not need to label in advance and is weak-supervised learning. The refined features extracted by the multi-attention CNN increase the sample details and contain the global information of the samples. The method proposed improves the accuracy and precision of music style recognition to a certain extent.

Design of music style recognition model and construction of interactive robot system

Music style recognition modeling by IndRNN

The music style describes the overall characteristics of a complete song. The dancing robots' performance style must match the music style. In recent years, researchers have applied neural networks (NNs) to audio signal processing (ASP) (Jiang, 2020; Er et al., 2021). Over time, many variants of RNN have been developed and applied to ASP. This work optimizes the RNN to use for music style recognition. Then, an endpoint detector algorithm (EDA) based on short-term energy difference is proposed. The starting point of notes can be determined by looking for the peak of short-term energy difference. Then, two layers of judgment are designed to determine the endpoint and reduce the dependence on the threshold (Chakraborty et al., 2021; Feng et al., 2021).

The principle of the proposed EDA is shown in Figure 1.

As in Figure 1, the music style recognition process is divided into two stages. The original signal is pre-processed in the training stage, and then the improved RNN is trained using the pre-processed data. In the testing stage, the audio to be tested is first pre-processed by simple data, and then the feature file is transformed (Mcauley et al., 2021; Wang et al., 2021).

DL is one of the main ways to lead to AI. DL is a branch of ML that essentially fits data to summarize the available laws. DL has successfully promoted science and technology and profoundly impacted big data analytics (BDA). A convolutional neural network (CNN) is one of the most important models in DL and lends well to image processing (IP). Combined with other technologies, CNN can be applied to many different fields. LeNet was the first real CNN proposed in 1998. This network has been widely used to recognize the handwritten font of Bank of America check, which has achieved good results. RNN is another commonly used DL structure and has a memory function. It is suitable for solving continuous sequence problems and is good at learning rules between samples with certain sequential significance. Unlike CNN, RNN is generally used in production and prediction, such as in Google Translate and some speech recognition applications (Mirza and Siddiqi, 2020; Wu, 2021). RNN is widely used in language models and text production, image description, video tagging, keyword extraction, and stock analysis. Meanwhile, RNN has a feedback structure. Its output relates both to the current input's weight and to the previous network's input. The difference between RNN and the traditional NN is that RNN has the concept of timing, and the state of the next moment will be affected by the current state. Some researchers also call recurrent networks deep networks, whose depth can be shown in input, output, and time-depth (Hernandez-Olivan et al., 2021; Parmiggiani et al., 2021). The RNN structure is given in Figure 2.

Equation (1) can calculate the hidden state h^t in the forward propagation of RNN:

$$h^t = \tanh\left(Ux^t + Wh^{t-1} + b\right) \tag{1}$$

The network output at a specific moment can be calculated by Equation (2)

$$o^t = Vh^t + c \tag{2}$$

The prediction output can be counted by Equation (3):

$$a^t = \text{softmax}\left(o^t\right) = \text{softmax}\left(Vh^t + c\right) \tag{3}$$

In Equations (1)–(3), $x^{(t)}$ represents the input of training samples when the sequence index number is t. $h^{(t)}$ denotes the hidden state of the model when the sequence index number is t. $h^{(t)}$ is jointly decided by $h^{(t)}$ and $x^{(t)}$. $o^{(t)}$ signifies the output of the model when the sequence index number is t. $o^{(t)}$ is only determined by the current hidden state $h^{(t)}$. $L^{(\mathrm{t})}$ refers to the model loss function when the sequence index number is t. signifies the true output of the training sample sequence when the sequence index number is t. Three matrices () are the model's linear relationship parameters, shared within RNN. The parameter sharing mechanism reflects the "recurrent feedback" of the RNN model.

For RNN, since there is a loss function at each position of the sequence, the final loss L can be explained by Equation (4):

$$L = \sum_{t=1}^{\Gamma} L^{(t)} \tag{4}$$

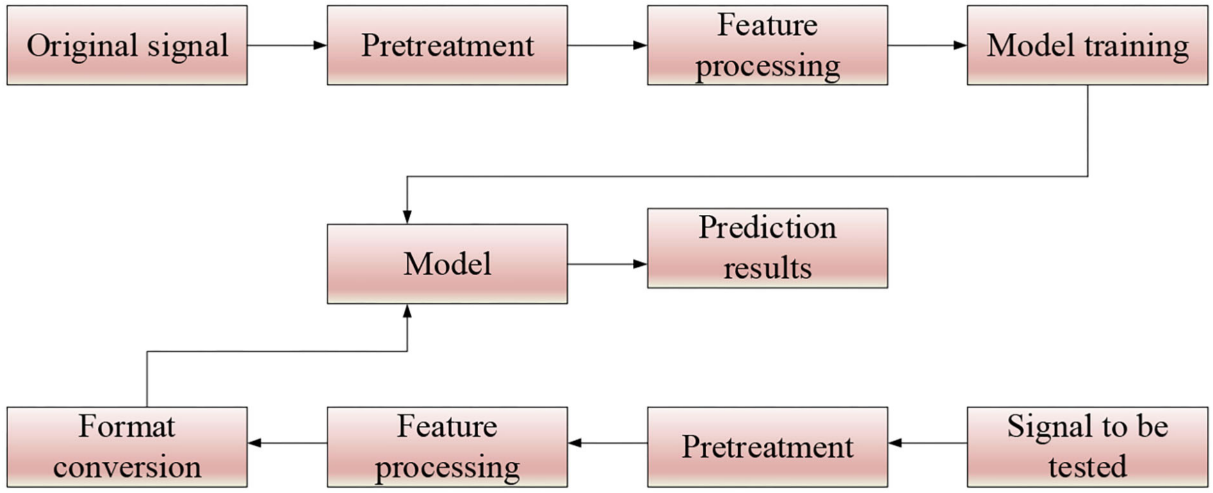


FIGURE 1
Framework of proposed EDA.

The parameter gradient calculation reads:

$$\frac{\partial L}{\partial c} = \sum_{t=1}^{\Gamma} \frac{\partial L^{(t)}}{\partial c} = \sum_{t=1}^{\Gamma} \hat{y}^{(t)} - y^{t} \tag{5}$$

$$\frac{\partial L}{\partial V} = \sum_{t=1}^{\Gamma} \frac{\partial L^{(t)}}{\partial V} = \sum_{t=1}^{\Gamma} \left(\hat{y}^{(t)} - y^{t}\right)\left(h^{(t)}\right)^{T} \tag{6}$$

RNN adds the concept of timing; thus, different input layers can be set according to the time node. Data can be entered in multiple ways. The number of hidden layers in the middle is the same as the number of time nodes, and the number of neurons per layer and independent variables are the same. The disadvantage of RNN is that it cannot solve the problem of long-term dependence, and there is a phenomenon of network gradient dissipation and explosion. Against the defect of RNN, the LSTM NN is proposed (Liu et al., 2021; Alfaro-Contreras et al., 2022), as drawn in Figure 3.

In LSTM, the forget and input gates are expressed by Equations (7) and (8), respectively. The short-term and long-term cell states are counted by Equations (9) and (10). The output gate is exhibited by Equation (11).

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right) \tag{7}$$

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right) \tag{8}$$

$$\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right) \tag{9}$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{10}$$

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right) \tag{11}$$

LSTM and gated recurrent unit (GRU) are the first proposed two RNN variants. However, the practical applications find that the Sigmoid and Tanh functions in LSTM and GRU will lead to gradient attenuation and significantly impact the input of long-term sequences. To solve the above problems, this section introduces an IndRNN (Tan et al., 2021; Wagener et al., 2021). The hidden state in traditional RNN is the input of the next state and is updated by Equation (12):

$$h_t = \sigma\left(WX_t + Uh_{t-1} + b\right) \tag{12}$$

In Equation (12), h_t is the hidden state at time t. h_{t-1} represents the hidden state at the previous moment. U is the weight of different stages.

According to relevant literature, the multiplication operation of recurrent weight causes gradient explosion or attenuation. The IndRNN adopts a new and independent RNN as the basic classification model. Unlike traditional RNN, IndRNN employs a different state update mechanism (Li and Zheng, 2021; Xu et al., 2021). Its recurrent input is processed by Hadamard product, as in Equation (13):

$$h_t = \sigma\left(WX_t + u \odot h_{t-1} + b\right) \tag{13}$$

In Equation (13), u is a recurrent weight. Its mathematical form is a vector. $\odot$ is a Hadamard product operation. The principle is to multiply the corresponding elements of the two matrices before and after the symbol.

At moment t, each neuron only accepts the input at this moment and its own hidden state as input information at moment $t-1$. The hidden state of the nth neuron is described by Equation (14):

$$h_{n,t} = \sigma\left(W_n X_t + u_n h_{n,t-1} + b_n\right) \tag{14}$$

In Equation (14), W_n, u_n is the nth line of input weight and recurrent weight. W and u input the spatial and temporal features, respectively.

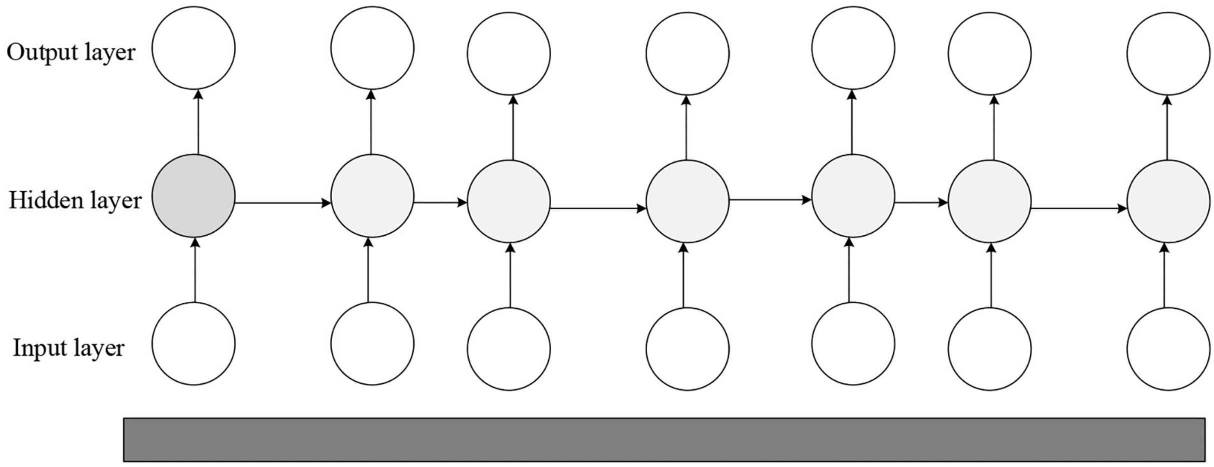


FIGURE 2
RNN structure.

The basic architecture of the IndRNN is depicted in Figure 4. Activation function (AF) chooses the rectified linear unit (ReLU). The IndRNN processes the input weight using the Recurrent + ReLU structure.

The IndRNN back propagates the gradient in each layer according to the temporal features. For the nth neuron $h_{n,t}$, if the optimization goal of T-time is J, then at t-time, the gradient reverse propagation process is described by Equations (15)–(18):

$$\frac{\partial J_n}{\partial h_{n,t}} = \frac{\partial J_n}{\partial h_{n,T}} \frac{\partial h_{n,T}}{\partial h_{n,t}} \tag{15}$$

$$\frac{\partial J_n}{\partial h_{n,t}} = \frac{\partial J_n}{\partial h_{n,T}} \prod_{k=t} \frac{\partial h_{n,k+1}}{\partial h_{n,k}} \tag{16}$$

$$\frac{\partial J_n}{\partial h_{n,t}} = \frac{\partial J_n}{\partial h_{n,T}} \prod_{k=t}^{T-1} \sigma'_{n,k+1} u_n \tag{17}$$

$$\frac{\partial J_n}{\partial h_{n,t}} = \frac{\partial J_n}{\partial h_{n,T}} u_n^{T-t} \prod_{k=t}^{T-1} \sigma'_{n,k+1} \tag{18}$$

$\sigma'_{n,k+1}$ is the AF in Equation (18).

According to Equation (18), the gradient of the IndRNN directly depends on u_n index, while the traditional RNN gradient is calculated by Equation (19):

$$\frac{\partial J_{n,T}}{\partial h_{n,t}} = \sum_{t=0}^{T} \frac{\partial J_{n,T}}{\partial \hat{y}_{n,T}} * \frac{\partial \hat{y}_{n,T}}{\partial h_{n,T}} \prod_{j=t+1}^{T} \frac{\partial h_{n,j}}{\partial h_{n,j-1}} \tag{19}$$

The traditional RNN determines the gradient by the Jacobian matrix. A slight change in the matrix might cause great fluctuation in the final output. In summary, compared with traditional RNN, IndRNN has many advantages in long-term sequence tasks. First, IndRNN can avoid gradient disappearance and explosion more effectively. Second, IndRNN can process long-term sequences better. Finally, IndRNN has a better explanation (Mussoi, 2021; Shalini et al., 2021; Xu, 2022).

Visual AM is a unique signal processing mechanism of the human brain. Human beings can choose areas of focus by observing global pictures (Mussoi, 2021; Zainab and Majid, 2021). Thereby, they devote more resources to the focus area than ordinary areas to obtain more detailed features while suppressing useless information. The essence of the AM is illustrated in Figure 5.

The input data of the input AM module is $X = \left(x^1, x^2, \cdots, x^k \cdots, x^n\right)$, representing n environmental variable sequences. $\boldsymbol{x}^k = \left(x_1^k, x_2^k, \cdots, x_t^k, \cdots x_T^k\right)$ denotes the kth environment variable sequence, and the time window size is T. The hidden state $\boldsymbol{h}_{t-1}$ of the previous time corresponding to the input of the LSTM unit and cell state $\boldsymbol{C}_{t-1}$-extracted environmental parameter weights are introduced into the input AM module. The calculation process is shown in Figure 6, where e_t^k and α_t^k are calculated by Equations (20) and (21):

$$e_t^k = \boldsymbol{V}_e \tanh\left(\boldsymbol{W}_e \left[\boldsymbol{h}_{t-1} : \boldsymbol{c}_{t-1}\right]\right) + \boldsymbol{U}_e \boldsymbol{x}^k + \boldsymbol{b}_e\Big) \tag{20}$$

$$\alpha_t^k = \frac{\exp\left(e_t^k\right)}{\sum_{i=1}^{n} \exp\left(e_t^i\right)} \tag{21}$$

In Equations (20) and (21), e_t^k represents the weight of the kth environmental parameter at time t. α_t^kk means the value of e_t^k normalized by softmax function. $\boldsymbol{V}_e$, $\boldsymbol{W}_e$, $\boldsymbol{U}_e$, and $\boldsymbol{b}_e$ are the parameter to be trained.

Equation (22) calculates the environmental parameter's vector $\tilde{\boldsymbol{x}}_t$ weighted by α_t^k at time t:

$$\tilde{\boldsymbol{x}}_t = \left(\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, \cdots, \alpha_t^k x_t^k, \cdots \alpha_t^n x_t^n\right) \tag{22}$$

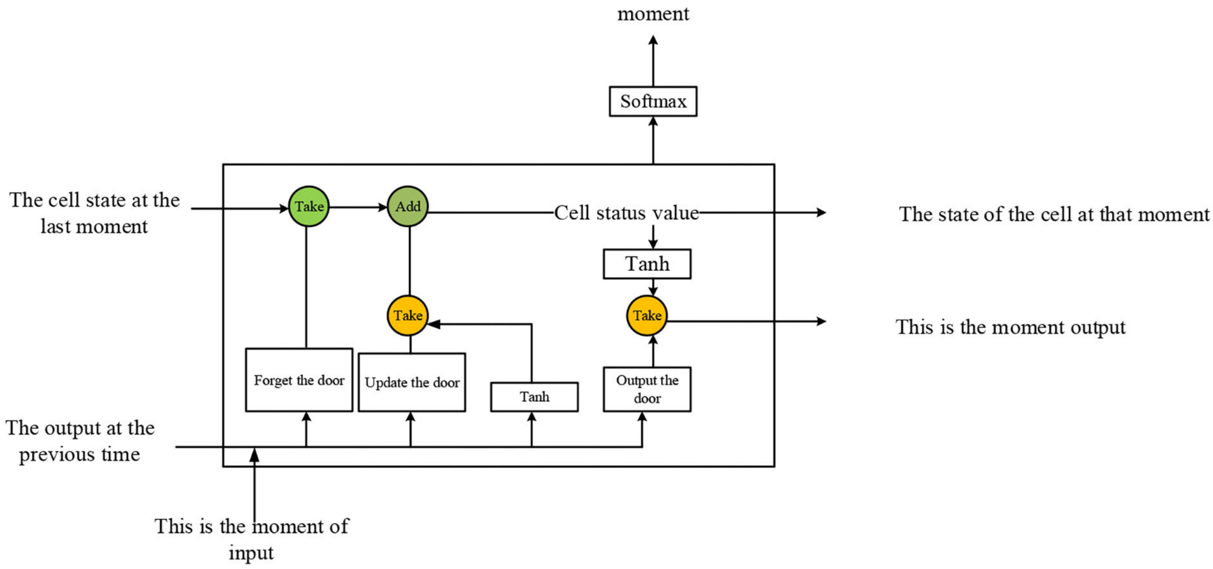

moment
Softmax
The cell state at the last moment
Take
Add
Cell status value
The state of the cell at that moment
Tanh
Take
Take
This is the moment output
Forget the door
Update the door
Tanh
Output the door
The output at the previous time
This is the moment of input

FIGURE 3
Original structure of the LSTM network.

The temporal AM extracts the importance of environmental variables at different times, and its calculation process is shown in Figure 7.

The input data are the output of the encoder module. That is the hidden state $\boldsymbol{h}_i^{'}$ of the weighted value $\tilde{\boldsymbol{x}}_t$ at time i of the environmental data sequence after passing through the LSTM unit. The temporal attention weight is calculated by Equations (23) and (24):

$$l_t^i = \boldsymbol{V}_d \tanh\left(\boldsymbol{W}_d\left[\boldsymbol{d}_{t-1}^{'} : \boldsymbol{s}_{t-1}^{'}\right]\right) + \boldsymbol{U}_d\boldsymbol{h}_i^{'}\right) \tag{23}$$

$$\beta_t^i = \frac{\exp\left(l_t^i\right)}{\sum_{j=1}^{T}\exp\left(l_t^j\right)} \tag{24}$$

In Equations (23) and (24), $1 < i \leqslant T, T$ is the size of the time window. $\boldsymbol{d}_{t-1}^{'}$ and $\boldsymbol{S}_{t-1}^{'}$ are the hidden state and cell state at $(t-1)$ time. $\boldsymbol{V}_d$, $\boldsymbol{W}_d$, and $\boldsymbol{U}_d$ are the parameter to be trained. β_t^i represents the normalized weight of the ith hidden state.

Output $\boldsymbol{O}_t$ of temporal AM module is calculated by Equation (25):

$$\boldsymbol{O}_t = \sum_{i=1}^{T}\beta_t^i\boldsymbol{h}_i^{'} \tag{25}$$

The decoder predicts the beat at time $T+1$ combined with the fully connected layer. $\boldsymbol{Y} = \left(y_1, y_2, \cdots, y_t, \cdots, y_T\right)$ represents the number of beats at each time in the T time window. The specific process is shown in Equations (26) and (27).

$$\tilde{y}_t = \boldsymbol{W}\left[y_t : \boldsymbol{O}_t\right] + \tilde{b} \tag{26}$$

$$\boldsymbol{d}_{t+1}^{'} = f\left(\boldsymbol{d}_t^{'}, \tilde{y}_t\right) \tag{27}$$

In Equations (26) and (27), $\boldsymbol{O}_t$ is the output of the temporal AM module. $\tilde{y}_t$ is a linear transformation of y_t. $\boldsymbol{d}_t^{'}$ represents the hidden state of the decoder at time t. $\boldsymbol{d}_{t+1}^{'}$ denotes the hidden state of the decoder at the time $(t+1)$. Function f is an LSTM computing unit. $\boldsymbol{W}$ and $\tilde{b}$ are parameters to be trained.

Design of interactive humanoid robot system

Interactive robots are dancing robots expressing artistic forms, such as action and language, using physical movements. The dancing robot can recognize the music style and beat for given music and display the right music style and beats. Generally, music style has two targets, namely, beat recognition and action performance. Beat recognition results can be expressed from the actions of interactive robots. Pose estimation is to extract dance movements to form a dance action database from many dance videos. Finally, the interactive robot performs specific actions according to the recognized beat. Dance movement extraction can generate intelligent choreography based on pose estimation technology. By comparison, pose estimation is a basic computer vision technology, the estimation of the human posture or the key points of the human body. The

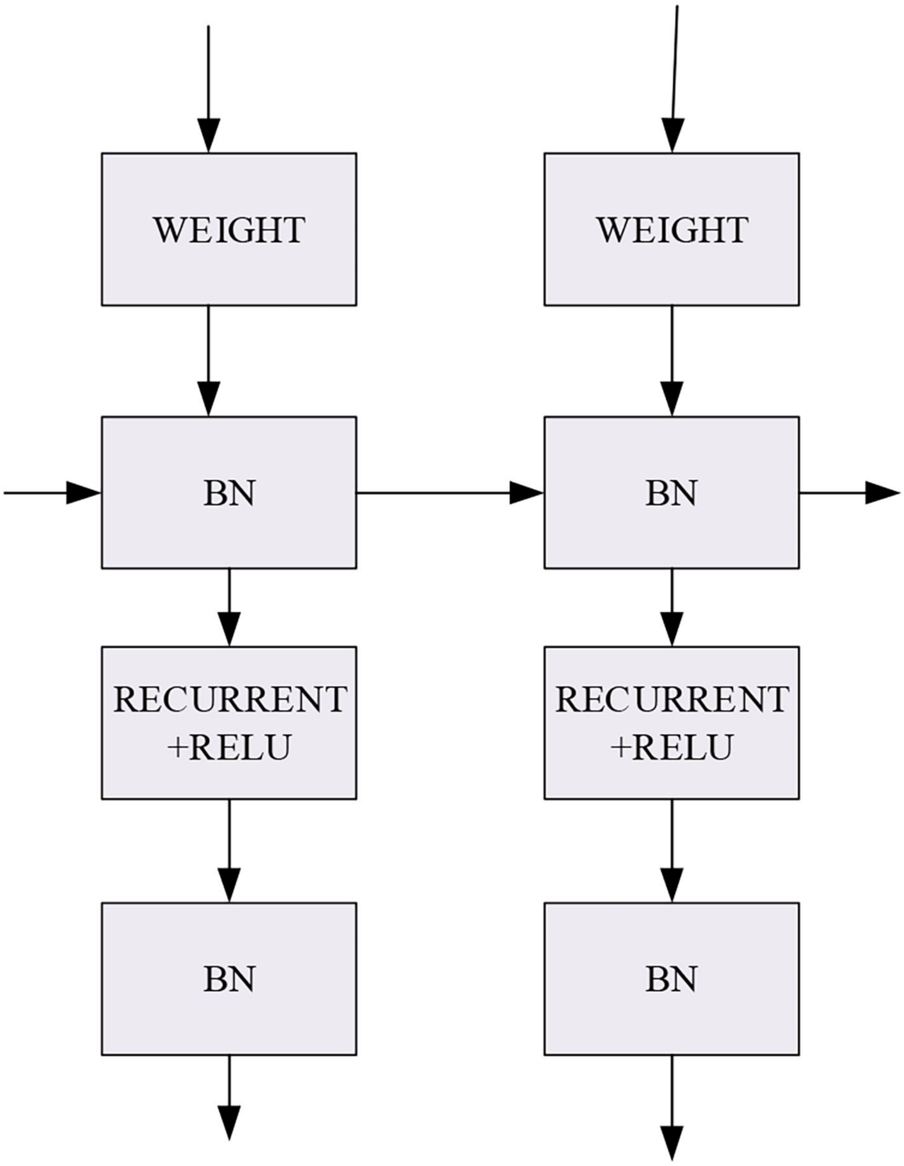
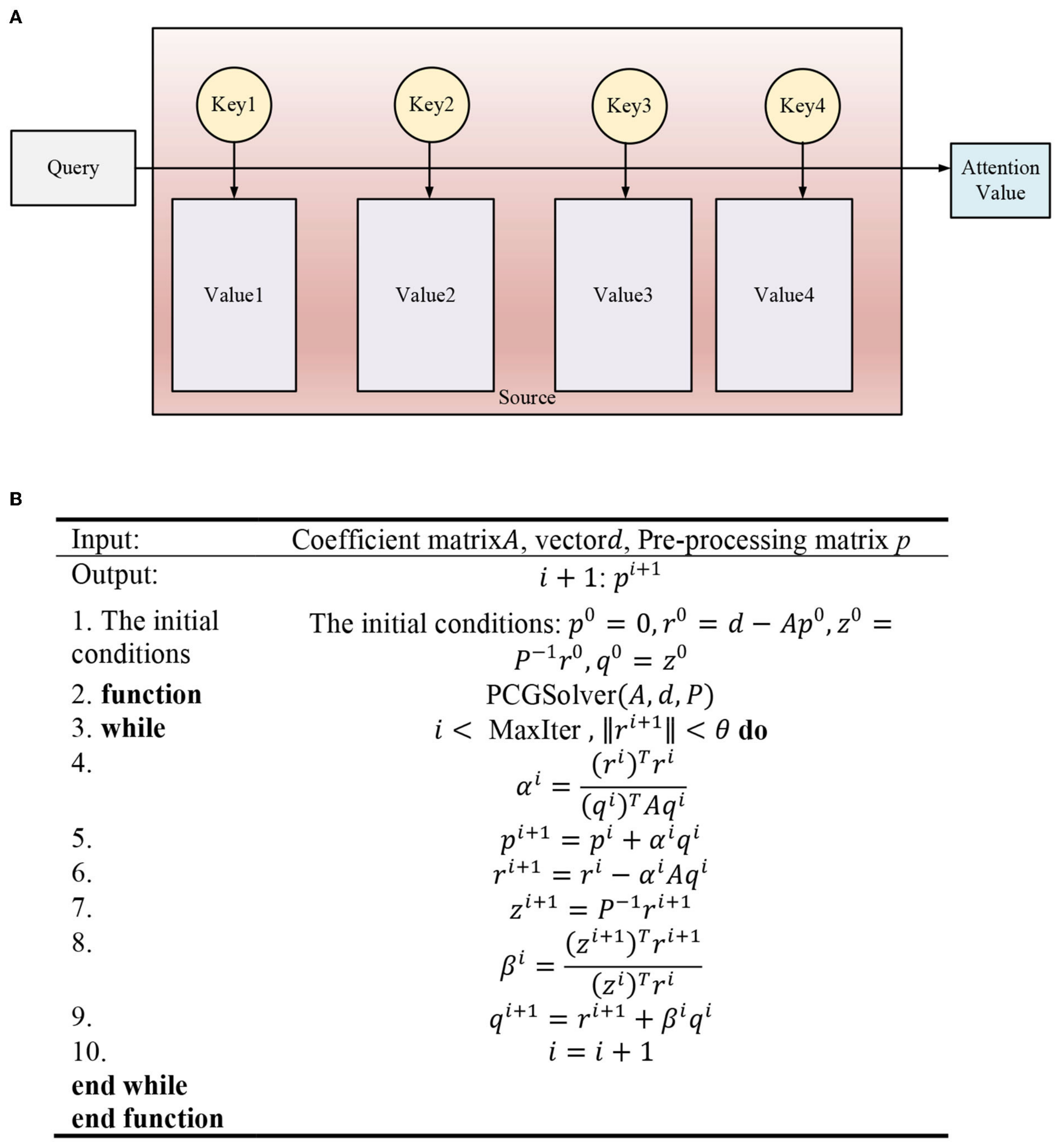


FIGURE 4
IndRNN.

dance action library comprises various stylish dances, divided into ten categories. The dance library is mainly divided into two parts, aiming at robots and non-professionals. The design framework of dancing robots is divided into three parts. The focus is on the design of the dance library, which is divided into three steps. First, the image is detected from the video frame. Second, 2D keypoint information is detected from the image. Third, the music style recognition system converts the 2D keypoint information into 3D information. Fourth, the 3D information is transformed into joint angle information recognizable by the robot motion model. Ultimately, the dance action library is enriched according to the actions obtained from different styles of dance videos. Dance pose estimation and the dance action classification modules in the HCI system are the key to background recognition. Accuracy and response time can evaluate the dance movements and test the feasibility of HCI systems based on dance education and action analysis and recognition.

The deep LSTM network architecture reported here contains a four-layer network structure. The first layer is the input, with 13 neuron nodes. The neurons in the middle two hidden layers are 128 and 32, respectively. The last layer is the output containing ten nodes corresponding to ten music styles.



```
Input:      Coefficient matrix$A$, vector$d$, Pre-processing matrix $p$
Output:     $i + 1$: $p^{i+1}$
The initial conditions
            The initial conditions: $p^0 = 0, r^0 = d - Ap^0, z^0 = P^{-1}r^0, q^0 = z^0$
function    PCGSolver($A, d, P$)
  while     $i <$ MaxIter , $\|r^{i+1}\| < \theta$ do
                $\alpha^i = \frac{(r^i)^T r^i}{(q^i)^T A q^i}$
                $p^{i+1} = p^i + \alpha^i q^i$
                $r^{i+1} = r^i - \alpha^i A q^i$
                $z^{i+1} = P^{-1} r^{i+1}$
                $\beta^i = \frac{(z^{i+1})^T r^{i+1}}{(z^i)^T r^i}$
                $q^{i+1} = r^{i+1} + \beta^i q^i$
                $i = i + 1$
  end while
end function
```

FIGURE 5
Thought of AM. **(A)** Network structure, **(B)** Algorithm pseudocode.

The dataset and the environment configuration of the experiment

The music style library used in this experiment is GTZAN. It is the western music style library used by Tzanetakis in his paper published in 2002, including ten music styles, namely, blues, classical music, country music, disco, hip-hop, jazz, metal, pop music, reggae, and rock, with 100 clips in each style, totalling 1,000 music clips. Each segment is a mono 16-bit *wav* file with a length of 30 s and a sampling rate of 22.05 kHz. Of these, 50 clips are selected from each style for training, 25 clips for verification, and 25 clips for testing.

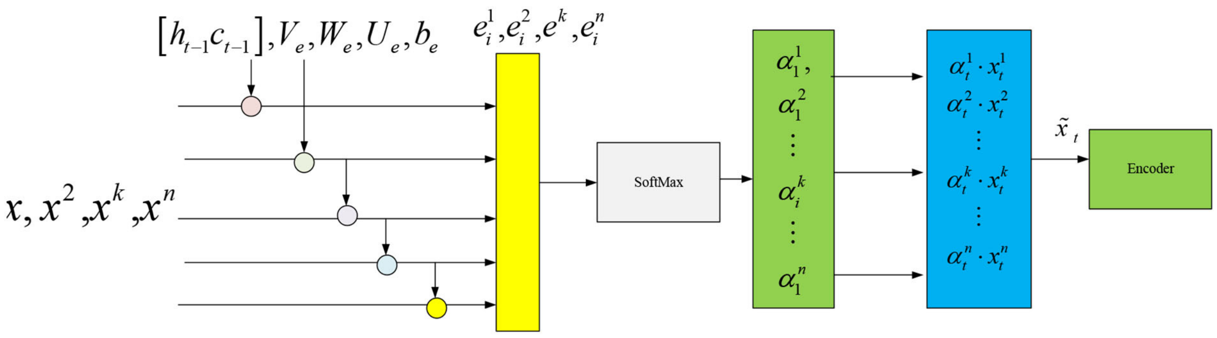

$[h_{t-1}c_{t-1}], V_e, W_e, U_e, b_e$
e_i^1, e_i^2, e^k, e_i^n
x, x^2, x^k, x^n
SoftMax
$\alpha_1^1, \alpha_1^2, \alpha_i^k, \alpha_1^n$
$\alpha_t^1 \cdot x_t^1$, $\alpha_t^2 \cdot x_t^2$, $\alpha_t^k \cdot x_t^k$, $\alpha_t^n \cdot x_t^n$
$\tilde{x}_t$
Encoder

FIGURE 6
Input AM module calculation process.

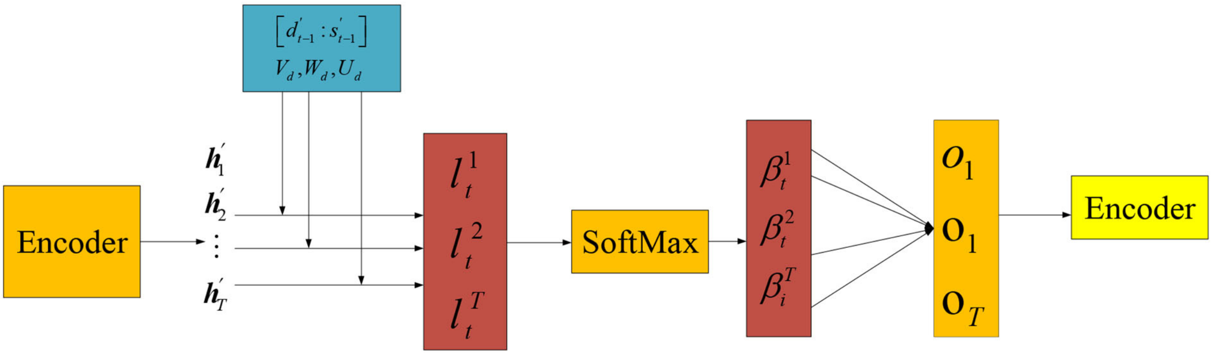


FIGURE 7
Temporal AM.

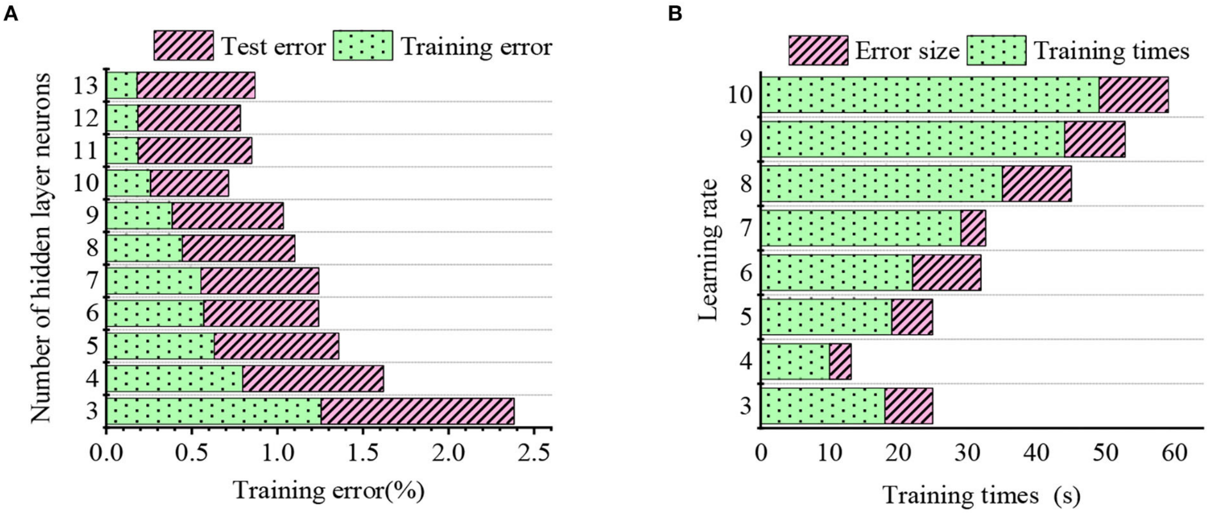


FIGURE 8
Relationship between the hidden layer neurons, the learning rate, and model error. **(A)** Indicates the relationship between the hidden layer neurons and error. **(B)** Shows the relationship between the learning rate and model error.

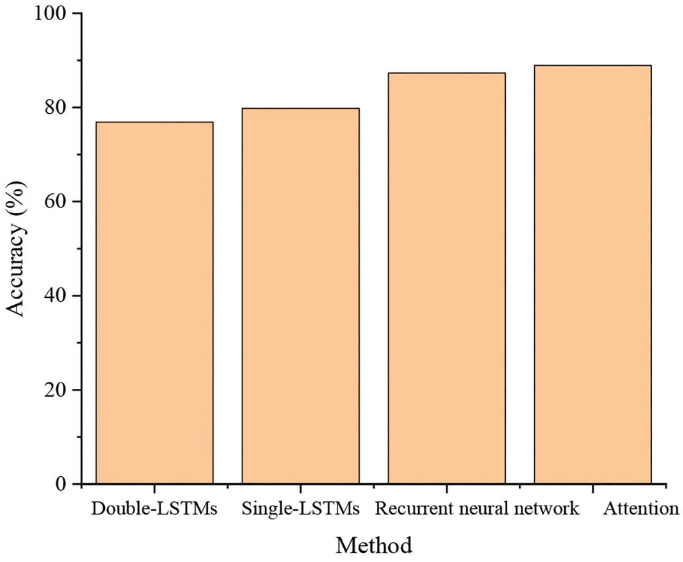


FIGURE 9
Model comparison results.

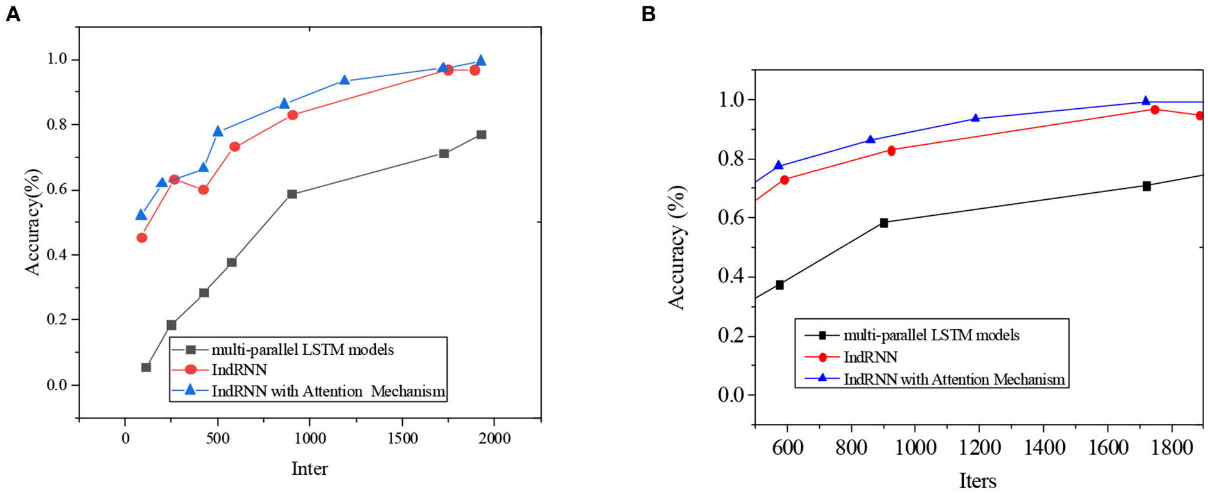


FIGURE 10
Comparison of multi-parallel LSTM models, IndRNN, and AM-IndRNN. **(A)** the first test; **(B)** the second test.

Analysis and discussion of experimental results

Experimental results of model parameter

The relationship between the hidden layer neurons, the learning rate, and model error is outlined in Figure 8.

Overall, the training error decreases when the hidden layer neurons increase from 3 to 13. In particular, when the hidden layer neurons increase from 10, 11, to 12, the training error increases first and then decreases. In conclusion, the optimal hidden layer nodes are 11. On the other hand, the learning rate directly affects the model learning and training efficiencies. Concretely, the prediction error fluctuates greatly given a large learning rate, and the model converges fast. By comparison, a low learning rate means some uncertainties and slow convergence. Finally, the optimal learning rate is determined as 0.01 by observing errors and the number of training.

Functional test of the algorithm

Figure 9 compares the music style and beats recognition accuracy of single LSTM, bi-directional LSTM, IndRNN, and AM-IndRNN.

Comparing the loss and prediction accuracy reveals that multi-parallel models have higher prediction accuracy than single ones. The recognition accuracy of the multi-parallel model reaches 79.8%, higher by 43.8% than the single LSTM model, and the model loss is only 6.85%. Overall, the recognition accuracy and loss of the AM-IndRNN reported here are optimal, reaching 88.9% and 7.48%, respectively. Therefore, the optimized LSTM has higher recognition accuracy and is more applicable for recognizing music styles and beats.

The result of note prediction accuracy is plotted in Figure 10.

In Figure 10, the abscissa means the number of iterations, and the ordinate denotes the note prediction accuracy. Apparently, note prediction accuracy increases with training iterations. Meanwhile, the accuracy of the proposed AM-IndRNN is always higher than that of multi-parallel LSTM models and IndRNN.

The experimental results show that compared with the model proposed by Soufineyestani et al. (2021), the AM-IndRNN reported here has a higher recognition rate on the GTZAN dataset. The experimental results of this paper are compelling. They can well complete the classification of music styles on the GTZAN dataset.

The advantage of the AM-IndRNN reported here is that the recognition accuracy and loss rate are optimal, reaching 88.9% and 0.0748, respectively. Compared with the non-optimized LSTM model, the optimized LSTM model has higher recognition accuracy.

The disadvantage of the AM-IndRNN reported here is that this work limits the research object to single music tone recognition. There are different musical instruments in different countries and nationalities. With the deepening of research, the recognition task may no longer be limited to specific musical instruments or single musical instrument performance. With the continuous expansion of instrumental music, the identification and discrimination work can eventually develop into the performance identification of multiple groups of musical instruments and even the music performance identification with vocal music elements.

Algorithm comparison

Under the same experimental conditions, the proposed AM-IndRNN model, deep Bach model, and BiLSTM- Generative adversarial network (GAN) model's note prediction accuracies are compared in Figure 11.

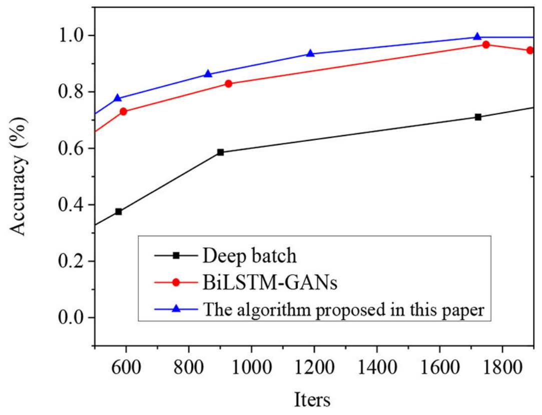


FIGURE 11
Comparison of the deep Bach model, BiSTM-GAN model, and the proposed AM-IndRNN model.

According to Figure 11, when the network iteration = 600, the music note prediction accuracy of the proposed AM-IndRNN, BiLSTM-GAN, and deep Bach model is 73, 65, and 33%, respectively. When the network iteration = 1,400, the accuracy of the above three models is 92, 85, and 53%, respectively. Therefore, with the increase in network iteration, the model's accuracy in predicting notes gradually increases. Meanwhile, the accuracy of the proposed AM-IndRNN to predict notes is always higher than deep Bach and BiLSTM-GAN models.

Conclusion

With the popularity of internet technology and multimedia equipment, online digital music has increased exponentially. Thus, it becomes extremely challenging to manually manage and classify massive numbers of online musical works. At the same time, users' needs for timely and accurate music information retrieval have become imminent. This requires the design of an accurate and effective music style and beats recognition and classification system to manage online music databases. The traditional music style classification methods need a priori knowledge, with complex feature extraction and fewer representative features. In particular, the DL classification model can be used for automatic music style classification. This work focuses on music style classification from audio feature extraction, classifier training, and music style prediction. As a result, a complete automatic music style recognition system is implemented. To do so, the LSTM model is selected over CNN. The sample music is divided into ten different styles. Meanwhile, a hierarchical classification is adopted to improve the classification accuracy further. Specifically, music is classified into strong and weak categories by the LSTM classifier and then divided into multiple subcategories. This new multi-stage classification method is used to classify different music styles. Experiments show that hierarchical multi-step can improve classification accuracy to a certain extent.

However, there are still some deficiencies. The music style recognition system reported here recognizes single notes, but multiple notes generally appear continuously in real music. Hence, future research work will continue to study the problem of multi-tone pitch recognition. In addition, due to the limitation of research time, the number of samples is small, and the experimental samples will continue to be expanded in future research.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

The author confirms being the sole contributor of this work and has approved it for publication.

Funding

This work was supported by the Research on Digital Inheritance and Protection of Intangible Cultural Heritage Music of Ethnic Minorities in Liaoning Province, 2019, Liaoning social science planning fund project (No. L19BMZ007).

Conflict of interest

The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Abbaspour, S., Fotouhi, F., Sedaghatbaf, A., Fotouhi, H., and Linden, M. et al. (2020). A comparative analysis of hybrid deep learning models for human activity recognition. *Sensors*, 20, 89–90. doi: 10.3390/s20195707

Alfaro-Contreras, M., Ríos-Vila, A., Valero-Mas, J. J., Inesta, J. M., and Calvo-Zaragova, J. (2022). Decoupling music notation to improve end-to-end Optical Music Recognition. *Patt. Recogn. Lett.* 158, 157–163. doi: 10.1016/j.patrec.2022.04.032

Briot, J. P., and Pachet, F. (2020). Music generation by deep learning—challenges and directions. *Neural Comput. Appl.* 32, 90–98. doi: 10.1007/s00521-018-3813-6

Chakraborty, S., Dutta, S., and Timoney, J. (2021). The cyborg philharmonic: synchronizing interactive musical performances between humans and machines. *Palgrave Commun.* 8, 51–56. doi: 10.1057/s41599-021-00751-8

Er, M., Cig, H., and Aydilek, B. B. A. (2021). new approach to recognition of human emotions using brain signals and music stimuli. *Appl. Acoust.* 175, 56–59. doi: 10.1016/j.apacoust.2020.107840

Feng, Y., Tao, X., and Lee, E. J. (2021). Classification of shellfish recognition based on improved faster r-cnn framework of deep learning. *Math. Prob. Eng.* 2021, 1–10. doi: 10.1155/2021/1966848

Hernandez-Olivan, C., Pinilla, I. Z., Hernandez-Lopez, C., and Beltran, J. R. et al. (2021). A Comparison of Deep Learning Methods for Timbre Analysis in Polyphonic Automatic Music Transcription. *Electronics*, 10, 810. doi: 10.3390/electronics10070810

Jiang, W. (2020). Evaluation of deep learning models for Urdu handwritten characters recognition. *J. Phys. Conf. Ser.* 1544, 012016. doi: 10.1088/1742-6596/1544/1/012016

Li, Y., and Zheng, W. (2021). Emotion recognition and regulation based on stacked sparse auto-encoder network and personalized reconfigurable music. *Mathematics* 9, 593. doi: 10.3390/math9060593

Liu, H., Zhou, A., Dong, Z., et al. (2021). M-gesture: Person-independent real-time in-air gesture recognition using commodity millimeter wave radar. *IEEE Internet Things J.* 9, 3397–3415. doi: 10.1109/JIOT.2021.3098338

Martin-Gutierrez, D., Penaloza, G. H., Belmonte-Hernandez, A., et al. (2020). A multimodal end-to-end deep learning architecture for music popularity prediction. *IEEE Access*, 2, 1. doi: 10.1109/ACCESS.2020.2976033

Mcauley, J. D., Shen, Y., Smith, T., and Kidd, G. R. (2021). Effects of speech-rhythm disruption on selective listening with a single background talker. *Atten. Percep. Psychophys.* 83, 2229–2240. doi: 10.3758/s13414-021-02298-x

Mirza, A., and Siddiqi, I. (2020). Recognition of cursive video text using a deep learning framework. *IET Image Process.* 14, 345–347. doi: 10.1049/iet-ipr.2019.1070

Mussoi, B. S. (2021). The impact of music training and working memory on speech recognition in older age. *J. Speech Lang. Hear. Res.* 64, 4524–4534. doi: 10.1044/2021_JSLHR-20-00426

Nam, J., Choi, K., Lee, J., Chou, S. Y., and Yang, Y. H., et al. (2019). Deep learning for audio-based music classification and tagging: teaching computers to distinguish rock from bach. *IEEE Signal Process. Mag.* 20, 89–90. doi: 10.1109/MSP.2018.2874383

Parmiggiani, N., Bulgarelli, A., Fioretti, V., Di Piano, A., Giuliani, A., Longo, F., et al. (2021). A deep learning method for AGILE-GRID gamma-ray burst detection. *Astrophys. J.* 914, 67. doi: 10.3847/1538-4357/abfa15

Rajesh, S., and Nalini, N. J. (2020). Musical instrument emotion recognition using deep recurrent neural network. *Procedia Comput. Sci.* 167, 16–25. doi: 10.1016/j.procs.2020.03.178

Shalini, S. K., Jaichandran, R., Leelavathy, S., Raviraghul, R., Ranjitha, J., and Saravanakumar, N. (2021). Facial Emotion Based Music Recommendation System using computer vision and machine learning techiniques. *Turk. J. Comput. Math. Educ.* 12, 912–917. doi: 10.17762/turcomat.v12i2.1101

Soufineyestani, M., Khan, A., and Sufineyestani, M. (2021). Impacts of music intervention on dementia: a review using meta-narrative method and agenda for future research. *Neurol. Int.* 13, 1–17. doi: 10.3390/neurolint13010001

Tan, Y., Sun, Z., Duan, F., Solé-Casals, J., and Caiafa, C. F. (2021). A multimodal emotion recognition method based on facial expressions and electroencephalography. *Biomedical Signal Processing and Control*, 70, 103029. doi: 10.1016/j.bspc.2021.103029

Wagener, G. L., Berning, M., Costa, A. P., Steffgen, G., and Melzer, A. (2021). Effects of emotional music on facial emotion recognition in children with Autism Spectrum Disorder (ASD). *J. Autism Dev. Disord.* 51, 3256–3265. doi: 10.1007/s10803-020-04781-0

Wang, J. G., Shao, H. M., Yao, Y., Liu, J. L., and Ma, S. W. (2021). A personalized feature extraction and classification method for motor imagery recognition. *Mobile Netw. Appl.* 26, 1359–1371. doi: 10.1007/s11036-021-01754-0

Wen, X. (2020). Using deep learning approach and IoT architecture to build the intelligent music recommendation system. *Soft Comput.* 2020, 1–10. doi: 10.1007/s00500-020-05364-y

Wu, R. (2021). Research on automatic recognition algorithm of piano music based on convolution neural network. *J. Phys. Conf. Ser.* 1941, 012086. doi: 10.1088/1742-6596/1941/1/012086

Xu, L., Wen, X., Shi, J., Li, S., Xiao, Y., Wan, Q., et al. (2021). Effects of individual factors on perceived emotion and felt emotion of music: based on machine learning methods. *Psychol. Music* 49, 1069–1087. doi: 10.1177/0305735620928422

Xu, Z. (2022). Construction of Intelligent Recognition and Learning Education Platform of National Music Genre under Deep Learning. *Front. Psychol.* 13, 843427–843427. doi: 10.3389/fpsyg.2022.843427

Zainab, R., and Majid, M. (2021). Emotion recognition based on EEG signals in response to bilingual music tracks]. *Int. Arab J. Inf. Technol.*, 18, 286–296. doi: 10.34028/iajit/18/3/4

Zhou, W., Ma, X., and Zhang, Y. (2020). Research on image preprocessing algorithm and deep learning of iris recognition. *J. Phys. Conf. Ser.* 1621, 8. doi: 10.1088/1742-6596/1621/1/v012008

frontiers | Frontiers in Neurorobotics

TYPE Brief Research Report
PUBLISHED 05 August 2022
DOI 10.3389/fnbot.2022.911469

OPEN ACCESS

EDITED BY
Mu-Yen Chen,
National Cheng Kung University, Taiwan

REVIEWED BY
Yuxuan Zhao,
Institute of Automation (CAS), China
Wei-Che Chien,
National Dong Hwa University, Taiwan

*CORRESPONDENCE
Young Chun Ko
ycko@sehan.ac.kr

RECEIVED 02 April 2022
ACCEPTED 28 June 2022
PUBLISHED 05 August 2022

CITATION
Liu X and Ko YC (2022) The use of deep learning technology in dance movement generation.
Front. Neurorobot. 16:911469.
doi: 10.3389/fnbot.2022.911469

The use of deep learning technology in dance movement generation

Xin Liu[1,2] and Young Chun Ko[3]*

[1]School of Music and Dance, Huaihua University, Huaihua, China, [2]Department of Education, Graduate School of Sehan University, Chonnam, South Korea, [3]Department of Teaching Profession, Sehan University, Chonnam, South Korea

The dance generated by the traditional music action matching and statistical mapping models is less consistent with the music itself. Moreover, new dance movements cannot be generated. A dance movement generation algorithm based on deep learning is designed to extract the mapping between sound and motion features to solve these problems. First, the sound and motion features are extracted from music and dance videos, and then, the model is built. In addition, a generator module, a discriminator module, and a self-encoder module are added to make the dance movement smoother and consistent with the music. The Pix2PixHD model is used to transform the dance pose sequence into a real version of the dance. Finally, the experiment takes the dance video on the network as the training data and trained 5,000 times. About 80% of the dance data are used as the training set and 20% as the test set. The experimental results show that Train, Valid, and Test values based on the Generator+Discriminator+Autoencoder model are 15.36, 17.19, and 19.12, respectively. The similarity between the generated dance sequence and the real dance sequence is 0.063, which shows that the proposed model can generate a dance more in line with the music. Moreover, the generated dance posture is closer to the real dance posture. The discussion has certain reference value for intelligent dance teaching, game field, cross-modal generation, and exploring the relationship between audio-visual information.

KEYWORDS

deep learning, dance movements, action characteristics, sound characteristics, dance sequence

Introduction

Dance is a carrier of performing arts that widely communicates and spreads characteristic culture through human body movements. It is an effective means to reflect cultural diversity and national characteristics. For example, the dance performers' gestures, eyes, and facial expressions, which are ever-changing postures, can represent people's seven emotions and six sensory pleasures. It can even represent natural scenery such as heaven, earth, mountains, and rivers and natural phenomena such as day and night. It is closely connected with music in structure, artistic expression, and interpretation (Minturn and Fowlin, 2020). The continuous development of science and technology provides a broad development platform for deep learning (DL) technology. How to apply this technique to the generation of dance movements is one of the problems that educational circles pay attention to. The dance movements are different due to cultural and ethnic differences, and the follow-up DL technology will also study it. To make dance and music a better fit, choreographers need to create dance movements according to the characteristics of music. Choreographers listen to music, analyze music types, characteristics, inner feelings, or information, and then design corresponding dance movements according to music information. This whole process is called choreography. It is an art of collecting and organizing movement sequences based on music to reflect or express the dancer's thoughts and emotions (Moreu et al., 2020).

The traditional dance generation algorithm usually constructs a music action database containing massive music action pairs. When a music segment is used as input, it will be divided into several small music segments. Each music segment can find the most similar segment in the database. Then, the system can provide corresponding dance action candidates and combine them into a new dance action. In recent years, with the development and popularization of DL, the artificial neural network has been successfully applied to the generation of dance movements. The significant advantage of using DL for dance generation is that it can extract high-level features directly from the original data. In addition, deep neural networks can create new dance movements (Shang and Sun, 2020; Gao and Xu, 2021). Li (2020) proposed a deep neural network, which is trained from zero in an end-to-end manner and generates faces directly from the original speech waveform without any additional identity information. Their model is trained in a self-supervised way by using the audio and video features naturally aligned in the video (Li, 2020). Thomas and Blanc (2021) proposed a cross-modal generation model based on a cyclic generation countermeasure network by considering a cross-modal cyclic generation countermeasure network and combining different generated subnetworks into a network. It further enhanced the effect of mutual generation between music and images (Thomas and Blanc, 2021). Griffin (2021) tried to use an Encoder-Decoder neural network model to learn the corresponding relationship between the original audio and video. This model uses the joint embedding of face and audio to generate a synthetic speech face video frame. The input of the model is a still image and audio segment of the target face. Then, the lip video of the target face is synchronized with the audio output (Griffin, 2021). Elst et al. (2021) combined the convolutional neural network model and generation countermeasure network model to produce a real face sequence synchronized by two networks and input audio (Elst et al., 2021). The limitation of the above dance action generation method is that due to the use of the end-to-end model, the consecutive frame of the generated dance may not be smooth, making the visual effect of the generated dance worse. Moreover, the dance directly generated by the algorithm is often difficult to match the music.

The purpose is to enhance the consistency between the dance generated by the model and the music itself and to increase the smoothness and rationality of the long-time dance sequence. This exploration designs a dance generation algorithm based on DL to extract the mapping between sound and motion features. First, the prosody features and audio beat features extracted from music are regarded as music features. The coordinates of human skeleton key points extracted from dance videos are trained as motion features. Then, the basic mapping between music and dance is realized through the generator module of the model to generate a smooth dance posture. The discriminator module is used to realize the consistency of dance and music. The audio features are more representative through the autoencoder module. The improved Pix2PixHD model is used to transform dance pose sequences into a real-life dance. Finally, the loss function of the model and the generation results of cross-modal dance sequences are analyzed through experiments, which proves that the scheme of the dance automatic generation model based on DL is scientific and effective. The advantage of its future work lies in providing a method reference and rich theoretical basis for the generation of subsequent dance movements and expanding the fields involved in the current DL technology.

Automatic generation algorithm of dance movement based on DL

Related technology

OpenPose is one of the most popular multi-person pose estimation algorithms. Like many bottom-up methods, it first detects the key point coordinates of all people in the image, and then assigns the detected key points to each corresponding person. In practice, the OpenPose network first uses the first few network layers of Visual Geometry Group-19 (VGG-19) to extract features from images. Next, these features are transmitted to two parallel convolution layer branches (Simpson et al., 2014; Kim et al., 2021). The first branch is used to predict 18 confidence maps, each representing a joint in the human skeleton. The second branch predicts a set containing 38 Part Affinity Fields (PAF), describing the connection degree among joints. Next, a series of steps are used to optimize the predicted value of each branch. With the joint confidence graph, a bipartite graph can be formed between each joint pair. Then, the PAF value is used to delete the weak connections in the bipartite graph to detect the key points of the human posture of all people in the graph.

The traditional deep neural network cannot effectively solve the problem of time-series format data. The Recurrent Neural Network (RNN) solves this problem. RNN has a closed loop, which can continuously input the time-series information into the network layer at different times. This circular structure

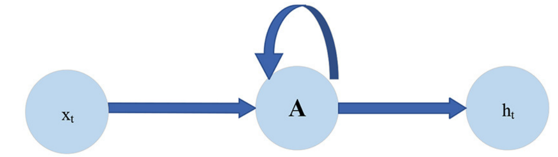


FIGURE 1
Structure diagram of RNN.

shows the close relationship between RNN and time-series data (Wang et al., 2021). Figure 1 is the structural diagram of RNN.

Long-Short Term Memory (LSTM) neural network is a special type of RNN. It is designed to solve the problem of long-term dependence on RNN. The network modules of all RNN internal loops are the same. In ordinary RNN, the structure of the loop module is very simple. However, the special RNN of LSTM is designed to avoid long-term dependence, so different but similar structures are designed, in which each module has different structures. They interact in a quite special way. The key to LSTM lies in the state of cells in each network layer and the horizontal line passing through cells. The cellular structure is similar to a conveyor belt structure. Data run directly on the whole chain, with only a small amount of linear interaction (Noumi et al., 2021; Venskus et al., 2021; Yang and Lee, 2021). Figure 2 displays the details.

LSTM first decides what information to discard from the cell. This decision is controlled by the Sigmoid layer of the forget gate. For each element in the cell state h_{t-1}, the forget gate outputs a number between 0 and 1 by inputting h_{t-1} and x_t, which represents the percentage f_t of information retained from the previous cell state c_{t-1} to the current cell. 1 means "keeping all this information," and 0 means "discarding all this information." The updated equation of f_t reads:

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right) \tag{1}$$

Then, the model will decide which new information to store in the cellular state. This step is divided into two parts. First, h_{t-1} and x_t are used to get *it* through an input gate to determine which information to update. Next, h_{t-1}, x_t, and a tanh layer are used to create a new cell vector candidate value $\tilde{C}_t$. This value may be updated to the cellular state. Equations (2) and (3) are i_t and $\tilde{C}_t$ updated equations.

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right) \tag{2}$$

$$\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right) \tag{3}$$

Next, the old cellular state C_{t-1} is updated to the new status C_t. The updated rule is to multiply the old state C_{t-1} of the previous time by the parameter f_t to forget part of the old cell state information. Then, the input gate added with a part of the candidate cell state $\tilde{C}_t$ information is used to update the state C_t. C_ The updated equation of C_t reads:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4}$$

Finally, after updating the cell state, it is essential to determine the final output according to the input h_{t-1} and x_t. The output will be based on the current cell state and some information will be filtered. First, the output gate of a Sigmoid layer is established to get the judgment conditions and determine which parts of the cell to output. Then, the cell state is passed through the tanh layer, so that the value of the output vector is between -1 and 1, and multiplied by the output gate. In this way, the final output result of the LSTM unit will be obtained. The updated equations read:

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right) \tag{5}$$

$$h_t = o_t \cdot \tanh\left(C_t\right) \tag{6}$$

The attention model is first used in machine translation, and now it has become an important concept and tool in DL. The attention mechanism is an important part of the neural network structure, which has many applications in natural language processing, machine learning, computer vision, and other fields (Han et al., 2021). The Sequence-to-Sequence model is a kind of End-to-End algorithm framework. It is also a transformation model framework from sequence to sequence. It is applied in machine translation, automatic response, and other scenarios. It consists of encoder-decoder architecture. The encoder is an RNN that accepts the input sequence $\{x_1, x_2, \ldots x_i\}$ (i is the length of the input sequence), and encodes it as a vector $\{\mathrm{h}_1, \mathrm{h}_2, \ldots \mathrm{h}_i\}$ of fixed length. The decoder is also an RNN, which takes a fixed-length vector h_i as the input to generate an output sequence $\{y_1, y_2, \ldots y_j\}$, where j is the length of the output sequence. At each time, h_i and S_j represent the hidden state of the encoder and decoder, respectively, which are called candidate state and query state, respectively (Luo et al., 2021; Zhou et al., 2021). Figure 3 is a schematic diagram of the network structure of the Sequence-to-Sequence model.

In the traditional encoder-decoder framework, the encoder must compress all input information into a fixed-length vector h_i. Then, it is passed to the decoder. Using a fixed-length vector to compress the input sequence may lead to a large amount of information loss. Moreover, it cannot model the alignment between input and output sequences. The attention model can effectively solve these two problems. Its core idea is to introduce attention weight α into the input sequence to give priority to the location of relevant information to generate the output the next time. The attention module in the network structure with the attention model is responsible for automatically learning attention weight α_{ij}, which can automatically capture the correlation between h_i and S_j (Gamal et al., 2020). These attention weights are then used to construct the content vector

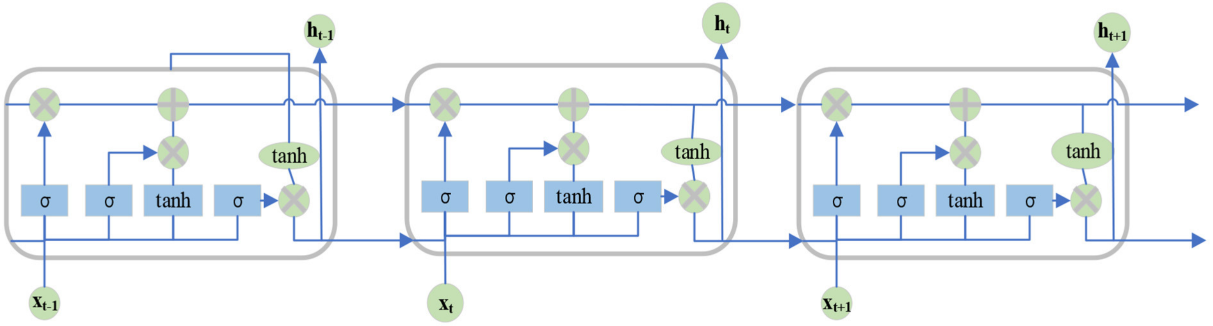


FIGURE 2
Schematic diagram of LSTM network structure.

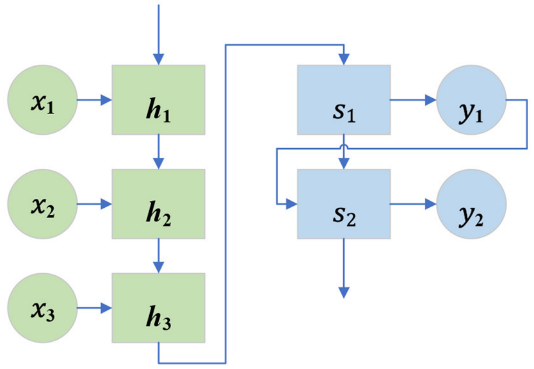


FIGURE 3
Network structure of the Sequence-to-Sequence model.

C, which is passed to the decoder as input. At each decoding position j, the content vector c_j is the weighted sum of all hidden states of the encoder and their corresponding attention weights. Equation (7) displays the details.

$$c_j = \sum_{i=1}^{T} \alpha_{ij} h_i \tag{7}$$

Attention weight is learned by adding the feedforward neural network to the Sequence-to-Sequence architecture. The feedforward network learns a special attention weight α_{ij}, uses h_i and S_j as the input of the neural network, and then learns the value of α_{ij} (Saka et al., 2021).

The autoencoder is a kind of neural network that uses a back propagation algorithm to iterate and make the output value equal to the input value. It first compresses the input information into the latent space representation and then reconstructs this representation into output. It is often used in dimensionality reduction and outlier detection (Samanta et al., 2020). Therefore, an autoencoder is actually a data compression algorithm, and its compression and decompression algorithms are realized through a neural network. It has three characteristics: (1) Data correlation. The autoencoder can only compress data similar to its previous training data. (2) Data loss. Compared with the original input, the output obtained by the autoencoder during decompression will have information loss. Hence, the autoencoder is a data lossy compression algorithm. (3) Automatic learning means that the autoencoder automatically learns from data samples, making it easy to train a specific encoder to input a specified class without any new work (Rahimzad et al., 2021; Verma et al., 2021).

Generative Adversarial Network (GAN) is a representative DL model. It makes the samples generated by the generated network obey the real data distribution through confrontation training (Yang et al., 2021). In GAN, there are two networks for confrontation training. One is the discriminative network. The goal is to judge whether a sample comes from real data or generated data from the network as accurately as possible and distinguish the generated data from the real data as much as possible. The other is the generative network. The goal is to generate real images as much as possible to deceive the discriminative network and make it unable to distinguish the samples from the source (Adamiak et al., 2021; Jeong et al., 2021). The final ideal result is that the model converges, and the discriminative network cannot judge the authenticity of the input samples, that is, the generated network can generate samples in line with the real data distribution.

Generally, image generation using GAN is to generate a random image according to random noise. Although the discriminator will judge the image's authenticity, the generated image is uncontrollable for the user. Pix2Pix model improves this problem. It uses paired data for training and realizes the mutual transformation of a group of images with some semantic relationship (Du et al., 2020). Pix2PixHD generates high-resolution and high-quality images based on Pix2Pix. With a source video and another target video, the goal is to generate a new video, so that the characters in the target video make the

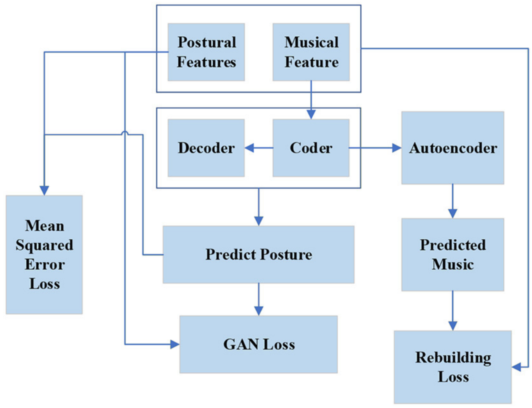


FIGURE 4
Schematic diagram of the network structure of the dance generation model.

same actions as those in the source video (Boni et al., 2020). The task is divided into three stages to complete: the human posture detection stage, the global posture normalization stage, and the mapping from the standardized human posture to the target character. The model uses the open-source human pose detection framework to create the human pose skeleton map from the source video in the pose detection stage. In the global posture standardization stage, the model considers the difference in spatial position between the characters in the source video and the characters in the target video (Shin et al., 2020). Finally, the model designs a countermeasure generation network to learn the mapping from the standardized human posture skeleton map to the real person image of the target character.

Model design

The dance generation algorithm based on the DL model is committed to generating realistic dance movements and matching the music as much as possible. The timing characteristics of dance and audio data should be considered in the model design. The focus of the model design is the combination of dance and music and ensuring a good dance generation effect. First, according to music and dance data characteristics, a feature extraction scheme is designed, including the extraction of prosodic features and rhythm features. The dance generation model of generating dance posture according to audio features is designed based on the feature extraction scheme. The model includes the generator, discriminator, and autoencoder modules. Then, according to the dance pose sequence generated by the model, a scheme of real dance transformation is designed. Figure 4 shows the overall design of the model.

In Figure 4, first, the audio features and motion features are extracted from the dance data, then the audio features are input into the dance generator to obtain the predicted dance posture, and Mean Square Error (MSE) Loss is made with the real dance posture. The reconstructed audio features are obtained through the Autoencoder module and the Loss of audio reconstruction is made. The predicted and real dance posture are sent to the discriminator together, and the discrimination is conducted through the anti-loss training model.

The Sequence-to-Sequence model has good sequence generation ability, so the generator module of the model is composed of a Sequence-to-Sequence model based on an attention mechanism. The generator model of the dance generation model with attention mechanism mainly includes three parts (Figure 5).

The encoder module is composed of multi-layer LSTM to extract long-term audio features. The input is the extracted audio feature vector and rhythm feature vector, and the output is the music context vector. The specific expression reads:

$$f_C = \mathrm{ReLU}\left(F_3 * \mathrm{ReLU}\left(F_2 * \mathrm{ReLU}\left(F_1 * E(X)\right)\right)\right) \tag{8}$$

$$H = \mathrm{EncoderRecurrency}\left(f_c\right) \tag{9}$$

F_1, F_2, and F_3 are three convolution kernels. ReLU is the non-linear activation on each convolution layer. *EncoderRecurrency* represents bidirectional LSTM. After feature extraction, the music feature sequence is first sent to three convolution layers to extract music context information, and then, sent to a bidirectional LSTM to generate the hidden state H of the encoder.

The encoder module and decoder module can calculate the hidden layer state $H = \{\mathrm{h}_1, \mathrm{h}_2, \ldots \mathrm{h}_i\}$ and hidden layer state $S = \{\mathrm{M}_1, \mathrm{M}_2, \ldots \mathrm{M}_j\}$, respectively. h_i and M_j are the hidden states of the coding layer and the hidden states of the decoding layer at each time step. Then, the attention weight is calculated and assigned to the music context vector to obtain the audio feature vector after assigning the weight. Attention calculation occurs at each decoder time step. The custom score function is calculated for the target hidden state and each source state to generate attention weight. To reduce the potential sub-sequence repetition or omission in the decoding process, the cumulative attention weight of the previous decoding process is considered as an additional feature to keep the model consistent when moving forward along the input sequence. Therefore, the model uses the attention mechanism based on location sensitivity to expand the previous attention mechanism. Equation (10) displays the details:

$$e_{i,j} = score\left(s_i, c\alpha_{i-1}, h_j\right) = v_a^T \tanh\left(Ws_i + Vh_j + Uf_{i,j} + b\right) \tag{10}$$

$f_{i,j}$ is the position feature obtained by convolution of the previous attention weight α_{i-1}. v_a^T, W, V, U, and b are the parameters to be trained. Through the attention calculation

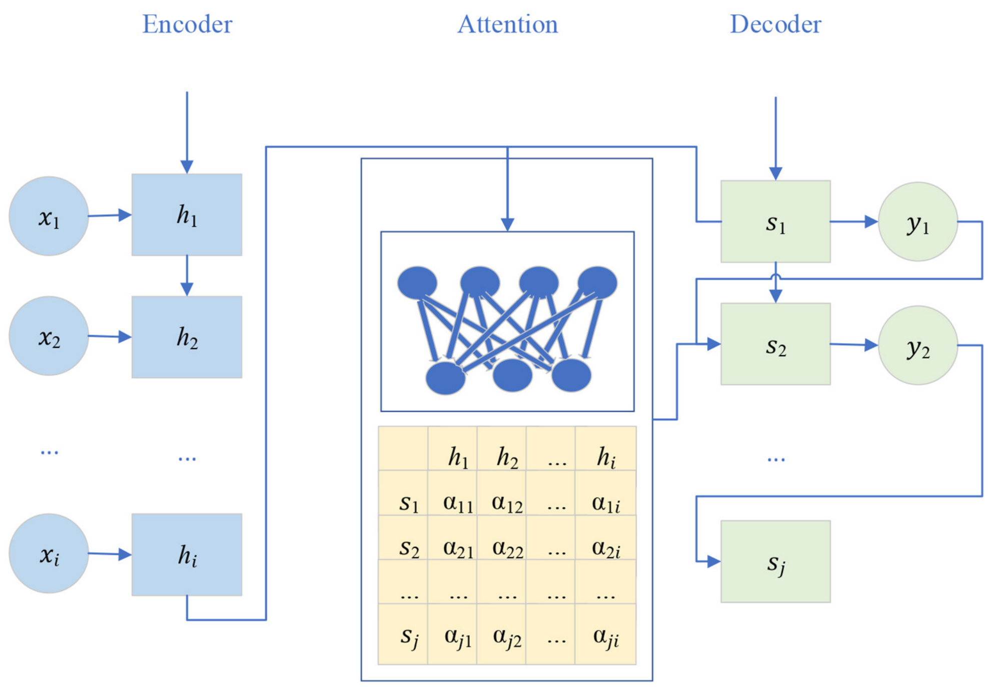

Encoder
Attention
Decoder
x_1
h_1
x_2
h_2
…
…
x_i
h_i
h_1 h_2 … h_i
s_1 α_{11} α_{12} … α_{1i}
s_2 α_{21} α_{22} … α_{2i}
… … … … …
s_j α_{j1} α_{j2} … α_{ji}
s_1
y_1
s_2
y_2
…
s_j

FIGURE 5
Schematic diagram of Sequence-to-Sequence model based on attention mechanism.

module, the attention weight between the hidden layer states h$_i$ and s$_j$ can be obtained.

The music training dataset is M = {M$_1$, M$_2$ … M$_n$}. M$_i$ is a sequence of audio feature vectors. The dance training dataset corresponding to music is $P = \{P_1, P_2 \ldots P_n\}$. P_i is the dance posture feature vector corresponding to M$_i$. {M$_i$,P_i} is the training data of a sample pair. M and P are obtained from live dance videos through specific feature extraction schemes. The goal of the model is to train a dance generator G and realize the mapping relationship between M and P. Equation (11) displays the details:

$$\mathcal{L}_{MSE}(G) = \frac{1}{N}\sum_{i=1}^{N} \parallel P_i - G\left(M_i\right) \parallel^2 \tag{11}$$

Equation (11) shows the specific process of generator training. First, the model is trained on {M$_i$,P_i} and MSE Loss is calculated between the dance G(M$_i$) generated by the model and the real dance P_i. After the training, the corresponding dance posture sequence can be obtained for any given music input.

A human posture sequence is a time-series that changes constantly. Therefore, the difference between the front and back frames of human posture can reflect the change process of action, not just considering the fixed posture action in time sequence. Such a change process can better represent the characteristics of dance posture. The discriminator input is set as posture vector $P = \{P_1, P_2 \ldots P_n\}$, frame difference vector of the pose before and after is set as M={P_1-P_2, P_2-P_3, … P_{n-1}-P_n} and the audio feature vector is set as M = {M$_1$, M$_2$ … M$_n$}. They are input into the discriminator together to judge whether the combination of pose vector and corresponding audio vector is true. Equation (12) shows the details:

$$\begin{gathered}\mathcal{L}_{GAN}(G, D) = \mathbb{E}_{(P,M)}[\log D((P, M))]+ \\ \tfrac{1}{2}\mathbb{E}_M[\log(1 - D((G(M), M))) + \log(1 - D((W, M)))]\end{gathered} \tag{12}$$

Equation (12) reveals that the generator G receives the music feature vector M and generates the predicted dance posture through C, which is recorded as $G(M)$. In the training phase, the generator G and the discriminator D are trained alternately in turn. D is a discriminative network, which is used to judge whether the generated dance is consistent with the music. $((G(M), M)$ sample pairs are set as a pair of negative samples. M is the other real dance pose that does not match the current audio. (W, M) is also set as a pair of negative samples. (P, M) is the sample pair composed of music and its corresponding real dance posture vector. It is set as a positive sample to train the discriminator. The output $D\ (G(M), M)$ represents the probability of mutual fit between the predicted dance and music. The closer the value of $D\ (G(M), M)$ is to 1, the better the fit between music and dance. The closer the output value is to 0, the more discordant the generated dance is with the music.

An Autoencoder module is added to the model to combine the music and dance posture more closely. The

TABLE 1 Selected dance types, nationalities, quantities and characteristics.

Type	Detailed	Quantity	Style characteristics
Mongolian dance	Drum dance	30	Bold and generous
	Chopsticks dance	30	
Hui dance	Banquet song	25	Cheerful and flexible
	Dancing flowers	25	
Uygur dance	Sanam	35	Warm, bold and delicate
	Dorang dance	35	
Yi dance	Cigarette box Dance	31	Positioning with “fire”
	Music and dance	31	
Zhuang dance	Shigong dance	29	Some labor actions
	Pole dance	29	

audio autoencoder module is a network designed for audio reconstruction. In the encoder stage of the generator module, the input audio features are encoded. When the coding is completed, in addition to the prediction of dance posture, the audio features are input into an autoencoder module and regressed with the audio features before coding. In this way, because the model’s loss function needs to consider the regression loss of dance posture and the regression loss of audio reconstruction simultaneously, it can ensure that the dance posture will be more consistent with the music when the model is predicted. Moreover, the encoded audio feature vector can better represent the original audio with the model’s training. The details are as follows:

$$f_i = Encoder\left(\mathrm{Concat}\left(M_i, B_i\right)\right) \tag{13}$$

$$M_i^{\sim} = Decoder\left(f_i\right) \tag{14}$$

The audio autoencoder takes the audio feature vector M*i* and the beat feature vector M*i* as inputs. The LSTM network structure is used to encode the audio features. Besides, the same and symmetrical network structure is used for decoding. The basic model structure is shown in the equation. M_i is the original audio feature vector, B_i is the rhythm eigenvector, and $M_i^{\sim}$ is the reconstructed audio feature vector. f is a dimension-reduced audio representation extracted from the audio feature vector. *Concat* is a vector splicing operation. *Encoder* and *Decoder* are neural networks to be learned. The loss function of the audio autoencoder is defined as the Euclidean distance between the original audio feature M_i and predicted audio features $M_i^{\sim}$. The Equation reads:

$$\mathcal{L}_{Recon}\left(\, Encoder,\ Decoder\,\right) = \frac{1}{N}\sum_{i=1}^{N} \parallel M_i^{\sim} - M_i \parallel^2 \tag{15}$$

To sum up, equation (16) displays the optimization objective of the model:

$$\min_G \max_D \mathcal{L}_{GAN}(G, D) + \lambda_1 \mathcal{L}_{MSE}(G) + \lambda_2 \mathcal{L}_{Recon}(Encoder,\ Decoder) \tag{16}$$

λ_1 and λ_2 are training parameters.

Experimental design

The dance synthesis results under different models and parameter settings are evaluated to verify the feasibility and effectiveness of the dance generation model. This experiment takes the dance data video screened and downloaded from the network as the experimental dataset for experiment and result analysis because the dance types of the platform are diverse and the research results will be more accurate. In addition, it is more convenient and quicker to choose. In the process of downloading, dance videos will also be screened to ensure that each type of dance is involved, so that their cultural characteristics can be captured from the dance. In the generator training phase, the model training times are set to 5,000 times. The model input dimension is 35, the encoder convolution layers are 3, and the maximum length of each convolution core is 5. The decoder dimension is 1,024, the prenet dimension is 256, the learning rate is set to 0.001, the gradient clipping threshold is set to 1, the weight attenuation is set to 1e-6, the batch size is set to 40, the seqlen is set to 125, and the optimizer is Adam. In the discriminator training phase, it is set that the discriminator is trained once every three training rounds of the generator. The learning rate of the discriminator is 0.001, the weight attenuation is 1e-6, and the optimizer uses Adam. The training set uses 80% of the dance data, and the test set uses 20% of the dance data. The experimental results are analyzed based on the model loss function and cross-modal dance sequence generation results. Table 1 shows the specific dance types selected.

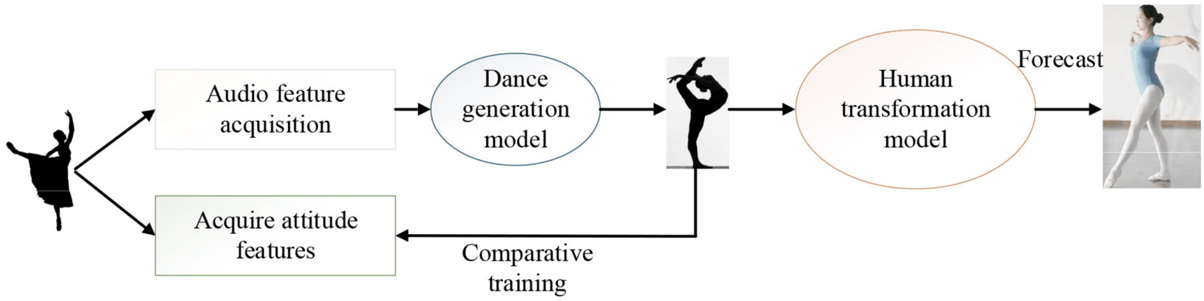

Audio feature acquisition
Dance generation model
Human transformation model
Forecast
Acquire attitude features
Comparative training

FIGURE 6
Experimental process of dance movement generation.

Figure 6 shows the specific experimental process.

Experimental results and analysis

Analysis results of the model loss function

In the stage of audio feature extraction, the influence of different data processing methods on the final loss function is analyzed, and the specific results are shown in Figure 7.

Figure 7 indicates that without filtering out the wrong data, the loss value of using the rhythm feature will be greater than that of not using the rhythm feature. This is also reasonable, because with the increase of the dimension of audio data, the noise may also increase, but the rhythm characteristics may be useful for the final generation effect of dance. The effect of using an interpolation function to supplement the missing values of dance data is better than not using it. It is very important to filter out the wrong data and it will enhance the final result.

In the model-building stage, on the premise of filtering out the wrong data, using rhythm characteristics and interpolating the missing values, the impact of different modules on the model loss is analyzed. The results are shown in Figure 8.

As Figure 8 presents, the loss value with disciplinarian is slightly greater than that without disciplinarian. This can be expected because the confrontation training will make the generator generate a new sequence that does not necessarily match the original dance posture sequence, which may deviate from the real value of the dance posture predicted by optimizing Euclidean loss only. After the comparative analysis of the above three types of models, it can be seen that the autoencoder module has a significant impact on the loss of the model.

Analysis of generation results of cross-modal dance sequences

First, the training set of the dance dataset is preprocessed. According to the settings of seqlen = 125 and batchsize = 40, music and dance sequences are segmented. Then, the segmented audio sequence features are extracted and projected into a dictionary together with their corresponding dance sequence. The dance generation model is used to generate dance from the music after segmentation. If the music segment does not appear in the previous training data, the most similar music segment is found in the audio feature vector dictionary obtained by the K-means algorithm and K-nearest neighbor algorithm. For K-means clustering, the number of clusters is set as k = 5. The similarity measure of the audio feature vector is Euclidean Distance. The similarity between the corresponding dance sequence and the generated dance sequence is calculated. If the music segment appears in the previous training data, the similarity between the music-generated dance sequence and the real dance sequence is directly calculated to measure the actual generation effect of the dance. Figure 9 shows the similarity between robot-generated dance and real human dance.

In Figure 9, the similarity between the robot-generated dance and the real human dance shows that the Generator+Discimінator+Autoencoder model has the best effect on the actual human dance generation. The dance effect of the network with a discriminator is better than that of the network without a discriminator. Moreover, the generator is superior to the generator of the LSTM series. To sum up, the dance generation model scheme based on Generator+Discriminator+Autoencoder can effectively extract music features and generate dance pose sequences that fit the music, which is feasible and effective.

To verify the actual effect of the design model, 200 dancers are recruited to evaluate the effect of dance movement generation. Figure 10 shows the specific results.

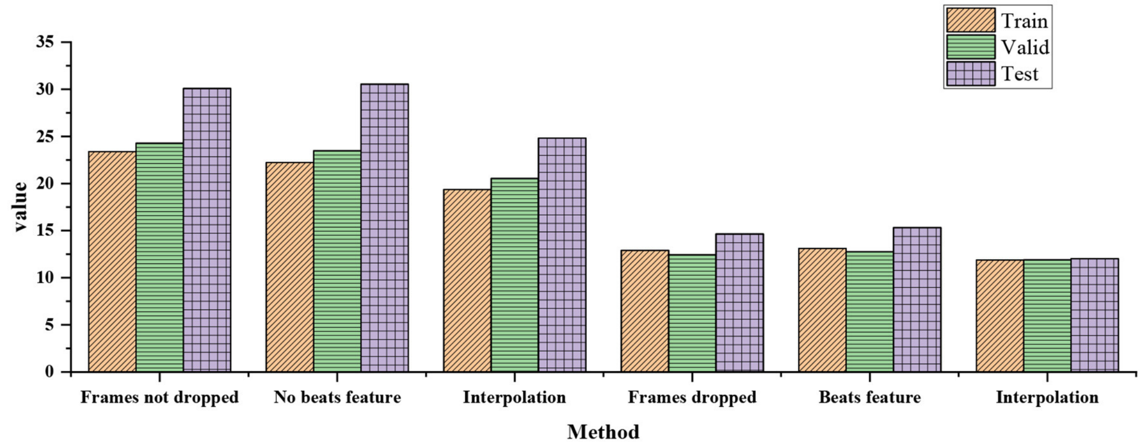


FIGURE 7
Influence of different data processing methods on final loss function in audio feature extraction stage.

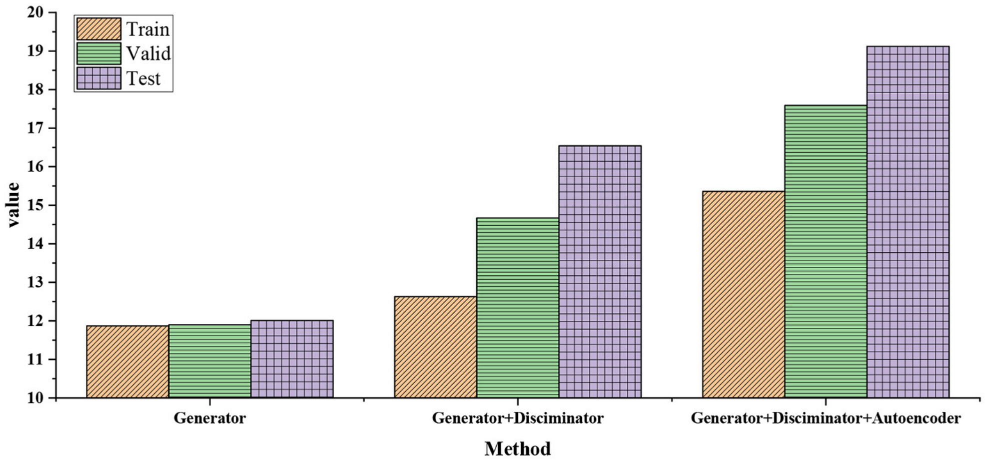


FIGURE 8
Impact of different modules on model loss in model building stage.

Figure 10 shows that 36.26% of the dancers think the model has a very good dance movement generation effect; 57.61% of the dancers think that the model has a better generation effect; 5.13% of the dancers hold that the model has a general generation effect; and 1% of the dancers think that the generation effect is poor. After that, 1% of the dancers are paid a return visit. They believe that the reason why they think the generation effect is poor is that the video generated by the model will get stuck due to the impact of the network environment. However, most of the dancers recognize the designed model. Therefore, in the follow-up study, it is necessary to strengthen the improvement of the network environment.

Discussion

With the development of science and technology, a variety of advanced technologies has been applied to the generation of dance movements. From the perspective of DL, this work studies the influence of this technology on the generation of dance movements. First, starting with the real dance scene, this work discusses and designs a DL-based dance action generation algorithm, which can extract the background music and dancers' actions. Second, it establishes the corresponding model. The selected data are applied to the model for research and analysis. Fink et al. (2021) tried to use the delayed LSTM artificial neural network to generate key points synchronized to

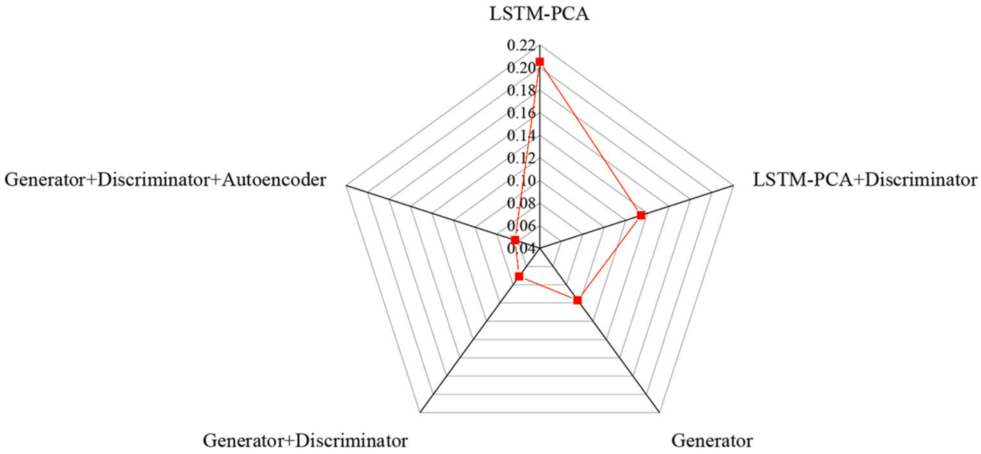


FIGURE 9
Similarity between robot-generated dance and real human dance.

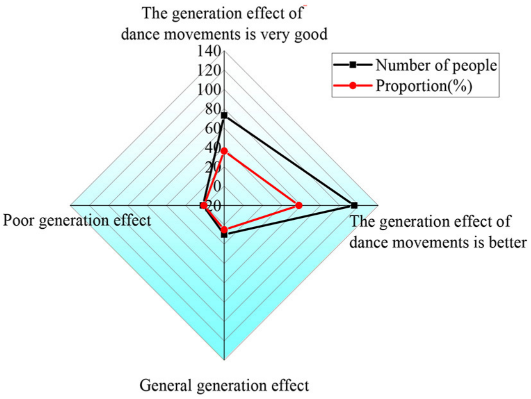


FIGURE 10
The dancer's evaluation of the dance action generation effect.

audio, and then, another network was used to generate video frames based on key points. It is a network architecture that takes any text as input, and then, generates the corresponding voice and synchronous photo display pure synchronous video. Different from other published methods, their methods are only composed of fully trainable neural networks and do not rely on any traditional computer graphics methods (Fink et al., 2021). Wang and Tong (2022) further proposed a time consistency method for dynamic pixel loss. Compared with the direct audio-to-image method, this cascading method avoids fitting the false correlation between audio-visual signals independent of speech content. To avoid these pixel jitter problems, they also emphasized the network's attention to audio-visual related areas, and proposed a new attention mechanism with dynamically adjustable pixel-level loss. In addition, to generate clearer images with well synchronized facial motion, they proposed a new regression-based discriminator structure, which considers sequence-level and frame-level information (Wang and Tong, 2022). The above two scholars discussed their methods of generating dance movements from different angles. This study draws lessons from the effective methods and designs a model based on Generator+Discriminator+Autoencoder to generate dance movements. This study has a certain reference value for intelligent dance teaching, cross-modal generation, and exploring the relationship between audio-visual information.

Conclusion

In the field of music-driven computer dance motion generation, there are many problems in the traditional music motion matching model and statistical mapping model. From the perspective of DL technology, this study studies the generation of dance movements, and mainly draws the following conclusions. (1) Taking 80% of the dance data as the training set and 20% of the dance data as the test set, it is found that in terms of the loss function, although the loss value based on Generator+Discriminator+Autoencoder model is higher than that of pure generator model, it can generate dance more in line with music. (2) In the evaluation of dance posture sequence, compared with other models, the dance posture generated by the model in this work is the closest to the real dance posture, whether for the music in the training set or the music in the test set. Experiments show that the scheme of the DL-based dance automatic generation model is scientific and effective. However, the amount of data selected here is less and there are some errors in the relevant test of the data. In addition, this study only analyzes the research of DL technology in dance movement generation but does not discuss the application of other technologies. In the future, a larger dance data set will be established to expand the training data, thereby, training a more representative dance movement generation model.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

Both authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

Funding

This work was supported by the Hunan Province Philosophy and Social Science Fund Project Research on the Art of Dance in Shangheyang Opera in Huaihua and its Creation and Communication (No. 19YBA269).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Adamiak, M., Bdkowski, K., and Majchrowska, A. (2021). Aerial imagery feature engineering using bidirectional generative adversarial networks: a case study of the pilica river region, poland. *Remote Sens*. 13, 306. doi: 10.3390/rs13020306

Boni, K., Klein, J., Vanquin, L., Wagner, A., Lacornerie, T., Pasquier, D., et al. (2020). MR to CT synthesis with multicenter data in the pelvic area using a conditional generative adversarial network. *Phys. Med. Biol.* 65, 7. doi: 10.1088/1361-6560/ab7633

Du, W. L., Zhou, Y., Zhao, J., and Tian, X. (2020). K-means clustering guided generative adversarial networks for SAR-optical image matching. *IEEE Access*. 8, 217554–217572. doi: 10.1109/ACCESS.2020.3042213

Elst, O., Vuust, P., and Kringelbach, M. L. (2021). Sweet anticipation and positive emotions in music, groove, and dance. *Curr. Opin. Behav. Sci*. 39, 79–84. doi: 10.1016/j.cobeha.2021.02.016

Fink, B., Bläsing, B., Ravignani, A., and Shackelford, T. K. (2021). Evolution and functions of human dance. *Evol. Hum. Behav*. 42, 351–360. doi: 10.1016/j.evolhumbehav.2021.01.003

Gamal, S., El-Nemr, M. K., and El-Kassas, A. M. (2020). Enhancement of Bi-objective function model to master straight- line facilities sequences using frequency from-to chart. *J. Facil. Manag*. 19, 12. doi: 10.1108/JFM-08-2020-0059

Gao, Y., and Xu, D. (2021). Application of posture recognition service system based on information fusion smart sensor in dance training. *J. Sens*. 2021, 1–7. doi: 10.1155/2021/4284249

Griffin, J. (2021). Commentary: music's digital dance: singing and swinging from product to service. *J. Mark*. 85, 223–226. doi: 10.1177/0022242920972704

Han, H., Choi, C., Jung, J., and Kim, H. S. (2021). Deep learning with long short term memory based sequence-to-sequence model for rainfall-runoff simulation. *Water*. 13, 437. doi: 10.3390/w13040437

Jeong, Y. J., Park, H. S., Ji, E. J., Yoon, H. J., Jeon, K., Cho, K., et al. (2021). Restoration of amyloid PET images obtained with short-time data using a generative adversarial networks framework. *Sci. Rep*. 11, 4825. doi: 10.1038/s41598-021-84358-8

Kim, W., Sung, J., Saakes, D., Huang, C., and Xiong, S. (2021). Ergonomic postural assessment using a new open-source human pose estimation technology (OpenPose). *Int. J. Ind. Ergon*. 84, 103164. doi: 10.1016/j.ergon.2021.103164

Li, F. (2020). Feasibility study on the "six in one" teaching mode in line dance. *Asian Soc. Sci*. 16, 138. doi: 10.5539/ass.v16n7p138

Luo, Y., Tang, N., Li, G., Tang, J., Chai, C., and Qin, X. (2021). Natural Language to visualization by neural machine translation. *IEEE Trans. Vis. Comput Graph*. 28, 217–226. doi: 10.1109/TVCG.2021.3114848

Minturn, M., and Fowlin, K. (2020). Embodied liberation: envisioning and manifesting a better world through dance. *Res. Dance Phy. Educ*. 4, 1–7. doi: 10.26584/RDPE.2020.12.4.2.1

Moreu, F., Maharjan, D., Zhu, C., and Wyckoff, E. (2020). monitoring human induced floor vibrations for quantifying dance moves: a study of human–structure interaction. *Front. Built Environ*. 6, 36. doi: 10.3389/fbuil.2020.00036

Noumi, T., Inoue, S., Fujita, H., Sadamitsu, K., Sakaguchi, M., Tenma, A., et al. (2021). Epitope prediction of antigen protein using attention-based LSTM network. *J. Inform. Process*. 29, 321–327. doi: 10.2197/ipsjjip.29.321

Rahimzad, M., Homayouni, S., Naeini, A. A., and Nadi, S. (2021). An efficient multi-sensor remote sensing image clustering in urban areas via boosted convolutional autoencoder (BCAE). *Remote Sens*. 13, 18. doi: 10.3390/rs131 32501

Saka, K., Kakuzaki, T., Metsugi, S., Kashiwagi, D., Yoshida, K., Wada, M., et al. (2021). Antibody design using LSTM based deep generative model from phage display library for affinity maturation. *Sci. Rep*. 11, 5852. doi: 10.1038/s41598-021-85274-7

Samanta, S., O'Hagan, S., Swainston, N., Roberts, T. J., and Kell, D. B. (2020). VAE-sim: a novel molecular similarity measure based on a variational autoencoder. *Molecules*. 25, 3446. doi: 10.3390/molecules25153446

Shang, R. A., and Sun, Y. (2020). So little time for so many ties: fit between the social capital embedded in enterprise social media and individual learning requirements. *Comput. Hum. Behav*. 120, 106615. doi: 10.1016/j.chb.2020.106615

Shin, G., Moon, Y. J., Park, E., Jeong, H. J., Lee, H., and Bae, S. H. (2020). Generation of high-resolution solar pseudo-magnetograms from Ca ii K images by deep learning. *Astrophys. J. Lett*. 895, 7. doi: 10.3847/2041-8213/ab9085

Simpson, T. T., Wiesner, S. L., and Bennett, B. C. (2014). Dance recognition system using lower body movement. *J. Appl. Biomech*. 30, 147–153. doi: 10.1123/jab.2012-0248

Thomas, N., and Blanc, V. (2021). Break it then build again: an arts based duoethnographic pilot reconstructing music therapy and dance-movement therapy histories.the arts in *Psychotherapy* 73, 101765. doi: 10.1016/j.aip.2021.101765

Venskus, J., Treigys, P., and Markeviiut, J. (2021). Unsupervised marine vessel trajectory prediction using LSTM network and wild bootstrapping techniques. *Nonlinear Anal. Model. Control*. 26, 718–737. doi: 10.15388/namc.2021.26.23056

Verma, S., Singh, S., and Majumdar, A. (2021). Multi-label LSTM autoencoder for non-intrusive appliance load monitoring. *Electr. Power Syst. Res*. 199, 107414. doi: 10.1016/j.epsr.2021.107414

Wang, S., and Tong, S. (2022). Analysis of high-level dance movements under deep learning and internet of things. *J Supercomput*. 59, 1–23. doi: 10.1007/s11227-022-04454-y

Wang, Z., Zhang, T., Shao, Y., and Ding, B. (2021). LSTM-convolutional-BLSTM Encoder-decoder network for minimum mean-square error approach to speech enhancement. *Appl. Acoust*. 172, 107647. doi: 10.1016/j.apacoust.2020.107647

Yang, B., Kang, Y., Yuan, Y. Y., Huang, X., and Li, H. (2021). ST-LBAGAN: spatio-temporal learnable bidirectional attention generative adversarial networks for missing traffic data imputation. *Knowl. Based Syst*. 215, 106705. doi: 10.1016/j.knosys.2020.106705

Yang, G., and Lee, B. (2021). Utilizing topic-based similar commit information and CNN-LSTM algorithm for bug localization. *Symmetry*. 13, 406. doi: 10.3390/sym130 30406

Zhou, P., Zhou, G., Wu, D., and Fei, M. (2021). Detecting multi-stage attacks using sequence-to-sequence model. *Comput. Secur*. 105, 102203. doi: 10.1016/j.cose.2021.102203

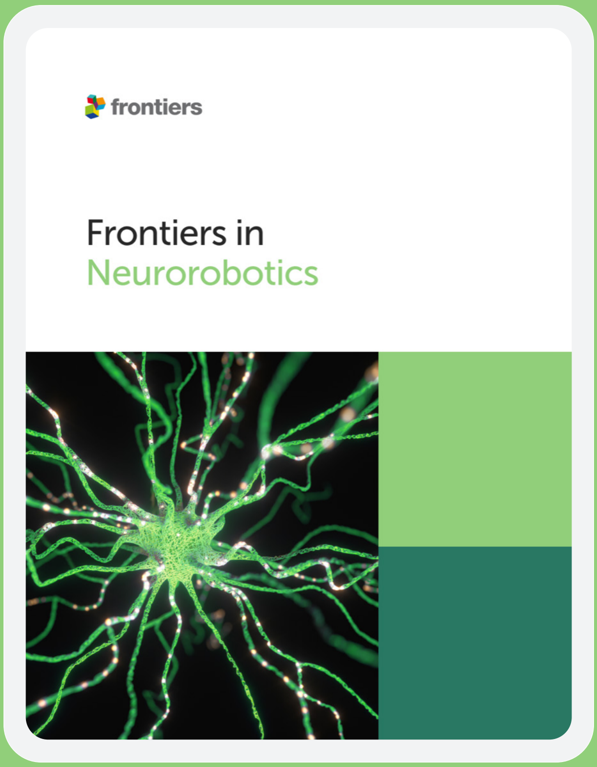
Frontiers in Neurorobotics

Investigates embodied autonomous neural systems and their impact on our lives

Part of the most cited neuroscience series, this journal advances understanding of neurorobotics - from prosthetic devices to brain machine interfaces, and wearable systems to home appliances.

Discover the latest Research Topics

See more →



frontiers

Frontiers in
Neurorobotics

Frontiers
Avenue du Tribunal-Fédéral 34
1005 Lausanne, Switzerland
frontiersin.org

Contact us
+41 (0)21 510 17 00
frontiersin.org/about/contact