

NiSNN-A: Noniterative Spiking Neural Network With Attention With Application to Motor Imagery EEG Classification

Chuhan Zhang^{1b}, Graduate Student Member, IEEE, Wei Pan^{1b}, Member, IEEE,
and Cosimo Della Santina^{1b}, Senior Member, IEEE

Abstract—Motor imagery (MI), an important category in electroencephalogram (EEG) research, often intersects with scenarios demanding low energy consumption, such as portable medical devices and isolated environment operations. Traditional deep learning (DL) algorithms, despite their effectiveness, are characterized by significant computational demands accompanied by high energy usage. As an alternative, spiking neural networks (SNNs), inspired by the biological functions of the brain, emerge as a promising energy-efficient solution. However, SNNs typically exhibit lower accuracy than their counterpart convolutional neural networks (CNNs). Although attention mechanisms successfully increase network accuracy by focusing on relevant features, their integration in the SNN framework remains an open question. In this work, we combine the SNN and the attention mechanisms for the EEG classification, aiming to improve precision and reduce energy consumption. To this end, we first propose a noniterative leaky integrate-and-fire (NiLIF) neuron model, overcoming the gradient issues in traditional SNNs that use iterative LIF neurons for long time steps. Then, we introduce the sequence-based attention mechanisms to refine the feature map. We evaluated the proposed noniterative SNN with attention (NiSNN-A) model on two MI EEG datasets, OpenBMI and BCIC IV 2a. Experimental results demonstrate that: 1) our model outperforms other SNN models by achieving higher accuracy and 2) our model increases energy efficiency compared with the counterpart CNN models (i.e., by 2.13 times) while maintaining comparable accuracy.

Index Terms—Attention mechanism, electroencephalogram (EEG) classification, motor imagery (MI), spiking neural networks (SNNs).

I. INTRODUCTION

ELECTROENCEPHALOGRAMS (EEGs), whether captured through noninvasive electrodes on the scalp or directly via invasive devices, are the cornerstone of the rapidly evolving domain of brain-computer interfaces (BCIs).

Received 9 December 2023; revised 5 September 2024; accepted 22 January 2025. Date of publication 14 March 2025; date of current version 6 August 2025. This work was supported in part by the European Union's Horizon Europe Program from Project EMERGE under Grant 101070918 and in part by the TU Delft AI Labs Programme. The work of Wei Pan was supported by the European Union's Horizon 2020 Research and Innovation Programme under Grant 951847. (Corresponding author: Wei Pan.)

Chuhan Zhang and Cosimo Della Santina are with the Department of Cognitive Robotics, Faculty of Mechanical Engineering, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: C.Zhang-8@tudelft.nl; C.DellaSantina@tudelft.nl).

Wei Pan is with the Department of Computer Science, The University of Manchester, M13 9PL Manchester, U.K. (e-mail: wei.pan@manchester.ac.uk). This article has supplementary downloadable material available at <https://doi.org/10.1109/TNNLS.2025.3538335>, provided by the authors.

Digital Object Identifier 10.1109/TNNLS.2025.3538335

Therefore, the accurate classification of EEG signals has attracted substantial attention over the years—with applications ranging from advanced neurorehabilitation techniques [1] to diagnostics and real-time health monitoring [2]. Within the realm of BCI, motor imagery (MI) holds a distinctive place [3]. MI refers to the mental imagination of specific movements by subjects, leading to the generation of distinct EEG patterns. Accurately classifying these EEG signals opens up application possibilities in advanced fields such as robot control and assistive technologies [4].

Recently, deep learning (DL) methods, such as convolutional neural networks (CNNs) [5], recurrent neural networks (RNNs) [6], and Transformers [7] have received increasing interest as a mean for classifying EEG signals [8], [9], [10], [11]. At present, in addition to conventional CNNs or RNNs, DL of EEG incorporates many other technologies [12], [13], [14], [15], [16]. Attention mechanisms [17], inspired by human cognitive processes, enhance the model's focus on relevant features, facilitating more efficient and accurate feature extraction [18]. Attention mechanisms have been successfully applied to EEG classification. More details are presented in Section II-B. However, all of these methods suffer from the current limitation of high energy consumption, which poses a significant barrier to deployment in low energy scenarios, such as edge devices for healthcare [19] or robot control [20].

In this work, we investigate the use of spiking neural networks (SNNs) for EEG classification. This technique mimics how biological neurons operate, allowing it to be used to interpret natural neuronal signals [21]. SNNs also offer a promising avenue for reducing energy consumption due to their event-based nature [22], [23], [24], which is attractive in view of edge applications. More details on the state of the art of SNN in EEG classification are provided in Section II-C. We propose a novel integration of SNNs and the attention mechanism, especially for EEG classification. At the core of our approach sits a newly proposed noniterative leaky integrate-and-fire (NiLIF) neuron, which approximates the neural dynamics of the biological LIF and mitigates the gradient problem by avoiding long-term dependencies. The second methodological contribution is the sequence-based attention model for EEG data, which can simultaneously obtain the attention scores of feature maps. The noniterative SNN with attention (NiSNN-A) models boost both the efficiency and accuracy of the execution process. It is

worth noting that some work has investigated attention SNNs, which, however, specifically targeted computer vision (CV) [25]. As we show in our experimental comparison, this method is not suitable for classifying long-term data such as EEG.

The contributions of this article can be summarized as follows.

- 1) We propose a novel NiLIF neuron model for SNNs, which eliminates the gradient problem caused by long-term dependencies while retaining the biology-inspired temporal properties of the LIF model.
- 2) We introduce a sequence-based attention mechanism for SNNs, improving the classification accuracy.
- 3) We show the combination of NiSNNs with attention mechanisms for MI EEG classification, simultaneously achieving high accuracy and reducing energy consumption.

The rest of this article is organized as follows. Section III introduces our proposed NiSNN-A models. Section IV gives the experimental details. The results and discussion are conducted in Section V. Section VI concludes this article.

II. RELATED WORKS

In this section, the related works are introduced. Section II-A illustrates the application of DL techniques specific to EEG signal processing. Subsequently, Section II-B illustrates the works with attention mechanisms. Finally, the applications of SNN in EEG classification are described in Section II-C.

A. DL Methods for EEG

In recent years, integrating DL techniques into EEG classification has gained significant traction [26]. CNN stands out among these techniques, particularly due to its ability to identify spatial-temporal patterns within the complex, multidimensional EEG data [27], [28], [29]. For instance, a temporal-spatial CNN was employed for EEG classification in [30] and [31]. The work in [32] introduced EEGNet, which integrates a separable convolutional layer following the temporal and spatial modules. Enhancing the CNN architecture, Huang et al. [33] and Humayun et al. [34] incorporated residual blocks for classification. Another noteworthy approach is presented in [35], where an autoencoder is built upon the CNN framework. Furthermore, this work adopted a subject-independent training paradigm, emphasizing its scalability on varied EEG data from different subjects. Apart from CNNs, long short-term memory (LSTM) networks have also been recognized for EEG classification due to their inherent ability to process time sequences effectively [36], [37], [38]. These methods are limited in high energy consumption, making them difficult to use on some edge devices with low-energy requirements.

B. Attention Mechanism

The Transformer architecture and attention mechanism, originally introduced in [17] for natural language processing tasks, have seen increasing adoption in many other machine learning tasks [39], [40]. These methods are now being explored in the field of EEG classification, given their ability

to handle time sequence data. The attention mechanism is particularly important for EEG data analysis. It holds the potential to enhance classification accuracy and emphasizes specific segments of the data, offering deeper insights into EEG signal characteristics. Various attention models have emerged in the field of EEG classification. For instance, Liu et al. [13] present a spatial and temporal attention model integrated with CNN. This approach leverages two distinct CNN modules to derive spatial and temporal attention scores separately, subsequently using four convolutional layers for classification. Meanwhile, Zhang et al. [41] apply a multihead attention module, as described in [17], combined with five convolutional layers to classify EEG signals. Direct applications of the Transformer attention structures from [17] for EEG classification are evident in [42] and [43]. Beyond solely leveraging CNN and attention mechanisms, some studies have integrated additional techniques. For instance, Luo et al. [44] introduce a mirrored input approach combined with an attention model that operates across each data record. In a more intricate approach, Ma et al. [45] deploy the spatial, spectral, and temporal Transformers, each catering to a different input data type. A popular EEG processing methodology, time-frequency common spatial pattern (TFCSP), is highlighted in [46]. This method intertwines a two-layer CNN and an attention model, feeding their concatenated outputs into a classifier. A notable trend is the use of global attention in conjunction with three sequential models, as shown in [47], [48], and [49]. These works employ three sequential attention models, each dedicated to a single dimension. By utilizing global average pooling (GAP), they efficiently diminish unrelated dimensions and consolidate the attention scores across the three output dimensions. These models effectively achieved the goal of EEG signal classification. However, these methods have the limitation of high energy consumption due to the use of attention CNN, making them difficult to use on some edge devices with low-energy requirements. Also, they lack a special focus on data from different channels and different time areas, which is important in EEG data. Attention mechanisms for SNNs specific to CV tasks [25] are discussed in Section III-B. Cai et al. [50] also proposed a vision-based SNN attention mechanism, in which spatial-channel-temporal-fused attention works at each time step and layer of the spike trains, providing strong robustness. However, Cai et al. [50] require attention calculation at each time step in an iterative manner, which takes a long time to execute due to a large number of loops.

C. Spiking Neural Networks

In recent years, SNNs have garnered increasing attention within the neural computing community with broad applications, such as CV and robot control [23], [51], [52], [53], [54]. Their growing significance can be attributed to their closer resemblance to biological neural systems than CNNs. SNNs, by simulating the discrete, spike-based communication found in actual neurons, promise enhanced efficiency and energy savings. However, this bioinspired approach comes with challenges, particularly during training. The gradient back-propagation, a staple in training CNNs, presents difficulties for SNNs due to their nondifferentiable spiking nature. To address

this, various training methods have been proposed [55], [56]: from leveraging evolutionary algorithms to adjust synaptic weights [57], to employing biologically inspired synaptic update rules like spiking-time dependent plasticity (STDP) [58]. Some researchers have also explored converting a well-trained CNN into their SNN counterparts [59], while others have advocated for using surrogate gradients for continuing backpropagation [60]. Given the biology-inspired and efficient attributes of SNNs, several works have successfully applied SNNs in the domain of EEG signal classification, showcasing their potential in real-world applications. For example, Antelis and Falcon [61] employed particle swarm optimization as an evolutionary algorithm for weight updates, combined with an unsupervised classifier like K -nearest neighbors and multilayer perceptron; however, their approach was not end-to-end and involved manual feature extraction. Studies that utilized STDP include [62], which integrated manual feature extraction methods such as fast Fourier transform and discrete wavelet transform with a 3-D SNN reservoir and supervised classifiers. Similarly, Kasabov and Capecci [63] adopted an unsupervised learning framework, implementing a 3-D SNN reservoir model. Wu et al. [64] combined the 3-D reservoir with a support vector machine as the classifier. Highlighting conversion techniques, Yan et al. [65] explored a tree structure, demonstrating the energy efficiency benefits of SNNs through a CNN-to-SNN conversion, while another work by Yan et al. [66] utilized power spectral density for feature extraction before such a conversion. Back propagation-like methodologies also found their adaptation in SNNs and EEG classification realm with work [67] using SpikeProp [68]. SpikeProp uses solutions of dynamics equations to represent the membrane potentials and uses backpropagation to calculate the timing of output spikes. However, SpikeProp considers a single input spike and output spike while our NiLIF model allows arbitrary inputs and multiple output spikes. In particular, the inputs to the NiLIF neuron can even be continuous values. Liao et al. [15] were notably the first to employ directly trained SNNs for EEG signal classification. However, these methods have limitations when deepening the networks and seeking to learn complex representations. There are also some works using parallel techniques in SNNs [69] considering the membrane potential without a reset mechanism, which is different from our method.

III. METHODS

In this section, we introduce the novel NiLIF neuron in Section III-A. Subsequently, the proposed attention models are delineated in Section III-B. Finally, Section III-C describes the network architecture of the proposed NiSNN-A.

A. LIF Neuron

Neurons serve as the fundamental components of neural networks. In this section, the iterative LIF neuron model is presented in Section III-A1. Then, our proposed NiLIF neuron model is detailed in Section III-A2. Finally, the comparisons are discussed in Section III-A3.

1) *Background (Iterative LIF Neuron Model)*: In the SNN community, the leaky integrate-and-fire (LIF) neuron model is widely used. It strikes a balance between simplicity and

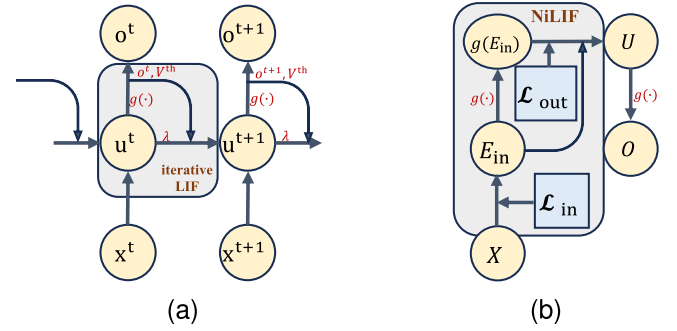


Fig. 1. Iterative LIF neuron model and the NiLIF neuron model. (a) Iterative LIF neuron model. The membrane potential u^{t+1} is computed recurrently, with each time step depending on its previous state u^t and the output spikes o^t from the preceding time step. (b) NiLIF neuron model. The matrix \mathcal{L}_{in} is used to calculate the input stimulus and accumulation processes, and the matrix \mathcal{L}_{out} is used to approximate the output spikes. The final membrane potential U and output spikes O are obtained simultaneously rather than iteratively.

biologically inspired characteristics. The LIF model can be described using the following equations:

Membrane Potential:

$$\begin{cases} \tau \frac{du(t_c)}{dt_c} = -u(t_c) + wx(t_c), & \text{if } u(t_c) \leq V_{th} \\ \lim_{\Delta \rightarrow 0^+} u(t_c + \Delta) = u_{res}, & \text{if } u(t_c) > V_{th} \end{cases}$$

$$u_{res} = \begin{cases} u(t_c) - V_{th}, & \text{with soft reset mechanism} \\ u_r, & \text{with hard reset mechanism} \end{cases}$$

Spike Generation:

$$o(t_c) = g(u_c)$$

$$g(a) = \begin{cases} 0, & \text{if } a \leq V_{th} \\ 1, & \text{if } a > V_{th} \end{cases} \quad (1)$$

where $\tau \in \mathbb{R}$ is the membrane time constant and $u(t_c) \in \mathbb{R}$ represents the neuron's membrane potential at continuous time $t_c \in \mathbb{R}$. $w x(t_c) \in \mathbb{R}$ is the input stimulus at time $t_c \in \mathbb{R}$, represented as the weighted input to the present layer in the neural network. w is the trainable parameter. $V_{th} \in \mathbb{R}$ is the membrane potential threshold. Specifically, when the membrane potential exceeds the threshold V_{th} , a spike is produced. When the membrane potential remains below the threshold, no spike is generated. After generating a spike, the membrane potential decreases to a reset potential $u_{res} \in \mathbb{R}$. Two types of reset mechanisms deal with the membrane potential after generating spikes: soft and hard reset. The soft reset mechanism resets the membrane potential by reducing the threshold potential V_{th} ; while the hard reset mechanism resets the membrane potential to a defined potential value $u_r \in \mathbb{R}$. $o(t_c) \in \mathbb{R}$ represents the output spike at time t_c . The function $g(\cdot)$ is the Heaviside step function, which describes the spike firing process.

To adapt to the requirements of backpropagation in neural networks, an iterative LIF neuron model [60] with the soft reset mechanism is introduced

$$\begin{aligned} u^{t+1} &= \lambda(u^t - V_{th}o^t) + wx^t \\ o^t &= g(u^t) \end{aligned} \quad (2)$$

where $\lambda \in \mathbb{R}$ denotes the decay rate of the membrane potential. We use u^t to represent $u(t_c = t)$, where $t \in \mathbb{N}$ represents the discrete time step in the iterative LIF neuron model. Similarly, o^t means $o(t_c = t)$ and x^t means $x(t_c = t)$. In this way, the membrane potential updates step by step, recurrently making the LIF neuron dynamics trainable by a network. However, the Heaviside step function $g(\cdot)$ in the firing process makes it nondifferentiable. This characteristic brings challenges for gradient backpropagation. To address this limitation, surrogate functions have been proposed [70]. Nowadays, the sigmoid functions are commonly employed as surrogate functions due to their capability to emulate the spike firing process, especially when associated with a high value of α : $\text{Sigmoid}(x) = (1/(1 + e^{-\alpha x}))$. Therefore, the gradient backpropagation in the iterative LIF with the sigmoid functions can be described as follows:

$$\frac{\partial o}{\partial u} = \frac{\partial \text{Sigmoid}(u)}{\partial u} = \text{Sigmoid}(u)(1 - \text{Sigmoid}(u))\alpha. \quad (3)$$

2) *Noniterative LIF Neuron Model*: As described in (2), the membrane potential u^{t+1} in iterative LIF depends on the output o^t and membrane potential u^t of the preceding time steps, which introduces complex neuron dynamics and long-term dependencies in gradient propagation [71]. Therefore, for cases with long time steps, the gradient problem caused by long-term dependencies hinders the model's high performance. Thus, we propose an NiLIF model designed to approximate the neuron dynamics and avoid long-term dependencies.

The LIF model has input stimulation, accumulation, and firing processes. To achieve these functions, the core idea behind the NiLIF model is first to consider the stimulation and accumulation effects over all time steps, then approximate the firing process with maximum sparsity, and finally obtain the approximate neuron dynamics without long-term dependencies.

The effects on time step t of the input stimulation without firing at time step i can be computed by solving different equation (1): $u^t = wx^i e^{-((t-i)/\tau)\delta t}$, where δt is the fixed time increment used to discretize the continuous time domain. We utilize the function $\mathcal{L}(\cdot)$ to represent the leaky component as $\mathcal{L}(t) = e^{-(t/\tau)\delta t}$. Therefore, the effects of all previous time steps on time step t is expressed as follows:

$$\epsilon_{\text{in}}^t = \sum_{i=0}^t wx^i \mathcal{L}(t-i) \quad (4)$$

where ϵ_{in}^t represents the input stimulus and accumulation effects without firing. Naturally, (4) can be represented using matrix format

$$E_{\text{in}} = X\mathcal{L}_{\text{in}} \quad (5)$$

where $E_{\text{in}} \in \mathbb{R}^{1 \times (t_n+1)}$ is defined as the effect matrix, $X \in \mathbb{R}^{1 \times (t_n+1)}$ is defined as the weighted input matrix, and $\mathcal{L}_{\text{in}} \in \mathbb{R}^{(t_n+1) \times (t_n+1)}$ is defined as the leaky matrix

$$\begin{aligned} E_{\text{in}} &= [\epsilon_{\text{in}}^0 \quad \epsilon_{\text{in}}^1 \quad \cdots \quad \epsilon_{\text{in}}^{t_n}] \\ X &= [wx^0 \quad wx^1 \quad \cdots \quad wx^{t_n}] \\ \mathcal{L}_{\text{in}} &= \begin{bmatrix} 1 & \mathcal{L}(1) & \cdots & \mathcal{L}(t_n) \\ 0 & 1 & \cdots & \mathcal{L}(t_n-1) \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}. \end{aligned} \quad (6)$$

Therefore, the stimulus and accumulation dynamics of the LIF neuron are represented.

In addition, we approximate the firing dynamics of the LIF neuron. We propose to use the term ϵ_{out} to represent the impact of output spikes from the previous time steps on the current time step t

$$\epsilon_{\text{out}}^t = \sum_{i=0}^{t-1} o^i V_{\text{th}} \mathcal{L}(t-i-1) \quad (7)$$

where $o^i V_{\text{th}}$ represents the soft reset mechanism. When no spike is generated ($o^i = 0$), $o^i V_{\text{th}}$ is 0, which means no output spike effect at this time step i . When a spike is generated at time step i ($o^i = 1$), the membrane potential in the next step will decrease by the threshold V_{th} , which will affect the current membrane potential with a leakage coefficient $\mathcal{L}(t-i-1)$. (7) can be represented using matrix format

$$E_{\text{out}} = O\mathcal{L}_{\text{out}} \quad (8)$$

where $E_{\text{out}} \in \mathbb{R}^{1 \times (t_n+1)}$ is defined as the output spikes effect matrix, $O \in \mathbb{R}^{1 \times (t_n+1)}$ is defined as the output spikes matrix, and $\mathcal{L}_{\text{out}} \in \mathbb{R}^{(t_n+1) \times (t_n+1)}$ is defined as the output leaky matrix

$$\begin{aligned} E_{\text{out}} &= [\epsilon_{\text{out}}^0 \quad \epsilon_{\text{out}}^1 \quad \cdots \quad \epsilon_{\text{out}}^{t_n}] \\ O &= [o^0 \quad o^1 \quad \cdots \quad o^{t_n}] \\ \mathcal{L}_{\text{out}} &= \begin{bmatrix} 0 & V_{\text{th}} & V_{\text{th}}\mathcal{L}(1) & \cdots & V_{\text{th}}\mathcal{L}(t_n-1) \\ 0 & 0 & V_{\text{th}} & \cdots & V_{\text{th}}\mathcal{L}(t_n-2) \\ 0 & 0 & 0 & \cdots & V_{\text{th}}\mathcal{L}(t_n-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}. \end{aligned} \quad (9)$$

Therefore, the firing process dynamics are represented. The neuron dynamics consisting of input stimulation, accumulation, and firing process are expressed as follows:

$$\begin{aligned} u^t &= \epsilon_{\text{in}}^t - \epsilon_{\text{out}}^t \\ &= \sum_{i=0}^t wx^i \mathcal{L}(t-i) - \sum_{i=0}^{t-1} o^i V_{\text{th}} \mathcal{L}(t-i-1) \end{aligned} \quad (10)$$

with the matrix format as follows:

$$U = E_{\text{in}} - E_{\text{out}} = E_{\text{in}} - O\mathcal{L}_{\text{out}}. \quad (11)$$

In (10), the output spikes o are the only part that needs to be solved. Given the accurate membrane potential U and the Heaviside function $g(\cdot)$, we have the following identity:

$$O = g(U) = g(E_{\text{in}} - O\mathcal{L}_{\text{out}}) \quad (12)$$

where only O is the unknown variable. To solve (11), we propose the following proposition.

Proposition 1: Given an LIF neuron with stimulus, accumulation, and firing dynamics (11), the inequality

$$g(E_{\text{in}} - I_1\mathcal{L}_{\text{out}}) \leq O \leq g(E_{\text{in}}) \quad (13)$$

always holds where $O \in \mathbb{R}^{1 \times (t_n+1)}$ is the output spike matrix, $I_1 \in \mathbb{R}^{1 \times (t_n+1)}$ is the all-ones matrix and \mathcal{L}_{out} is defined in (6). *Proof*: We use (12) to prove Proposition 1. Because $O = g(U)$, we have $\min(g(U)) \leq O \leq \max(g(U))$. This implies $g(\min(U)) \leq O \leq g(\max(U))$. Given $U = E_{\text{in}} - O\mathcal{L}_{\text{out}}$, we can write

$$g(\min(E_{\text{in}} - O\mathcal{L}_{\text{out}})) \leq O \leq g(\max(E_{\text{in}} - O\mathcal{L}_{\text{out}})). \quad (14)$$

Then, we have

$$g(E_{\text{in}} - \max(O)\mathcal{L}_{\text{out}}) \leq O \leq g(E_{\text{in}} - \min(O)\mathcal{L}_{\text{out}}). \quad (15)$$

Since O represents the binary output spikes matrix, it follows that $I_0 \leq O \leq I_1$. Thus, we obtain

$$g(E_{\text{in}} - I_1\mathcal{L}_{\text{out}}) \leq O \leq g(E_{\text{in}} - I_0\mathcal{L}_{\text{out}}) \quad (16)$$

which finally simplifies as Proposition 1. \square

Thus, we use Proposition 1 to estimate O in (11) as $U = E_{\text{in}} - \hat{O}\mathcal{L}_{\text{out}}$, where \hat{O} is the estimated output spikes. To ensure the sparsity of the output spikes (i.e., to minimize O), it is essential to consider the case of minimal U . Therefore, the membrane potential U is estimated as follows:

$$\begin{aligned} U &= E_{\text{in}} - \max(O)\mathcal{L}_{\text{out}} = E_{\text{in}} - g(E_{\text{in}})\mathcal{L}_{\text{out}} \\ O &= g(U). \end{aligned} \quad (17)$$

Therefore, the neuron dynamics of the NiLIF model with the soft reset mechanism can be represented as follows:

$$\begin{aligned} u^t &= \sum_{i=0}^t wx^i \mathcal{L}(t-i) \\ &\quad - \sum_{i=0}^{t-1} g\left(\sum_{k=0}^i wx^k \mathcal{L}(i-k)\right) V_{\text{th}} \mathcal{L}(t-1-i), \\ o^t &= g(u^t). \end{aligned} \quad (18)$$

We aim to preserve the input gradient values without introducing significant neuron changes. Therefore, we utilize the derivative as follows during backpropagation:

$$\frac{\partial o}{\partial u} = \begin{cases} 1, & \text{if } 0 < u < 1 \\ 0, & \text{else.} \end{cases} \quad (19)$$

The pseudocode of the NiLIF model is shown in Algorithm 1. We present the derivations of the dynamics in this section.

3) *Comparison (Iterative LIF and NiLIF Neurons)*: The flow diagrams for the two neurons are shown in Fig. 1. The iterative LIF neuron operates in a recurrent manner, relying on the output from the preceding time step for its next computation. Conversely, the NiLIF model simultaneously processes data from all time steps, requiring only a few matrix operations to determine the membrane potential and output spikes for all time steps. The nonloop characteristic of the NiLIF neuron could avoid the gradient issues caused by long-term dependencies. We assume the loss is $L \in \mathbb{R}^{1 \times (t_n+1)}$, which is equal to the summation of the loss $l^t \in \mathbb{R}$ of each time step t . The gradient derivation for both LIF neuron models during backpropagation is shown as follows:

$$\frac{\partial L}{\partial w} = \sum_{t=0}^{t_n} \frac{\partial l_t}{\partial w} = \sum_{t=0}^{t_n} \frac{\partial l_t}{\partial o^t} \frac{\partial o^t}{\partial u^t} \frac{\partial u^t}{\partial w}. \quad (20)$$

Therefore, we introduce two propositions regarding $(\partial u^t / \partial w)$ in two LIF neuron models, respectively, to compare the gradient issues. Proposition 2 shows the gradient equation of the iterative LIF neuron, and Proposition 3 introduces the gradient equation of the NiLIF neuron.

Algorithm 1 Pseudocode for the NiLIF Model

Input Time steps t , threshold v , decay constant τ , input x .
function L_IN_MATRIX(t, τ)
 $e \leftarrow \text{zeros}((t, t))$
for $i, j \in \text{range}(t)$ **do**
 if $j > i$ **then**
 $e[i][j] \leftarrow \exp(-((j-i)/\tau))$
 end if
 if $i = j$ **then** $e[i][j] = 1$ **end if**
end for
return e
end function
function L_OUT_MATRIX(e, v)
 $s \leftarrow \text{zeros}(e)$
for $i, j \in \text{range}(t)$ **do**
 if $j > i$ **then** $s[i][j] = e[i][j-1]v$ **end if**
end for
return s
end function
procedure Ni_LIF
 function __INIT__(t, v)
 $e \leftarrow \text{L_IN_MATRIX}(t, \tau)$, $s \leftarrow \text{L_OUT_MATRIX}(e, v)$
 end function
 function FORWARD(x)
 $u \leftarrow xe$, $o \leftarrow (u > v)s$, $u \leftarrow u - o$, $o \leftarrow (u > v)$
 return o
 end function
end procedure

Proposition 2: Given an iterative LIF neuron with the dynamics in (2), the limit condition

$$\forall u_1^t, \quad \lim_{t \rightarrow \infty} \frac{\partial u_1^t}{\partial w} = 0 \quad (21)$$

always holds, where u_1^t is the membrane potential of the iterative LIF neuron at the time step t .

Proof: In the iterative LIF neuron model, $(\partial u_1^t / \partial w)$ is derived in (26), as shown at the bottom of the next page, which is composed of a summation of two parts of accumulated gradient multiplication $\prod_{i=0}^{t-1} (\partial u_1^{i+1} / \partial u_1^i)$. According to (2), $(\partial u_1^{i+1} / \partial u_1^i)$ is equal to the decay rate $\lambda \in (0, 1)$ which is less than 1. Therefore, $\lim_{t \rightarrow \infty} \prod_{i=0}^{t-1} (\partial u_1^{i+1} / \partial u_1^i) = \lim_{t \rightarrow \infty} \lambda^t = 0$, which causes $\lim_{t \rightarrow \infty} (\partial u_1^t / \partial w) = 0$. \square

Proposition 3: Given a NiLIF neuron with the dynamics in (18), the limit condition

$$\exists u_{\text{Ni}}^t, \quad \text{s.t.} \quad \lim_{t \rightarrow \infty} \frac{\partial u_{\text{Ni}}^t}{\partial w} \neq 0 \quad (22)$$

always holds where u_{Ni}^t is the membrane potential of the NiLIF neuron at the time step t .

Proof: We construct a specific example to prove $\exists u_{\text{Ni}}^t$, s.t. $\lim_{t \rightarrow \infty} (\partial u_{\text{Ni}}^t / \partial w) \neq 0$. We use the symbol $f(n)$ to represent the term $\sum_{i=0}^n wx^i \mathcal{L}(t-i)$ in (18). Therefore, the membrane potential u_{Ni}^t can be represented as follows:

$$u_{\text{Ni}}^t = f(t) - \sum_{i=0}^{t-1} g(f(i)) V_{\text{th}} \mathcal{L}(t-1-i). \quad (23)$$

Therefore, $(\partial u_{Ni}^t / \partial w)$ is derived as follows:

$$\frac{\partial u_{Ni}^t}{\partial w} = \frac{\partial f(t)}{\partial w} - \sum_{i=0}^{t-1} \frac{\partial g(f(i))}{\partial f(i)} \frac{\partial f(i)}{\partial w} V_{th} \mathcal{L}(t-1-i). \quad (24)$$

There exists $w > 0$, $x^t > 0$, $\forall i < t, x^i < 0$ such that $\forall i < t, f(i) < 0$ and $f(t) \in (0, 1)$. In this case, $(\partial g(f(i)) / \partial f(i))$ is equal to 0 based on (19). Equation (24) is derived as follows:

$$\frac{\partial u_{Ni}^t}{\partial w} = \frac{\partial f(t)}{\partial w} = \sum_{i=0}^t x^i \mathcal{L}(t-i). \quad (25)$$

In the case of $\sum_{i=0}^t w x^i \mathcal{L}(t-i)$ is in $(0, 1)$ and $w > 0$, $\sum_{i=0}^t x^i \mathcal{L}(t-i)$ is not 0. Therefore, we prove that $\exists u_{Ni}^t$, s.t. $\lim_{t_n \rightarrow \infty} (\partial u_{Ni}^{t_n} / \partial w) \neq 0$. \square

Propositions 2 and 3 show that when the number of time steps is large, the iterative LIF neuron has the vanishing gradient problem, but the NiLIF neuron does not. As well known in [71], the presence of a continuously accumulated gradient multiplication part, $\prod_{i=1}^{t_n-1} (\partial u_1^{i+1} / \partial u_1^i)$, can cause gradient vanishing issues during training. Conversely, the gradient equations of the proposed NiLIF only contain the summation term, which can avoid the gradient issue caused by accumulated multiplication. A detailed comparison of gradient problems is in the Appendix (see Supplementary Material).

In our NiLIF model, the final membrane potential U is decided by the approximate term $g(E_{in})$ and takes the minimal border of the actual situation. E_{in} overestimates the output spikes, resulting in $g(E_{in})$ being greater than it should be, yielding a reduced final membrane potential U and fewer output spikes in O . Therefore, NiLIF exhibits greater sparsity than the iterative LIF model, as shown in Fig. 2. Given the same input spike train, the NiLIF neuron produces fewer

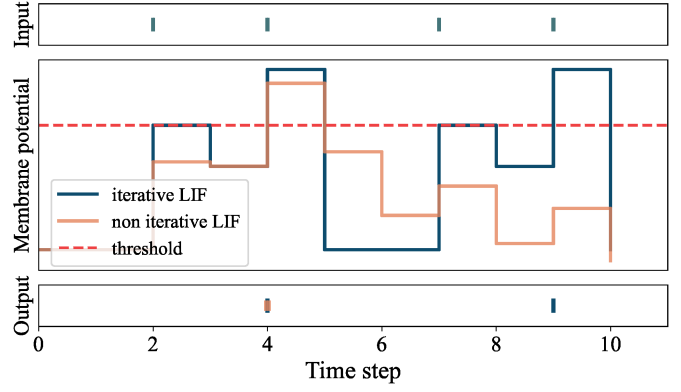


Fig. 2. Illustrative comparison between the iterative LIF and NiLIF models. Both models utilize the same input spike train as shown in the uppermost figure, however, with different outputs. From the third figure, both models produce an output spike at time step 4. Notably, the NiLIF model does not generate a second spike at time step 9 due to higher sparsity.

spikes than its iterative counterpart. The sparsity brings more energy efficiency during execution.

B. Attention Model

In this section, we present the proposed attention models. The fundamental goal of the attention mechanism is to employ neural networks to compute the attention score, which is applied across the entire feature map, assigning weights to the features and extracting useful ones. In the context of EEG signals, the original input data are shaped as $\mathbb{R}^{B \times C \times D}$, where $B \in \mathbb{N}$ is the batch size, $C \in \mathbb{N}$ represents the channel size and $D \in \mathbb{N}$ is the length of data for each channel. We segment the data into timepieces, resulting in a new size of $\mathbb{R}^{B \times C \times S \times T}$, where S is the number of timepieces and $T \in \mathbb{N}$ is the number of time steps in each segment. It is important to note that the multiplication result of S and

$$\begin{aligned} \frac{\partial u_1^t}{\partial w} &= \frac{\partial u_1^t}{\partial u_1^{t-1}} \frac{\partial u_1^{t-1}}{\partial w} + \frac{\partial u_1^t}{\partial o^{t-1}} \frac{\partial o^{t-1}}{\partial w} + x^{t-1} \\ &= \frac{\partial u_1^t}{\partial u_1^{t-1}} \left[\frac{\partial u_1^{t-1}}{\partial u_1^{t-2}} \frac{\partial u_1^{t-2}}{\partial w} + \frac{\partial u_1^{t-2}}{\partial o^{t-2}} \frac{\partial o^{t-2}}{\partial w} + x^{t-2} \right] + \frac{\partial u_1^t}{\partial o^{t-1}} \frac{\partial o^{t-1}}{\partial w} + x^{t-1} \\ &= \frac{\partial u_1^t}{\partial u_1^{t-1}} \left[\frac{\partial u_1^{t-1}}{\partial u_1^{t-2}} \left[\frac{\partial u_1^{t-2}}{\partial u_1^{t-3}} \left[\dots \left[\frac{\partial u_1^2}{\partial u_1^1} \left[\frac{\partial u_1^1}{\partial u_1^0} \frac{\partial u_1^0}{\partial w} + \frac{\partial u_1^1}{\partial o^0} \frac{\partial o^0}{\partial w} + x^0 \right] + \frac{\partial u_1^2}{\partial o^1} \frac{\partial o^1}{\partial w} + x^1 \right] + \dots \right] + \frac{\partial u_1^{t-2}}{\partial o^{t-3}} \frac{\partial o^{t-3}}{\partial w} + x^{t-3} \right] \right. \\ &\quad \left. + \frac{\partial u_1^{t-1}}{\partial o^{t-2}} \frac{\partial o^{t-2}}{\partial w} + x^{t-2} \right] + \frac{\partial u_1^t}{\partial o^{t-1}} \frac{\partial o^{t-1}}{\partial w} + x^{t-1} \\ &= \sum_{i=1}^{t_n} \left[\prod_{k=i}^{t_n-1} \frac{\partial u_1^{k+1}}{\partial u_1^k} \right] x^{i-1} + \sum_{i=2}^{t_n} \left[\prod_{k=i}^{t_n-1} \frac{\partial u_1^{k+1}}{\partial u_1^k} \right] \frac{\partial u_1^i}{\partial o^{i-1}} \frac{\partial o^{i-1}}{\partial w} = \sum_{i=2}^{t_n} \sum_{k=i}^{t_n} \underbrace{\left[\prod_{k=i}^{t_n-1} \frac{\partial u_1^{k+1}}{\partial u_1^k} \right]}_{\text{Accumulating gradient multiplication}} \left[x^{i-1} + \frac{\partial u_1^i}{\partial o^{i-1}} \underbrace{\frac{\partial o^{i-1}}{\partial w}}_{\text{Accumulating gradient multiplication}} \right] \\ &\quad + \underbrace{\prod_{i=1}^{t_n-1} \frac{\partial u_1^{i+1}}{\partial u_1^i}}_{\text{Accumulating gradient multiplication}} x^0 \end{aligned} \quad (26)$$

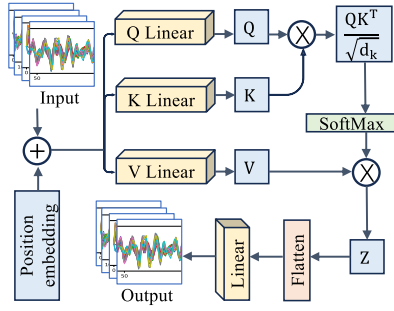


Fig. 3. Illustration of the linear attention model. The linear attention model integrates position embedding with the input data. This model employs three linear layers to obtain the query, key, and value matrices. The attention score is computed through the multiplication of the query and key matrices, then undergoing normalization through a Softmax function. To produce the final enhanced output, an additional linear layer is utilized in the final step.

T should equal D . Given that EEG data typically presents as long temporal sequences, it is important to determine which timepieces are crucial for classification. Thus, our attention model places particular emphasis on the dimension S . We introduce two distinct attention mechanisms: Sequence attention (Seq-attention), as described in Section III-B1, and channel sequence attention (ChanSeq-attention), as detailed in Section III-B2. Section III-B3 presents global-attention, a special case of ChanSeq-attention.

Contrary to the sequential attention models like those in [25], our intention is to utilize a single model to capture the attention score simultaneously. We introduce two model architectures for each attention mechanism: the linear architecture and the convolutional architecture. Fig. 3 illustrates how the architecture of the linear attention differs from the attention model described in Fig. 4. The distinction between these architectures lies in their methods for attention score computation: the linear architecture incorporates fully connected layers, whereas the convolutional architecture employs convolutional layers. Notably, both Seq-attention and ChanSeq-attention have linear and convolutional versions.

1) *Seq-Attention Mechanism*: The sequence attention mechanism mainly focuses on identifying which timepieces require attention. The linear Seq-attention (Linear-Seq-attention) model is given as follows after reshaping the input data into dimension $\mathbb{R}^{B \times S \times (C \times T)}$:

$$\begin{aligned}
 q &= Q_{fc}(x + p), & Q_{fc} : \mathbb{R}^{B \times S \times (C \times T)} &\rightarrow \mathbb{R}^{B \times d_1 \times S \times d_2} \\
 k &= K_{fc}(x + p), & K_{fc} : \mathbb{R}^{B \times S \times (C \times T)} &\rightarrow \mathbb{R}^{B \times d_1 \times S \times d_2} \\
 v &= V_{fc}(x + p), & V_{fc} : \mathbb{R}^{B \times S \times (C \times T)} &\rightarrow \mathbb{R}^{B \times d_1 \times S \times d_2} \\
 A &= \text{Softmax}\left(\frac{qk^T}{\sqrt{d_k}}\right), & A &\in \mathbb{R}^{B \times d_1 \times S \times S} \\
 \hat{x} &= Av, & \hat{x} &\in \mathbb{R}^{B \times d_1 \times S \times d_2} \\
 h_{\text{linear-seq}}(x) &= \text{FC}_{\text{seq}}(\hat{x}) \\
 \text{FC}_{\text{seq}} : \mathbb{R}^{B \times d_1 \times S \times d_2} &\rightarrow \mathbb{R}^{B \times C \times S \times T}
 \end{aligned} \tag{27}$$

where Q_{fc} , K_{fc} and V_{fc} are the three linear layers to generate the query matrix q , the key matrix k , and the value matrix v , respectively. In particular, these matrices adhere to dimensions $\mathbb{R}^{B \times d_1 \times S \times d_2}$, where d_1 and d_2 represent hyperparameters. p is the position embedding [17]. Subsequently, the attention score A is the matrix product derived by qk^T , followed by

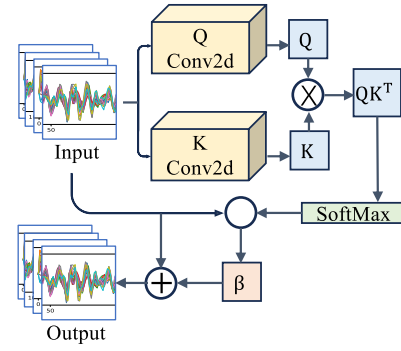


Fig. 4. Illustration of convolutional attention model. Two convolutional layers are employed to get the query and key matrices in the convolutional attention model. The attention score is computed through matrix multiplication and subsequently normalized by the Softmax function. A matrix operation then integrates the attention score with the input data. Depending on the model variant, this can manifest as matrix multiplication in the Seq-attention model (refer to Section III-B1) and the ChanSeq-attention model (detailed in Section III-B2), or as an elementwise product in the Global-attention model (see Section III-B3). In the final step, a trainable parameter, denoted by β , is introduced to balance the original and refined features.

normalization. By applying the Softmax function to A , weights are assigned to the values in v . Then, a fully connected layer produces the final output $h_{\text{linear-seq}}(x)$, which represents the input data enhanced with attention.

The convolutional Seq-attention (Conv-Seq-attention) model is presented as follows:

$$\begin{aligned}
 q &= Q_{\text{conv}}(x), & Q_{\text{conv}} : \mathbb{R}^{B \times S \times C \times T} &\rightarrow \mathbb{R}^{B \times S \times (d \times C \times T)} \\
 k &= K_{\text{conv}}(x), & K_{\text{conv}} : \mathbb{R}^{B \times S \times C \times T} &\rightarrow \mathbb{R}^{B \times S \times (d \times C \times T)} \\
 A &= \text{Softmax}(qk^T), & A &\in \mathbb{R}^{B \times S \times S} \\
 \hat{x} &= Ax, & \hat{x} &\in \mathbb{R}^{B \times C \times S \times T} \\
 h_{\text{conv-seq}}(x) &= \alpha \hat{x} + x
 \end{aligned} \tag{28}$$

where Q_{conv} and K_{conv} are the two convolutional layers with reshape techniques. Similar to the Linear-Seq-attention mechanism, they are designated to generate the query matrix q and key matrix k , respectively. Within this model, d serves as a hyperparameter. Unlike its linear counterpart, the Conv-Seq-attention model omits the computation of the value matrix and instead directly calculates the attention score A by multiplying the matrix. Subsequent to the Softmax function, these weights are integrated with the input data via matrix multiplication. Finally, the Conv-Seq-attention introduces a trainable parameter α to modulate the balance between the attention-enhanced result and the original input data.

Both the Linear-Seq-attention and Conv-Seq-attention mechanisms yield attention scores of size $\mathbb{R}^{S \times S}$ in the last two dimensions. Then, they utilize matrix multiplication to produce the final enhanced feature map. In this way, attention is exclusively directed toward different timepieces and their interaction without consideration of information from other dimensions.

2) *ChanSeq-Attention Mechanism*: Typically, EEG signals are collected from multiple channels. For example, the OpenBMI dataset that we use in this work has 62 channels [72]. Beyond directing attention to timepieces, it is also valuable to be concerned with which channels receive the most attention. Thus, we introduce an attention mechanism named

the ChanSeq-attention mechanism, designed to determine when and where the features must be focused.

The equations for linear ChanSeq-attention (Linear-ChanSeq-attention) are as follows:

$$\begin{aligned}
 q &= Q_{fc}(x + p), & Q_{fc} &: \mathbb{R}^{B \times C \times S \times T} \rightarrow \mathbb{R}^{B \times C \times d_1 \times S \times d_2} \\
 k &= K_{fc}(x + p), & K_{fc} &: \mathbb{R}^{B \times C \times S \times T} \rightarrow \mathbb{R}^{B \times C \times d_1 \times S \times d_2} \\
 v &= V_{fc}(x + p), & V_{fc} &: \mathbb{R}^{B \times C \times S \times T} \rightarrow \mathbb{R}^{B \times C \times d_1 \times S \times d_2} \\
 A &= \text{Softmax}\left(\frac{qk^\top}{\sqrt{d_k}}\right), & A &\in \mathbb{R}^{B \times C \times d_1 \times S \times S} \\
 \hat{x} &= Av, & \hat{x} &\in \mathbb{R}^{B \times C \times d_1 \times S \times d_2} \\
 h_{\text{linear-chanseq}}(x) &= \text{FC}_{\text{chanseq}}(\hat{x}) \\
 \text{FC}_{\text{chanseq}} &: \mathbb{R}^{B \times C \times d_1 \times S \times d_2} \rightarrow \mathbb{R}^{B \times C \times S \times T}
 \end{aligned} \tag{29}$$

where the meaning of the parameters is the same as in Linear-Seq-attention. However, it is worth noting that to implement attention across both channels and timepieces simultaneously, the dimensions of Q_{fc} , K_{fc} , and V_{fc} are adjusted to $\mathbb{R}^{B \times C \times d_1 \times S \times d_2}$ instead of $\mathbb{R}^{B \times d_1 \times S \times d_2}$. The new channel dimension C is consistently maintained throughout the entirety of the model.

The equations of corresponding convolutional Conv-ChanSeq-attention are as follows:

$$\begin{aligned}
 q &= Q_{\text{conv}}(x), & Q_{\text{conv}} &: \mathbb{R}^{B \times C \times S \times T} \rightarrow \mathbb{R}^{B \times C \times S \times (d \times T)} \\
 k &= K_{\text{conv}}(x), & K_{\text{conv}} &: \mathbb{R}^{B \times C \times S \times T} \rightarrow \mathbb{R}^{B \times C \times S \times (d \times T)} \\
 A &= \text{Softmax}(qk^\top), & A &\in \mathbb{R}^{B \times C \times S \times S} \\
 \hat{x} &= Ax, & \hat{x} &\in \mathbb{R}^{B \times C \times S \times T} \\
 h_{\text{conv-chanseq}}(x) &= \alpha \hat{x} + x
 \end{aligned} \tag{30}$$

where the parameters in this context hold the same representations as those in the Conv-Seq-attention. Mirroring the adjustments in Linear-ChanSeq-attention, Q_{conv} and K_{conv} have dimensions $\mathbb{R}^{B \times C \times S \times (d \times T)}$, which is different from the previous approach of merging the C and T dimensions. As a result, the attention score A is characterized by a size of $\mathbb{R}^{B \times C \times S \times S}$, facilitating the attention across various channels while accounting for different timepieces.

3) *Global-Attention Mechanism*: In this section, we go further from the ChanSeq-attention. Given input data with dimensions $\mathbb{R}^{B \times C \times S \times T}$, not only are channels and timepieces considered but also the specific time steps are important. To address this, we introduce the global-attention mechanism, which operates attention across all three dimensions: C , S , and T . It decides when, where, and which feature is essential.

The global-attention can be viewed as a special case of the Conv-ChanSeq-attention when the dimensions of S and T are equal. The corresponding equations are presented as follows:

$$\begin{aligned}
 q &= Q_{\text{conv}}(x), & Q_{\text{conv}} &: \mathbb{R}^{B \times C \times S \times T} \rightarrow \mathbb{R}^{B \times C \times S \times (d \times T)} \\
 k &= K_{\text{conv}}(x), & K_{\text{conv}} &: \mathbb{R}^{B \times C \times S \times T} \rightarrow \mathbb{R}^{B \times C \times T \times (d \times S)} \\
 A &= \text{Softmax}(qk^\top), & A &\in \mathbb{R}^{B \times C \times S \times T} \\
 \hat{x} &= A \odot x, & \hat{x} &\in \mathbb{R}^{B \times C \times S \times T} \\
 h_{\text{conv-global}}(x) &= \alpha \hat{x} + x.
 \end{aligned} \tag{31}$$

In this article, the parameters align with those defined in the Conv-ChanSeq-attention. In particular, when S and T

have same dimension sizes, both A and x have the size of $\mathbb{R}^{B \times C \times S \times T}$. Contrary to the Conv-ChanSeq-attention, the Global-attention employs elementwise product instead of matrix multiplication when calculating enhanced feature map \hat{x} in (31). Consequently, with A having dimensions $\mathbb{R}^{B \times C \times S \times T}$, each time step in each timepiece of each channel receives a specific attention score to improve the feature map. This method distinguishes itself from other attention models by directly enhancing the data.

C. Network Architecture

In this section, we present the network architectures shown in Fig. 5. Fig. 5(a) illustrates the architecture of NiSNN-A, which utilizes a two-layer residual spiking convolutional framework. The first spiking layer can be seen as a spiking encoder as proposed in [15]. The membrane potential batch normalization introduced in [73] is also embedded in the NiLIF neurons. The NiLIF neuron yields a binary sequence as the output. To preserve the binary nature of the data stream, a max pooling layer is used to reduce dimensionality. After the second spiking layer, the proposed attention model is integrated. Finally, two linear layers without activation functions are employed to classify the output labels.

To compare and verify that the proposed attention mechanism can also be applied to the CNN network, we show the attention CNN counterpart to the NiSNN-A in Fig. 5(b). The CNN architecture mirrors the SNNs to regulate extraneous variables, encompassing a two-layer convolutional residual framework. After each convolutional layer, a batch normalization layer is applied, followed by the rectified linear unit (ReLU) activation function. The average pooling layer is then used to reduce the dimension.

In particular, since the SNN network has the characteristics of binary data flow, the input data for both the second spiking layer and the first linear layer consists of accumulator operations (ACs). In contrast, all operations within the CNN are multiplicative and accumulate (MAC) operations.

IV. EXPERIMENTS

In this section, we outline the details of the experiment. Details about the dataset and its processing can be found in Section IV-A. The network configuration is elaborated in Section IV-B. Finally, the approach to energy analysis is presented in Section IV-C.

A. Dataset

In this study, we utilized the BCIC IV 2a dataset [74] and the OpenBMI dataset [72] to evaluate the performance of the NiSNN-A.

The BCIC IV 2a dataset consists of EEG MI recordings from nine participants, with each participant having 288 trials across 20 channels. Each trial lasts 3 s and is recorded at a sampling frequency of 250 Hz. The OpenBMI dataset, which is larger in scale, includes EEG MI signals from 54 participants, with each participant providing 400 trials across 62 channels. These trials are 4 s with a sampling frequency of 1000 Hz. The channels selected from OpenBMI

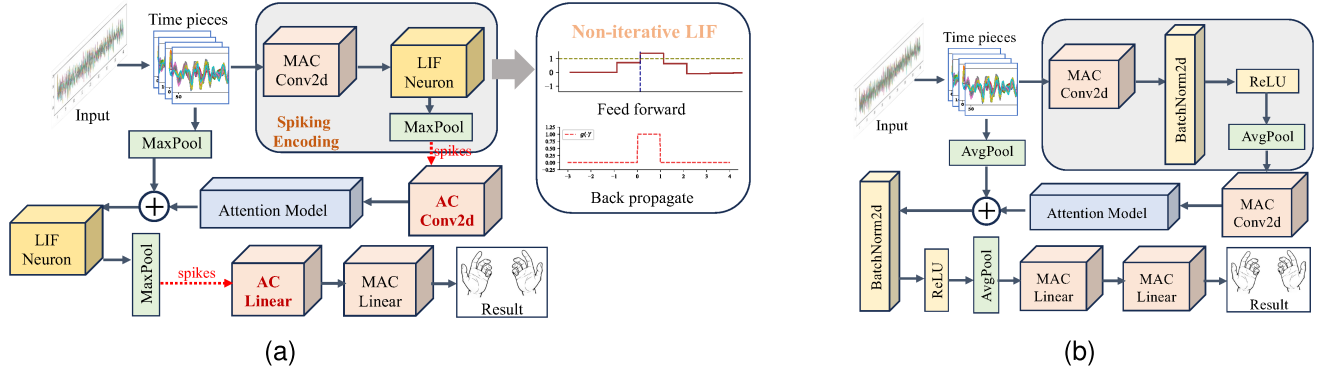


Fig. 5. Network architecture overview. (a) NiSNN-A architecture. The SNN has a residual block containing two spiking layers. Instead of traditional batch normalization layers and activation functions, the model employs LIF neurons to transform real-valued data streams into spikes. Notably, the proposed NiLIF neurons are utilized, leveraging the Heaviside function during the feed-forward phase and the surrogate derivation during backpropagation. Max pooling layers are adopted after spiking layers to maintain a binary data flow. After the second spiking layer, an attention mechanism is integrated to refine the feature maps. The final stage of the model consists of two fully connected layers for classification. (b) Attention CNN architecture. The overarching architecture of the CNN closely mirrors that of the SNN, encompassing a single residual block with two 2-D convolutional layers. A batch normalization layer follows the convolutional layer and utilizes the ReLU function as its activation function. The network employs average pooling layers for downsampling.

TABLE I
ARCHITECTURE OF NETWORKS DURING TRAINING. B IS THE BATCH SIZE, C IS THE CHANNEL SIZE, S IS THE NUMBER OF TIMEPIECES, AND T IS THE NUMBER OF TIME POINTS IN EACH TIMEPIECE

Block	SNN layer	CNN layer	Filters	Size/padding	Output
Spike Encoding	Input	Input	-	-	(B, C, S, T)
	Clone residual	Clone residual	-	-	(B, C, S, T)
	Conv2d	Conv2d	C	(1, 5)/same	(B, C, S, T)
	-	BatchNorm2d	C	-	(B, C, S, T)
	-	ReLU	-	-	(B, C, S, T)
	LIF	-	-	-	(B, C, S, T)
Classifier	MaxPool2d	AvgPool2d	-	(2, 2)	($B, C, S/2, T/2$)
	AC-Conv	MAC-Conv2d	C	(10, 10)/same	($B, C, S/2, T/2$)
	Attention	Attention	-	-	($B, C, S/2, T/2$)
	Add residual	Add residual	-	-	($B, C, S/2, T/2$)
	-	BatchNorm2d	C	-	($B, C, S/2, T/2$)
	-	ReLU	-	-	($B, C, S/2, T/2$)
	LIF	-	-	-	($B, C, S/2, T/2$)
	MaxPool2d	AvgPool2d	-	(2, 2)	($B, C, S/4, T/4$)
	Flatten	Flatten	-	-	($B, C \times S/4 \times T/4$)
	AC-Linear	MAC-Linear	-	-	($B, 20$)
	Linear	Linear	-	-	($B, 2$)

include FC-5/3/1/2/4/6, C-5/3/1/z/2/4/6, and CP-5/3/1/z/2/4/6, whereas the selected channels for BCIC IV 2a are FC-3/1/z/2/4, C-5/3/1/z/2/4/6, CP-3/1/z/2/4, and P-1/z/2. The tasks of both datasets are to classify MI EEG signals for left-hand and right-hand movements.

The experiments train the networks to identify the common features in EEG signals. We take the subject-independent approach as utilized in [35], which reserves data from one subject for testing and uses data from the remaining subjects for training. Specifically, for OpenBMI, the training data includes 53 subjects' recordings, totaling 21 200 trials, and the test data comprises 200 trials. For BCIC IV 2a, the training data includes eight subjects' trials, totaling 2304 trials, with the test dataset comprising 144 trials. For both datasets, each trial contains 400 time steps after downsampling. Therefore, this strategy ensures that the model is tested on unseen data, reinforcing its scalability and practical applicability.

Throughout the training phase, we cycle each subject as a test case. The overall performance is then calculated by averaging the accuracy from all cycles. Therefore, each model

undergoes training and testing 54 times for OpenBMI and nine times for BCIC IV 2a, respectively, to ascertain the final result and remove some uncertainty.

B. Network Setups

As described in Section IV-A, the input data have a dimension size of $\mathbb{R}^{B \times C \times S \times T}$, where B denotes the batch size, C is the channel size, S indicates the number of timepieces, and T is the number of time steps within each timepiece. In the experiment, the batch size is 64 and the channel size is 20. Each trial is segmented into 20 timepieces, rendering S and T as 20. Also, we set the threshold V_{th} in the NiLIF as 0.5.

The network parameters are shown in Table I. The first convolutional layer acts as a spiking encoder, only considering the information within each timepiece with a kernel size of (1, 5). The second convolutional layer plays the role of classifier, taking into account both intratimepieces and inner timepiece information with a kernel size of (10, 10). All pooling layers employ the kernel size (2, 2). The first linear layer reduces the flattened data dimension to 20, and the

TABLE II
FLOPS OF MODEL EXECUTIONS. CNN MODELS ONLY HAVE MAC OPERATION, AND SNN MODELS HAVE MAC, AC_{conv} , AND AC_{fc} OPERATIONS. x IS 4 810 040, y IS 1 170 040, a IS 4 000 000 AND b IS 10 000

Reference	Method	Type	MAC	Ratio _{MAC}	AC_{conv}	AC_{fc}	Ratio _{AC}
Autthasan et al. [35]	vanilla CNN	CNN	x	baseline	-	-	-
Huang et al. [33]	vanilla CNN	CNN	x	-0%	-	-	-
Liu et al. [13]	Sequence + Temporal attention	CNN	$x+1,644,200$	↑ 34.2%	-	-	-
Luo et al. [44]	Sequence + Temporal attention	CNN	$x+2,469,600$	↑ 51.3%	-	-	-
Fan et al. [49], Wang et al. [47]	Channel + Sequence + Temporal attention	CNN	$x+45,200$	↑ 0.9%	-	-	-
Zhang et al. [46]	Channel attention	CNN	$x+180,000$	↑ 3.7%	-	-	-
Wu et al. [60]	vanilla SNN (Iterative LIF)	SNN	$y-180,000$	↓ 79.4%	0.807a	0.255b	baseline
Hu et al. [75]	vanilla SNN	SNN	y	↓ 75.7%	0.352a	0.383b	↓ 56.36%
Yao et al. [25]	Channel attention	SNN	$y+3,600$	↓ 75.6%	0.433a	0.356b	↓ 46.26%
Yao et al. [25]	Sequence attention	SNN	$y+2,400$	↓ 75.6%	0.407a	0.425b	↓ 49.51%
Yao et al. [25]	Temporal attention	SNN	$y+2,400$	↓ 75.6%	0.459a	0.331b	↓ 43.12%
Yao et al. [25]	Channel + Temporal attention	SNN	$y+6,000$	↓ 75.6%	0.457a	0.384b	↓ 43.37%
Yao et al. [25]	Channel + Sequence attention	SNN	$y+6,000$	↓ 75.6%	0.449a	0.362b	↓ 44.30%
Yao et al. [25]	Sequence + Temporal attention	SNN	$y+4,800$	↓ 75.6%	0.484a	0.367b	↓ 39.93%
Yao et al. [25]	Channel + Sequence + Temporal attention	SNN	$y+8,400$	↓ 75.5%	0.500a	0.272b	↓ 38.09%
Ours	Linear-Seq-attention	SNN	$y+97,800$	↓ 73.6%	0.397a	0.312b	↓ 50.80%
Ours	Conv-Seq-attention	SNN	$y+822,100$	↓ 58.6%	0.358a	0.419b	↓ 55.58%
Ours	Linear-ChanSeq-attention	SNN	$y+496,800$	↓ 65.3%	0.382a	0.326b	↓ 52.61%
Ours	Conv-ChanSeq-attention	SNN	$y+824,000$	↓ 58.5%	0.345a	0.403b	↓ 57.17%
Ours	Global-attention	SNN	$y+806,000$	↓ 58.9%	0.348a	0.413b	↓ 56.76%

* Only the first vanilla SNN uses the Iterative LIF neurons (specified already), otherwise all SNN models use the proposed Non-iterative LIF neuron model.

final linear layer reduces it further to 2, thereby having the final classification result. Within the attention layer, the hyperparameters d_1 and d_2 are set to 6 and 20, respectively, for all linear attention models. For convolutional attention models, the hyperparameter d is set to 8. The models are trained for 20 epochs. We employ a well-trained CNN model as a pretrained network for its corresponding SNN model to accelerate the training procedure. During the training process, we adopted the Adam optimizer with a learning rate of 0.001. The cross-entropy loss is utilized as the loss function

$$CE_Loss(y, p) = - \sum_n y_n \log(p_n) \quad (32)$$

where y represents the label of the data and p is the network's output.

C. Energy Analysis

For analyzing the energy consumption of CNN and SNN models, we adopt the same energy analysis method in [25], which calculates the network's floating point operations (FLOPs).

The main operations within neural networks in this context can be categorically divided into three primary types: the convolutional layer, the linear layer, and matrix multiplication. The FLOPs associated with each of these operations can be described as follows:

$$\begin{aligned} FLOPs_{conv}^n &= k_0^n k_1^n h^n w^n c^n c^{n-1} \\ FLOPs_{fc}^n &= i^n o^n \\ FLOPs_{mm} &= m_1 n m_2 \end{aligned} \quad (33)$$

where for the n th convolutional layer, k_0^n and k_1^n denote the dimensions of the convolutional kernel, while h^n and w^n represent the dimensions of the output map. In addition, c^n and c^{n-1} specify the size of the input and output data

channels, respectively. For the n th linear layer, i^n represents the input size and o^n denotes the output size. Finally, for matrix multiplication involving two matrices of dimensions $[m_1, n]$ and $[n, m_2]$, the FLOPs are determined as the product of these three dimensions.

In the CNN models, all network data consist of real-valued numbers, which makes all operations MAC. The energy analysis is divided into the FLOPs corresponding to the standard vanilla CNN model, denoted by x , and the FLOPs for the additional attention model. Given our consistent network architecture for all models, the value of x remains constant, 4 810 040. In the SNN models, the FLOPs analysis is more complicated. As elaborated in Section III-C, the inputs to the second convolutional and first linear layers are all binary. Consequently, these layers employ AC operations, which accumulate weight directly without involving multiplications. It is worth noting that the number of AC operations is contingent upon the spike rate, given that accumulation happens only when the input is 1. Thus, the spike rates of these two layers emerge as important parameters when quantifying the AC operations within SNN models. The rest of the SNN retains the use of MAC operations, mirroring the corresponding CNN models. Therefore, the FLOPs assessment for SNN models can be partitioned into three parts: AC of the binary layers, MAC of the rest of the SNN model, and the MAC of the additional attention model. The MAC of the rest of the SNN model could be divided into two parts: the MAC of LIF neurons and the MAC of the rest of the NiSNN model. We present the MAC of the rest SNN model with NiLIF neurons as y , which is a constant across all SNN models with NiLIF neurons, calculated as 1 170 040. The MAC of SNN with iterative LIF neurons is less than y , which is calculated as $y - 180 000$. It should be highlighted that while the FLOPs associated with iterative LIF neurons are fewer than those of NiLIF neurons, the latter are implemented using matrix

TABLE III
ACCURACY AND ENERGY PERFORMANCE OF LEFT H VERSUS RIGHT H. SUBJECT-INDEPENDENT CLASSIFICATION
USING THE BCIC IV 2A DATASET FOR SNNs WITH ATTENTION MECHANISMS

Reference	Method	Type	Accuracy	AC _{conv}	AC _{fc}	Energy (μJ)
Autthasan et al. [35]	vanilla CNN	CNN	0.6545 +/- 0.1053	-	-	23.569
Huang et al. [33]	vanilla CNN	CNN	0.6866 +/- 0.0885	-	-	23.569
Liu et al. [13]	Sequence + Temporal attention	CNN	0.6641 +/- 0.1011	-	-	31.626
Luo et al. [44]	Sequence + Temporal attention	CNN	0.6667 +/- 0.1191	-	-	35.670
Fan et al. [49], Wang et al. [47]	Channel + Sequence + Temporal attention	CNN	0.6745 +/- 0.1144	-	-	23.791
Zhang et al. [46]	Channel attention	CNN	0.6875 +/- 0.0880	-	-	24.451
Wu et al. [60]	vanilla SNN (with the Iterative LIF neuron)	SNN	0.5182 +/- 0.0607	0.8034	0.1050	7.744
Hu et al. [75]	vanilla SNN	SNN	0.6120 +/- 0.0842	0.3966	0.4311	7.165
Yao et al. [25]	Channel attention	SNN	0.5703 +/- 0.0614	0.4891	0.4159	7.515
Yao et al. [25]	Sequence attention	SNN	0.5773 +/- 0.0478	0.4607	0.4423	7.407
Yao et al. [25]	Temporal attention	SNN	0.6094 +/- 0.0794	0.5284	0.4246	7.651
Yao et al. [25]	Channel + Temporal attention	SNN	0.5885 +/- 0.0961	0.5093	0.4278	7.600
Yao et al. [25]	Channel + Sequence attention	SNN	0.5920 +/- 0.0746	0.4803	0.4256	7.496
Yao et al. [25]	Sequence + Temporal attention	SNN	0.5521 +/- 0.0576	0.5687	0.4558	7.808
Yao et al. [25]	Channel + Sequence + Temporal attention	SNN	0.5512 +/- 0.0817	0.5365	0.4218	7.710
Ours	Linear-Seq-attention	SNN	0.6259 +/- 0.1020	0.4200	0.3218	7.727
Ours	Conv-Seq-attention	SNN	0.6493 +/- 0.1030	0.3778	0.4452	11.126
Ours	Linear-ChanSeq-attention	SNN	0.6241 +/- 0.0862	0.4056	0.3380	9.631
Ours	Conv-ChanSeq-attention	SNN	0.6580 +/- 0.1079	0.3701	0.4363	11.107
Ours	Global-attention	SNN	0.6615 +/- 0.1219	0.3784	0.4433	11.049

* Only the first vanilla SNN uses the Iterative LIF neurons (specified already). Otherwise, all SNN models use the proposed Non-iterative LIF neuron model.

TABLE IV
ACCURACY AND ENERGY PERFORMANCE OF LEFT H VERSUS RIGHT H. SUBJECT-INDEPENDENT CLASSIFICATION
USING THE OPENBMI DATASET FOR SNNs WITH ATTENTION MECHANISMS

Reference	Method	Type	Accuracy	AC _{conv}	AC _{fc}	Energy (μJ)
Autthasan et al. [35]	vanilla CNN	CNN	0.7368 +/- 0.1320	-	-	23.569
Huang et al. [33]	vanilla CNN	CNN	0.7370 +/- 0.1314	-	-	23.569
Liu et al. [13]	Sequence + Temporal attention	CNN	0.7394 +/- 0.1280	-	-	31.626
Luo et al. [44]	Sequence + Temporal attention	CNN	0.7138 +/- 0.1411	-	-	35.670
Fan et al. [49], Wang et al. [47]	Channel + Sequence + Temporal attention	CNN	0.7386 +/- 0.1279	-	-	23.791
Zhang et al. [46]	Channel attention	CNN	0.7405 +/- 0.1362	-	-	24.451
Wu et al. [60]	vanilla SNN (with the Iterative LIF neuron)	SNN	0.5036 +/- 0.0053	0.8114	0.4040	7.776
Hu et al. [75]	vanilla SNN	SNN	0.6971 +/- 0.1329	0.3067	0.3340	6.840
Yao et al. [25]	Channel attention	SNN	0.6757 +/- 0.1279	0.3776	0.2966	7.113
Yao et al. [25]	Sequence attention	SNN	0.6978 +/- 0.1257	0.3531	0.4083	7.020
Yao et al. [25]	Temporal attention	SNN	0.6975 +/- 0.1309	0.3891	0.2369	7.148
Yao et al. [25]	Channel + Temporal attention	SNN	0.6819 +/- 0.1265	0.4040	0.3395	7.220
Yao et al. [25]	Channel + Sequence attention	SNN	0.6848 +/- 0.1297	0.4180	0.2993	7.270
Yao et al. [25]	Sequence + Temporal attention	SNN	0.6934 +/- 0.1256	0.4002	0.2791	7.200
Yao et al. [25]	Channel + Sequence + Temporal attention	SNN	0.6774 +/- 0.1318	0.4626	0.1231	7.441
Ours	Linear-Seq-attention	SNN	0.6843 +/- 0.1254	0.3736	0.3016	7.560
Ours	Conv-Seq-attention	SNN	0.7073 +/- 0.1397	0.3379	0.3936	10.981
Ours	Linear-ChanSeq-attention	SNN	0.6845 +/- 0.1204	0.3587	0.3145	9.462
Ours	Conv-ChanSeq-attention	SNN	0.7083 +/- 0.1402	0.3201	0.3688	10.926
Ours	Global-attention	SNN	0.7051 +/- 0.1377	0.3183	0.3824	10.832

* Only the first vanilla SNN uses the Iterative LIF neurons (specified already), otherwise all SNN models use the proposed Non-iterative LIF neuron model.

operations, whereas the former rely on loop structures, leading to increased execution time. The original AC for the second convolutional layer is 4 000 000, denoted by a , while for the first linear layer, it is 10 000, represented as b .

The final FLOPs comparison of models is shown in Table II. Details on model selection are provided in Section V-A. We choose the vanilla CNN and SNN as baseline models for comparing MAC and AC operations, respectively. All spiking rates are calculated by the average of two dataset experiments.

For each experiment, the spiking rates are calculated over all subjects. Table II shows that the proposed model can reduce the energy consumption from both MAC and AC calculation. After the FLOPs analysis, we could calculate the total energy cost. We adopt the same assumption as in [25] that the data for various operations are implemented as floating point 32 bits in 45-nm technology, where the MAC energy is 4.6 pJ and the AC energy is 0.9 pJ. The detailed energy is listed in Tables III and IV.

V. RESULTS AND DISCUSSION

In this section, we describe the results of the proposed attention models which are delineated in Section V-A. Then, a discussion along with the result visualization is provided in Section V-B.

A. Comparison With State-of-the-Art Methods

Tables III and IV list the comparison performances of various attention mechanisms with CNNs and SNNs on two EEG datasets. We have chosen seven CNN models with attention mechanisms for EEG classification as our benchmark. Autthasam et al. [35] have proposed a vanilla CNN model, employing a subject-independent approach for training and testing phases. Furthermore, Huang et al. [33] incorporated residual blocks into the CNN model. Of the seven baseline models, five models employ attention mechanisms. It is worth noting that [47] and [49] employ a global-attention mechanism that encompasses all three dimensions. However, their approach is characterized by extracting attention scores individually for each dimension after using the pooling methods to minimize unrelated dimensions. This stands differently from the methodology of our proposed global-attention model. A total of ten models were chosen as the SNN baselines. It is worth noting that in the context of EEG signal processing, the iterative LIF model struggles with the long time steps, potentially leading to gradient issues. Therefore, we adopted the proposed NiLIF neuron in other baseline models. Concerning the attention mechanism, the baseline models are from the image attention SNN model developed for CV, as introduced in [25]. This approach composes various dimensional attention components sequentially to achieve the attention score, diverging from our proposed models that utilize a singular model.

The accuracy metric was determined by the ratio of correctly classified samples to the total number of samples, as given by

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (34)$$

where TP, TN, FP, and FN represent true positive, true negative, false positive, and false negative, respectively. For energy analysis, we adopted the unit microjoules to quantify the energy consumption of the networks as discussed in Section IV-C.

The experimental results show that the proposed NiSNN-A can achieve the best performance among SNN models, which are 0.6615 and 0.7 for the BCIC IV 2a dataset and OpenBMI dataset, respectively. This accuracy is comparable with the CNN models, with the additional advantage of consuming around two times less energy.

Notably, compared with the iterative LIF neuron model, the proposed NiLIF model improves the accuracy by 0.1 for BCIC IV 2a dataset and 0.2 for OpenBMI dataset, respectively, while reducing the energy cost. Tables III and IV show that the firing rate of NiLIF neurons is much lower than of iterative LIF neurons, which illustrates the sparsity of our proposed method. The proposed NiSNN and attention mechanisms can be used separately. We present the comparison results of NiSNN and the proposed attention mechanisms on various datasets in the Appendix (see Supplementary Material), showing the proposed methods' feasibility.

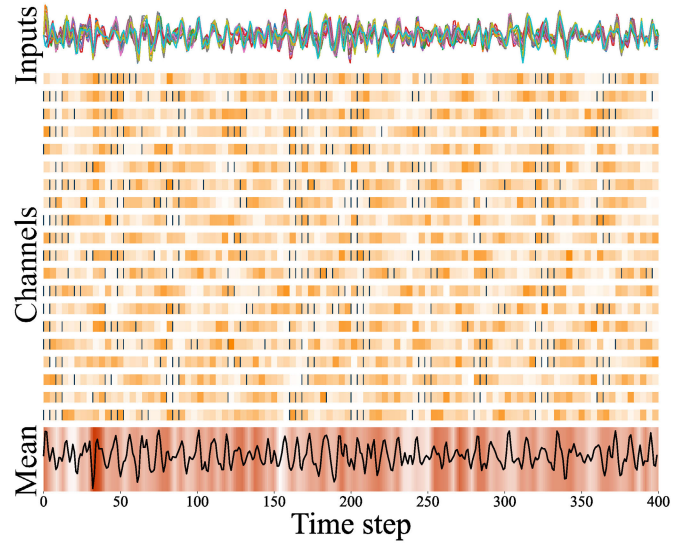


Fig. 6. Generated attention visualization example, using our global NiSNN-A model. The top figure shows the input EEG signals. In the following figure, black scatters represent the spikes after encoding. The shade of orange blocks represents the degree of attention. The darker orange block shows a higher attention score, and the lighter orange block represents a lower one. In the last figure, the black curve represents the average value of the input EEG on 20 channels.

B. Discussion

The experimental results highlight the potential of the attention mechanism in improving the classification accuracy of EEG signals, especially for SNN models. To understand more clearly how the attention mechanism works, we provide a visualization that illustrates the role of the attention score A in the entire feature map. Fig. 6 intuitively represents the attention score A of the global-attention model. The top part of the figure depicts the input 20-channel EEG signals in a numeric format. The second part shows the spikes after the spiking encoder, represented by the black raster. The orange blocks represent the attention scores after normalization. Darker orange indicates higher attention scores, while lighter orange indicates lower ones. It illustrates the inner operation of the attention mechanism, highlighting the model's ability to allocate variable attention to different regions of the EEG signal, answering when, where, and what information is relevant. However, unlike understandable visualizations of attention mechanisms in visual tasks, the physiological meaning of attention scores is not intuitive. Therefore, we show the averaging effect in the third part of the figure. We show the mean of the 20-channel EEG signal, represented by the black line. The orange shading represents higher or lower attention scores. Interestingly, the attention mechanism emphasizes time steps around 30–40, corresponding to the most apparent drop in the EEG signal, revealing the importance of this sudden change in the EEG signal. By employing such a dynamic weighting method, the model ensures that the important regions of the input are prioritized, potentially contributing to the high accuracy in classification tasks.

From an energy consumption point of view, the results highlight the significance of employing SNNs, which achieve accuracy comparable with CNN models and offer a twofold

reduction in energy use. This efficiency can be important in applications where power consumption is a concern, such as portable EEG devices or real-time EEG monitoring systems. Therefore, SNNs present promising potential for these energy-conscious edge devices.

VI. CONCLUSION

This article introduced an NiSNN-A model, encompassing NiLIF neurons and diverse attention mechanisms. The newly proposed NiLIF neuron retains the biological attributes of traditional LIF neurons while efficiently handling long temporal data. This design avoids long loops in execution and gradient challenges by approximating the neuron dynamics and calculating synchronously. Subsequently, the attention mechanism emphasizes important parts of the feature map. Notably, all our proposed attention models integrate computations within one singular model instead of using sequential architectures. We employed the BCIC IV 2a and OpenBMI datasets for validation, adopting a subject-independent approach to demonstrate the model's capabilities in uniformed feature extraction for unfamiliar participants. The results indicate that our approach surpasses other SNN models in accuracy performance. It achieves accuracy comparable to its CNN counterparts but improves energy efficiency. Furthermore, our attention visualization results reveal that our model improves the classification task's accuracy and offers deeper insights into EEG signal interpretation.

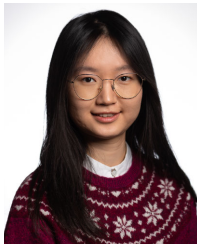
This article has provided a way for novel methodologies in EEG classification, focusing on potential cooperation between attention mechanisms and SNN architectures. In the future, as the field of EEG signal processing continues to evolve, our findings require continued innovation and adaptive strategies to address challenges.

REFERENCES

- [1] A. Tsiamalou et al., "EEG in neurorehabilitation: A bibliometric analysis and content review," *Neurol. Int.*, vol. 14, no. 4, pp. 1046–1061, Dec. 2022.
- [2] C. Del Percio et al., "What a single electroencephalographic (EEG) channel can tell us about Alzheimer's disease patients with mild cognitive impairment," *Clin. EEG Neurosci.*, vol. 54, no. 1, pp. 21–35, 2023.
- [3] A. Singh, A. A. Hussain, S. Lal, and H. W. Guesgen, "A comprehensive review on critical issues and possible solutions of motor imagery based electroencephalography brain-computer interface," *Sensors*, vol. 21, no. 6, p. 2173, Mar. 2021.
- [4] J. Zhang and M. Wang, "A survey on robots controlled by motor imagery brain-computer interfaces," *Cognit. Robot.*, vol. 1, pp. 12–24, Jan. 2021.
- [5] S. Rajwal and S. Aggarwal, "Convolutional neural network-based EEG signal analysis: A systematic review," *Arch. Comput. Methods Eng.*, vol. 30, no. 6, pp. 3585–3615, 2023.
- [6] S. Alhagry, A. A. Fahmy, and R. A. El-Khoribi, "Emotion recognition based on EEG using LSTM recurrent neural network," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 10, pp. 355–358, 2017.
- [7] Y. Song, Q. Zheng, B. Liu, and X. Gao, "EEG conformer: Convolutional transformer for EEG decoding and visualization," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 710–719, 2022.
- [8] O.-Y. Kwon, M.-H. Lee, C. Guan, and S.-W. Lee, "Subject-independent brain-computer interfaces based on deep convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 3839–3852, Oct. 2020.
- [9] J.-S. Bang, M.-H. Lee, S. Fazli, C. Guan, and S.-W. Lee, "Spatio-spectral feature representation for motor imagery classification using convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 7, pp. 3038–3049, Jul. 2022.
- [10] S. Zhang et al., "An explainable and generalizable recurrent neural network approach for differentiating human brain states on EEG dataset," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 6, pp. 7339–7350, Jun. 2024.
- [11] Y. Wu, Y. Yang, J. S. Xiao, C. Zhou, H. Sui, and H. Li, "Invariant spatiotemporal representation learning for cross-patient seizure classification," 2025. [Online]. Available: <https://openreview.net/forum?id=TkbjqexD8w>
- [12] C. Ju and C. Guan, "Tensor-CSPNet: A novel geometric deep learning framework for motor imagery classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 10955–10969, Dec. 2023.
- [13] X. Liu, Y. Shen, J. Liu, J. Yang, P. Xiong, and F. Lin, "Parallel spatial-temporal self-attention CNN-based motor imagery classification for BCI," *Frontiers Neurosci.*, vol. 14, Dec. 2020, Art. no. 587520.
- [14] E. Eldele et al., "An attention-based deep learning approach for sleep stage classification with single-channel EEG," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 809–818, 2021.
- [15] X. Liao, Y. Wu, Z. Wang, D. Wang, and H. Zhang, "A convolutional spiking neural network with adaptive coding for motor imagery classification," *Neurocomputing*, vol. 549, Sep. 2023, Art. no. 126470.
- [16] Y. Zhang et al., "Improving EEG decoding via clustering-based multitask feature learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 8, pp. 3587–3597, Aug. 2022.
- [17] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–15.
- [18] S. Li, Q. Yan, and P. Liu, "An efficient fire detection method based on multiscale feature extraction, implicit deep supervision and channel attention mechanism," *IEEE Trans. Image Process.*, vol. 29, pp. 8467–8475, 2020.
- [19] S. Abirami and P. Chitra, "Energy-efficient edge based real-time healthcare support system," in *Advances in Computers*, vol. 117, no. 1. Amsterdam, The Netherlands: Elsevier, 2020, pp. 339–368.
- [20] Z. Huang and M. Wang, "A review of electroencephalogram signal processing methods for brain-controlled robots," *Cognit. Robot.*, vol. 1, pp. 111–124, Jan. 2021.
- [21] S. Woźniak, A. Pantazi, T. Bohnstingl, and E. Eleftheriou, "Deep learning incorporating biologically inspired neural dynamics and in-memory computing," *Nat. Mach. Intell.*, vol. 2, no. 6, pp. 325–336, 2020.
- [22] C. D. Schuman, S. R. Kulkarni, M. Parsa, J. P. Mitchell, P. Date, and B. Kay, "Opportunities for neuromorphic computing algorithms and applications," *Nature Comput. Sci.*, vol. 2, no. 1, pp. 10–19, Jan. 2022.
- [23] Y. Liu and W. Pan, "Spiking neural-networks-based data-driven control," *Electronics*, vol. 12, no. 2, p. 310, Jan. 2023.
- [24] M. Ali Siddiqi et al., "A lightweight architecture for real-time neuronal-spike classification," 2023, *arXiv:2311.04808*.
- [25] M. Yao et al., "Attention spiking neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 8, pp. 9393–9410, Aug. 2023.
- [26] A. Al-Saegh, S. A. Dawwd, and J. M. Abdul-Jabbar, "Deep learning for motor imagery EEG-based classification: A review," *Biomed. Signal Process. Control*, vol. 63, Jan. 2021, Art. no. 102172.
- [27] J. Cui, Z. Lan, O. Sourina, and W. Müller-Wittig, "EEG-based cross-subject driver drowsiness recognition with an interpretable convolutional neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7921–7933, Oct. 2023.
- [28] H. Dong, A. Supratak, W. Pan, C. Wu, P. M. Matthews, and Y. Guo, "Mixed neural network approach for temporal sleep stage classification," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 26, no. 2, pp. 324–333, Feb. 2018.
- [29] J. Wang, R. Gao, H. Zheng, H. Zhu, and C.-J. R. Shi, "SSGCNet: A sparse spectra graph convolutional network for epileptic EEG signal classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 9, pp. 12157–12171, Sep. 2024.
- [30] H. Dose, J. S. Møller, H. K. Iversen, and S. Puthusserypady, "An end-to-end deep learning approach to MI-EEG signal classification for BCIs," *Exp. Syst. Appl.*, vol. 114, pp. 532–542, Dec. 2018.
- [31] Y. Li, L. Guo, Y. Liu, J. Liu, and F. Meng, "A temporal-spectral-based squeeze-and-excitation feature fusion network for motor imagery EEG decoding," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 1534–1545, 2021.
- [32] V. Lawhern, A. Solon, N. Waytowich, S. M. Gordon, C. Hung, and B. J. Lance, "EEGNet: A compact convolutional neural network for EEG-based brain-computer interfaces," *J. Neural Eng.*, vol. 15, no. 5, 2018, Art. no. 056013.

- [33] J.-S. Huang, W.-S. Liu, B. Yao, Z.-X. Wang, S.-F. Chen, and W.-F. Sun, "Electroencephalogram-based motor imagery classification using deep residual convolutional networks," *Frontiers Neurosci.*, vol. 15, Nov. 2021, Art. no. 774857.
- [34] A. I. Humayun, A. S. Sushmit, T. Hasan, and M. I. H. Bhuiyan, "End-to-end sleep staging with raw single channel EEG using deep residual ConvNets," in *Proc. IEEE EMBS Int. Conf. Biomed. Health Informat. (BHI)*, May 2019, pp. 1–5.
- [35] P. Autthasan et al., "MIN2Net: End-to-end multi-task learning for subject-independent motor imagery EEG classification," *IEEE Trans. Biomed. Eng.*, vol. 69, no. 6, pp. 2105–2118, Jun. 2022.
- [36] X. Ma, S. Qiu, C. Du, J. Xing, and H. He, "Improving EEG-based motor imagery classification via spatial and temporal recurrent neural networks," in *Proc. 40th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2018, pp. 1903–1906.
- [37] L.-M. Zhao, X. Yan, and B.-L. Lu, "Plug-and-play domain adaptation for cross-subject EEG-based emotion recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 1, pp. 863–870.
- [38] L. Xia et al., "MuLHiTA: A novel multiclass classification framework with multibranch LSTM and hierarchical temporal attention for early detection of mental stress," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 9657–9670, Dec. 2023.
- [39] T. Brown et al., "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1877–1901.
- [40] J. Achiam et al., "GPT-4 technical report," 2023, *arXiv:2303.08774*.
- [41] L. Zhang, F. Xiao, and Z. Cao, "Multi-channel EEG signals classification via CNN and multi-head self-attention on evidence theory," *Inf. Sci.*, vol. 642, Sep. 2023, Art. no. 119107.
- [42] X. Wang and Z. Wang, "CNN with self-attention in EEG classification," in *Proc. Int. Conf. Human-Comput. Interact.* Cham, Switzerland: Springer, Jan. 2022, pp. 512–526.
- [43] Y. Wen, W. He, and Y. Zhang, "A new attention-based 3D densely connected cross-stage-partial network for motor imagery classification in BCI," *J. Neural Eng.*, vol. 19, no. 5, Oct. 2022, Art. no. 056026.
- [44] J. Luo et al., "A shallow mirror transformer for subject-independent motor imagery BCI," *Comput. Biol. Med.*, vol. 164, Sep. 2023, Art. no. 107254.
- [45] Y. Ma, Y. Song, and F. Gao, "A novel hybrid CNN-transformer model for EEG motor imagery classification," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2022, pp. 1–8.
- [46] R. Zhang, G. Liu, Y. Wen, and W. Zhou, "Self-attention-based convolutional neural network and time-frequency common spatial pattern for enhanced motor imagery classification," *J. Neurosci. Methods*, vol. 398, Oct. 2023, Art. no. 109953.
- [47] T. Wang, J. Mao, R. Xiao, W. Wang, G. Ding, and Z. Zhang, "Residual learning attention CNN for motion intention recognition based on EEG data," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, Oct. 2021, pp. 1–6.
- [48] H. Li, H. Chen, Z. Jia, R. Zhang, and F. Yin, "A parallel multi-scale time-frequency block convolutional neural network based on channel attention module for motor imagery classification," *Biomed. Signal Process. Control*, vol. 79, Jan. 2023, Art. no. 104066.
- [49] C.-C. Fan, H. Yang, Z.-G. Hou, Z.-L. Ni, S. Chen, and Z. Fang, "Bilinear neural network with 3-D attention for brain decoding of motor imagery movements from the human EEG," *Cognit. Neurodynamics*, vol. 15, no. 1, pp. 181–189, Feb. 2021.
- [50] W. Cai et al., "A spatial-channel-temporal-fused attention for spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 10, pp. 14315–14329, Oct. 2024.
- [51] A. Barton, E. Volna, M. Kotyrba, and R. Jarusek, "Proposal of a control algorithm for multiagent cooperation using spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 4, pp. 2016–2027, Apr. 2023.
- [52] Q. Liu, G. Pan, H. Ruan, D. Xing, Q. Xu, and H. Tang, "Unsupervised AER object recognition based on multiscale spatio-temporal features and spiking neurons," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5300–5311, Dec. 2020.
- [53] D. Liu, N. Bellotto, and S. Yue, "Deep spiking neural network for video-based disguise face recognition based on dynamic facial movements," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 1843–1855, Jun. 2020.
- [54] A. Safa et al., "Improving the accuracy of spiking neural networks for radar gesture recognition through preprocessing," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 6, pp. 2869–2881, Jun. 2023.
- [55] M. Dampfhofer, T. Mesquida, A. Valentian, and L. Anghel, "Backpropagation-based learning techniques for deep spiking neural networks: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 9, pp. 11906–11921, Sep. 2024.
- [56] T. Zhang, S. Jia, X. Cheng, and B. Xu, "Tuning convolutional spiking neural network with biologically plausible reward propagation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 7621–7631, Dec. 2022.
- [57] S. Schliebs and N. Kasabov, "Evolving spiking neural network—A survey," *Evolving Syst.*, vol. 4, no. 2, pp. 87–98, Jun. 2013.
- [58] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-based spiking deep convolutional neural networks for object recognition," *Neural Netw.*, vol. 99, pp. 56–67, Mar. 2018.
- [59] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Front. Neurosci.*, vol. 11, p. 682, Dec. 2017.
- [60] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Front. Neurosci.*, vol. 12, p. 331, May 2018.
- [61] J. M. Antelis and L. E. Falcón, "Spiking neural networks applied to the classification of motor tasks in EEG signals," *Neural Netw.*, vol. 122, pp. 130–143, Feb. 2020.
- [62] Y. Luo et al., "EEG-based emotion classification using spiking neural networks," *IEEE Access*, vol. 8, pp. 46007–46016, 2020.
- [63] N. Kasabov and E. Capecci, "Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes," *Inf. Sci.*, vol. 294, pp. 565–575, Feb. 2015.
- [64] X. Wu et al., "Improving NeuCube spiking neural network for EEG-based pattern recognition using transfer learning," *Neurocomputing*, vol. 529, pp. 222–235, Apr. 2023.
- [65] Z. Yan, J. Zhou, and W.-F. Wong, "Energy efficient ECG classification with spiking neural network," *Biomed. Signal Process. Control*, vol. 63, Jan. 2021, Art. no. 102170.
- [66] Z. Yan, J. Zhou, and W.-F. Wong, "EEG classification with spiking neural network: Smaller, better, more energy efficient," *Smart Health*, vol. 24, Jun. 2022, Art. no. 100261.
- [67] S. Ghosh-Dastidar and H. Adeli, "Improved spiking neural networks for EEG classification and epilepsy and seizure detection," *Integr. Comput.-Aided Eng.*, vol. 14, no. 3, pp. 187–212, May 2007.
- [68] S. M. Bohte, J. N. Kok, and H. La Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, nos. 1–4, pp. 17–37, Oct. 2002.
- [69] W. Fang et al., "Parallel spiking neurons with high efficiency and ability to learn long-term dependencies," *Adv. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 53674–53687.
- [70] E. O. Nefci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 51–63, Nov. 2019.
- [71] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, *Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies*. IEEE Press, 2001. [Online]. Available: https://www.researchgate.net/profile/Y-Bengio/publication/2839938_Gradient_Flow_in_Recurrent_Nets_the_Difficulty_of_Learning_Long-Term_Dependencies/links/546cd26e0cf2193b94c577c2/Gradient-Flow-in-Recurrent-Nets-the-Difficulty-of-Learning-Long-Term-Dependencies.pdf
- [72] M.-H. Lee et al., "EEG dataset and OpenBMI toolbox for three BCI paradigms: An investigation into BCI illiteracy," *GigaScience*, vol. 8, no. 5, May 2019, Art. no. giz002.
- [73] Y. Guo et al., "Membrane potential batch normalization for spiking neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2023, pp. 19420–19430.
- [74] M. Tangermann et al., "Review of the BCI competition IV," *Front. Neurosci.*, vol. 6, p. 55, Jul. 2012.
- [75] Y. Hu, Y. Wu, L. Deng, and G. Li, "Advancing residual learning towards powerful deep spiking neural networks," 2021, *arXiv:2112.08954*.
- [76] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers Neurosci.*, vol. 9, p. 437, Nov. 2015.
- [77] H. Li, H. Liu, X. Ji, G. Li, and L. Shi, "CIFAR10-DVS: An event-stream dataset for object classification," *Frontiers Neurosci.*, vol. 11, p. 309, May 2017.

- [78] A. Amir et al., "A low power, fully event-based gesture recognition system," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7243–7252.
- [79] J. K. Holland, E. K. Kemsley, and R. H. Wilson, "Use of Fourier transform infrared spectroscopy and partial least squares regression for the detection of adulteration of strawberry purées," *J. Sci. Food Agricult.*, vol. 76, no. 2, pp. 263–269, Feb. 1998.
- [80] H. A. Dau et al., "The UCR time series archive," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1293–1305, Nov. 2019.



Chuhan Zhang (Graduate Student Member, IEEE) received the B.Eng. degree in computer science from Harbin Institute of Technology, Harbin, China, in 2020, and the M.Eng. degree in computer science from KU Leuven, Leuven, Belgium, in 2022. She is currently pursuing Ph.D. degree with TU Delft, Delft, The Netherlands.

Her research interests include spiking neural networks (SNNs) and learning-based robot control.



Wei Pan (Member, IEEE) received the Ph.D. degree in bioengineering from the Imperial College London, London, U.K., in 2016.

He was an Assistant Professor in robot dynamics with the Department of Cognitive Robotics and the Co-Director of Delft SELF AI Laboratory, TU Delft, Delft, The Netherlands. He is currently a Senior Lecturer in machine learning with the Department of Computer Science and a member of the Centre for AI Fundamentals and the Centre for Robotics and AI, The University of Manchester, Manchester, U.K.

His research interests include machine learning and control theory with applications in robotics.



Cosimo Della Santina (Senior Member, IEEE) received the Ph.D. degree in robotics from the University of Pisa, Pisa, Italy, in 2019.

He was a Visiting Ph.D. Student and a Post-Doctoral Researcher with Massachusetts Institute of Technology, Boston, MA, USA, from 2017 to 2019. Since 2020, he has been with the German Aerospace Centre, Weßling, Germany, as an External Research Scientist. He is currently an Associate Professor with the Department of Cognitive Robotics, Delft University of Technology, Delft, The Netherlands.

His research interests include modeling and control of soft robots, and learning dynamical systems.