frontiers | Frontiers in Neuroinformatics

TYPE Original Research
PUBLISHED 24 March 2023
DOI 10.3389/fninf.2023.1081160

OPEN ACCESS

EDITED BY
Jeny Rajan,
National Institute of Technology, India

REVIEWED BY
Rashid Mehmood,
King Abdulaziz University, Saudi Arabia
Latha S,
SRM Institute of Science and Technology, India
Seyed Mohammad Sadegh Movahed,
Shahid Beheshti University, Iran

*CORRESPONDENCE
Sahaj Anilbhai Patel
✉ sahaj432@uab.edu

RECEIVED 26 October 2022
ACCEPTED 08 March 2023
PUBLISHED 24 March 2023

CITATION
Patel SA and Yildirim A (2023) Non-stationary neural signal to image conversion framework for image-based deep learning algorithms.
Front. Neuroinform. 17:1081160.
doi: 10.3389/fninf.2023.1081160

Non-stationary neural signal to image conversion framework for image-based deep learning algorithms

Sahaj Anilbhai Patel* and Abidin Yildirim

Department of Electrical and Computer Engineering, The University of Alabama at Birmingham, Birmingham, AL, United States

This paper presents a time-efficient preprocessing framework that converts any given 1D physiological signal recordings into a 2D image representation for training image-based deep learning models. The non-stationary signal is rasterized into the 2D image using Bresenham's line algorithm with time complexity O(n). The robustness of the proposed approach is evaluated based on two publicly available datasets. This study classified three different neural spikes (multi-class) and EEG epileptic seizure and non-seizure (binary class) based on shapes using a modified 2D Convolution Neural Network (2D CNN). The multi-class dataset consists of artificially simulated neural recordings with different Signal-to-Noise Ratios (SNR). The 2D CNN architecture showed significant performance for all individual SNRs scores with (SNR/ACC): 0.5/99.69, 0.75/99.69, 1.0/99.49, 1.25/98.85, 1.5/97.43, 1.75/95.20 and 2.0/91.98. Additionally, the binary class dataset also achieved 97.52% accuracy by outperforming several others proposed algorithms. Likewise, this approach could be employed on other biomedical signals such as Electrocardiograph (EKG) and Electromyography (EMG).

KEYWORDS

biomedical signals, non-stationary signal to 2D image representation, 2D convolution neural network (2D CNN), Bresenham's line algorithm, electroencephalogram (EEG)

1. Introduction

The Brian-Machine Interface (BMI) domain is a rapidly evolving multidisciplinary field that aims to bridge the neuron's electrical response from extracellular or intracellular electrode arrays to man-made devices, commonly referred to as prosthetic devices (Carmena et al., 2003). The neural activities in different brain regions correlate to their conclusive motor control functions. For BMI, it is essential to decode neural signals accurately and classify them based on specific features. This would help to localize a specific type of neuron population in the brain, which is critical for implantable devices (Lebedev et al., 2011).

Since recording in the 1929s (Adrian and Bronk, 1929), many different methods have been developed to classify neural signals. Simple threshold methods have been used to detect the action potentials (Gerstein and Clark, 1964). Despite the simplicity, this method is efficient only with a high SNR. With advanced computerization and programming tools, various computational algorithms have been developed to classify efficiently and accurately the different types of neural activities (Lewicki, 1998). The primary two criteria for spike sorting or "clustering" consists of three basic steps: Filtering the signal, Feature representation or extraction, and Classification. Selecting an optimum preprocessing method with less computation time would inevitably help to perform the signal classification in real-time, which is one of BMI's fundamental requirements.

In the preprocessing feature extraction or representation step, most researchers employed different Frequency and Time-Frequency (TF) domain methods that convert 1D non-stationary signals into a 2D image representation. These 2D images represent the different properties of a given neural recording signal. The Fast Fourier Transform (FFT) and Short-Time Fourier Transform (STFT) algorithms are commonly used frequency-based approaches for feature representation from neural recording (Boashash and Sucic, 2003). However, the FFT can only represent spectral resolution, and it is highly computationally intense compared to other TF methods with a time complexity of O(N log N) (Peng, 2020). The Short-Time Fourier Transform (STFT) overcomes the problem of the FFT method by adding temporal and spectral resolution (Mandhouj et al., 2021). Chaudhary et al. (2019) presented the Continues-Wavelet method that outperforms the STFT for classifying different Electroencephalogram (EEG) signal classes. However, selecting the mother wavelet according to the classification application is crucial while applying the wavelet method. In addition, the scale parameter of the mother wavelet set by the user determines the signal's low or high temporal resolution.

Besides various TF representation methods (Brynolfsson and Sandsten, 2017; Khan et al., 2022), Anwar et al. proposed a new approach by placing multiple 2D EEG topographic maps of skulls into a single image from a given 1D segmented EEG signal (Anwar and Eldeib, 2020). Bizopoulos et al. (2019) proposed a "Signal-to-Image Module" that converts raw 1D signal samples into 2D Spectrogram images following up with one layer Convolution Neural Network (CNN) (Bizopoulos et al., 2019). However, the "Signal-to-Image Module" performance was inefficient without the CNN layer. Furthermore, adding one CNN layer increases computational time to O((M*N)*k2), where "M" and "N" represents image size while "k" represents kernel size. Zhao et al. (2021) proposed a Signal-To-Image Mapping (STIM) technique where the 2D image is formed by calculating the correlation between each sample point in the time series. The correlated sample points represent the grayscale by normalizing the data between 0 – 255. However, the converted 2D image does not preserve the signal space characteristics. Kavasidis et al. (2017) proposed the "*Brain2Image*" methodology using a deep learning model. The *Brain2Images* uses Long Short Term Memory (LSTM) autoencoder for converting multi-channel EEG signals into an image for 2D CNN model classification.

For systems with dynamic behaviors, many different methodologies are also proposed to analyze the behaviors of non-stationary signals in multi-dimensional space, such as Lyapunov exponents and fractal dimensions (Donner and Barbosa, 2008). Generally, such dynamical systems are utilized for analyzing the chaotic systems in low dimensions by reconstructing the phase–space (Packard et al., 1980). Such phase spaces are mapped into one dimension by extracting the positive characteristic exponent of the attractor. Over time, many scholars have developed different approaches to integrating the time series of dynamic systems with field graph theory (Takens, 1981; Yang and Yang, 2008). In graph theory, various methods have been developed to convert time series into graph networks, such as visibility graphs (represents condition on the time series amplitudes) (Lacasa et al., 2008). Later, the graph network is converted into a 2D image (such as Adjacency, Degree, and Laplacian matrix) representing the whole graph's features and the time series non-linear properties (Donner et al., 2010). For example, Marwan et al. (2009) represented a complex network into a 2D recurrence matrix with its logistic map, where the recurrence matrix represents the neighbors in phase space. These 2D log-mapped recurrence matrices measured significant sensitivity to change in the dynamics. Shimada et al. (Myers et al., 2019) proposed a k-nearest neighbors approach in their 2D phase space matrix that represents fixed k numbers of single observations in given environments. The single observation can be fixed-phase space distance (constant volume, density). However, such a matrix does not preserve temporal information of given time series.

The conversion of 1D vector signals to 2D images is performed by implementing various "Signal Processing" methodologies such as FFT, Spectrogram, Empirical Mode Decomposition, and Wavelet. These methodologies and some others are time intensive and require high computational power. However, in computer graphics applications, many algorithms for image formation do not require high computing power. The computer graphics domain comprises four significant areas: Image Rendering, Modeling, Animation, and Postprocessing (Wolfe et al., 2000). For instance, in computer graphics, to render a line in 1D or 2D space (bitmap), two distinct pixel locations are selected, and the pixel locations between them are approximated. The Digital Differential Analyzer (DDA), a digital integrating computer (Bartee et al., 1962), is a line drawing algorithm. The DDA draws the lines that utilize the slope equation and approximates the next pixel value based on previous results. However, DDA only works in the first cartesian quadrant, and operating with floating-point numbers will increase the computational time. To overcome DDA, Bresenham (1965) proposed a method that approximates n-points in any given quadrant by calculating simple arithmetic integer operations such as addition, subtraction, and bit-shifting, that decreases computational time by O(n) where n represents the number of samples.

Generally, after converting the non-stationary signal into 2D image representation in BMI applications, the images or extracted features are directly fed into the classifying model to identify the different classes. In feature extraction, the most dominant features are extracted from multi-dimensional 2D images, which leads to reducing the input image size for classifying. For instance, the machine learning domain's most common feature extraction technique is Principal Component Analysis (PCA) (Jolliffe, 2002). In PCA, the user selects the first few dominant features (known as Principal Components) for classification. After the state-of-the-art algorithm by Hilton. et.al (Krizhevsky et al., 2017), the CNN Deep Learning techniques are more commonly used for image classification tasks. They have been implemented in various applications such as Speech recognition, Self-Driving car, and classifying biomedical signals. Additionally, in recent years, CNN architecture has been improved by methods such as GoogleNet (Szegedy et al., 2015), Highway Network

(Srivastava et al., 2015), VGG (Simonyan and Zisserman, 2014), and Xception (Chollet, 2017).

This paper presents a robust preprocessing approach to classify neural spikes in 2D CNN by converting 1D non-stationary neural recording into a 2D image representation. Figure 1 demonstrates the complete process of the proposed method. First, the simulated neural recordings are normalized between 0 and 1. Next, the entire recording is divided into smaller segments the size of 1 × 56. Then, the 1D segmented windows are converted into a 2D image by assigning each sample amplitude and interconnecting the sample points using Bresenham's line algorithm (Bresenham, 1965). Finally, the 2D images are fed into the 2D CNN model for classification. The following sections of the paper are as follows: Section 2 presents the dataset. Section 3 explains the overall methodology, followed by three subsections – Signal Pre-processing, Rasterization, and Feature Extraction and Classification. The experimental result and discussion are described in section 4. Finally, section 5 concludes this paper.

2. Dataset

2.1. Dataset – 1

The dataset-1 consists of simulated action potentials with added noise (Bernert and Yvert, 2019). The timing information of the action potentials was also given. The data was generated by using equation (Eq.) 1 (Adamos et al., 2008).

$$V\left(t\right)=Acos\left(2\pi\frac{t-\tau_{ph}}{\tau_1}\right)exp\left(-\left(\frac{2.3548t}{\tau_2}\right)^2\right) \tag{1}$$

The dataset contains three types of action potentials with seven different SNRs ranging from 0.5 to 2. The parameters (A, τ_1, τ_2, τ_{ph}) for generating three action potentials (spikes) are presented in Bernert and Yvert (2019). Each SNR trial contains ten recordings with three types of spikes, which are randomly distributed over a period of 200 s. The signal is stimulated at a sampling frequency (fs) of 20 kHz with a mean firing rate of 3.3 Hz. Table 1 illustrates the number of segmented windows for each class with different SNR levels. Note that each window is made of 56 samples. The number of windows (Table 1) for each spike type was configured to be nearly the same to prevent the deep learning model (2D CNN) from being biased toward classes with more segmented windows while training and testing the model.

2.2. Dataset – 2

Dataset-2 was collected from the University of California at Irvine (UCI) - Machine Learning Repository, which is utilized for Epileptic Seizure Recognition. The original dataset consists of brain activity from five subjects with five sets (100 channels/set) and 23.6 seconds per recording (Andrzejak et al., 2001). The dataset on UCI was already pre-processed with 178 samples per window and five classes to classify. In this paper, the dataset was converted from a multiclass category to binary classification (Seizure, Non-Seizure activity). The total number of windows and the number of classifying categories is presented in Table 2.

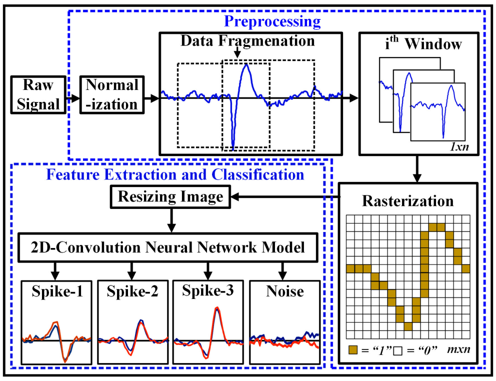

FIGURE 1
Overall block diagram of the proposed framework for neural spikes classification.

TABLE 1 Number of windows for each class per SNRs in dataset 1.

SNR	Spike-1	Spike-2	Spike-3	Noise
0.5	6,423	6,595	6,597	7,759
0.75	6,444	6,591	6,597	7,759
1.0	6,419	6,591	6,597	7,761
1.25	6,394	6,587	6,597	7,758
1.5	6,279	6,585	6,596	6,500
1.75	5,987	6,574	6,597	7,756
2.0	5,553	6,633	6,597	7,750

TABLE 2 Number of windows for each class in dataset 2.

Non-seizure activity	Seizure activity
9,200	2,300

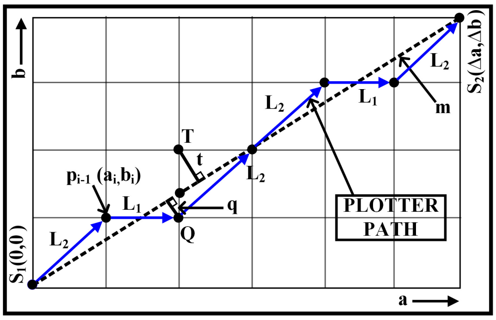

FIGURE 2
Bresenham's line algorithm for first octant.

3. Proposed methodology

3.1. Signal normalization and segmentation

Initially, each sample value of the raw signal is scaled by the unit L2 norm (ranging between 0 and 1) (Bhatia, 2013). The vector

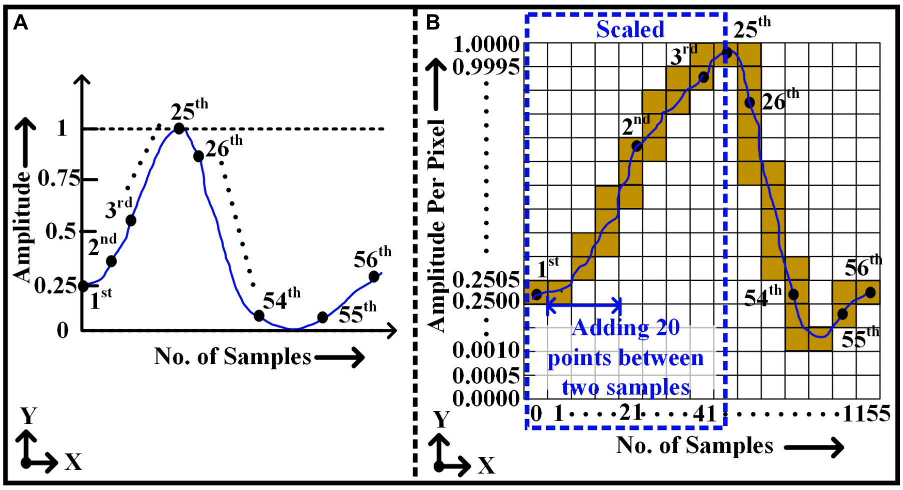

FIGURE 3
Rasterization approach - converting 1D vector of non-stationary signal to 2D pixel image. **(A)** 1D window frame – 1 × 56 size. **(B)** 2D window frame after rasterization - 2001 × 1155 size.

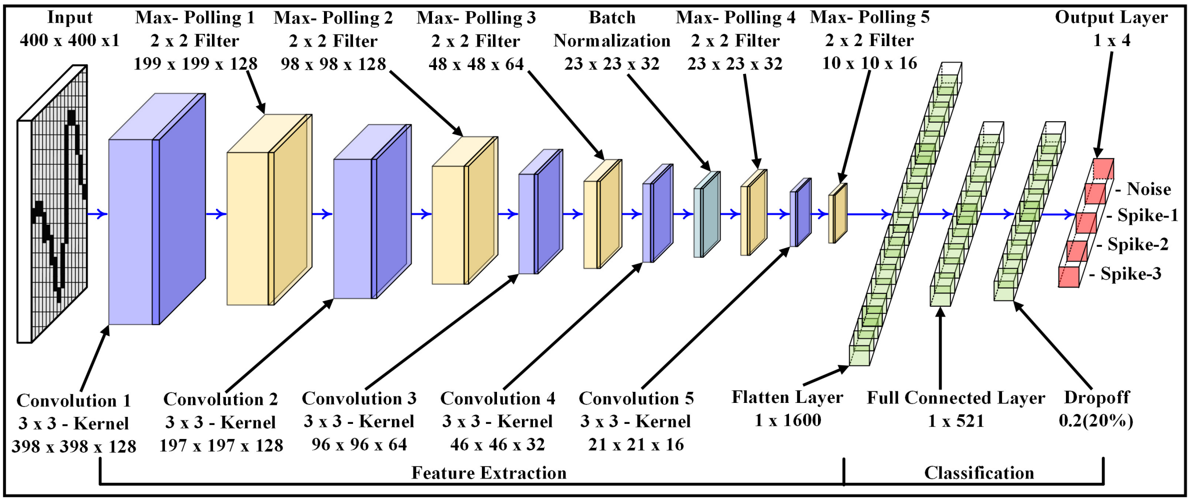

FIGURE 4
Modified 2D CNN architecture for feature extraction and classification of four classes.

normalization of the raw signal was performed by using,

$$y = \frac{x}{\sqrt{\sum_{i=1}^{d} x_i^2}} \tag{2}$$

where x is 1-D input samples values in a vector, y is 1-D normalized output in vector, and its L2 norm is $\sqrt{\sum_{i=1}^{d} x_i^2}$. Later, the normalized input vector is segmented into fixed 1 X n window sizes. Each window carries 56 sample values which are equal to 2.8 milliseconds.

3.2. Rasterization

Rasterization is a process of converting any vector graphic shape into a pixel image (Worboys and Duckham, 2004). For instance, Bresenham's line algorithm approximates n-points between given two points of interest or locations of interest in any coordinates of Euclidean space.

TABLE 3 Number of training and testing dataset per SNRs in dataset 1.

SNR	Train	Test	Total
0.5	8,211	19,160	27,371
0.75	8,217	19,174	27,391
1.0	8,210	19,158	27,368
1.25	8,200	19,136	27,336
1.5	7,788	18,172	25,960
1.75	8,074	18,840	26,914
2.0	7,959	18,574	26,533

TABLE 4 Number of training and testing dataset for dataset 2.

	Train	Test
Normal Activity	2,744	6,456
Seizure Activity	706	1,594

The prediction for the next n^{th} point is determined by calculating the least distances, i.e., q and t, which are perpendicular to the slope line - m of given two-pixel positions.

For instance, in Figure 2, points S_1 and S_2 are placed adjutant to each other in the first octant. Therefore, to predict the next decision parameter for the point, the movement can be either L_1 (to point Q) if $q < t$ or L_2 (to the point T) if $t \geq q$ respectively. Therefore, the difference between q and t is calculated to compute the next move. If we define the difference as: $\nabla_i = q - t$, then;

$$if \begin{cases} \nabla_i < 0, & execute\ L1 \\ \nabla_i \geq 0, & execute\ L2 \end{cases}$$

The initial point S_1 where $a = 0$ and $b = 0$ the ∇_i will as expressed in equation 3, and the points p_{i+1} which is progressing toward S_2 after the initial stage is calculated based on ∇_{i+1} which is Eq. 4 and Eq. 5.

$$\nabla_i = 2\Delta b - \Delta a \quad (3)$$

If $\nabla_i \geq 0$,

$$\nabla_{i+1} = \nabla_i + 2\Delta b - 2\Delta a \quad (4)$$

If $\nabla_i < 0$,

$$\nabla_{i+1} = \nabla_i + 2\Delta b \quad (5)$$

where,

$$\Delta a = x_2 - x_1$$

$$\Delta b = y_2 - y_1$$

Figure 3A illustrates the normalized window with a total of 56 sample points (i.e., y-axis) over time (i.e., x-axis). Figure 3B shows the conversion of (1 × 56) window size to an image (2,001 pixel × 1,155 pixel) format applying Bresenham's line algorithm. Each pixel in the direction of the y-axis of the image represents the amplitude value of the action potential with a resolution of 0.0005. Likewise, the column pixels in the x-axis direction represent the sample locations. Initially, all samples of 1D window representation are placed in their appropriate locations in 2D images as a pixel value of 1 or 0 accordingly. As shown in Figure 3B, to reconstruct a continuous uniformity pattern for the signal, two consequent real sample locations are filled with the dummy pixels. For instance, it is assumed that each window includes 20 empty dummy samples over an interval between each two real sample locations. The assumption of dummy samples resulting a satisfactory representation of the signal curve by connecting the points with line segments. However, the length of intermediates samples can be varied to reduce the overall size of the image. Later, using Bresenham's line algorithm, all true sample values are interconnected, and it represents a complete shape or pattern of the signal in a 2D image.

3.3. Feature extraction and classification

The 2D CNN architecture is used to extract the features (in the spatial domain) of the signals and then classify them according to different signal types and different SNRs. The proposed approach was performed on custom-built 2D CNN architecture [extended LeNet-5 architecture (LeCun et al., 1998)], as shown in Figure 4. The 2D CNN model consists of five different sizes of convolution layers and five max-pooling layers. Each convolution layer extracts/preserves different spatial features, which are also called feature maps. These feature maps are obtained by convoluting the input (output of the previous layer) with a

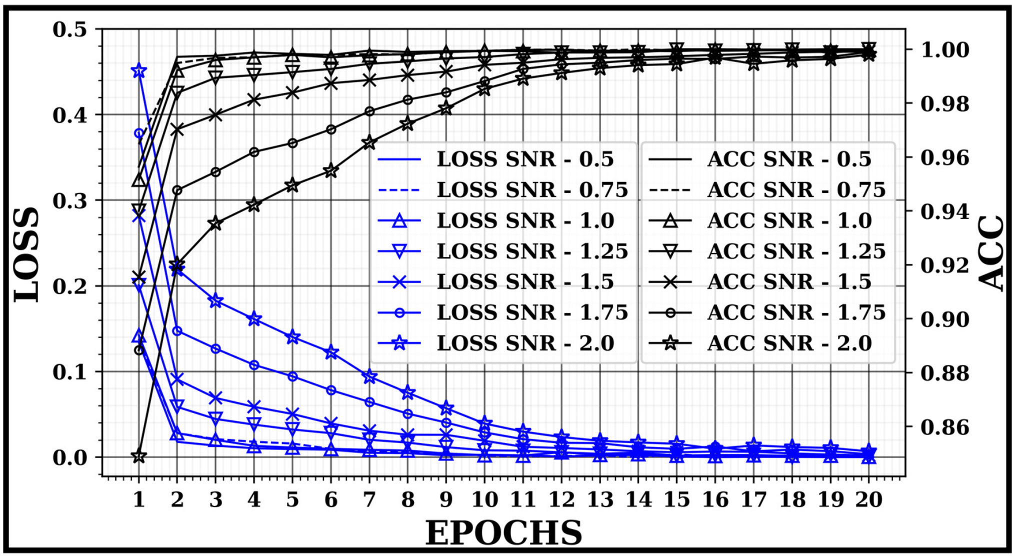

FIGURE 5
Loss vs. no. of epochs per SNR's and accuracy vs. epochs.

TABLE 5 Performance of proposed approach with 2D CNN for different SNR signals – Precision ratio, recall ratio, F1-score and accuracy rate.

SNR	Class	Precision ratio (%)	Recall ratio (%)	F1-score (%)	Accuracy rate (%)
0.5	Noise	99.81	99.63	99.72	99.77
	Spike-1	99.73	99.80	99.76	
	Spike-2	99.69	99.89	99.79	
	Spike-3	99.84	99.80	99.82	
0.75	Noise	99.68	99.63	99.66	99.69
	Spike-1	99.75	99.84	99.80	
	Spike-2	99.52	99.67	99.59	
	Spike-3	99.80	99.63	99.71	
1.0	Noise	99.74	99.57	99.65	99.49
	Spike-1	99.79	99.84	99.82	
	Spike-2	99.60	98.79	99.19	
	Spike-3	98.83	99.76	99.29	
1.25	Noise	99.79	98.64	99.21	98.85
	Spike-1	98.94	99.88	99.41	
	Spike-2	98.20	98.20	98.20	
	Spike-3	98.31	98.77	98.54	
1.5	Noise	98.83	97.90	98.36	97.43
	Spike-1	99.06	99.61	99.21	
	Spike-2	96.55	94.71	95.63	
	Spike-3	95.38	97.84	96.59	
1.75	Noise	95.11	98.15	96.61	95.20
	Spike-1	98.89	96.63	97.75	
	Spike-2	92.35	92.07	92.21	
	Spike-3	94.90	93.53	94.53	
2.0	Noise	94.61	94.03	94.32	91.98
	Spike-1	95.78	96.36	96.07	
	Spike-2	90.21	83.73	86.85	
	Spike-3	87.64	94.12	90.76	

TABLE 6 Performance of proposed approach with 2D CNN for dataset 2 – Precision ratio, recall ratio, F1-score and accuracy rate.

Class	Precision ratio (%)	Recall ratio (%)	F1-score (%)	Accuracy rate (%)
Seizure activity	98.33	98.59	98.46	97.52
Normal activity	94.22	93.22	93.73	

3 × 3 kernel size (Eq. 6) followed by a non-linear Rectified Linear Activation Unit (ReLU) activation function (Eq. 7). The non-linear activation function extracts more complex features. The output of each convolution layer is connected to the max-pooling layer. The purpose of the max-pooling layers is to extract maximum spatial values from each feature map. In addition, these reduce the convolution layer size by half, which leads to a decrease in the number of training weights parameters for the 2D CNN model. The output size of each feature map and pooling layer can be calculated by Eq. 8 and Eq. 9 accordingly.

$$COV_{ij}^{L} = \sum_{a=0}^{k-1}\sum_{b=0}^{k-1} \omega_{ab} COV_{(i+a)(j+b)}^{L-1} \tag{6}$$

where,

COV_{ij}^{L} = Output per pixel in convolution layer, ω_{ab} = weights per pixel in convolution layer, k = Kernel size, L = Convolution layer.

$$Y_{ij}^{L} = f(COV_{ij}^{L}) \tag{7}$$

where,

f = ReLU or SoftMax activation function,

$$COV_{out} = \left[\frac{N_c + 2P_c + K_c}{S_c}\right] + 1 \tag{8}$$

where,

COV_{out} = Feature map output size, N_c = Input size (Width/Height), P_c = Convolution Padding size, K_c = Convolution Kernel size and S_c = Convolution Stride size.

$$P_{out} = \left[\frac{N_p - K_p}{S_p}\right] + 1 \tag{9}$$

where,

P_{out} = Pooling layer output size, N_p = Input size (Width/Height), K_p = Pooling Kernel size, S_c = Pooling Stride size.

In the initial stage of convolution layers, the number of feature maps is higher than in the end convolution layers. For instance, Convolution 1 has 128 feature maps, while Convolution 5 has only 16 feature maps. Therefore, the arrangement of such layer size was used to extract larger patterns compared to finer patterns. The batch normalization is placed between the fourth convolution layer and the fourth max-pooling layer. It potentially helps to converge loss values of the deep learning model much faster in mini-batches, which contain small bundles of data. This leads to a reduction in time consumption during the training phase of the deep learning model.

After the fifth max-pooling layer, the 3D tensor feature maps are converted to 1 × 1600 1D vector space, called a flattened layer. The next layer with the size of 1 × 512 is called the Fully Connected Layer (FCL), where each artificial neuron in the FCL layer is interconnected with the artificial neurons of the previous or next layer. The following layer is called the Dropout Layer (DL), which is connected to the FCL and has the same vector size as FCL. The DL functions as regularization, which prevents the 2D CNN model from over-fitting the training dataset while training the weight parameters. In DL, the random neurons in the layer and their supportive links are eliminated.

Finally, the DL is connected with the Output Layer with the size of 1 × 4 for Dataset-1 and 1 × 2 for Dataset – 2. The vector size of the output layer represents the number of classes to be identified by the 2D CNN model. For the output layer, the SoftMax activation function is deployed. The SoftMax activation function determines the outcome probabilities of each class.

The RMSprop optimizer (Tieleman and Hinton, 2012) is used while training the 2D CNN model to reduce classification loss error (E). This error rate can be reduced by backpropagation. In backpropagation, the weight parameters for each layer are

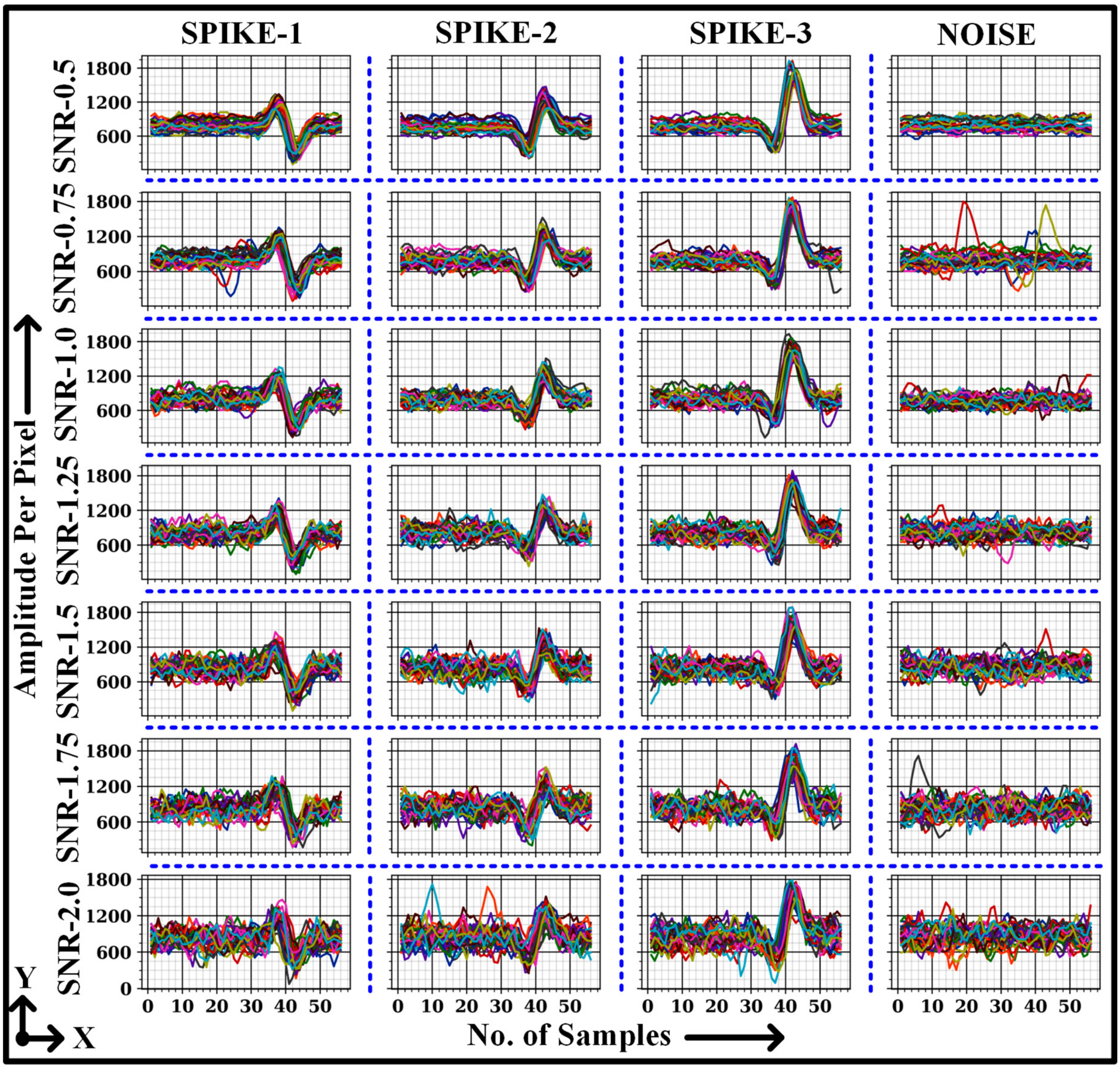

FIGURE 6
2D CNN model predicted output on test dataset for various images with SNR signals.

updated by finding the partial derivative of E w.r.t individual weight parameter per layer that is $\left(\frac{\partial E}{\partial COV_{ij}^{L}}\right)$.

4. Results and discussion

The proposed framework was executed on the University of Alabama at Birmingham supercomputer. The supercomputer is configured with 24 Cores and 23 GB of memory per Core. The method was trained and evaluated using a 2D CNN model with various SNRs. Additionally, to demonstrate the robust performance of the proposed approach for other applications, the 2D CNN model was also trained and evaluated on the EEG dataset (Dataset-2). During each trial, the parameters for 2D CNN architecture are kept the same, such as the number of epochs, batch size, optimizer, input image size, and split ratio for training and testing datasets except output classes. The total number of windows for Dataset-1 and Dataset-2 is split into 30% training and 70% test dataset, respectively. Table 3 and Table 4 illustrate the total number of training and test sample windows for each SNR and dataset-2 accordingly.

Figure 5 demonstrates the accuracy (ACC) and loss of the 2D-CNN model for each SNR according to the number of epochs while training the training dataset. The optimizer was set with a 0.001 learning rate and 20 epochs. Figure 5 also shows that in images with higher SNR, the loss converges faster to a minimum in a few epochs, in contrast to images with low SNR. In this study, to verify the overall performance of the proposed approach on different images with different SNRs, the number of epochs is kept similar for all trials while training the 2D CNN model.

The data splitting approach was implemented for cross-validating the 2D CNN model. To prevent the 2D CNN model

TABLE 7 Comparison of proposed approach with other studies.

Studies	ACC
K-nearest neighbor (Resque et al., 2019)	90.01
Multi-layer perceptron (Resque et al., 2019)	93.53
Naive-Bayes (Resque et al., 2019)	95.98
Random forest (Resque et al., 2019)	97.08
1D CNN (Xu et al., 2020)	97.13
Support vector machine (Resque et al., 2019)	97.31
Our method	**97.52**
CNN-LSTM (Xu et al., 2020)	99.39

from being biased toward any of the classes, the input data was shuffled and divided almost equally for each class except for Dataset 2. The performance of the approach was calculated based on results predicted by a trained 2D CNN model on 70% of the test data by comparing its ground truth results. **Tables 5**, **6** represent the precision ratio, recall ratio, F1-score for each class, and overall ACC for each SNR (Dataset-1) and Dataset-2, respectively. The precision ratio, recall ratio, F1- score, and ACC are calculated using Eq. 10, Eq. 11, Eq. 12, and Eq. 13, respectively.

$$Precision = \frac{TP}{TP + FP} * 100 \tag{10}$$

$$Recall = \frac{TP}{TP + FN} * 100 \tag{11}$$

$$F1 - Score = \frac{2 * \left(Precision * Recall\right)}{\left(Precisioin + Recall\right)} * 100 \tag{12}$$

$$Accurary = \frac{TP + TN}{TP + FP + TN + FN} * 100 \tag{13}$$

where,

TP = True Positive, FP = False Positive, TN = True Negative, FN = False Negative.

The overall ACC for images with signals 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, and 2.0 SNR was above 90%. Despite this, the average ACC for 2.0 SNR was lower than other SNRs because of the higher number of False Positive detection for class "Spike-2". **Figure 6** illustrates a few classified results of the testing dataset with images having different SNRs. It also demonstrates a few misclassified outputs in the "Noise" class at "SNR 0.75." The overall ACC for images in Dataset-2 was 97.52%. The performance of this study was compared with other studies in related fields based on Dataset-2, as shown in **Table 7**.

5. Conclusion

This work demonstrated an efficient preprocessing framework to convert any non-stationary signal into a 2D image using Bresenham's line algorithm. The 2D images can utilize any image classifying algorithm or image-based deep learning models. The experiment was conducted on neural data with various SNR and EEG recordings. The proposed approach showed empirical evidence of performance accuracy with various SNRs and fewer data samples for training. The overall classification accuracy for each SNR outcome (SNR/ACC) as: 0.5/99.69, 0.75/99.69, 1.0/99.49, 1.25/98.85, 1.5/97.43, 1.75/95.20 and 2.0/91.98. In addition, the 2D CNN also performed significantly well on the EEG dataset compared to other methods, with 97.52% ACC. The proposed pre-processing framework could be used in real-time because of the advantage of the low computation time of Bresenham's line algorithm.

Data availability statement

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

Author contributions

SP developed the framework for this research and drafted the manuscript. AY helped to revise the drafted manuscript and provided valuable advice for this research. Both authors read and approved the final manuscript.

Acknowledgments

The author thank Dr. Frank Amthor for providing valuable insights and his expertise.

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/fninf.2023.1081160/full#supplementary-material

References

Adamos, D. A., Kosmidis, E. K., and Theophilidis, G. (2008). Performance evaluation of PCA-based spike sorting algorithms. *Comput. Methods Programs Biomed.* 91, 232–244. doi: 10.1016/j.cmpb.2008.04.011

Adrian, E. D., and Bronk, D. W. (1929). The discharge of impulses in motor nerve fibres: Part II. The frequency of discharge in reflex and voluntary contractions. *J. Physiol.* 67, i3–i151. doi: 10.1113/jphysiol.1929.sp002557

Andrzejak, R. G., Lehnertz, K., Mormann, F., Rieke, C., David, P., and Elger, C. E. (2001). Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Phys. Rev. E* 64:061907. doi: 10.1103/PhysRevE.64.061907

Anwar, A. M., and Eldeib, A. M. (2020). "EEG signal classification using convolutional neural networks on combined spatial and temporal dimensions for BCI systems," in *Proceedings of the 2020 42nd annual international conference of the IEEE Engineering in medicine & biology society (EMBC)*, (Montreal: IEEE), 434–437. doi: 10.1109/EMBC44109.2020.9175894

Bartee, T. C., Lebow, I. L., and Reed, I. S. (1962). *Theory and design of digital machines*. New York, NY: McGraw-Hill.

Bernert, M., and Yvert, B. (2019). An attention-based spiking neural network for unsupervised spike-sorting. *Int. J. Neural Syst.* 29:1850059. doi: 10.1142/S0129065718500594

Bhatia, R. (2013). *Matrix analysis*, Vol. 169. Berlin: Springer Science & Business Media.

Bizopoulos, P., Lambrou, G. I., and Koutsouris, D. (2019). "Signal2image modules in deep neural networks for EEG classification," in *Proceedings of the 2019 41st annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, (Berlin: IEEE), 702–705. doi: 10.1109/EMBC.2019.8856620

Boashash, B., and Sucic, V. (2003). Resolution measure criteria for the objective assessment of the performance of quadratic time-frequency distributions. *IEEE Trans. Signal Process.* 51, 1253–1263. doi: 10.1109/TSP.2003.810300

Bresenham, J. E. (1965). Algorithm for computer control of a digital plotter. *IBM Syst. J.* 4, 25–30. doi: 10.1147/sj.41.0025

Brynolfsson, J., and Sandsten, M. (2017). "Classification of one-dimensional non-stationary signals using the Wigner-Ville distribution in convolutional neural networks," in *Proceedings of the 2017 25th European signal processing conference (EUSIPCO)*, (Kos: IEEE), 326–330. doi: 10.23919/EUSIPCO.2017.8081222

Carmena, J. M., Lebedev, M. A., Crist, R. E., O'Doherty, J. E., Santucci, D. M., Dimitrov, D. F., et al. (2003). Learning to control a brain–machine interface for reaching and grasping by primates. *PLoS Biol.* 1:e42. doi: 10.1371/journal.pbio.0000042

Chaudhary, S., Taran, S., Bajaj, V., and Sengur, A. (2019). Convolutional neural network based approach towards motor imagery tasks EEG signals classification. *IEEE Sens. J.* 19, 4494–4500. doi: 10.1109/JSEN.2019.2899645

Chollet, F. (2017). "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (Honolulu, HI: IEEE), 1251–1258. doi: 10.1109/CVPR.2017.195

Donner, R. V., and Barbosa, S. M. (2008). Nonlinear time series analysis in the geosciences. *Lect. Notes Earth Sci.* 112:37. doi: 10.1007/978-3-540-78938-3

Donner, R. V., Zou, Y., Donges, J. F., Marwan, N., and Kurths, J. (2010). Recurrence networks—a novel paradigm for nonlinear time series analysis. *New J. Phys.* 12:033025. doi: 10.1088/1367-2630/12/3/033025

Gerstein, G. L., and Clark, W. A. (1964). Simultaneous studies of firing patterns in several neurons. *Science* 143, 1325–1327. doi: 10.1126/science.143.3612.1325

Jolliffe, I. T. (2002). *Principal component analysis for special types of data*. New York, NY: Springer, 338–372.

Kavasidis, I., Palazzo, S., Spampinato, C., Giordano, D., and Shah, M. (2017). "Brain2image: Converting brain signals into images," in *Proceedings of the 25th ACM international conference on multimedia*, (New York, NY: Association for Computing Machinery), 1809–1817. doi: 10.1145/3123266.3127907

Khan, N. A., Mohammadi, M., Ghafoor, M., and Tariq, S. A. (2022). Convolutional neural networks based time-frequency image enhancement for the analysis of EEG signals. *Multidimens. Syst. Signal Process.* 33, 863–877. doi: 10.1007/s11045-022-00822-2

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

Lacasa, L., Luque, B., Ballesteros, F., Luque, J., and Nuno, J. C. (2008). From time series to complex networks: The visibility graph. *Proc. Natl. Acad.Sci. U.S.A.* 105, 4972–4975. doi: 10.1073/pnas.0709247105

Lebedev, M. A., Tate, A. J., Hanson, T. L., Li, Z., O'Doherty, J. E., Winans, J. A., et al. (2011). Future developments in brain-machine interface research. *Clinics* 66, 25–32. doi: 10.1590/S1807-59322011001300004

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 2278–2324. doi: 10.1109/5.726791

Lewicki, M. S. (1998). A review of methods for spike sorting: The detection and classification of neural action potentials. *Network* 9, R53–R78. doi: 10.1088/0954-898X_9_4_001

Mandhouj, B., Cherni, M. A., and Sayadi, M. (2021). An automated classification of EEG signals based on spectrogram and CNN for epilepsy diagnosis. *Analog Integr. Circuits Signal Process.* 108, 101–110. doi: 10.1007/s10470-021-01805-2

Marwan, N., Donges, J. F., Zou, Y., Donner, R. V., and Kurths, J. (2009). Complex network approach for recurrence analysis of time series. *Phys. Lett. A* 373, 4246–4254. doi: 10.1016/j.physleta.2009.09.042

Myers, A., Munch, E., and Khasawneh, F. A. (2019). Persistent homology of complex networks for dynamic state detection. *Phys. Rev. E* 100:022314. doi: 10.1103/PhysRevE.100.022314

Packard, N. H., Crutchfield, J. P., Farmer, J. D., and Shaw, R. S. (1980). Geometry from a time series. *Phys. Rev. Lett.* 45, 712–716. doi: 10.1103/PhysRevLett.45.712

Peng, R. D. (2020). *A very short course on time series analysis*. Available online at: https://bookdown.org/rdpeng/timeseriesbook/the-fast-fourier-transform-fft.html (accessed August 21, 2022).

Resque, P., Barros, A., Rosário, D., and Cerqueira, E. (2019). "An investigation of different machine learning approaches for epileptic seizure detection," in *Proceedings of the 2019 15th international wireless communications & mobile computing conference (IWCMC)*, (Tangier: IEEE), 301–306. doi: 10.1109/IWCMC.2019.8766652

Simonyan, K., and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv [Preprint]*

Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015). Highway networks. *arXiv [Preprint]*

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: IEEE), 1–9. doi: 10.1109/CVPR.2015.7298594

Takens, F. (1981). "Detecting strange attractors in turbulence," in *Dynamical systems and turbulence, Warwick 1980 lecture notes in mathematics*, Vol. 898, eds D. Rand and L. Young (Berlin: Springer), 366–381. doi: 10.1007/BFb0091924

Tieleman, T., and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *Coursera* 4, 26–31.

Wolfe, R., Lowther, J. L., and Shene, C. K. (2000). Rendering+ modeling+ animation+ postprocessing = computer graphics. *ACM SIGGRAPH Comput. Graph.* 34, 15–18. doi: 10.1145/369215.369224

Worboys, M. F., and Duckham, M. (2004). *GIS: A computing perspective*. Boca Raton, FL: CRC press. doi: 10.4324/9780203481554

Xu, G., Ren, T., Chen, Y., and Che, W. (2020). A one-dimensional CNN-LSTM model for epileptic seizure recognition using EEG signal analysis. *Front. Neurosci.* 14:578126. doi: 10.3389/fnins.2020.578126

Yang, Y., and Yang, H. (2008). Complex network-based time series analysis. *Phys. A Stat. Mech. Appl.* 387, 1381–1386. doi: 10.1016/j.physa.2007.10.055

Zhao, J., Yang, S., Li, Q., Liu, Y., Gu, X., and Liu, W. (2021). A new bearing fault diagnosis method based on signal-to-image mapping and convolutional neural network. *Measurement* 176:109088. doi: 10.1016/j.measurement.2021.109088