frontiers in Neuroscience

ORIGINAL RESEARCH
published: 19 February 2021
doi: 10.3389/fnins.2021.582608

# *nMNSD*—A Spiking Neuron-Based Classifier That Combines Weight-Adjustment and Delay-Shift

Gianluca Susi [1,2,3]*, Luis F. Antón-Toro [1,2], Fernando Maestú [1,2,4], Ernesto Pereda [1,5] and Claudio Mirasso [6]

[1] UPM-UCM Laboratory of Cognitive and Computational Neuroscience, Centro de Tecnologia Biomedica, Madrid, Spain, [2] Departamento de Psicología Experimental, Facultad de Psicología, Universidad Complutense de Madrid, Madrid, Spain, [3] Department of Civil Engineering and Computer Science, University of Rome "Tor Vergata", Rome, Italy, [4] CIBER-BBN: Networking Research Center on Bioengineering, Biomaterials and Nanomedicine, Madrid, Spain, [5] Departamento de Ingeniería Industrial & IUNE & ITB. Universidad de La Laguna, Tenerife, Spain, [6] Instituto de Física Interdisciplinar y Sistemas Complejos (IFISC, UIB-CSIC), Palma de Mallorca, Spain

OPEN ACCESS

**Edited by:**
Anirban Bandyopadhyay,
National Institute for Materials Science, Japan

**Reviewed by:**
Pathik Sahoo,
National Institute for Materials Science, Japan
Giacomo Capizzi,
University of Catania, Italy

***Correspondence:**
Gianluca Susi
gianluca.susi@ctb.upm.es

**Specialty section:**
This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience

**Received:** 12 November 2020
**Accepted:** 15 January 2021
**Published:** 19 February 2021

**Citation:**
Susi G, Antón-Toro LF, Maestú F, Pereda E and Mirasso C (2021) nMNSD—A Spiking Neuron-Based Classifier That Combines Weight-Adjustment and Delay-Shift. Front. Neurosci. 15:582608. doi: 10.3389/fnins.2021.582608

The recent "multi-neuronal spike sequence detector" (MNSD) architecture integrates the weight- and delay-adjustment methods by combining heterosynaptic plasticity with the neurocomputational feature spike latency, representing a new opportunity to understand the mechanisms underlying biological learning. Unfortunately, the range of problems to which this topology can be applied is limited because of the low cardinality of the parallel spike trains that it can process, and the lack of a visualization mechanism to understand its internal operation. We present here the nMNSD structure, which is a generalization of the MNSD to any number of inputs. The mathematical framework of the structure is introduced, together with the "trapezoid method," that is a reduced method to analyze the recognition mechanism operated by the nMNSD in response to a specific input parallel spike train. We apply the nMNSD to a classification problem previously faced with the classical MNSD from the same authors, showing the new possibilities the nMNSD opens, with associated improvement in classification performances. Finally, we benchmark the nMNSD on the classification of static inputs (MNIST database) obtaining state-of-the-art accuracies together with advantageous aspects in terms of time- and energy-efficiency if compared to similar classification methods.

**Keywords: classification, delay learning, MNSD, online learning, spike latency, heterosynaptic plasticity, MNIST database**

## 1. INTRODUCTION

In the last few years, diverse machine learning (ML) methods have been proposed for the recognition of spike patterns generated by neural populations (Ambard and Rotter, 2012; Tapson et al., 2013; Grassia et al., 2017; Nazari and Faes, 2019). The ability to learn and decode spike patterns is not only useful for the interpretation of biological mechanisms (Koyama et al., 2010; Rudnicki et al., 2012; Heelan et al., 2019) but also for engineering applications, such as artificial vision and hearing (Nogueira et al., 2007; Zai et al., 2015; Schofield et al., 2018) analysis of brain signals (Susi et al., 2018), forecasting of energy consumption (Kulkarni et al., 2013), and so on. Most of such ML methods are based on neural networks, and specifically on the bio-inspired spiking neural networks (SNNs) (Maass, 1997; Florian, 2012).

In the literature, there are many learning methods for SNNs that make use of biologically plausible strategies. While most of these methods are based on synaptic learning rules aimed at modifying the weights (i.e., *weight adjustment*), only few of them consider the modulation of the delay time to achieve learning (i.e., *delay shift*, see Brückmann et al., 2004; Adibi et al., 2005; Taherkhani et al., 2015; Matsubara, 2017; Hwu et al., 2018; Wang et al., 2019). Interestingly, it has been demonstrated that the alteration of delays has advantages in forming spatiotemporal memories, over altering synaptic weights (Izhikevich, 2006; Hwu et al., 2018). Incidentally, experimental research proved that delays are widely present in biological neural networks and contribute to encode information (Chase and Young, 2007; Minneci et al., 2012), and various biological justifications have been attributed to the delay adjustment processes, among which the activity-dependent myelination (Mount and Monje, 2017) (which, in turn, results in the modulation of conduction velocities) and the spike latency tuning (see Fields, 2008, 2015; Zhou et al., 2012; Matsubara, 2017; Hwu et al., 2018; Wang et al., 2019).

The recently developed multi-neuronal spike sequence detector (MNSD) architecture (Susi et al., 2018) is a simple but effective bio-inspired topology specialized in online learning and recognition of parallel spike trains. Such tool integrates the weight- and delay-adjustment methods by means of the spike timing-dependent plasticity (STDP) and the well-known mechanism of spike latency (i.e., the neuron's intrinsic potential-dependent delay time between the overcoming of the "threshold" and the actual spike generation, Izhikevich, 2004; Salerno et al., 2011), representing a new opportunity to understand the mechanisms underlying biological learning.

In its original form, the MNSD architecture (**Figure 1**) is composed of:

- A layer of *delay neurons* $D_1, D_2, D_3$ (termed delay layer), which receive the parallel spike train (composed of the external spikes $ES_1, ES_2, ES_3$). Such neurons are characterized by nearest-neighbor excitatory interactions mediated by heterosynaptic STDP (the dashed links in **Figure 1**); this mechanism allows the synaptic weights of the neurons to change on the basis of the current parallel spike train. Through the spike latency feature, the change in the weight reflects on the modulation of the delay in relaying the external spike received on the *i-th* branch, toward next stages of the structure. In this way, the MNSD is able to learn the input parallel spike train, which can be represented into a multi-dimensional, temporal, feature space (Susi et al., 2018);
- One *target neuron T*, which performs the summation of the outputs of the three delay neurons and acts as readout neuron, signaling the recognition of a specific parallel spike train. In order for the target to produce a spike, a synchrony of the contributions arriving from previous steps of the structure is required. It happens only if the delays introduced by the *delay neurons* are able to compensate the initial lags among the spikes of the input parallel spike train;
- Three sets of weights, each one for a family of connections. The *input weight set* (i.e., the set of variables $\langle w_{D_1,ES_1}, w_{D_2,ES_2}, ..., w_{D_n,ES_n} \rangle$, each one representing an *input weight*), that is where the learning is finally encoded; the input weights, that are subject to the action of STDP, are placed between the input terminals and the delay layer. The *heterosynaptic weight set* (i.e., the set of variables $\langle w_{D_1,D_2},\ w_{D_2,D_1}, w_{D_2,D_3},\ w_{D_3,D_2}, ..., w_{D_{n-1},D_n},\ w_{D_n,D_{n-1}} \rangle$ each one representing a *heterosynaptic weight*); the heterosynaptic weights, which are placed between adjacent delay neurons, are characterized by a very low value and serve for the STDP-based adjustment mechanism of the input weights. The *output weight set* (i.e., the set of variables $\langle w_{T,D_1}, w_{T,D_2}, ..., w_{T,D_n} \rangle$, each one representing an *output weight*); the output weights are placed between the delay layer and the target neuron, and allow us to control the target summation. This is done by assigning the relevance of each feature in the definition of the pertaining class. In other words, the output weight set establishes the shape of the delimiter of a class in the feature space.

The described structure is able to perform online learning and recognition. Obviously, it can also be used envisaging separately a learning phase (with STDP activated) and a recognition phase (with STDP disabled).

Unfortunately, the range of problems to which the original version of the MNSD can be applied is limited because of two reasons: the low cardinality of the parallel spike trains that can be processed (3 branches, i.e., 3 features per class) and the lack of a tool able to represent the internal computation in order to set the parameters in accord to the specification of the problem in a knowingly manner.

In this work, we present various novelties:

- The nMNSD structure *per se* (**Figure 2**), which is a generalization of the MNSD to any number of inputs. The analysis of the internal operation of the structure is presented, considering the two operating modalities highlighted in Susi et al. (2018):
  - *Static behavior*, i.e., how an nMNSD with a specific weight configuration will react to a new given input sequence (disregarding the action of STPD);
  - *Dynamic behavior*, i.e., how an nMNSD with a specific set of STDP parameters will change its input weights in consequence to a new given input sequence.

  To improve the practical usability of the nMNSD as a classifier tool, we will discuss two possibilities: (1) to regulate the impact of each feature in the target summation for the determination of the class (i.e., *feature relevance* property), and (2) to configure different nMNSDs to be used in a multiclass problem where the number of branches is the number of features, and the number of nMNSD structures used is the number of classes.
- The *trapezoid method*, i.e., a reduced method to analyze the recognition mechanism operated by the nMNSD in response to a specific input sequence. This serves as design support regarding the static behavior of the nMNSD, and results necessary to represent its internal processing with

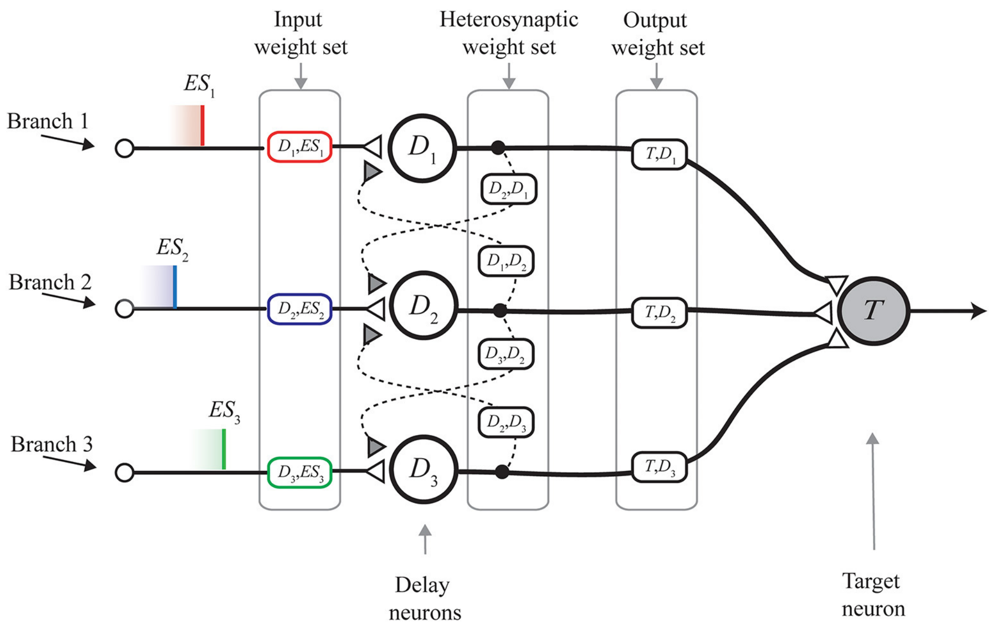

**FIGURE 1 |** Multi-neuronal spike sequence detector (MNSD) structure hit by the parallel spike train $ES_1, ES_2, ES_3$. The different sections of the structure are shown. From **left** to **right**: input weight set, heterosynaptic weight set, output weight set; delay and target neuron are indicated with $D_i$ and $T$, respectively. Modified from Susi et al. (2018).

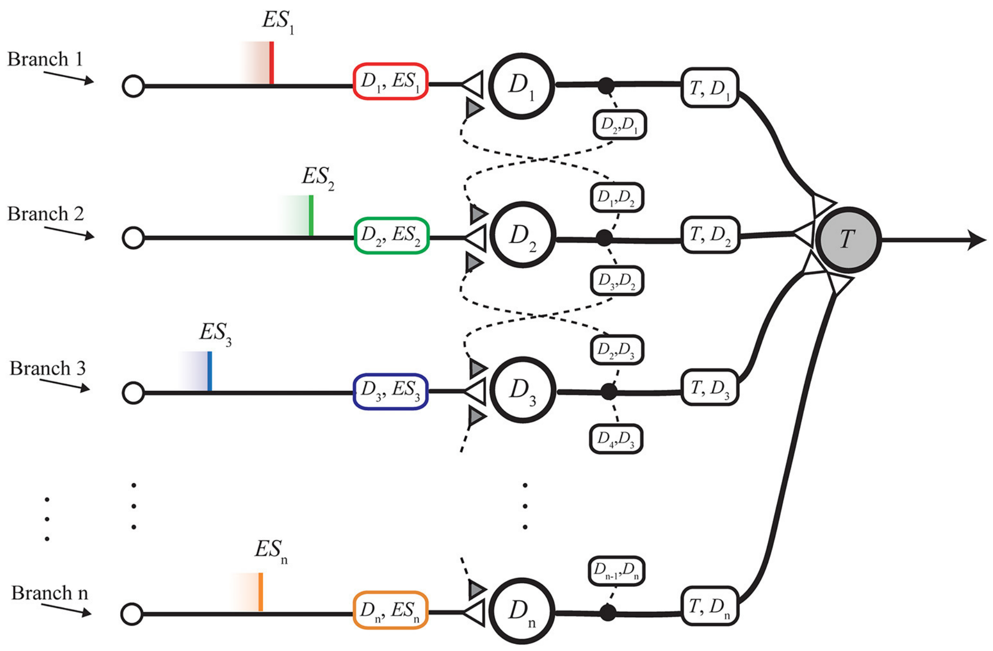

**FIGURE 2 |** nMNSD hit by a parallel spike train.

a number of branches greater than 3, since in this case the feature space is not trivially representable (more than 3 dimensions). We provide a visualization toolbox based on this method, which allows the user to consciously customize nMNSD-based classification systems for specific classification problems.

Finally, we present 2 applications of the nMNSD. We apply our extended method to a classification problem previously faced with the classical MNSD from the same authors, showing the new possibilities the nMNSD opens, with associated improvement in classification performances. Finally, we benchmark the nMNSD on the classification of handwritten digits from the MNIST database, obtaining state of the art accuracies together with advantageous aspects in terms of time- and energy-efficiency if compared to similar classification methods.

The visualization toolbox, based on a dedicated nomogram (see Appendix 1 in **Supplementary Material**), can be found at the following weblink: www.github.com/LCCN/Frontiers2021.

## 2. MATERIALS AND METHODS

### 2.1. A Brief Resume on the LIFL Neuron Model

The LIFL neuron model (Cardarilli et al., 2013; Susi et al., 2018) is similar to the classical Leaky Integrate-and-Fire (LIF), but it is characterized by the presence of the *spike latency* neurocomputational feature (Izhikevich, 2004; Salerno et al., 2011), a real neuron characteristic that has been extracted from the Hodgkin–Huxley equations (Salerno et al., 2011). In a nutshell, it consists of the neuron's intrinsic potential-dependent delay time between the overcoming of the "threshold" and the actual spike generation, allowing the neuron to encode the strength of the input in the spike times. The LIFL neuron model is characterized by an internal state $S$ that represents the membrane potential of the biological counterpart. $S$ conventionally ranges from 0 (representing the resting value, $S_0$) to a maximum value $S_{max}$ (at most $\infty$), and a fixed threshold $S_{th}$, slightly greater than 1. The value of $S$ with respect to $S_{th}$ demarcates two different working modes of the neuron:

- the *passive mode*, for $S < S_{th}$. Here, the evolution of $S$ during time is characterized by a spontaneous decay. Although the equations of the LIFL neuron are compatible with different decay types, we will consider here for simplicity a linear subthreshold decay (asin Susi et al., 2018; Mattia and Del Giudice, 2000). Accordingly, given a temporal distance $\Delta t$ between two consecutive incoming spikes, $S$ experiences a decrease, such that:

$$S_{new} = S_{old} - L_d \cdot \Delta t \tag{1}$$

  being $L_d$ a non-negative quantity called *decay parameter*.
- the *active mode*, for $S \geq S_{th}$. Pnce $S$ crosses the value $S_{th}$, the neuron is ready to fire; however, firing is not instantaneous, but it occurs after a continuous-time delay, the model equivalent of the spike latency feature of real neuron, that we call *time-to-fire* and indicate with $ttf$:

$$ttf = \frac{1}{S-1} \tag{2}$$

  The latter defines the relationship between $S$ and $ttf$.

  Here, the evolution of $S$ is characterized by a spontaneous growth:

$$S_{new} = \frac{(S_{old}-1)^2 \Delta t}{1-(S_{old}-1)\Delta t} \tag{3}$$

**TABLE 1 |** Recommended values for the nMNSD structure parameters (see the reference paper Susi et al., 2018).

| Parameter | Value / Condition |
|---|---|
| Neuron parameters | Identical for all neurons: |
| | $S_0 = 0$ |
| | $L_d \leq 0.15$ |
| | $S_{th} = 1.04$ (*) |
| ES amplitudes | $A_{ES_i} = 1.00$ (**) |
| Input weights (starting value) | $w_{D_i,ES_i} \simeq 1.08$ (***) |
| Target weights | $\sum_{i=1}^{n} w_{T,D_i} \geq S_{th}$ |
| STDP parameters | $A_+ = -A_- \leq 0.01; \tau+ = \tau_- \simeq [2-10]$ |
| Heterosynaptic weights | $w_{D_i,D_i+1} = w_{D_{i+1},D_1} \simeq 0$ (****) |
| Connection delays | 0 (instantaneous) |

*(*) Able to ensure a value of $ttf_{max}$ sufficiently high to differentiate the input patterns ($ttf_{max\,D_i} = 25$ ms). (**) Giving the same weight for all the input spikes is a simplifying but not unrealistic assumption, since spike amplitude has been observed to change mostly as function of the firing rate of the spiking neuron's activity (Stratton et al., 2012), which in our experiments can be considered very low and quite constant. (***) Chosen to let $D_i$ generate a spike around the center of the latency range (i.e., $ttf(1.08) = ttf_{max\,D_i}/2 = 12.5$ ms), then obtain a large variation margin for the weight adjustments during learning. (****) Lateral contributions are considered weak.*

Obviously, in the case of a transition from passive to active mode, Equation (1) is applied (Equation 3 if vice versa, although we will not consider this case since inhibitory contributions are not envisaged in this work).

The firing threshold is written as:

$$S_{th} = 1 + d \tag{4}$$

where $d$ is a positive value called *threshold constant*, which fixes a bound for the maximum value of $ttf$. According to Equation (2), when $S = S_{th}$, $ttf$ is maximum, and equals to:

$$ttf_{max} = \frac{1}{d} \tag{5}$$

$ttf_{max}$ represents the upper bound of the time-to-fire and is a measure of the finite maximum spike latency of the biological counterpart (FitzHugh, 1955).

Simple Dirac delta functions (representing the action potentials) are exchanged between neurons in form of pulses or pulse trains.

### 2.2. Pattern Recognition in a Trained nMNSD

The nMNSD is an extension of the MNSD to an arbitrary number of branches (**Figure 2**). This gives the possibility to face classification problems characterized by an arbitrary number of features, $n$.

We give in this section a summary of the static behavior of an nMNSD structure (i.e., as multi-neuronal spike pattern detector, disregarding the action of the STDP) and evaluate how the structure will react with regard to a specific input parallel pattern, in accord to the structure's input weight set.

Assumed the same amplitude $A_{ES_i} = 1$ for each single external input spike $ES_i$ (see **Table 1**), a parallel spike train of

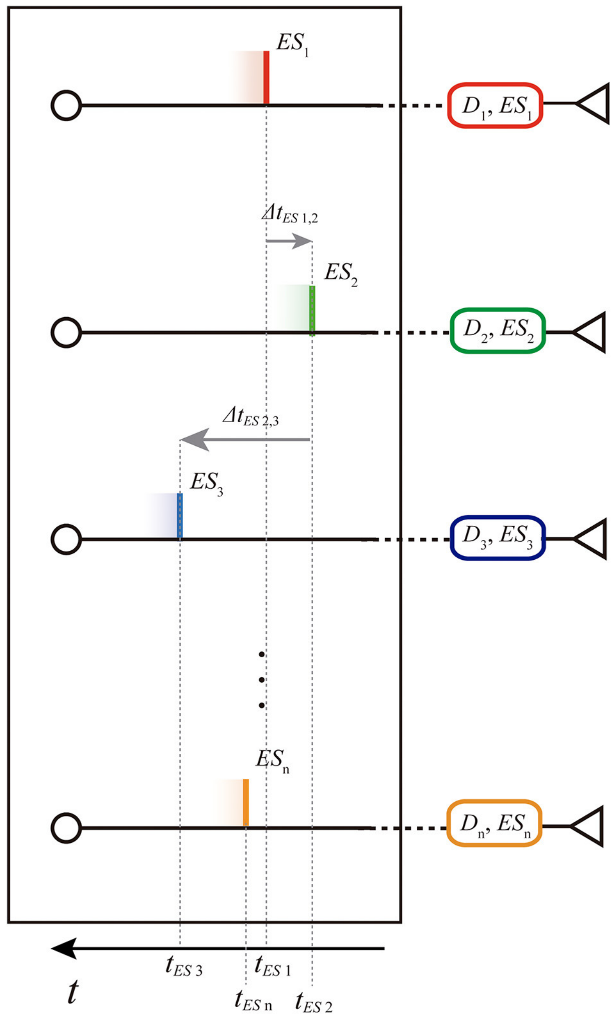

**FIGURE 3 |** Representation of the spike train shown in **Figure 2** on the entryway of the structure: absolute spike times ($t_{ES1}$, $t_{ES2}$,..., $t_{ESn}$) and relative intervals ($\Delta t_{ES1,2}$, $\Delta t_{ES2,3}$, ..., $\Delta t_{ESn-1,n}$). Note that the time axis is reversed since we are representing the parallel spike train in motion toward right.

order $n$ is characterized by a vector of $n$ external *absolute spike times* of consecutive branches $\langle t_{ES_1}, t_{ES_2}, ..., t_{ES_i}, ..., t_{ES_n} \rangle$ (positive values), one for each component spike. In order to make our procedure independent of the initial time offset, and for ease representation, in some parts of this document the parallel spike train can be equivalently defined by the vector of $n - 1$ intervals between spike events of consecutive branches, i.e., *relative intervals* $\langle \Delta t_{ES_{1,2}}, \Delta t_{ES_{2,3}}, ..., \Delta t_{ES_{n-1,n}} \rangle$ (which components can assume positive or negative values) (see **Figure 3**).

When the single external input spike goes through the delay layer, the corresponding input weight attenuates/amplifies the pulses accordingly (**Figure 4**). Hypothesizing the delay neuron $D_i$ is in the resting condition $S_{D_i} = S_0 = 0$ (where $S_0$ is the resting potential, see **Table 1**); after the reception of the spike, the following value will be reached:

$$S_{D_i} = A_{ES_i} \cdot w_{D_i,ES_i} \tag{6}$$

if the following condition is satisfied for the internal state of $D_i$:

$$S_{D_i} \geq S_{th} \tag{7}$$

the neuron will produce a spike. It will be done after its time to fire, $ttf_{D_i}$, evoked by the internal state reached by the neuron $D_i$ in according to Equation (2).

Through the latency the LIFL neuron has the extraordinary ability to perform a strength-to-delay transformation, but in the event that all the input weights had the same value and with the assumptions above, the delay neurons would introduce the same lag to each single input spike. In contrast, different weights give rise to different delays: the higher the weight $w_{D_i,ES_i}$, the greater the input to the delay neuron (Equation 6) and the lower the delay involved in the spike generation.

Considering an nMNSD of $n$ branches, according to its input weight set it presents a *preferential parallel train* (PPT) with respect to the activation of its target, which acts as readout neuron generating a spike in case of recognition. Considering the structure at rest, the arrival of a specific input parallel pattern should produce to its target the same response irrespectively to when it arrives. For this reason, the structure's PPT is defined as a set of preferential relative intervals for each couple of adjacent branches, i.e., the vector of $n - 1$ values $\langle \overline{\Delta t_{ES_{1,2}}}, \overline{\Delta t_{ES_{2,3}}}, ..., \overline{\Delta t_{ES_{n-1,n}}} \rangle$, instead of absolute ones. The PPT depends on its input weight set, which reflects how the structure has evolved during the previous learning. Considering all the connections instantaneous (see **Table 1**) as in the reference work of Susi et al. (2018), the only delays present in the structure are those introduced by the latency feature. Therefore, as a consequence to the introduction of a multi-neuronal pattern to the structure, two different responses are possible:

- The structure is not able to detect the specific input multi-neuronal spike sequence; this is because the delays produced by the input weight set, applied to the current input, do not result in a synchronous target summation;
- The input weight set of the structure generates a set of delays that, in combination with the relative intervals of the current input pattern, verify the *simultaneity condition* at the target, making the target spike, then revealing the detection of the specific input train.

The *simultaneity condition* at the target occurs when the characteristic intervals of the input multi-neuronal spike sequence is coincident (or quasi-coincident) with the PPT of the structure. The more the input characteristic set of intervals fits the structure's PPT, the more the *maximum target summation peak* $S_{p,M}$ (i.e., the maximum $S$ achieved by the target during the summation) will be higher (see **Figure 4**).

Although the input weight set mediates the mapping of the input pattern in the feature space, the output weight set allows to define the boundaries of the classes. Those 2 weight sets reflect on the positioning and on the shape of the classes in the feature space, respectively (see section 2.6). The PPT consists of a line in the n-dimensional temporal features space, and the output weight set modulates the confidence intervals for a parallel spike train to be recognized with respect to the PPT of the structure. The output

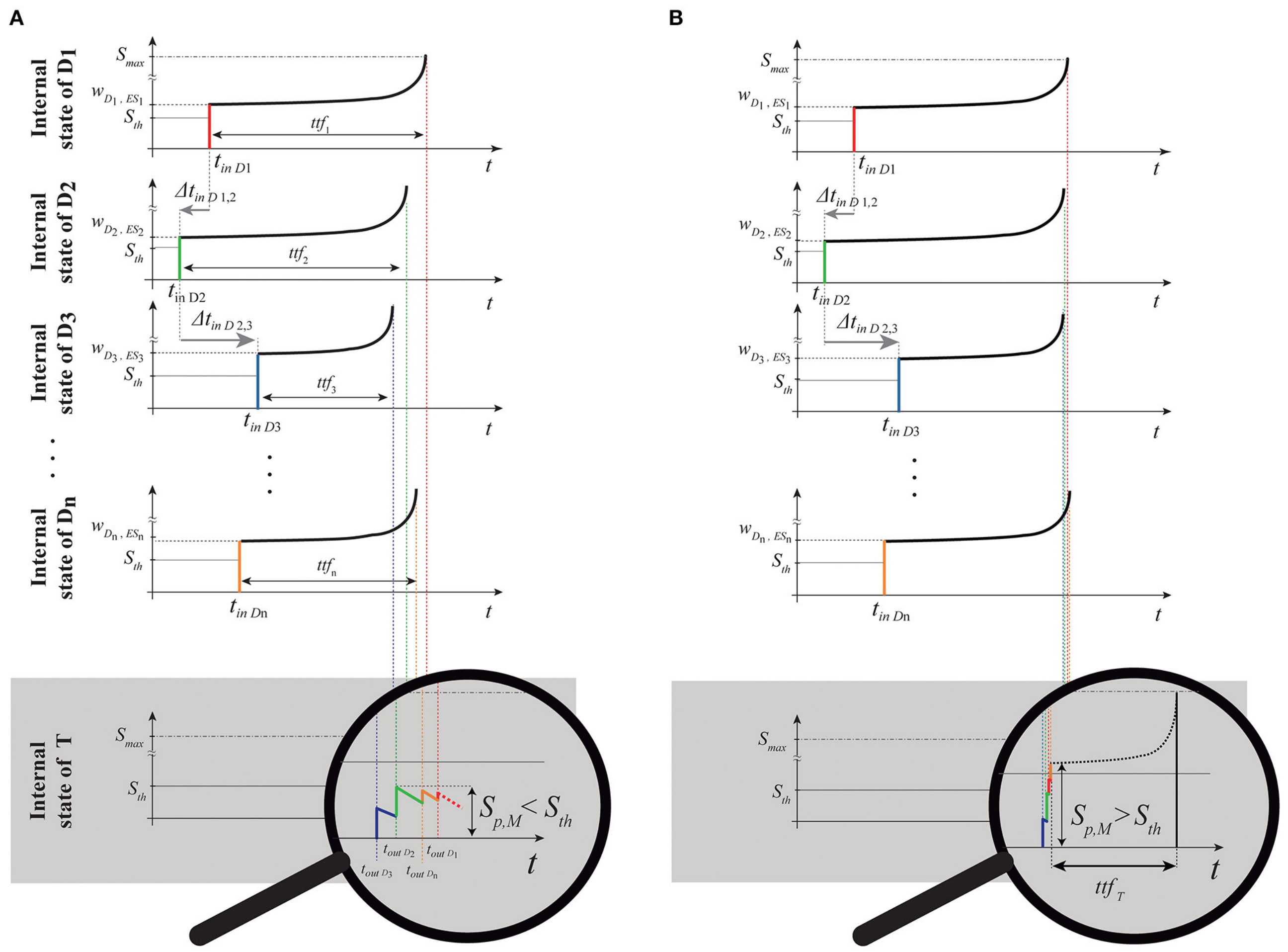

**FIGURE 4 |** Representation of the same spike train shown in **Figures 2**, **3** passing through the delay layer and related impact on the internal state of the delay neurons $D_i$ and target neuron $T$. **(A)** Internal neuron representation ($S_{D_i}$) of the absolute spike times ($t_{inD_1}, t_{inD_2}, ..., t_{inD_n}$) and relative intervals ($\Delta t_{inD_{1,2}}, \Delta t_{inD_{2,3}}, ..., \Delta t_{inD_{n-1,n}}$) of the parallel spike train. In this case, the target will not produce a spike because the simultaneity condition at the target is not reached (relative intervals of the input are not coincident with the PPT of the structure); **(B)** we show the response of a structure with a different PPT, to the same parallel input train, which summation tends to the *simultaneity condition*. Consequently, such pattern is detected, and then the spike is generated. It happens after a proper time-to-fire $ttf_T$. To show the summation process to the reader, the diagram of the internal state of the target neuron is zoomed for ease of illustration. Note that the temporal values in this internal representation are symmetrical with respect to **Figures 2**, **3** since we are representing here the Cartesian temporal axis of times and not a "photography" of the parallel spike train entering the structure.

weights are chosen so that their sum (i.e., the *target activity level*) is greater or equals to $S_{th}$:

$$\sum_{i=1}^{n} w_{T,D_i} \geq S_{th} \tag{8}$$

i.e., in order to allow the target spike when the favorable "simultaneity condition" is verified. The easier choice for the output weights is to set them to the same value ($w_{T,D_1} = w_{T,D_2} = ... = w_{T,D_i}$), so that the features have equal weight in the target summation. Alternatively, we can differentiate the degree of importance of each feature in determining a class by giving different output weights to each of the branches (feature relevance).

## 2.3. How the Structure Adapts to New Incoming Patterns

When STDP is active, the structure is ready to learn a new parallel spike train. In this way, one nMNSD is able to identify one class, shaping its boundaries in the feature space according to the examples presented to its input during the training. We define in this section how the structure with STDP activated behaves when a parallel spike train is presented to its input (dynamic behavior).

The adaptive core of the structure resides in the interplay of spike latency and plasticity: the delay $ttf_i$ that characterizes the neuronal pathway $i$ is due to the spike latency of the delay neuron $w_{D_i,ES_i}$, which in turn is modulated by the neighboring branch(es) through heterosynaptic STDP (an in-depth analysis of such interaction is shown in Susi et al., 2018) when the plasticity is active. In facts, in this case the weight $w_{D_i,ES_i}$ is instantaneously influenced in response to a new input parallel spike in the following way:

- Influence of $D_{i+1}$ on $D_i$

$$\begin{cases} \Delta w(D_i, ES_i) = A_+ e^{-\frac{\Delta t\, out_{D_{i+1},D_i}}{\tau_+}}, & \text{for } \Delta t\, out_{D_{i+1},D_i} > 0 \\ \Delta w(D_i, ES_i) = 0, & \text{for } \Delta t\, out_{D_{i+1},D_i} = 0 \\ \Delta w(D_i, ES_i) = A_- e^{\frac{\Delta t\, out_{D_{i+1},D_i}}{\tau_-}}, & \text{for } \Delta t\, out_{D_{i+1},D_i} < 0 \end{cases} \quad (9)$$

- Influence of $D_{i-1}$ on $D_i$

$$\begin{cases} \Delta w(D_i, ES_i) = A_+ e^{-\frac{\Delta t\, out_{D_{i-1},D_i}}{\tau_+}}, & \text{for } \Delta t\, out_{D_{i-1},D_i} > 0 \\ \Delta w(D_i, ES_i) = 0, & \text{for } \Delta t\, out_{D_{i-1},D_i} = 0 \\ \Delta w(D_i, ES_i) = A_- e^{\frac{\Delta t\, out_{D_{i-1},D_i}}{\tau_-}}, & \text{for } \Delta t\, out_{D_{i-1},D_i} < 0 \end{cases} \quad (10)$$

As extension of the MNSD, we apply these equations to all delay neurons. Note that, for the first and last branches of the structure, we apply only eq.9 or eq.10, respectively, since they have only one neighbor.

## 2.4. Neuron and Structure Settings

The neuron model used and most of the nMNSD settings presented in this work are based on the reference paper (Susi et al., 2018) and summarized in **Table 1**.

Importantly, as in Susi et al. (2018), we make two additional assumptions: (1) every time a new parallel input pattern arrives to the structure, all the neurons are at the resting potential $S_0$, and (2) we consider the STDP constants sufficiently small to avoid interaction among subsequent input sequences. These make possible to analyze the effect of each parallel input to the nMNSD structure separately, both for static and dynamic behaviors.

## 2.5. The Trapezoid Method

The introduction of a new input pattern to the nMNSD structure may make $T$ spike. In the affirmative, the spike will be generated after an interval depending on the *ttf*s introduced by delay neurons and target neuron. Then, a *multiple-input single-output* transfer function is associated to each nMNSD.

The *trapezoid method* provides an intuitive representation of the internal mechanism of the structure that allows us to geometrically decompose such transfer function. Looking at the decomposed version of the structure response, we can instantaneously know if a new parallel input train will make the nMNSD target spike, or how we can modify the nMNSD to make it happen. Using a dedicated 2-dimensional nomogram (see Appendix 1 in **Supplementary Material**), this geometrical method makes possible to evaluate the impact of an input pattern on the $S_T$ of a trained nMNSD of any dimensionality, directly at the input of the structure, without having to execute intermediate steps. This new representation of the detection process provides us with a visual feedback to easily customize the structure's settings to better fit the characteristics of the problem we are facing (e.g., degree of importance of single features in the class definition, compensation of expected feature variance, and so on).

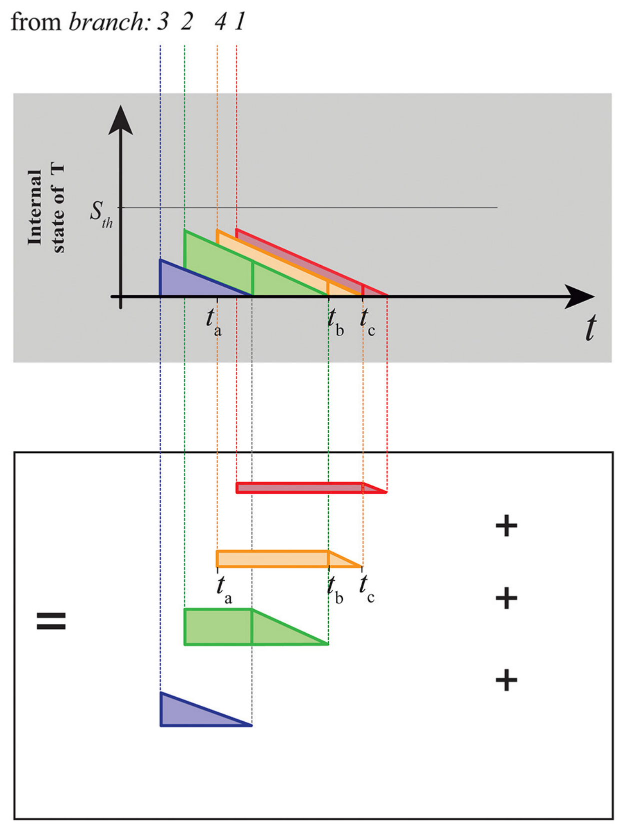

**FIGURE 5 |** Decomposition of the area subtended by the internal state of the target neuron shown in **Figure 4A** in terms of trapezoids. Considering the arrival of a generic contribution to the target neuron (e.g., the orange one in A), its advent (at the time $t_a$) will cause an immediate increase of the value of $S_T$ equals to the target weight associated to the afferent contribution. Its contribution on the $S_T$ can be considered an additive constant, until the decay due to the previous contribution is exhausted (interval $[t_a, t_b]$, i.e., rectangle part). After this, its contribution will start to decrease linearly (with slope $L_d$) until the resting potential is reached (interval $[t_b, t_c]$, i.e., triangle part).

The method envisages as first step to represent all the structure parameters that define its static behavior (as neuron parameters, PPT, and feature relevances), as well as the input arrivals to the target, at the input of the structure through $n$ right trapezoids, lying each one on a semi-plane with a real horizontal axis characterized by reversed times (increasing values toward the left, since we are portraying the parallel spike train on the entryway of the structure, see **Figure 3**). This method allows us to evaluate in a differentiated manner the contribution that each single i-th component of the current input parallel spike train (i.e., $ES_i$) will produce on the $S_T$. The possibility to analyze the nMNSD operation and to optimize the recognition using the simplified visual feedback given by the trapezoids lays the groundwork for the design of stratified nMNSD-based classification systems.

We will present in this section the rationale underlying the trapezoid decomposition, and how it can be obtained. Then, we will show the iter envisaged by the trapezoid method to execute the summation directly at the input of the structure, in two different versions: graphical method and analytical method.

#### 2.5.1. How the Trapezoids Are Drawn?

The trapezoids are geometrical entities that allow us to easily decompose the integration process that takes place into the LIFL in the underthreshold range, as shown in **Figure 5**. Each trapezoid incorporates information of both the branch of an nMNSD structure (neuronal parameters and input and output weights) and the parallel input that is going through the structure.

Indeed, the generic trapezoid associated to the branch $i$ is composed of a rectangle (on the left side) whose base $rect_i$ depends on the current input, adjacent to a right triangle (on the right side) whose base $tri_i$ depends on the structure parameters. The height of the *ith* trapezoid equals to the value of the corresponding *ith* output weight $w_{T,D_i}$. The abscissa (time point) of the left limit of the *ith* trapezoid, $\overline{v_i}$, is related to the input weights of the delay neurons. Taken as a whole, the $\overline{v_i}$ set represents the PPT of the structure.

Like the PPT of the structure, the trapezoids are not constrained to a fixed time point, since they should act at any time, i.e., irrespectively of when the parallel spike train is entering to the network (as noted in section 2.2). Then, without loss of generality, we represent the left extremity of the first trapezoid $\overline{v_1}$ as placed to the fictive value $c$ and relate the abscissae of the left sides of all the other trapezoids to this value, in the following manner:

$$\langle \overline{v_1}, \overline{v_2}, ..., \overline{v_i}, ..., \overline{v_n} \rangle = \tag{11a}$$

$$= \langle c, c + \overline{\Delta t_{ES_{1,2}}}, c + \overline{\Delta t_{ES_{1,3}}}, ..., c + \overline{\Delta t_{ES_{1,n}}} \rangle = \tag{11b}$$

$$= \langle c, c + \overline{\Delta t_{ES_{1,2}}}, c + \overline{\Delta t_{ES_{1,2}}} + \overline{\Delta t_{ES_{2,3}}}, ..., c + \overline{\Delta t_{ES_{1,2}}} + \overline{\Delta t_{ES_{2,3}}} + ... + \overline{\Delta t_{ES_{n-1,n}}} \rangle \tag{11c}$$

where:

$$\overline{\Delta t_{ES_{i,j}}} = \frac{1}{w_{D_j\ ES_j} - 1} - \frac{1}{w_{D_i\ ES_i} - 1} \tag{12}$$

(see Appendix 2 in **Supplementary Material**). Note that using Equation (11a) and (11c), we can relate the abscissae of the trapezoids to the PPT of the structure, in terms of preferential relative intervals $\langle \overline{\Delta t_{ES_{1,2}}}, \overline{\Delta t_{ES_{2,3}}}, ..., \overline{\Delta t_{ES_{n-1,n}}} \rangle$. In this way, we are operating a geometrical transformation to represent the $S_T$ diagram to the input of the nMNSD, taking in account the effect the *ES*s will undergo once they cross the structure.

We introduce the concept of *crossing order* $\langle ref_1, ref_2, ..., ref_k, ..., ref_n \rangle$, i.e., the set of integer numbers which indicates the sequence of the branch indices which spikes will progressively arrive to the target neuron (to give an example, $\langle 3, 2, 4, 1 \rangle$ in **Figure 5**). To decompose the target summation in trapezoids, the first geometrical object to be drawn is the one associated to the branch $ref_1$, i.e., the branch which contribution will arrive for first to the target. It will lack of the rectangle part, consisting then on a simple right triangle. It has a slope equals to the underthreshold decay of $T$ (i.e., $L_d$), as all the other triangles of the chart, so that:

$$tri_{ref_k} = \frac{w_{T,D_{ref_k}}}{L_d} \tag{13}$$

To draw the set of rectangles, we have to take in account the arrival order of the contributions to the target. Considering **Figure 5**, we note that the length of the rectangle associated to a generic branch is given by the interval between the arrival of its own contribution to the target and the end of the previously arrived trapezoid/triangle (i.e., $t_a$ and $t_b$ respectively, if we consider the orange trapezoid in **Figure 5**). Considering the correlate of such temporal distance at the input of the structure (transformed by the eq.12), we obtain the set of rectangle lengths by iterating the following formula, for $k = 2, 3, ..., n$:

$$rect_{ref_k} = rect_{ref_{k-1}} + tri_{ref_{k-1}} - (\overline{v_{ref_{k-1}}} - \overline{v_{ref_k}} + t_{ES_{ref_k}} - t_{ES_{ref_{k-1}}}) \tag{14}$$

where, obviously, $rect_{ref_{k-1}} = 0$ when we compute $rect_{ref_2}$. If the interval between two arrivals is long enough to allow $S_T$ to fully discharge, then the related trapezoid will lack of the rectangle part, as the first trapezoid. To obtain the complete set of trapezoids, we have just to complement the trapezoids, adding the triangles defined above at the right end of the rectangles generated.

#### 2.5.2. Iter Description

Now that we know how to draw the trapezoids on our chart, we can go back to analyze the passage of the train in the structure looking at the input of the structure only. We conceive the $n$ trapezoids placed on the abscissae $\langle \overline{v_1}, \overline{v_2}, ..., \overline{v_i}, ..., \overline{v_n} \rangle$, and the current input parallel spike train, characterized by the absolute times $\langle t_{ES_1}, t_{ES_2}, ..., t_{ES_n} \rangle$. For ease of representation, we consider the spike train in motion toward the structure (from left to right, see **Figure 6A**, left), locked on their Cartesian references, as if they too were moving toward the nMNSD.

Following this method, we developed an interactive visualization and optimization system of nMNSD structures, available at www.github.com/LCCN/Frontiers2021.

##### *2.5.2.1. Graphical Method*

From the parameters of the nMNSD, we are able to draw the PPT of the structure on the trapezoid chart; once we know the input parallel input pattern, we can complement the graph with the related trapezoid set as described above. The method envisages as preliminary step the detection of the *crossing order*. We can alternatively visualize it as the sequence of the branch indices which input spikes will progressively cross the left side of their related trapezoid, during the entrance toward the structure. The crossing event related to $ES_i$ allows us to represent the spike contribution on the target neuron from the branch $i$. Note that the crossing order does not reflect the indices of the branches, which *ESs* ordinately arrive to the related delay neuron, but the sequence of the target arrivals evoked by the *ESs* (the one thing does not imply the other). The crossing order can be graphically individuated by rigidly translating the parallel input train toward the trapezoids, and

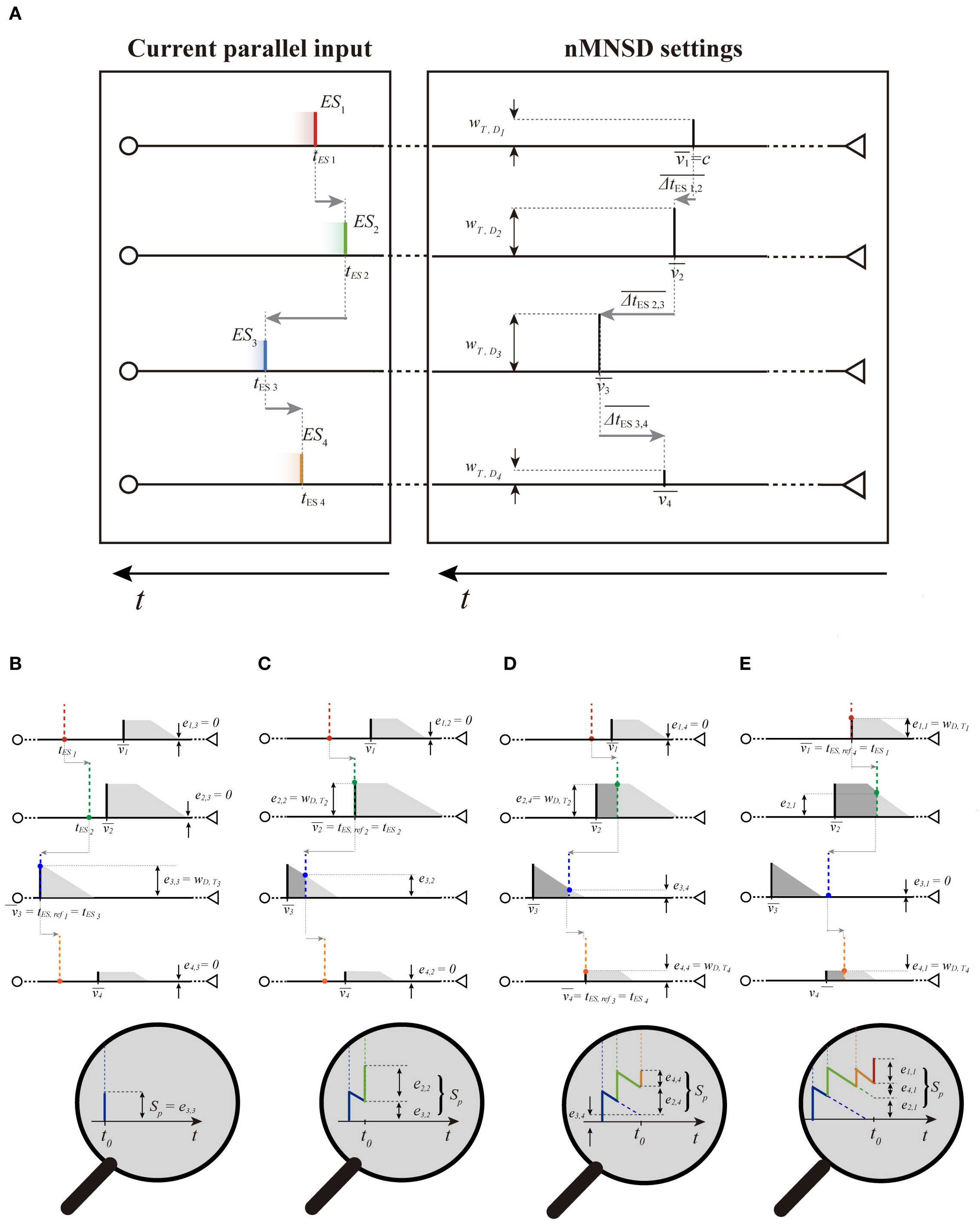

**FIGURE 6 |** Target summation executed with the trapezoid method. **(A)** The PPT of a parallel spike train of order 4 (left) moving toward the preferential parallel train (PPT) of the nMNSD settings; the two blocks are here represented separately for ease of understanding. Note that the time axis has to be considered reversed in both the blocks (as in **Figure 3**), since we are portraying the parallel spike train in motion. Envisaging the parallel train sliding toward right we can notice that ESs will cross the PPT of the structure with the following ES order (i.e., the following crossing order): $\langle 3, 2, 4, 1 \rangle$. In **(B–E)**, we give a "stop-motion" representation of the target summation executed with the trapezoid method, showing both the parallel spike train and the trapezoid set of the structure on the same chart (with related zoomed representation of $S_T$ below). Except for the trapezoid associated to the first crossing step (which always lacks the rectangle part, as explained in the text), a generic trapezoid has the rectangle length determined by the time difference between its arrival and the end of the trapezoid associated to the last contribution arrived to the target; once the rectangle parts have been drawn, the triangle parts can consequently traced to the right of the rectangle parts to obtain the complete trapezoids. Using the trapezoid method, the efficacies $e_{i,ref_k}$ to the state of the target neuron $S_T$ are directly noticeable on the input of the structure, since they are represented by the heights of the intersections between the ES and the upper perimeter of the related trapezoid (colored dots). At this point, the summation can be easily decomposed in the contribution of each branch. In the zoom below, the reader can ascertain for the first three crossing steps ($ref_1 = 3$, $ref_2 = 2$, $ref_3 = 4$) that the computation of the $S_p$ through the trapezoid method is equivalent to the one obtained by the classical target summation at the target neuron.

reporting the sequence of crossing (see **Figures 6B–E**). Each *crossing step* of the $ES_i$ on the associated trapezoid (left extremity) represents the arrival of the contribution from the branch of order $i$ to the target; each crossing step corresponds to a relative maximum on the membrane potential of the target $S_T$ (i.e., a summation peak $S_p$). Since $S_T$ depends in general also on the past arrivals, taking in account a generic crossing step $ref_k$, we can represent the related $S_{p,ref_k}$ as the sum of the residual contributions given by each target arrival. Such contributions are called *target efficacies* (indicated as $e_{ES_{i,ref_k}}$) and are represented by the heights of the intersections of the prolongations of each $ES$ of the train with the related trapezoid. Note that a set of target efficacies $\langle e_{ES_{1,ref_k}}, e_{ES_{2,ref_k}}, ..., e_{ES_{n,refn}} \rangle$ (and consequently a $S_p$) can be calculated for each one of the $n$ crossing steps. Obviously, since the value $ref_k$ is the branch number which related spike arrives $k - th$ to $T$, when we consider the crossing step associated to $ES_i$, its target efficacy is represented by the full $w_{T,D_i}$ (i.e., $e_{ES_{i,ref_k}} = w_{T,D_{ref_k}}$ when $ref_k = i$). That said, for each crossing step, $S_{p,ref_k}$ can be calculated by summing up the $e_{i,ref_k}$ related to all the parallel spike train components. For a complete parallel spike train, we call $S_{p,M}$ the maximum among all the $S_p$s. If in almost one of these steps, the value $S_{p,ref_k}$ is greater than $S_{th}$, then the target neuron will produce a spike, signaling the detection of the input pattern by the nMNSD structure. The process is illustrated graphically in **Figure 6**, and analytically in **Supplementary Table 1** of the Appendix 2.

### 2.5.3. Will the Target Produce a Spike? And When? Beyond the Behavior of an Isolated nMNSD

As already shown in section 2.2, a spike will be produced by the target neuron only if, at least for one of the crossing events, results that $S_{p,ref_k} > S_{th}$, signaling the recognition of the pattern. Analytically speaking, each new incoming pattern will be able to evoke an output spike on the target only if the following condition is satisfied:

$$S_{p,M} = max(S_{p,ref_1}, S_{p,ref_2}, ..., S_{p,ref_k}, ..., S_{p,ref_n}) \geq S_{th} \tag{15}$$

The spike will be generated after a proper time-to-fire $ttf_T$ (Equation 2). To calculate this value, we have to discern among two cases:

- Case 1: No $ES$ of the parallel spike train will arrive to $T$ during it is in the active mode. In this case, the $ttf_T$ can be easily calculated using Equation (2), considering $S_{p,M}$ as $S$.
- Case 2: Some $ES$ of the parallel spike train will arrive to $T$ during it is already in the active mode (see **Figure 7**). Such case necessitates further attention; see (Susi et al., 2018, **Supplementary Material**, Par.1).

In order to avoid overcomplicating the analysis, we study here the first case, assured by imposing the following condition on the output weights:

$$(\sum_{i=1}^{n} w_{T,D_i}) - w_{T,D_j} < S_{th}, \text{ for each } j \text{ chosen between } [1, n] \tag{16}$$

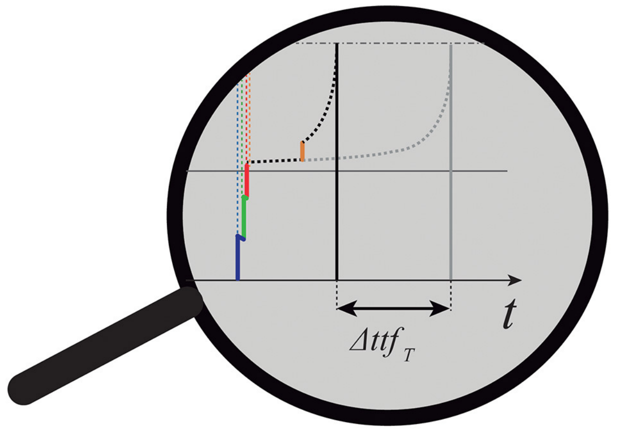

**FIGURE 7 |** Peculiar case of target summation in which one *ES* arrives to *T* while it is already in the active mode.

On the other hand, their sum should be at least equal to $S_{th}$, as previously stated (see Equation 8). The time of spike generation gives us a score of the goodness of the input, and it can be used as further information to build more complex configurations of nMNSD, as discussed in section 2.7.

## 2.6. Shape of the Hypervolume

Among the many architectural parameters of the nMNSD, we can differentiate between those involved in the recognition of a pattern and those involved in its subsequent signaling. As indicated in section 2.3, the input weights represent the PPT of the nMNSD, then they have to do with the actual recognition of the input pattern. In the n-dimensional feature space, the PPT of the nMNSD is represented by the line $\zeta$, with slope of 45° with respect of each of the axes and passing through the point determined by the following coordinates:

$$\frac{1}{w_{D_1,ES_1} - 1}, \frac{1}{w_{D_2,ES_2} - 1}, \frac{1}{w_{D_3,ES_3} - 1}, ..., \frac{1}{w_{D_n,ES_n} - 1} \tag{17}$$

(for an in-depth explanation, see Susi et al. (2018)). To understand the signaling phase of a pattern, let us consider a volume in the n-dimensional feature space (i.e., a hypervolume), consisting of an augmentation of the PPT such that if the set of arrival times of a pattern falls into it, the MNSD produces a spike. Such augmentation represents the tolerance of the structure, i.e., the error margin the nMNSD admits from the PPT of the parallel input to consider it recognized. To modify the hypervolume, we can act on the target weights and $L_d$ (of the target neuron):

- Target weights allow to introduce a selective tolerance with respect to a single feature. This is useful when we have fluctuations of the values of a determined involved feature;
- $L_d$ modulates the tolerance of the structure with respect to all its features: the higher (lower) the $L_d$, the more selective (robust) the structure becomes to the jitter. If we have equals $w_{T,D_1}$, we have a hypercylinder as hypervolume, whose radius depends on $L_d$.

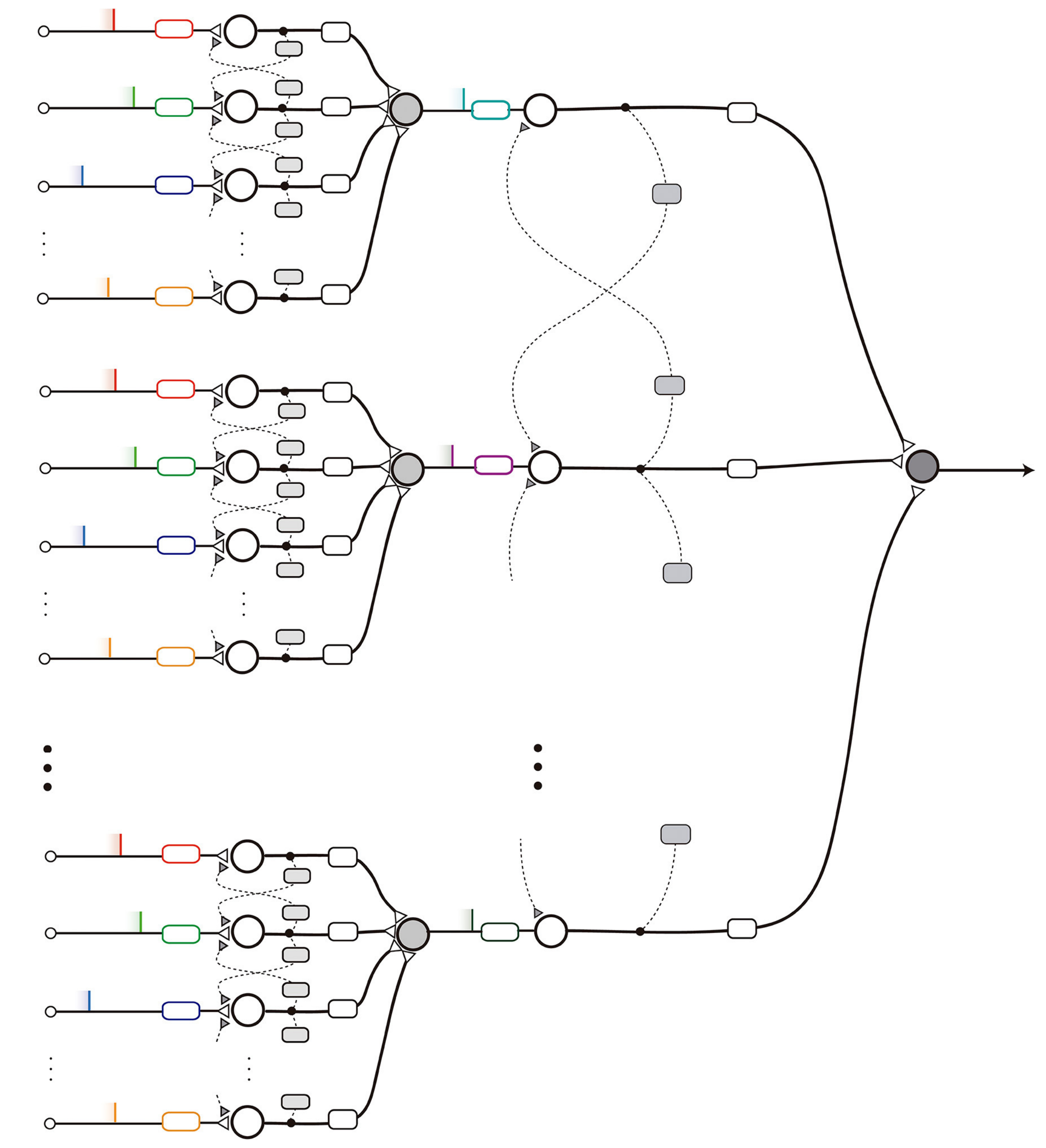

**FIGURE 8 |** Scheme of a layered nMNSD-based system.

## 2.7. Toward Complex nMNSD-Based Classification Systems: Multiclass Classification and Layering

The nMNSDs can be used for multiclass classification by assigning an nMNSD to each class of the problem. The training can be made separately for each nMNSD structure, paying attention to sufficiently differentiate the domains of the classes, to avoid that more than a target will fire for the same input, causing indeterminacy of the belonging of the pattern to one of the two classes. Since each structure is represented by its own trapezoid set, the refinement of the hypervolumes associated to the classes can be achieved by minimizing the intersection area between the different trapezoid sets. In cases where the intersection of classes is unavoidable, the evaluation of the $ttf_T$ can be informative since an anticipated spike is often representative of a better fit with the class represented by the structure that produced it.

Another configuration can be obtained by parallelizing the nMNSD structures to analyze their output together. Each nMNSD of the same layer will produce a spike on a certain instant, which together will form a new parallel spike sequence.

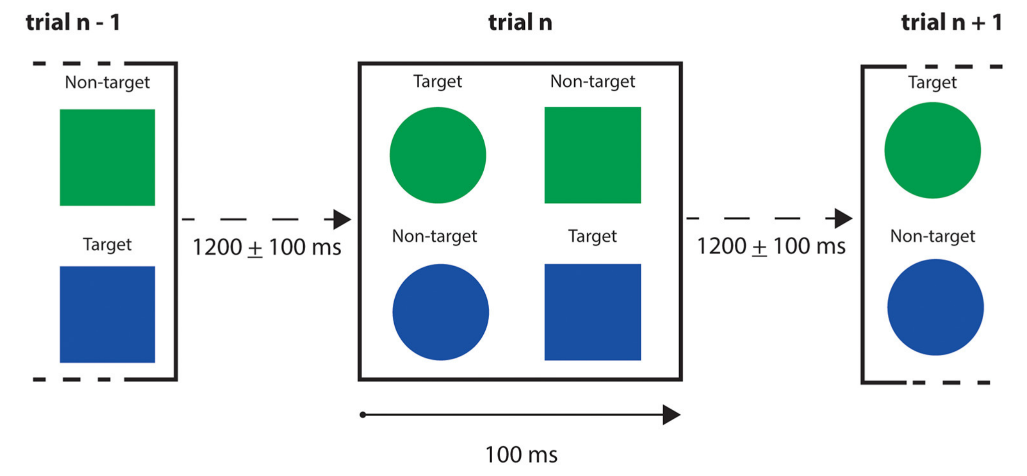

**FIGURE 9 |** Go/No-Go trials schematic representation. Trial's exposition time for the stimuli was 100 ms, with a *stimulus onset asynchrony* (i.e., the time interval between two trials) between 1,100 and 1,300 ms. The stimuli presented were blue and green circles and squares. Participant was instructed to press a button when a blue square or a green circle appeared, and not to press for green squares and blue circles.

nMNSDs of subsequent layers will be gradually trained on the parallel spike sequences evoked by previous levels. Note that in each layer one should adapt the parallel patterns to fall within the proper range of action (see **Figure 8**).

## 3. RESULTS

In order to show the improved usability of the nMNSD tool to tackle pattern recognition problems with respect to the MNSD, we test the structure on two specific datasets. First, we face again the classification problem presented in Susi et al. (2018) (section 3.2) concerning the recognition of No-Go patterns from a patient executing the Go/No-Go paradigm, and finally we benchmark the nMNSD on static inputs using the MNIST dataset of handwritten digits to find pros and cons compared to other SNN-based classification methods. The simulations are performed using an event-driven implementation of the nMNSD structures in Matlab2019b environment.

### 3.1. Go/No-Go Paradigm

The data of this first test concern a Go/No-Go paradigm characterized by a 70/30 presentation ratio. The study is ongoing and conducted at the Laboratory of Cognitive and Computational Neuroscience of Madrid. It involves 67 healthy and right-handed subjects (age range 13–17 years old), without previous history of psychiatric or neurological conditions, neither psychopharmacological treatment nor drug intake. High-density magnetoencephalography (MEG) signals were obtained from 306 channels (102 pairs of planar gradiometers and 102 magnetometers) with an Elekta Neuromag Vectorview system situated in a magnetically and electrically shielded room. Only the 102 Magnetometers were used to carry out the analysis.

The MEG data from the subject with the highest performance in both Go and No-Go conditions have been chosen for further analysis (we selected the best performer to minimize the risk of having unintentional or random responses in our dataset). We finally considered for each trial the time interval $< 400ms$ after the stimulus presentation to exclude the premotor response (Deecke et al., 1976; Ikeda et al., 2000). A schematic representation of the task is given in **Figure 9**, and a resume of the subject performance in **Table 2**.

**TABLE 2 |** Go/No-Go task.

| Total inhibitions | Correct inhibitions | Accuracy inhibitions | Accuracy response | Averaged reaction time |
|---|---|---|---|---|
| 103 | 101 | 98.05% | 100% | 546.43 ms |

*Resume of the performance of the selected participant.*

We preliminarily selected for each of the two classes (No-Go and Go trials) the networks of informative magnetometers, evaluating the patterns of occurrence of absolute maxima in the reference time window. We evaluated their discriminative power by computing their intra-class stability and inter-class independence. In **Figure 10**, the processing pipeline performed for each of the classes is illustrated. On the bases of previous studies (Amirali et al., 2018), we quantified stability and discriminative power of the magnetometers using different versions of the time series (broadband unfiltered time-series, *alpha*-filtered time series, and *theta*-filtered time series), obtaining better results with the original broadband time series. To differentiate the two classes, we finally chosen the eight magnetometers indicated with the IDs listed in **Table 3**.

The nMNSD has been trained (75 trials for each class) and then tested (20 trials for each class) to recognize No-Go patterns using the instants corresponding to the maximum amplitudes in the time window of the premotor response. We repeated a whole training cycle (learning + validation) varying the parameters $L_d$ [0.005, 0.08], $A_{+/-}$ [0.001, 0.04] and $\tau_{+/-}$ [5, 8] to find the optimal nMNSD settings (which we call *best learners*). Using an

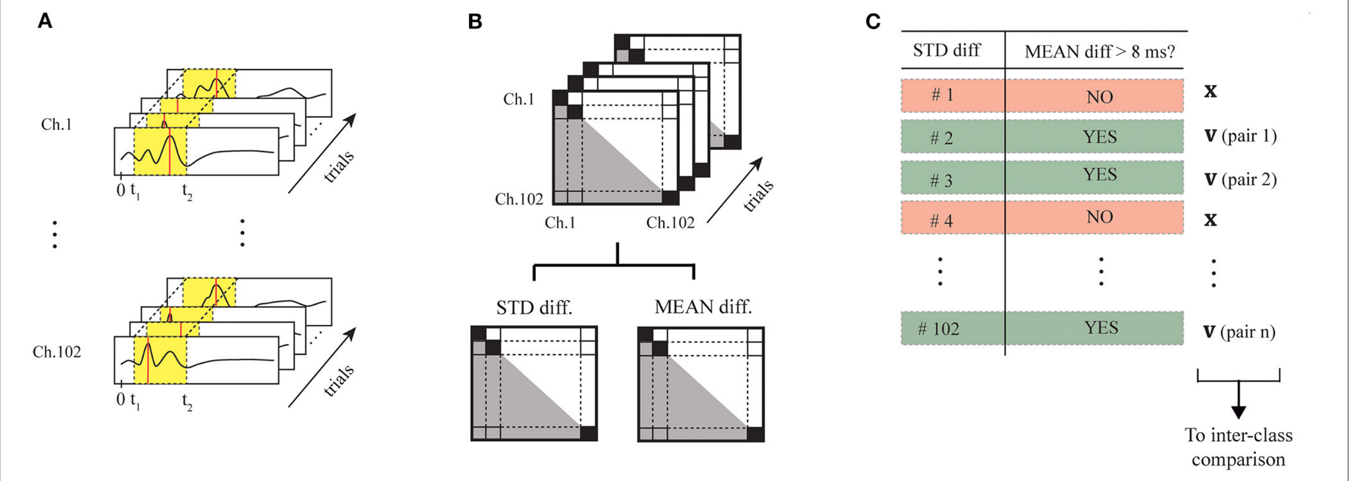

**FIGURE 10 |** Processing pipeline for the selection of the eight informative magnetometers. Considering the time series during the window of interest (in yellow), we executed the following steps first for the No-Go trials and then for the Go trials: **(A)** detected the maximum peak on each MEG channel, for each of the trials; **(B)** we computed the time difference considering each combination of magnetometers, and extracted mean and standard deviation of the time difference on each pair of MEG sensors along the trials; **(C)** we selected the couples of sensors starting from those characterized by the lowest standard deviations and a mean $\geq 10ms$. Inter-class discriminative power has been then guaranteed by comparing the 2 sets of sensor couples selected for the 2 classes, and removing possible pairs of sensors duplicated in the 2 classes if they presented the same mean.

**TABLE 3 |** List of the IDs of the selected MEG magnetometers representative of the two classes: No-Go and Go.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Magnetometer ID:** | 0231 | 0241 | 0911 | 0921 | 1521 | 1541 | 1631 | 1911 |

*Intel(R) Core(TM) i5-8250U CPU, 8 GB RAM*, each complete training cycle lasted an average of 10 ms, and the recognition of a single No-Go pattern lasted only 0.1 ms. The simulations individuated a rich area with a satisfactory trade-off between average Accuracy, Precision, and Recall (up to 0.75, 0.76, and 0.72, respectively).

In **Figure 11**, we represent the classification performance obtained with the different settings of the trained nMNSD, and in **Figure 12** the representation on the trapezoid chart of the trained No-Go structure, when crossed with a No-Go trial and then with a Go trial.

Beside the fact that this represents an improvement with respect to the recognition performances achieved in our previous work, we presented a new methodology, able to take into account a greater number of features, and the representation of the trained structure in the trapezoid chart. Importantly, since we analyzed the time series during the premotor response ($< 400ms$ while average response of the subject happened at 546.4 ms), and the delay of our system is negligible, our method allows to perform the classification before the action is made from the subject. This opens up interesting scenarios for the action control in critical decisions.

In addition to the visualization of the nMNSD settings with respect to a given parallel spike train, the realized toolbox leaves room for the improvement of the tuning of the structure through the positions of the *trapezoids* and allows the user to analyze the *feature relevances* to differentiate the impacts evoked by the different features of the problem. Obviously, this method can be extended to other types of problems, even in areas other than neuroscience.

## 3.2. MNIST Database

In this second test, we adapted the nMNSD structure to classify 2D static inputs using the grayscale images of MNIST database (LeCunn et al., 1998). The original images are composed of 28 × 28 pixels and each pixel spans the range [0–255]. We created two additional subsampled versions of the images by grouping neighboring pixels in *fields*. Images are encoded in the input layer through the time-to-first-spike of the assigned neurons, using the following intensity-to-latency relation:

$$t_i = \frac{I_{max} - I_i}{I_{max}} \cdot 25 \tag{18}$$

where $I_{max}$ is the maximum value of intensity of a pixel/field, $I_i$ the actual value of intensity of the pixel/field, and 25 ms of simulated time is the maximum latency encoded (corresponding to an "unwritten" pixel/field ). This does not need any of the preprocessing steps commonly used in SNNs (e.g., Gabor filters). We repeated a whole training cycle (learning + validation) varying the parameters $L_d$, $A_{+/-}$ and $\tau_{+/-}$ in the same ranges cited in section 3.1 to find the optimal nMNSD settings (i.e., the parameters of the *best learners*).

We implemented three versions of the nMNSD network (i.e., A, B, and C), illustrated in **Figure 13**. While in configuration A we simply attributed each pixel of the original image to one different input neuron of the nMNSD, in configurations B and C we grouped the branches in adjacent fields composed of 4 × 4 and 7 × 7 pixel each, respectively, and assigned to them one different input neuron of the nMNSD, using the mean value

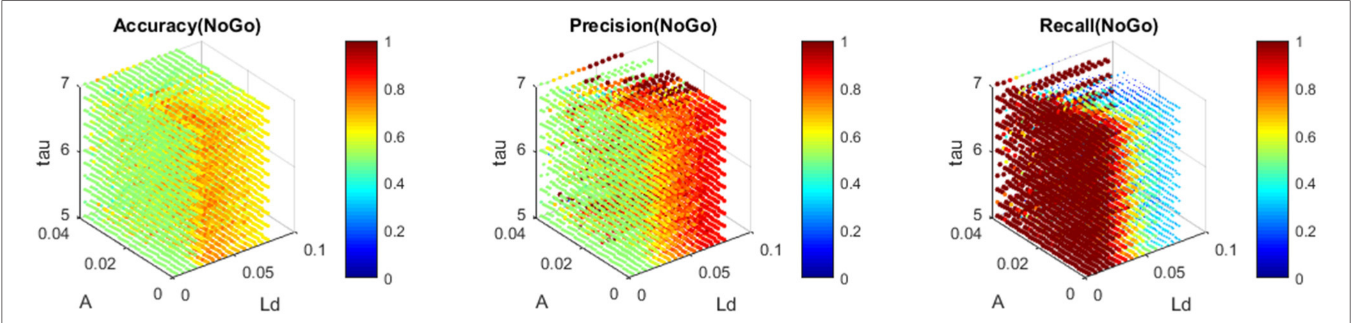

**FIGURE 11 |** Accuracy, Precision, and Recall obtained with the trained nMNSD (No-Go structure) around the optimal parameters. Each of the colored point represents the performance of the nMNSD trained with the parameters indicated in the axes.

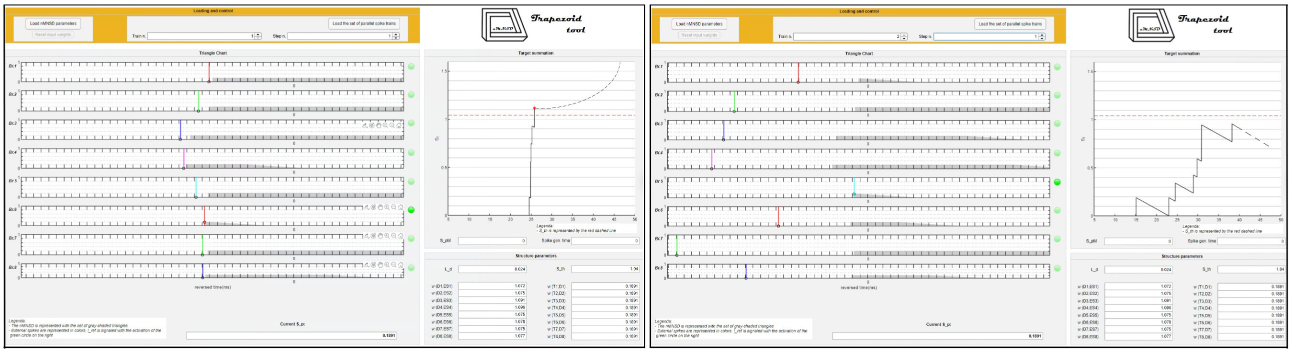

**FIGURE 12 |** Representation on the trapezoid chart of the No-Go structure after the training, when crossed by 2 parallel spike trains. **Left**: The No-Go structure is stimulated with a No-Go trial (correctly recognized). **Right**: The No-Go structure is stimulated with a Go trial (that is not recognized by the structure).

of intensity of the field and coded this number to the input neurons using Equation (18). Heterosynaptic lateral connections are activated between adjacent neurons during learning and disabled during test.

We tested the nMNSD considering the two strategies *one-versus-all* (OvA) and *one-versus-one* (OvO).

For the OvA, the recognition system is composed of one nMNSD structure, which is trained to detect a digit vs. the complementary ones. We considered the digit "1" for simplicity, since it is the one with the greater number of examples: $n_{LEARNING}$=6742, $n_{TEST}$=1134. For each combination of the parameters $L_d$, $A_{+/-}$, and $\tau_{+/-}$ explored, we executed both training and test with balanced quantities of examples for preferred and non-preferred classes (for learning: 6742 images for digit "1" and as many images among the digits $[0, 2-9]$, considering the first 60000 trials of the MNIST dataset; for test: 1134 images for digit "1," and as many among the remaining trials for the other digits, considering the last 10000 trials of the dataset). Once chosen the structure parameters and the input weights (best learners), we optimized the output weights using the *fminsearch* algorithm (based on the *Nelder–Mead* search method Nelder and Mead, 1965; Lagarias et al., 1998), obtaining an average improvement of 5% in accuracy, arriving to values of 79, 82, and 93% for the A, B, C topologies, respectively. The results are shown in **Table 4**.

For the OvO, the recognition system is here composed of two parallel 4 × 4 nMNSD structures of type C, each one trained to recognize one digit. We executed the training considering 5421 images per class. Then we optimized the output weights, and finally tested the network using 837 images per class. To resolve the cases in which more than 1 target produces a spike during the recognition, we evaluated two different strategies: (1) we considered recognizing the digit associated to the target which produces a spike for first; (2) we used 5 redundancy nMNSDs for each class and provided each target with a bundle of "muting" connections that aim to inhibit the complementary final target during its latency period. The topology for this latter variant, which we call *OvO enhanced*, is described in **Figure 13E**. In this second case, the learning has been carried out initializing input weights and internal states to random values to broaden the differentiation between the redundancy modules. Both these strategies increased the system performances; while the first strategy results faster to train, the second needs an additional supervised step (for the inhibitory neurons), but leads to better results. The latter structure achieved an accuracy of 95%.

The learning times for each training cycle (using the same computer specified in section 3.1) showed significant variations with respect to the topology used. The results are shown in **Table 5**. Differently, the classification time for each presented pattern do not varies substantially with respect to the topology used and is equal to $27 \pm 8\mu s$.

Regarding the OvA configuration, the grouping operation leads to a drastic reduction of the learning times and to an increase of accuracy. Regarding the OvO configuration, the

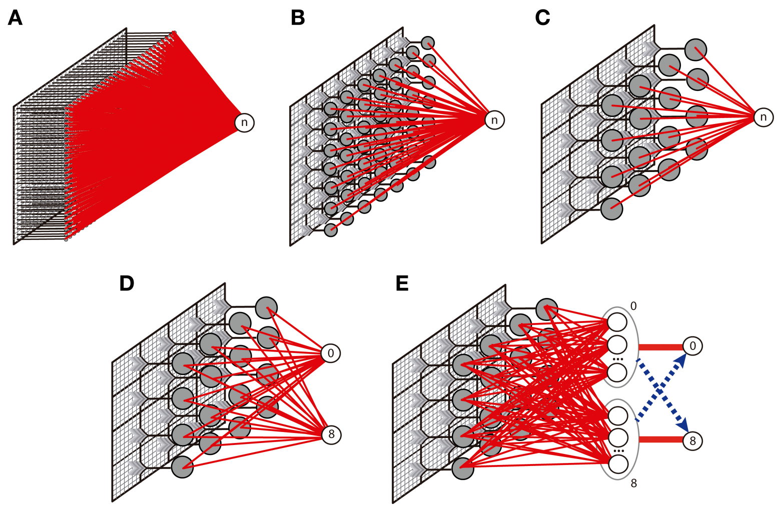

**FIGURE 13 |** Description of the different network topologies tested. In **(A)**, the nMNSD is fed with the original image (28 × 28 pixels); in **(B)**, the pixels are grouped in 49 (7 × 7) fields; in **(C)**, the pixels are grouped in 16 (4 × 4) fields. one-versus-one (OvO): **(D)** Original configuration and **(E)** enhanced version (with inhibitory bundles in blue).

**TABLE 4 |** Comparison of one-versus-all (OvA) performance obtained with the three structures implemented.

| Configuration | Learning time (for training cycle) | Accuracy |
|---|---|---|
| A | 300*s* | 0.79 |
| B | 1.35*s* | 0.82 |
| C | 0.4*s* | 0.93 |

**TABLE 5 |** Comparison of one-versus-one (OvO) performances obtained with the two methodologies implemented (best accuracies).

| Configuration | Learning + optimization times (for training cycle) | Accuracy |
|---|---|---|
| D | [0.4 + 5]*s* | 0.91 |
| E (OvO enhanced) | [0.45 + 6.5]*s* | 0.95 |

enhanced version reported the best results. The values of accuracy are notable considering the very simple network configuration (17 neurons and 32 connections for the OvA-*C*, 28 neurons and 180 connections for the OvO-*E*).

Compared to existing character recognition systems based on SNN, the nMNSD showed a state of the art level accuracy. At the same time, the system is composed of a very limited number of neurons and synapses if compared to today's SNN-based recognition systems (see Kheradpisheh and Masquelier, 2020) for a useful report) and the recognition process consists of only one spike per neuron. Taken together, these characteristics testify the proneness of the nMNSD to time- and energy-efficiency, two of the most desirable features in electronic systems (see Göltz et al., 2020).

## 4. DISCUSSIONS AND CONCLUSIONS

We presented the nMNSD structure, which is a generalization of the MNSD to any number of inputs. After the explication of the structure operation, we discussed some aspects to improve the practical usability of the nMNSD as classifier tool. In addition, we have illustrated the *trapezoid method* that is a reduced method to analyze the recognition mechanism operated by the nMNSD in response to a specific input sequence. This method is useful to represent the internal processing mechanism of a specific nMNSD, especially when the number of branches is greater than three, i.e., when the feature space becomes not trivially representable. A visualization toolbox based on this method is available at www.github.com/LCCN/Frontiers2021.

The first application showed in this paper regards the classification of brain signals. We applied the nMNSD to a problem previously faced with the classical MNSD from the same authors, showing the improvement in classification performances achieved with the nMNSD structure with respect to the simple MNSD, due to the possibility of encoding a greater number of features, and the support in the "neural design" brought by the visualization tool realized. Finally, we tested the performance of the nMNSD on the MNIST dataset. Regarding this application, we developed different versions of the nMNSD, including one based on a redundancy layer, which showed the best performance in terms of accuracy, at the cost of a greater number of spikes in the network, which would results in greater power dissipation in hardware implementations. Nevertheless, considering that the encoding strategy of the nMNSD is based on single spikes and not on spike-rate, the dissipated power would still be low and the impact is minimal, so the benefits of this change far outweigh the negatives.

Among the most important highlights on the presented method, there are:

- The possible use of the information given by the latency of the target neuron in case of indetermination in a multiclass scenario,
- The possibility to encode the impact of each feature in the target summation for the determination of a class (i.e., *feature relevance* property),
- The configuration of different nMNSDs to be used in a multiclass problem.

Recent trend in sensor technologies are pointed to resource-efficient algorithms for high-speed learning and online recognition, to be implemented directly on the sensors (Kasetty et al., 2008; Iacoviello et al., 2015; Cardarilli et al., 2020). Our results show that the nMNSD proficuously taken in consideration for pattern recognition applications requires low power consumption and low training times.The simplicity and low computational cost of this methodology suggest a large-scale implementation for real-time learning and recognition applications in several areas.

## DATA AVAILABILITY STATEMENT

The datasets generated for this study are available on request from the corresponding author.

## ETHICS STATEMENT

The studies involving human participants were reviewed and approved by Complutense University of Madrid. Written informed consent to participate in this study was provided by the participants' legal guardian/next of kin.

## AUTHOR CONTRIBUTIONS

GS and CM designed the computational framework. GS, FM, and EP conceptualized the paper. LA-T and FM provided and analyzed brain data. GS, LA-T, FM, EP, and CM wrote the paper. All authors contributed to the article and approved the submitted version.

## FUNDING

LA-T acknowledges financial support by UCM-SANTANDER scholarship to PhD students (Ref. C2017). FM acknowledges the Community of Madrid through the project Madrid Neurocentro (B2017/BMD-3760). EP and CM acknowledge support from MINECO and Ministerio de Ciencia e Innovación (Spain) through projects TEC2016-80063-C3 and PID2019-111537GB-C22/AEI/10.13039/501100011033. The work of CM is partially supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 899265-ADOPD. CM acknowledges the Spanish State Research Agency, through the María de Maeztu Program for Units of Excellence in R&D (MDM-2017-0711).

## ACKNOWLEDGMENTS

We wanted to thank the colleagues of CTB and IFISC for their helpful comments.

## SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnins.2021.582608/full#supplementary-material

## REFERENCES

Adibi, P., Meybodi, M. R., and Safabakhsh, R. (2005). Unsupervised learning of synaptic delays based on learning automata in an RBF-like network of spiking neurons for data clustering. *Neurocomputing* 64, 335–357. doi: 10.1016/j.neucom.2004.10.111

Ambard, M., and Rotter, S. (2012). Support vector machines for spike pattern classification with a leaky integrate-and-fire neuron. *Front. Comput. Neurosci.* 6:78. doi: 10.3389/fncom.2012.00078

Amirali, V., Moritz, M., Andres, N., Ann-Kathrin, S., and Christian, B. (2018). Machine learning provides novel neurophysiological features that predict performance to inhibit automated responses. *Sci. Rep.* 8:16235. doi: 10.1038/s41598-018-34727-7

Brückmann, A., Klefenz, F., and Wünsche, A. (2004). A neural net for 2D-slope and sinusoidal shape detection. *Int. Sci. J. Comput.* 3, 21–26. Available online at: https://www.frontiersin.org/articles/10.3389/fncom.2018.00037/full

Cardarilli, G. C., Cristini, A., Di Nunzio, L., Re, M., Salerno, M., and Susi, G. (2013). "Spiking neural networks based on LIF with latency: simulation and synchronization effects," in *2013 Asilomar Conference on Signals, Systems and Computers* (Pacific Grove, CA: IEEE), 1838–1842. doi: 10.1109/ACSSC.2013.6810620

Cardarilli, G. C., Di Nunzio, L., Fazzolari, R., Re, M., and Spano, S. (2020). AW-SOM, an algorithm for high-speed learning in hardware self-organizing maps. *IEEE Trans. Circ. Syst. II Express Briefs* 67, 380–384. doi: 10.1109/TCSII.2019.2909117

Chase, S. M., and Young, E. D. (2007). First-spike latency information in single neurons increases when referenced to population onset. *Proc. Natl. Acad. Sci. U.S.A.* 104, 5175–5180. doi: 10.1073/pnas.0610368104

Deecke, E., Grozinger, B., and Kornhuber, H. H. (1976). Voluntary finger movements in man: cerebral potentials and theory. *Biol. Cybern.* 23, 99–119. doi: 10.1007/BF00336013

Fields, R. (2008). White matter in learning, cognition and psychiatric disorders. *Trends in Neurosci.* 31, 361–370. doi: 10.1016/j.tins.2008.04.001

Fields, R. (2015). A new mechanism of nervous system plasticity: activity-dependent myelination. *Nat. Rev. Neurosci.* 16, 756–767. doi: 10.1038/nrn4023

FitzHugh, R. (1955). Mathematical models of threshold phenomena in the nerve membrane. *Bull. Math. Biol.* 17, 257–278. doi: 10.1007/BF02477753

Florian, R. V. (2012). The chronotron: a neuron that learns to fire temporally precise spike patterns. *PLoS ONE* 7:e40233. doi: 10.1371/journal.pone.0040233

Göltz, J., Baumbach, A., Billaudelle, S., Kungl, A. F., Breitwieser, O., Meier, K., et al. (2020). "Fast and deep neuromorphic learning with first-spike coding," in *NICE '20: Proceedings of the Neuro-inspired Computational Elements Workshop* (Heidelberg: ACM). doi: 10.1145/3381755

Grassia, F., Levi, T., Doukkali, E., and Kohno, T. (2017). Spike pattern recognition using artificial neuron and spike-timing-dependent plasticity implemented on a multi-core embedded platform. *Artif. Life Robot.* 23, 200–204. doi: 10.1007/s10015-017-0421-y

Heelan, C., Lee, J., Shea, R., Lynch, L., Brandman, D. M., Truccolo, W., et al. (2019). Decoding speech from spike-based neural population recordings in secondary auditory cortex of non-human primates. *Commun. Biol.* 2:466. doi: 10.1038/s42003-019-0707-9

Hwu, T., Wang, A., Oros, N., and Krichmar, J. (2018). Adaptive robot path planning using a spike neuron algorithm with axonal delays. *IEEE Trans. Cogn. Dev. Syst*. 10, 126–137. doi: 10.1109/TCDS.2017.2655539

Iacoviello, D., Petracca, A., Spezialetti, M., and Placidi, G. (2015). A real-time classification algorithm for eeg-based bci driven by self-induced emotions. *Comput. Methods Prog. Biomed*. 122, 293–303. doi: 10.1016/j.cmpb.2015.08.011

Ikeda, A., Ohara, S., Matsumoto, R., Kunieda, T., Nagamine, T., Miyamoto, S., et al. (2000). Role of primary sensorimotor cortices in generating inhibitory motor response in humans. *Brain* 123, 1710–21. doi: 10.1093/brain/123.8.1710

Izhikevich, E. M. (2004). Which model to use for cortical spiking neurons? *IEEE Trans. Neural Netw*. 15, 1063–1070. doi: 10.1109/TNN.2004.832719

Izhikevich, E. M. (2006). Polychronization: computation with spikes. *Neural Comput*. 18, 245–282. doi: 10.1162/089976606775093882

Kasetty, S., Stafforda, C., Walkera, G., Wang, X., and Keogh, E. (2008). "Real-time classification of streaming sensor data," in *20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2008)* (Dayton, OH: IEEE). doi: 10.1109/ICTAI.2008.143

Kheradpisheh, S., 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

Koyama, S., Eden, U. T., Brown, E. N., and Kass, R. E. (2010). Bayesian decoding of neural spike trains. *Ann. Inst. Stat. Math*. 62, 37–59. doi: 10.1007/s10463-009-0249-x

Kulkarni, S., Simon, S., and Sundareswaran, K. (2013). A spiking neural network (SNN) forecast engine for short-term electrical load forecasting. *Appl. Soft Comput*. 13, 3628–3635. doi: 10.1016/j.asoc.2013.04.007

Lagarias, J., Reeds, M., Wright, M., and Wright, P. (1998). Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM J. Optimizat*. 9, 112–147. doi: 10.1137/S1052623496303470

LeCunn, Y., Cortes, C., and Burgues, C. (1998). *The MNIST Database*. Available online at: http://yann.lecun.com/exdb/mnist/

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

Matsubara, T. (2017). Conduction delay learning model for unsupervised and supervised classification of spatio-temporal spike patterns. *Front. Comput. Neurosci*. 11:104. doi: 10.3389/fncom.2017.00104

Mattia, M., and Del Giudice, P. (2000). Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses. *Neural Comput*. 12, 2305–2329. doi: 10.1162/089976600300014953

Minneci, F., Kanichay, R. T., and Silver, R. A. (2012). Estimation of the time course of neurotransmitter release at central synapses from the first latency of postsynaptic currents. *J. Neurosci. Methods* 205, 49–64. doi: 10.1016/j.jneumeth.2011.12.015

Mount, C., and Monje, M. (2017). Wrapped to adapt: Experience-dependent myelination. *Neuron* 95, 743–756. doi: 10.1016/j.neuron.2017.07.009

Nazari, S., and Faes, K. (2019). Spiking pattern recognition using informative signal of image and unsupervised biologically plausible learning. *Neurocomputing* 330, 196–211. doi: 10.1016/j.neucom.2018.10.066

Nelder, J., and Mead, R. (1965). A simplex method for function minimization. *Comput. J*. 7, 308–313. doi: 10.1093/comjnl/7.4.308

Nogueira, W., Katai, A., Harczos, T., Klefenz, F., Buechner, A., and Edler, B. (2007). "An auditory model based strategy for cochlear implants," in *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (Lyon), 4127–4130. doi: 10.1109/IEMBS.2007.4353244

Rudnicki, M., Zuffo, M. K., and Hemmert, W. (2012). Sound decoding from auditory nerve activity. *Front. Comput. Neurosci*. 92. doi: 10.3389/conf.fncom.2012.55.00092

Salerno, M., Susi, G., and Cristini, A. (2011). "Accurate latency characterization for very large asynchronous spiking neural networks," in *Bioinformatics 2011 - Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms*, eds M. Pellegrini, A. L. N. Fred, J. Filipe, and H. Gamboa (Rome: SciTePress), 116–124.

Schofield, A. J., Gilchrist, I. D., Bloj, M., Leonardis, A., and Bellotto, N. (2018). Understanding images in biological and computer vision. *Interface Focus* 8:20180027. doi: 10.1098/rsfs.2018.0027

Stratton, P., Cheung, A., Wiles, J., Kiyatkin, E., Sah, P., and Windels, F. (2012). Action potential waveform variability limits multi-unit separation in freely behaving rats. *PLoS ONE* 7:e38482. doi: 10.1371/journal.pone.0038482

Susi, G., Antón Toro, L., Canuet, L., López, M. E., Maestú, F., Mirasso, C. R., et al. (2018). A neuro-inspired system for online learning and recognition of parallel spike trains, based on spike latency, and heterosynaptic STDP. *Front. Neurosci*. 12:780. doi: 10.3389/fnins.2018.00780

Taherkhani, A., Belatreche, A., Li, Y., and Maguire, L. P. (2015). DL-resume: a delay learning-based remote supervised method for spiking neurons. *IEEE Trans. Neural Netw. Learn. Syst*. 26, 3137–3149. doi: 10.1109/TNNLS.2015.2404938

Tapson, J., Cohen, G., Afshar, S., Stiefel, K., Buskila, Y., Hamilton, T., et al. (2013). Synthesis of neural networks for spatio-temporal spike pattern recognition and processing. *Front. Neurosci*. 7:153. doi: 10.3389/fnins.2013.00153

Wang, X., Lin, X., and Dang, X. (2019). A delay learning algorithm based on spike train kernels for spiking neurons. *Front. Neurosci*. 13:252. doi: 10.3389/fnins.2019.00252

Zai, A. T., Bhargava, S., Mesgarani, N., and Liu, S.-C. (2015). Reconstruction of audio waveforms from spike trains of artificial cochlea models. *Front. Neurosci*. 9:347. doi: 10.3389/fnins.2015.00347

Zhou, Y., Mesik, L., Sun, Y. J., Liang, F., Xiao, Z., Tao, H. W., et al. (2012). Generation of spike latency tuning by thalamocortical circuits in auditory cortex. *J. Neurosci*. 32, 9969–9980. doi: 10.1523/JNEUROSCI.1384-12.2012

**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 GC declared a past co-authorship with one of the authors GS to the handling editor.