STT-BP: Training Kernel-Learnable LIF for Efficient Neuromorphic Vision and Audio Recognition via Spatio-Temporal Tilted Backpropagation

Haoran Gao
School of Microelectronics and Communication Engineering
Chongqing University
Chongqing, China
gaohaoran@cqu.edu.cn

Xiang Fu
School of Microelectronics and Communication Engineering
Chongqing University
Chongqing, China
fuxiang@cqu.edu.cn

Yihang Chen
School of Microelectronics and Communication Engineering
Chongqing University
Chongqing, China
chenyihang@cqu.edu.cn

Xinyu Li
School of Microelectronics and Communication Engineering
Chongqing University
Chongqing, China
lixinyu@stu.cqu.edu.cn

Cong Shi*
School of Microelectronics and Communication Engineering
Chongqing University
Chongqing, China
shicong@cqu.edu.cn

***Abstract*—Spiking Neural Networks (SNNs) offer a promising avenue for energy-efficient edge intelligence, yet their performance is often bottlenecked by the rigid, fixed dynamics of traditional neuron models like LIF and CuBa-LIF. These models typically rely on pre-defined exponential decay kernels, which fail to capture the diverse multi-scale temporal dependencies present in varying neuromorphic modalities. To overcome this limitation, we propose the Kernel-Learnable LIF (KL-LIF) neuron, a generalized model capable of end-to-end learning arbitrary temporal response kernels (Integrated and Reset kernels). To train deep networks composed of KL-LIF neurons efficiently, we derive the Spatio-temporal Tilted Backpropagation (STT-BP) algorithm. By exploiting the duality between forward spike propagation and backward error flow, STT-BP establishes a tilted gradient path via kernelized error gradients, enabling unified optimization of synaptic weights and temporal kernels without complex state unrolling. Extensive experiments on neuromorphic vision (DVS-Gesture, DVS-CIFAR10) and spiking audio (SHD, SSC) datasets demonstrate that STT-BP significantly outperforms state-of-the-art methods. Notably, our approach reveals distinct optimal temporal scales for different modalities—shorter kernels for vision and longer kernels for audio—achieving top-tier accuracy.**

Keywords—Spiking Neural Networks, Learnable Kernel, Spatio-temporal Tilted Backpropagation, Event-based Vision, Spiking Audio Recognition

I. INTRODUCTION

Deep Neural Networks (DNNs) have achieved remarkable success in a wide range of fields, including computer vision and natural language processing [1]. However, their reliance on continuous-valued tensor multiplications and high-precision memory access has led to prohibitive energy consumption and computational costs, posing significant challenges for deployment on resource-constrained edge devices [2]. As Moore's Law slows down, Spiking Neural Networks (SNNs), often regarded as the third generation of neural networks, have emerged as a promising alternative for realizing energy-efficient machine intelligence [3]. SNNs mimic the information processing mechanism of the biological brain by using discrete, binary spikes for communication. This event-driven nature allows SNNs to exploit high Spatio-temporal sparsity, significantly reducing energy consumption when implemented on neuromorphic hardware. Furthermore, SNNs are inherently well-suited for processing data from neuromorphic sensors such as Dynamic Vision Sensors (DVS) [4]. Unlike traditional frame-based cameras that suffer from high latency and motion blur, these sensors generate asynchronous Spatio-temporal event streams with microsecond-level temporal resolution. SNNs can theoretically leverage the high temporal resolution and rich temporal dynamics embedded in these sensory streams for rapid and efficient recognition [5].

Despite these advantages, training high-performance deep SNNs remains a challenging task. Existing training methods can be broadly categorized into three approaches. The first is ANN-to-SNN conversion, which maps weights from a pre-trained ANN to an SNN [6]. While recent works have achieved nearly lossless conversion [7], [8], this method typically requires long simulation time-steps to approximate high-precision activation values, sacrificing the low-latency advantage of SNNs and struggling to handle native temporal dynamics of neuromorphic datasets. The second category is based on Spike-Timing-Dependent Plasticity (STDP), a biologically plausible unsupervised learning rule [9]. Despite being energy-efficient, STDP relies on local synaptic updates, making it difficult to perform global error assignment in multi-layer deep networks, thus limiting its performance on complex classification tasks [10], [11]. The third and currently most effective approach is Backpropagation-based (BP-based), like the Spatio-temporal Backpropagation (STBP) [12] or the Backpropagation Through Time (BPTT) [13]. By utilizing surrogate gradients to overcome the non-differentiability of spike generation, these methods allow for the direct training of deep SNNs [14]. Notable works such as SLAYER [15], various STBP improvements [16], [17] and STOP [18] have achieved state-of-the-art results.

Although BP-based training achieves many advancements, a fundamental limitation persists in the intrinsic dynamics of their reliance on the widely used Leaky Integrate-and-Fire (LIF) and Current-based Leaky Integrate-and-Fire (CuBa-LIF) models. While computationally simple, this fixed exponential

decay parameter imposes a rigid memory structure on the neuron, forcing all temporal information to decay at a uniform, preset rate. This lack of flexibility is particularly detrimental when processing neuromorphic sensory data, such as DVS vision streams or spiking audio, which inherently contain multi-scale temporal features and complex long-term dependencies. Consequently, neurons with fixed dynamics act as static low-pass filters, lacking the adaptability required to extract optimal temporal features from diverse input patterns, thereby bottlenecking the representational power of deep SNNs.

To break the constraints of fixed dynamics, we propose a novel neuron model that generalizes the standard LIF and CuBa-LIF neuron into a learnable form. By relaxing the restriction on the kernel shape, we introduce the Kernel-Learnable LIF (KL-LIF) neuron. Unlike traditional models that rely on hyperparameter-tuned decay constants, KL-LIF parameterizes the temporal response kernels (including an integrated kernel and a reset kernel) as learnable parameters. This generalization endows each neuron with the capability to adaptively learn arbitrary kernel shapes, effectively transforming them into trainable temporal feature extractors tailored to the input data. To efficiently train deep networks constructed with these generalized neurons, we derive the Spatio-temporal Tilted Backpropagation (STT-BP) algorithm. Instead of relying on conventional BPTT/STBP which effectively unrolls fixed recursive equations, STT-BP exploits the duality between spike-based forward propagation and error-based backward propagation. It establishes a "tilted" gradient propagation path via Kernelized Error Gradients, enabling efficient end-to-end learning of both synaptic weights and temporal kernels within a unified framework.

The main contributions of this work are summarized as follows:

1. A novel Kernel-Learnable LIF (KL-LIF) neuron model is proposed. This model generalizes the traditional LIF and Cuba-LIF neurons to support the learning of arbitrary-shaped temporal response kernels (i.e., Integrated Kernel and Reset Kernel) in deep SNNs, endowing the network with adaptive temporal feature extraction capabilities.

2. The Spatio-temporal Tilted Backpropagation (STT-BP) algorithm is derived to enable efficient training of SNN composed by the KL-LIF neurons. By establishing a tilted gradient flow via kernelized error gradients, this algorithm solves the challenge of training deep SNNs based on arbitrary kernel formulations, providing a unified and efficient learning framework without the need for iterative state expansion.

3. Extensive experiments on multiple neuromorphic image and audio datasets, including DVS-Gesture, DVS-CIFAR10, SHD, and SSC, demonstrate the superiority of our approach. The results indicate that the proposed framework outperforms existing SOTA methods in recognition accuracy, showcasing the distinct advantage of using KL-LIF and STT-BP for processing complex Spatio-temporal dynamics.

II. PRELIMINARIES

A. Standard LIF Neuron

The most commonly used spiking neuron model in SNNs is the LIF neuron, characteristic of a good balance between biological plausibility and computational efficiency. The LIF spiking neuron continuously integrates weighted incoming (i.e., presynaptic) spikes onto its membrane potential via a set of synapses. Once its membrane potential crosses the firing threshold, the neuron issues an outgoing (i.e., postsynaptic) spike, and immediately resets itself by subtracting the threshold from the membrane potential. Meanwhile, the membrane potential is exponentially leaking all the time. More particularly, the temporal dynamics of a LIF neuron in the discrete time domain can be formulated as:

$$U_j^l[t] = \alpha(U_j^l[t-1] - \theta s_j^l[t-1]) + \sum_i w_{ji}^l s_i^{l-1}[t] \tag{1}$$

$$s_j^l[t] = H(U_j^l[t] - \theta) \tag{2}$$

where t represents the discrete time-step, U_j^l, s_j^l, and θ ($\theta > 0$) are the membrane potential, the fired binary spike, and the firing threshold of neuron j in layer l, respectively, while α ($0 \le \alpha \le 1$) means the leakage factor, w_{ji}^l denotes the weight of the synapse connecting this neuron and a particular presynaptic neuron indexed as i in the preceding layer, and $H(\cdot)$ delegates the firing action based on the Heavyside step function: $H(x) = 1$ or 0 when $x \ge 0$ or $x < 0$, respectively. If we expand Eq. 1 along the temporal dimension, the membrane potential can be expressed as a convolution of input spikes and an implicit exponential kernel:

$$\begin{aligned} U_j^l[t] &= \sum_i w_{ji}^l \sum_{t' \le t} s_i^{l-1}[t'] \alpha^{t-t'} - \theta \sum_{t' < t} s_j^l[t'] \alpha^{t-t'} \\ &= \sum_i w_{ji}^l (\alpha^t \otimes s_i^{l-1}[t]) - \theta(\alpha^t \otimes s_j^l[t]) \end{aligned} \tag{3}$$

where $\otimes$ represents temporal convolution, α^t acts as a fixed exponential decay kernel.

B. CuBa-LIF Neuron

Another common neuron model is the CuBa-LIF neuron model [19], introducing a synaptic current state, leading to a second-order dynamic system. The membrane of the CuBa-LIF can be expressed as follows:

$$U_j^l[t] = \sum_i w_{ji}^l ((\alpha_1^t - \alpha_2^t) \otimes s_i^{l-1}[t]) - \theta(\alpha_3^t \otimes s_j^l[t]) \tag{4}$$

where $0 \le \alpha_2 < \alpha_1 \le \alpha_3 \le 1$, and α_1^t, α_2^t, α_3^t are different exponential decay kernels. Especially, when $\alpha_2 = 0$ and $\alpha_1 = \alpha_3$, the CuBa-LIF neuron will turn into an LIF neuron.

As can be seen from Eq. 3 and Eq. 4, both standard LIF and CuBa-LIF neurons rely on pre-defined, fixed exponential decay kernel combinations, which restricts the representative capability of spiking neuron models. The situation can be alleviated by employing higher-order and more complex exponential decay kernels combinations, but manually adjusting these parameters will bring huge computational overheads.

III. METHODOLOGY

A. Kernel-Learnable LIF neuron

Instead of restricting the neuron to fixed exponential decay kernel combinations, we propose to train the kernel directly to enhance the adaptive ability of neuron model. As such, we generalize kernel-based forms of LIF and CuBa-LIF neurons into the Kernel-Learnable LIF (KL-LIF), as shown in Fig. 1. The membrane potential of a KL-LIF neuron is defined as:

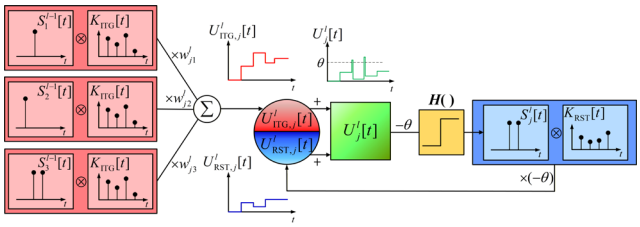

Fig. 1. Kernel-Learnable LIF (KL-LIF) neuron model

$$\begin{aligned} U_j^l[t] &= U_{\mathrm{ITG},j}^l[t] + U_{\mathrm{RST},j}^l[t] \\ &= \sum_i w_{ji}^l (K_{\mathrm{ITG}}[t] \otimes s_i^{l-1}[t]) - \theta (K_{\mathrm{RST}}[t] \otimes s_j^l[t]) \\ &= \sum_i w_{ji}^l KS_{\mathrm{ITG},i}^{l-1}[t] - \theta KS_{\mathrm{RST},j}^l[t] \end{aligned} \tag{5}$$

where $\otimes$ represents temporal convolution, $U_{\mathrm{ITG},j}^l$ represents integrated membrane potential component, $U_{\mathrm{RST},j}^l$ represents reset membrane potential component, K_{ITG} represents integrated kernel, K_{RST} represents reset kernel. $KS_{\mathrm{ITG},i}^{l-1}[t]$, $KS_{\mathrm{RST},j}^l[t]$ represent integrated and reset kernelized spike map, respectively, whose nature is temporal convolution result of the spike sequence and kernel. And its spike firing function is same with Eq. 2. Different from LIF and CuBa-LIF, the kernel $K_{\mathrm{ITG}}[t]$, $K_{\mathrm{RST}}[t]$ of KL-LIF can be learned rather than assigned with fixed exponential decay kernel combinations. The learning process will be discussed in the following sections.

B. Basic Concept of STT-BP

The primary motivation and fundamental principle of the proposed STT-BP algorithm are to construct the gradient flow of KL-LIF neurons by exploiting the duality between spike-based forward propagation and error-based backward propagation. Instead of treating the temporal dynamics strictly as a recurrent dependency that must be unrolled step-by-step, STT-BP establishes a "tilted" gradient propagation path via Kernelized Error Gradients. This formulation allows the gradients for both spatial weights and temporal kernels to be derived efficiently from a unified Spatio-temporal representation. Specifically, to achieve this goal, we start from the total output loss E^*, and investigate its gradient with respect to a generic parameter, say v_j^l in neuron j of layer l:

$$\begin{aligned} \Delta v_j^l &= \frac{\partial E^*}{\partial v_j^l} = \sum_t \frac{\partial E[t]}{\partial v_j^l} \\ &= \sum_t \frac{\partial E[t]}{\partial U_j^l[t]} \frac{\partial U_j^l[t]}{\partial v_j^l} \end{aligned} \tag{6}$$

Here, we also define

$$\tilde{v}_j^l[t] = \frac{\partial E[t]}{\partial v_j^l[t]} \tag{7}$$

as the *instantaneous error* of v_j^l for a single time-step t. As a result, the error gradient at the output spike s_j^l of neuron j in layer l at time-step t is:

$$\tilde{s}_j^l[t] = \frac{\partial E[t]}{\partial s_j^l[t]} \tag{8}$$

While the error gradients at spikes $\tilde{s}_j^l[t]$ alone are insufficient for parameter optimization, as the learnable components—namely the synaptic weights and temporal kernels—modulate the network output indirectly by influencing the continuous membrane potential. Consequently, to close the learning loop, we must propagate the error from the discrete spike domain back to the membrane potential domain via the chain rule. This derived gradient with respect to the membrane potential constitutes the fundamental part in our STT-BP framework, acting as the nexus that distributes error gradients to both spatial and temporal parameters.

In the remainder of this section, we solve the neuron errors $\tilde{s}_j^l[t]$, membrane potential errors $\tilde{U}_j^l[t]$, and figure out the expected updates of the synaptic weights, the integrated kernel and the reset kernel.

C. Neuron Error Computation

We solve the instantaneous neuron errors $\tilde{s}_j^l[t]$ of all neurons via Spatio-temporal BP procedure. For the output layer $l = L$, by taking into account Eq. 2 and 5, we obtain:

$$\begin{aligned} \tilde{U}_j^L[t] &= \frac{\partial E[t]}{\partial U_j^L[t]} = \frac{\partial E[t]}{\partial s_j^L[t]} \frac{\partial s_j^L[t]}{\partial U_j^L[t]} \\ &= \tilde{s}_j^l[t] \varphi_{\mathrm{SG}}(U_j^L[t] - \theta) \end{aligned} \tag{9}$$

where $\varphi_{\mathrm{SG}}(\cdot)$ is surrogate gradient function to handle the non-differentiability of spikes, the loss function $E[t]$ can use cross-entropy or mean square error. Similarly, for any layer l, we can derive the error gradient of the membrane potential as:

$$\tilde{U}_j^l[t] = \frac{\partial E[t]}{\partial s_j^l[t]} \frac{\partial s_j^l[t]}{\partial U_j^l[t]} = \tilde{s}_j^l[t] \frac{\partial s_j^l[t]}{\partial U_j^l[t]} = \tilde{s}_j^l[t] \varphi_{\mathrm{SG}}(U_j^l[t] - \theta) \tag{10}$$

And based on Eq. 5, we can get:

$$\tilde{U}_{\mathrm{ITG},j}^l[t] = \frac{\partial E[t]}{\partial U_{\mathrm{ITG},j}^l[t]} = \frac{\partial E[t]}{\partial U_j^l[t]} \frac{\partial U_j^l[t]}{\partial U_{ITG,j}^l[t]} = \tilde{U}_j^l[t] \tag{11}$$

$$\tilde{U}_{\mathrm{RST},j}^l[t] = \frac{\partial E[t]}{\partial U_{\mathrm{RST},j}^l[t]} = \frac{\partial E[t]}{\partial U_j^l[t]} \frac{\partial U_j^l[t]}{\partial U_{\mathrm{RST},j}^l[t]} = \tilde{U}_j^l[t] \tag{12}$$

Inspired by Eq. 5, the error gradient $\tilde{s}_j^l[t]$ at the neuron i in layer l-1 comprises the component backpropagated through the integrated component U_{ITG} in layer l across the Spatio-temporal domain, and the reset component backpropagated through the U_{RST} of neuron i itself in the temporal domain. Thus, $\tilde{s}_i^{l-1}[t]$ can be inferenced as:

$$\begin{aligned} \tilde{s}_i^{l-1}[t] &= \sum_{t' \geq t} \frac{\partial E[t']}{\partial s_i^{l-1}[t]} \\ &= \sum_{t' \geq t} \sum_j \frac{\partial E[t']}{\partial U_{\mathrm{ITG},j}^l[t']} \frac{\partial U_{\mathrm{ITG},j}^l[t']}{\partial s_i^{l-1}[t]} + \sum_{t' > t} \frac{\partial E[t']}{\partial U_{\mathrm{RST},i}^{l-1}[t']} \frac{\partial U_{\mathrm{RST},i}^{l-1}[t']}{\partial s_i^{l-1}[t]} \\ &= \sum_{t' \geq t} \sum_j \tilde{U}_j^l[t'] \frac{\partial U_{\mathrm{ITG},j}^l[t']}{\partial s_i^{l-1}[t]} + \sum_{t' > t} \tilde{U}_i^{l-1}[t'] \frac{\partial U_{\mathrm{RST},i}^{l-1}[t']}{\partial s_i^{l-1}[t]} \end{aligned} \tag{13}$$

where the partial derivative terms in the two summation terms in the last row of the above equation respectively represent: i) the contribution of the spike fired by the neuron i in the preceding layer l-1 at time t to the U_{ITG} component of the neuron j in the succeeding layer l at time t', and 2) the contribution of that same spike to the U_{RST} component of neuron i itself at time t'. Based on the definitions of the kernel K_{ITG} and K_{RST} mentioned in Eq. 5, as well as the Linear Time-Invariant (LTI) properties of the spike responses for U_{ITG} and U_{RST}, we can derive the following:

$$\frac{\partial U^{l}_{\mathrm{ITG},j}[t']}{\partial s^{l-1}_{i}[t]} = w^{l}_{ji} K_{\mathrm{ITG}}[t'-t] \tag{14}$$

$$\frac{\partial U^{l-1}_{\mathrm{RST},i}[t']}{\partial s^{l-1}_{i}[t]} = -\theta K_{\mathrm{RST}}[t'-t] \tag{15}$$

Taking Eq. 14 and 15 into Eq. 13, we will derive：

$$\begin{aligned}\tilde{s}^{l-1}_{i}[t] &= \sum_{t'\geq t}\sum_{j} \tilde{U}^{l}_{j}[t'] w^{l}_{ji} K_{\mathrm{ITG}}[t'-t] - \sum_{t'>t} \tilde{U}^{l-1}_{i}[t'] \theta K_{\mathrm{RST}}[t'-t] \\ &= \sum_{j} w^{l}_{ji} \sum_{t'\geq t} \tilde{U}^{l}_{j}[t'] K_{\mathrm{ITG}}[t'-t] - \theta \sum_{t'>t} \tilde{U}^{l-1}_{i}[t'] K_{\mathrm{RST}}[t'-t] \\ &= \sum_{j} w^{l}_{ji} KU^{l}_{\mathrm{ITG},j}[t] - \theta KU^{l-1}_{\mathrm{RST},i}[t]\end{aligned} \tag{16}$$

where we define $KU^{l}_{\mathrm{ITG},j}[t] = \sum_{t'\geq t} \tilde{U}^{l}_{j}[t'] K_{\mathrm{ITG}}[t'-t]$ and $KU^{l-1}_{\mathrm{RST},i}[t] = \sum_{t'>t} \tilde{U}^{l-1}_{i}[t'] K_{\mathrm{RST}}[t'-t]$ as the corresponding kernelized error maps, which are essentially the time-domain correlation calculations between the membrane potential errors and the kernel functions. Then we define the corresponding kernelized error gradients as:

$$\tilde{s}^{l-1}_{\mathrm{ITG},i}[t] = \sum_{j} w^{l}_{ji} KU^{l}_{\mathrm{ITG},j}[t] \tag{17}$$

$$\tilde{s}^{l-1}_{\mathrm{RST},i}[t] = -\theta KU^{l-1}_{\mathrm{RST},i}[t] \tag{18}$$

And Eq. 16 becomes as:

$$\tilde{s}^{l-1}_{i}[t] = \tilde{s}^{l-1}_{\mathrm{ITG},i}[t] + \tilde{s}^{l-1}_{\mathrm{RST},i}[t] \tag{19}$$

Eq. 10 and Eq. 17-19 constitute the error gradient propagation process of STT-BP. The calculation commences from the output layer L at the final time step $t = T$ to obtain the neuron errors for each layer. It then proceeds to calculate the output layer errors and the errors of preceding layers at time $t = T - 1$, continuing iteratively until the error calculations for all layers at the first time-step are completed. The data flow dependencies of the error gradients like $\tilde{s}^{l-1}_{\mathrm{ITG},i}[t]$ and $\tilde{s}^{l-1}_{\mathrm{RST},i}[t]$ during this backpropagation process are illustrated in the ITG Backward Propagation and RST Backward Propagation paths in Figure 2. It is evident that the majority of the error data flow propagates in a tilted manner across the Spatio-temporal domain.

Similar to the forward propagation phase, during backpropagation, at each time step t, KU_{ITG} for any neuron in the succeeding layer needs to be calculated only once to form a KU_{ITG} kernelized error map. This map is then transmitted to the preceding layer to perform spatial dimension operations with its synaptic weights (at this stage, the synaptic weight requires spatial transposition, analogous to the backpropagation process in ANNs).

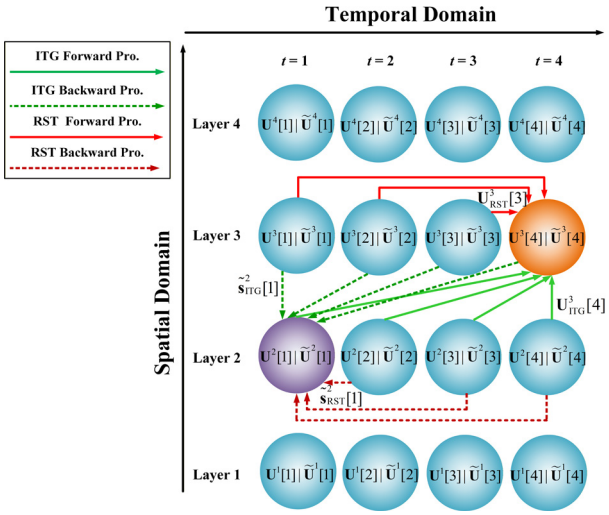

Fig. 2. Spatio-temporal paths of forward signal computation and backward error propagation in STT-BP learning.

D. Update of Weight

Based on Eq. 5 and 6, the weights are updated as follows:

$$\Delta w^{l}_{ij} = \sum_{t} \frac{\partial E[t]}{\partial w^{l}_{ij}} == \sum_{t} \frac{\partial E[t]}{\partial U^{l}_{\mathrm{ITG},j}[t]} \frac{\partial U^{l}_{\mathrm{ITG},j}[t]}{\partial w^{l}_{ij}} \tag{20}$$

Inspired from Eq. 5 and 11，we can obtain：

$$\frac{\partial U^{l}_{\mathrm{ITG},j}[t]}{\partial w^{l}_{ij}} = (K_{\mathrm{ITG}}[t] \otimes s^{l-1}_{i}[t]) = KS^{l-1}_{\mathrm{ITG},i}[t] \tag{21}$$

Taking Eq. 21 into Eq. 20, and we reach：

$$\Delta w^{l}_{ij} = \sum_{t} \frac{\partial E[t]}{\partial w^{l}_{ij}} == \sum_{t} \tilde{U}^{l}_{j}[t] KS^{l-1}_{\mathrm{ITG},i}[t] \tag{22}$$

For convolutional layers, multiple positions on the KS_{ITG} kernelized spike map of the preceding layer involve convolution calculations using shared convolutional kernel weights to generate the U_{ITG} component of the succeeding layer. Since the weights of the convolution kernel are shared, each weight within the kernel influences the membrane potentials at multiple locations in the subsequent layer. Therefore, we must accumulate the gradient contributions from the membrane potentials of all output neurons in the succeeding layer to the same convolution kernel weight, as described in Eq. 22. The weights are then updated based on these accumulated values. This process is similar to the weight error gradient calculation in ANNs. Overall, it can be viewed as the convolution of the $\tilde{U}^{l}_{j}[t]$ and $KS^{l-1}_{\mathrm{ITG},i}[t]$ to obtain the weight error gradient at time t.

E. Update of Integrated Kernel and Reset Kernel

The essence of the temporal kernels is that they act as weights in the temporal dimension. Just as synaptic weights in the spatial dimension can be acquired through learning, kernel functions can also be learned. We assume that neurons within each layer share the same set of the kernels $K^l_{\mathrm{ITG}}[t]$ and $K^l_{\mathrm{RST}}[t]$. First, we need to calculate the error gradient of the loss at time step t with respect to the element p of $K^l_{\mathrm{ITG}}[t]$, denoted as $K^l_{\mathrm{ITG}}[p]$. Combining this with Eq. 5, we obtain:

$$\begin{aligned}\frac{\partial E[t]}{\partial K^l_{\mathrm{ITG}}[p]} &= \sum_j \frac{\partial E[t]}{\partial U^l_{\mathrm{ITG},j}[t]} \sum_i \frac{\partial U^l_{\mathrm{ITG},j}[t]}{\partial KS^{l-1}_{\mathrm{ITG},i}[t]} \frac{\partial KS^{l-1}_{\mathrm{ITG},i}[t]}{\partial K^l_{\mathrm{ITG}}[p]} \\ &= \sum_j \widetilde{U}^l_j[t] \sum_i w^l_{ji} \frac{\partial KS^{l-1}_{\mathrm{ITG},i}[t]}{\partial K^l_{\mathrm{ITG}}[p]}\end{aligned} \tag{23}$$

Assuming $t' = t - p$, then we get：

$$\begin{aligned}\frac{\partial KS^{l-1}_{\mathrm{ITG},i}[t]}{\partial K^l_{\mathrm{ITG}}[p]} &= \frac{\partial(\sum_{t'\le t} s^{l-1}_i[t']K_{\mathrm{ITG}}[t-t'])}{\partial K^l_{\mathrm{ITG}}[p]} \\ &= \begin{cases} s^{l-1}_i[t-p], & t \ge p \\ 0 \quad\quad , & t < p \end{cases}\end{aligned} \tag{24}$$

Substituting Eq. 24 into Eq. 23 leads to：

$$\frac{\partial E[t]}{\partial K^l_{\mathrm{ITG}}[p]} = \begin{cases} \sum_j \widetilde{U}^l_j[t] \sum_i w^l_{ji} s^{l-1}_i[t-p], & t \ge p \\ 0 \quad\quad\quad\quad\quad\quad , & t < p \end{cases} \tag{25}$$

Finally, the update for the integrated kernel $K^l_{\mathrm{ITG}}[p]$ is obtained:

$$\Delta K^l_{\mathrm{ITG}}[p] = \sum_t \frac{\partial E[t]}{\partial K^l_{\mathrm{ITG}}[p]} = \sum_{t\ge p}\sum_j \widetilde{U}^l_j[t] \sum_i w^l_{ji} s^{l-1}_i[t-p] \tag{26}$$

Similarly, by calculating the error gradient of the loss at a single time step t with respect to the element q of $K^l_{\mathrm{RST}}[t]$, denoted as $K^l_{\mathrm{RST}}[q]$, and combining it with Eq. 5, we obtain:

$$\begin{aligned}\frac{\partial E[t]}{\partial K^l_{\mathrm{RST}}[q]} &= \sum_j \frac{\partial E[t]}{\partial U^l_{\mathrm{RST},j}[t]} \frac{\partial U^l_{\mathrm{RST},j}[t]}{\partial KS^l_{\mathrm{RST},j}[t]} \frac{\partial KS^l_{\mathrm{RST},j}[t]}{\partial K^l_{\mathrm{RST}}[q]} \\ &= -\theta \sum_j \widetilde{U}^l_j[t] \frac{\partial KS^l_{\mathrm{RST},j}[t]}{\partial K^l_{\mathrm{RST}}[q]}\end{aligned} \tag{27}$$

Assuming $t' = t - q$, then we get：

$$\begin{aligned}\frac{\partial KS^l_{\mathrm{RST},j}[t]}{\partial K^l_{\mathrm{RST}}[q]} &= \frac{\partial(\sum_{t'<t} s^l_j[t']K_{\mathrm{RST}}[t-t'])}{\partial K^l_{\mathrm{RST}}[q]} \\ &= \begin{cases} s^l_j[t-q], & t > q \\ 0, \quad t \le q \end{cases}\end{aligned} \tag{28}$$

Taking Eq. 28 into Eq. 27：

$$\frac{\partial E[t]}{\partial K^l_{\mathrm{RST}}[q]} = \begin{cases} -\theta \sum_j \widetilde{U}^l_j[t] s^l_j[t-q], & t > q \\ 0, \quad t \le q \end{cases} \tag{29}$$

Eventually, the update for the reset kernel $K^l_{\mathrm{RST}}[q]$ is :

$$\Delta K^l_{\mathrm{RST}}[p] = \sum_t \frac{\partial E[t]}{\partial K^l_{\mathrm{RST}}[p]} = -\theta \sum_{t>p}\sum_j \widetilde{U}^l_j[t] s^l_j[t-p] \tag{30}$$

IV. Experiments

A. Experimental Setup

To evaluate the capability of the proposed STT-BP algorithm in handling tasks involving complex Spatio-temporal dynamics, we conducted experiments on two types of neuromorphic datasets: event-based vision datasets (DVS-Gesture [20], DVS-CIFAR10 [21]) and spiking audio datasets (Spiking Heidelberg Digits (SHD) and Spiking Speech Commands (SSC) [22]). These datasets were selected as they inherently contain rich temporal information and long-term dependencies.

For the DVS-CIFAR10 and DVS-Gesture datasets, we adopted the direct input encoding scheme consistent with previous works [23]. The asynchronous event streams were segmented into T fixed-length slices, where each slice contains an equal number of events. The events within each slice were integrated pixel-wisely to form a frame-like tensor, which was then fed into the network at each time-step. We set the time-steps T to 10 for DVS-CIFAR10 and 20 for DVS-Gesture. For the SHD and SSC datasets, we followed the preprocessing protocol described in [24]. The original input dimension of 700 channels was reduced to 140 by binning every 5 spatially adjacent input channels. In the temporal dimension, we utilized a discrete time-step of t = 10 ms.

Correspondingly, we employed varying network architectures tailored to the complexity of these tasks. For the DVS-CIFAR10 and DVS-Gesture datasets, we utilized a streamlined VGG-11 architecture (64C3-128C3-256C3-256C3-512C3-512C3-512C3-512C3-AveragePooling-200-100-Class) and VGG-9 (only has one fully-connected layer). Conversely, for the SHD and SSC tasks, we designed 1D convolutional spiking neural networks to process the temporal audio features. Specifically, the SHD network consists of three 1D convolutional layers with channel dimensions of 256, 256, and 20 (corresponding to the number of classes), while the SSC network employs a deeper structure with four 1D convolutional layers utilizing channel dimensions of 512, 512, 512, and 35, respectively.

All experiments were implemented using the PyTorch framework. During the training phase, we utilized the Cross-Entropy (CE) loss function. To address the non-differentiability of spike generation during backpropagation, the Sigmoid function was employed as the surrogate gradient. Regarding the optimization strategy, we applied the SGD optimizer for the DVS vision datasets and the Adam optimizer for the spiking audio datasets. For all tasks, the momentum was set to 0.9, and a Cosine Annealing scheduler was applied to dynamically adjust the learning rate throughout the training epochs. Finally, valid initialization is vital for our STT-BP framework. The learnable kernels K_{ITG}、K_{RST} of the KL-LIF neurons were explicitly initialized to exponential decay dynamics. This strategy ensures the network begins with biologically plausible behavior before progressively learning more complex kernel shapes driven by the task requirements. All experiments were conducted on an NVIDIA RTX A6000 GPU.

B. *Validation Results*

To verify the effectiveness of the proposed Kernel-Learnable LIF (KL-LIF) neuron and investigate the impact of the kernel length K on temporal feature extraction, we conducted ablation studies across different modalities, including neuromorphic vision (DVS-Gesture, DVS-CIFAR10) and spiking audio (SHD, SSC) datasets. The kernel length determines the neuron's temporal receptive field, which is crucial for capturing task-specific temporal dependencies.

Figure 3 illustrates classification accuracies with various temporal kernel lengths. We observed distinct behaviors for vision and audio tasks, supporting the necessity of adaptive kernel learning. For DVS-based datasets, the optimal kernel length tends to be relatively short. On DVS-Gesture, the accuracy peaks at 98.96% when $K = 4$. Further increasing the length to 6 leads to a slight degradation 98.26%, and a very short kernel $K = 2$ yields 94.44%. This indicates that a moderate temporal window is sufficient to capture gesture motion patterns without introducing redundant noise. For DVS-CIFAR10, the highest accuracy of 83.50% is achieved at a short kernel length of $K = 2$, with performance dropping to 81.20% ($K = 4$) and 80.70% ($K = 6$) as the kernel lengthens. This suggests that for object recognition from accumulated event frames, the spatial features are dominant, and excessively long temporal integration may blur critical instantaneous cues.

In contrast, spiking audio datasets exhibit a clear preference for longer temporal receptive fields, reflecting the inherent long-term dependencies in speech signals. On the SHD dataset, increasing K from 5 to 10 results in a substantial accuracy boost from 91.21% to 94.74%. Similarly, on the SSC dataset, the accuracy improves from 76.74% ($K = 5$) to 79.32% ($K = 10$). However, blindly increasing the length (e.g., $K = 15$) causes performance saturation or decline (SHD: 92.80%, SSC: 78.80%), likely due to optimization difficulties or overfitting with overly complex kernels.

These results quantitatively demonstrate that different neuromorphic tasks require distinct temporal scales. The fixed decay dynamics in traditional LIF neurons cannot adapt to such diverse requirements. Our STT-BP framework with KL-LIF neurons successfully identifies the optimal operating point for each task—shorter kernels for rapid visual processing and longer kernels for temporal auditory integration—thereby maximizing recognition performance.

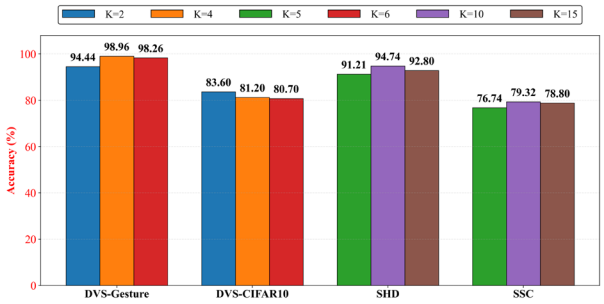

Fig. 3. Learning Accuracy across Various Datasets with Different Kernel Lengths.

TABLE I. CLASSIFICATION ACCURACY ON DVS-GESTURE AND DVS-CIFAR10

Dataset	Training Method	Spiking Neuron Model	SNN Architecture	#Time-steps	Acc. (%)
DVS-Gesture	STBP [13]	LIF	VGG-11	20	93.40[1]
	STBP + tdBN [17]	LIF	ResNet-17	40	96.87
	SLTT [23]	LIF	VGG-11	20	97.92
	STOP [18]	LIF	VGG-11	20	98.26
	STBP + KLIF [25]	KLIF	VGG-11	12	94.10
	STBP + PLIF [26]	PLIF	VGG-11	20	97.57
	STT-BP (ours)	**KL-LIF**	**VGG-11**	**20**	**98.96**
	STT-BP (ours)	**KL-LIF**	**VGG-9**	**20**	**98.96**
DVS-CIFAR10	EIHL [56]	LIF	VGG-11	6	62.45
	STBP [13]	LIF	VGG-11	10	67.40[1]
	STBP + tdBN [17]	LIF	ResNet-19	10	67.80
	SLTT [23]	LIF	VGG-11	10	77.30
	TET [36]	LIF	VGG-11	10	77.33
	STOP [18]	LIF	VGG-11	10	77.10
	STBP + PLIF [26]	PLIF	VGG-11	20	74.80
	STT-BP (ours)	**KL-LIF**	**VGG-11**	**10**	**83.60**
	STT-BP (ours)	**KL-LIF**	**VGG-9**	**10**	**83.50**

[1] accuracy obtained from [18]

C. *Comparison with Previous Work*

Table I compares the DVS-Gesture and DVS-CIFAR10 classification accuracies of the proposed STT-BP algorithm with prior state-of-the-art deep SNN training methods. It indicates that our STT-BP algorithm achieves superior performance across both dataset tasks. For DVS-Gesture**,** we achieved a top-1 accuracy of 98.96% even using a more compact VGG-9 SNN. This result outperforms the fully synergistic STOP algorithm (98.26%) [18] and the SLTT method (97.92%) [23], demonstrating the advantage of our learnable kernel formulation in capturing the dynamic features of moving gestures. As for DVS-CIFAR10, the superiority of STT-BP is even more pronounced. It reached an accuracy of 83.50% with the smaller VGG-9 SNN, surpassing the TET (77.33%) [27] using a deeper VGG-11 structure by a significant margin of +6.17%. Notably, compared to STBP + tdBN (67.80%) [17] and STOP (77.10%) [18], our approach demonstrates that relaxing the constraints on neuron dynamics via kernel learning is crucial for handling complex event streams.

Table II compares classification accuracies on spiking audio datasets SHD and SSC. Such audio tasks require the network to process long-term temporal dependencies with high precision. Associated with used model parameter sizes are also reported in Table II, as such metric is a key indicator in this domain. For the SHD dataset, STT-BP achieved 94.74% accuracy, outperforming the STBP + RadLIF (94.62%) [38] and STBP + Dense Conv Delays (93.44%) [24]. Similarly, on the SSC dataset, our method set a new hallmark with 79.32% accuracy, exceeding the STBP + Dense Conv Delays (78.44%). Another key advantage of our STT-BP framework is its relatively low parameter overhead. As shown in Table II, for the SSC task, the STBP + Dense Conv Delays requires 19.4 MB of parameters to model temporal delays explicitly. In contrast, our STT-BP achieves higher accuracy with only

TABLE II. CLASSIFICATION ACCURACY ON SHD, SSC DATASET

Dataset	Method	Spiking Neuron Model	#Params (MB)	Acc. (%)
SHD	BPTT + Hete. RSNN [28]	Cuba-LIF	0.11	82.7
	EventProp [29]	LIF	1.78	84.8
	BPTT + SpikGRU [30]	Cuba-LIF	0.28	87.8
	BPTT + Adaptive SRNN [31]	ALIF	0.14	90.4
	BPTT + Delays [32]	LIF	0.10	90.4
	STBP + TA [33]	LIAF	0.11	91.0
	STBP + STSC [34]	LIF	2.27	92.3
	BPTT + Adaptive Delays [35]	SRM	0.11	92.4
	STBP + RPSU [36]	RPSU	1.78	92.4
	STBP + Dloss [37]	LIF	0.21	92.5
	STBP + Dense Conv Delays	LIF	2.66	93.4
	STBP + RadLIF [38]	adLIF	3.89	94.6
	STT-BP (ours)	**KL-LIF**	**0.52**	**94.7**
SSC	BPTT + Hete. RSNN [28]	Cuba-LIF	0.11	57.3
	STBP + Spiking CNN [39]	LIF	0.61	72.0
	BPTT + Adaptive SRNN [31]	ALIF	0.77	74.2
	BPTT + SpikGRU [30]	Cuba-LIF	0.28	77.0
	STBP + RadLIF [38]	adLIF	3.9	77.4
	STBP + Dense Conv Delays	LIF	19.4	78.4
	STT-BP (ours)	**KL-LIF**	**2.93**	**79.3**

2.93 MB of parameters—an approximate 6.6 times reduction in the model size.

The above efficiency of our method in coping with both temporal visual and audio neuromorphic streams stems from the KL-LIF neuron's ability to embed temporal dynamics directly into the learnable kernels $K^l_{\mathrm{ITG}}[t]$ and $K^l_{\mathrm{RST}}[t]$ rather than relying on parameter-heavy dense connections or explicit delay buffers. By the titled backpropagation path, STT-BP optimizes these compact temporal kernels end-to-end, enabling the network to learn rich temporal features with a minimal footprint. This characteristic makes STT-BP particularly suitable for deployment on resource-constrained neuromorphic edge devices.

V. CONCLUSION

In this work, we present a novel learning method for deep SNNs that addresses the limitations of fixed neuronal dynamics. By introducing the KL-LIF neuron and the STT-BP algorithm, we enable the adaptive learning of temporal response kernels alongside synaptic weights. Our theoretical formulation of tilted gradient propagation simplifies the training of complex temporal dynamics into efficient kernelized operations. Experimental results across diverse neuromorphic benchmarks validated the superiority of our approach. We quantitatively demonstrated that visual tasks benefit from compact temporal kernels, whereas auditory tasks require extended kernels to capture long-term dependencies. The STT-BP algorithm successfully discovers these optimal operating points, achieving state-of-the-art accuracy on DVS-Gesture, DVS-CIFAR10, SHD, and SSC datasets. Furthermore, our method exhibits remarkable parameter efficiency, reducing the model size for complex audio tasks by approximately 6.6 times compared to explicit delay modeling approaches. These findings suggest that STT-BP offers a scalable and efficient solution for deploying high-performance SNNs on resource-constrained neuromorphic hardware, paving the way for adaptive edge intelligence capable of handling complex Spatio-temporal data streams.

ACKNOWLEDGMENT

This work was funded in part by the National Natural Science Foundation of China under Grant No. 92464103 and 62334008, in part by the Graduate Research and Innovation Foundation of Chongqing, China (Grant No. CYB240027), in part by Chongqing Overseas Personnel Returning to China for Entrepreneurship and Innovation Support Program under Grant No. cx2025012, and in part by College Students' Innovative Entrepreneurial Training Plan Program (Grant No. 202610611147). We also appreciate the participation of Yi Chen in the experiments.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] V. Sze, Y. Chen, T. Yang, and J. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[3] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019.

[4] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128× 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[5] G. Gallego *et al.*, "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, 2020.

[6] P. U. Diehl, D. Neil, J. Binas, et al., "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," *2015 Internation al Joint Conference on Neural Networks* (*IJCNN*), Killarney, Ireland, pp. 1-8, Jul. 2015.

[7] B. Han, G. Srinivasan, and K. Roy, "RMP-SNN: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 13555–13564.

[8] H. Gao, J. He, H. Wang, *et al.*, "High-accuracy deep ANN-to-SNN conversion using quantization-aware training framework and calcium-gated bipolar leaky integrate and fire neuron," *Frontiers in Neuroscience*, vol. 17, pp.1-11, Mar. 2023.

[9] N. Caporale and Y. Dan, "Spike timing-dependent plasticity: a Hebbian learning rule," *Annual Review of Neuroscience*, vol. 31, pp. 25–46, 2008.

[10] M. Mozafari, S. R. Kheradpisheh, T. Masquelier, A. Nowzari-Dalini, and M. Ganjtabesh, "Firstspike-based visual categorization using reward-modulated STDP," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 6178–6190, 2018.

[11] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-based spiking deep convolutional neural networks for object recognition," *Neural Networks*, vol. 99, pp. 56–67, 2018.

[12] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[13] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-Temporal backpropagation for training highperformance spiking neural networks," *Frontiers in Neuroscience*, vol. 12, p. 331, 2018.

[14] E. O. Neftci, 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 Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[15] S. B. Shrestha and G. Orchard, "SLAYER: Spike layer error reassignment in time," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 1412–1421.

[16] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi, "Direct training for spiking neural networks: Faster, larger, better," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 1311–1318, 2019.

[18] H. Gao, X. Zhou, Y. Lin, *et al.*, "STOP: spatiotemporal orthogonal propagation for weight-threshold-leakage synergistic training of deep spiking," *Neuromorphic computing and Engineering*, vol. 5, no. 4, pp. 1–18, 2025.

[19] R**.** Gütig, "Spiking neurons can discover predictive features by aggregate-label learning," *Science*, vol. 351, no. 6277, Mar. 2016.

[20] A. Amir, B. Taba, D. Berg, *et al.*, "A low power, fully event-based gesture recognition system," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, pp. 7388-7397, Jul. 2017.

[21] H. Li, H. Liu, X. Ji, G. Li, and L. Shi, "CIFAR10-DVS: an event-stream dataset for object classification," *Frontiers in neuroscience*, vol. 11, pp. 1-10, Jun. 2017.

[22] B. Cramer, Y. Stradmann, J. Schemmel, and F. Zenke. "The heidelberg spiking data sets for the systematic evaluation of spiking neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 7, pp. 2744–2757, Jul. 2022.

[23] Q. Meng, M. Xiao, S. Yan, *et al.*, "Towards Memory- and Time-Efficient Backpropagation for Training Spiking Neural Networks," *2023 IEEE/CVF International Conference on Computer Vision* (*ICCV*), Paris, France, pp. 6143-6153, Oct. 2023.

[24] I. Hammouamri, I. Khalfaoui-Hassani and T. Masquelier, "Learning Delays in Spiking Neural Networks using Dilated Convolutions with Learnable Spacings," *2024 International Conference on Representation Learning* (*ICLR*), Vienna, Austria, pp. 1-14, May 2024.

[25] C. Jiang and Y. Zhang, "Klif: An optimized spiking neuron unit for tuning surrogate gradient slope and membrane potential," *Neural Computation*, vol. 36, no. 12, Nov. 2024.

[26] W. Fang, Z. Yu, Y. Chen, T. Masquelier, T. Huang, and Y. Tian, "Incorporating learnable membrane time constant to enhance learning of spiking neural networks," in *Proceedings of the IEEE/CVF international conference on computer vision* (*ICCV*), pp. 2661–2671, Oct. 2021.

[27] X. Yao, F. Li, Z. Mo, and J. Cheng, "GLIF: A unified gated leaky integrateand-fire neuron for spiking neural networks," *Advances in Neural Information Processing Systems* (*NeurIPS*), New Orleans, pp. 1-12, 2022.

[28] N. Perez-Nieves, V. C. Leung, P. L. Dragotti, and D. F. Goodman, "Neural heterogeneity promotes robust learning," *Nature communications*, vol. 12, no. 1, pp. 1-9, Oct. 2021.

[29] T. Nowotny, J. P. Turner, and J. C. Knight, "Loss shaping enhances exact gradient learning with eventprop in spiking neural networks," *Neuromorphic Computing and Engineering*, vo. 5, no.1, pp. 1-21, Jan. 2025.

[30] M. Dampfhoffer, T. Mesquida, A. Valentian, and L. Anghel, "Investigating current-based and gating approaches for accurate and energy-efficient spiking recurrent neural networks," *Artificial Neural Networks and Machine Learning* (*ICANN*), Bristol, United Kingdom, pp. 359–370, Sep. 2022.

[31] B. Yin, F. Corradi, and S. M. Bohte, "Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks," *Nature Machine Intelligence*, vol.3, pp. 905-913, Oct. 2021.

[32] A. P.-Saucedo, A. Yousefzadeh, G. Tang, *et al.*, "Empirical study on the efficiency of spiking neural networks with axonal delays, and algorithm-hardware benchmarking." In *2023 IEEE International Symposium on Circuits and Systems* (*ISCAS*), Monterey, CA, USA, pp. 1–5, May 2023.

[33] M. Yao, H. Gao, G. Zhao, *et al.*, "Temporal-wise attention spiking neural networks for event streams classification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (*ICCV*), pp. 10221–10230, Oct. 2021.

[34] C. Yu, Z. Gu, D. Li, *et al.*, "STSC-SNN: Spatio-Temporal Synaptic Connection with temporal convolution and attention for spiking neural networks," *Frontiers in Neuroscience*, vol. 16, pp. 1-15, Dec. 2022.

[35] P. Sun, E. Eqlimi, Y. Chua, P. Devos, and D. Botteldooren, "Adaptive axonal delays in feedforward spiking neural networks for accurate spoken word recognition," *2023 IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), Rhodes Island, Greece, pp. 1-4, Jun. 2023.

[17] H. Zheng, Y. Wu, L. Deng, Y. Hu, and G. Li, "Going deeper with directly-trained larger spiking neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, pp. 11062–11070, 2021.

[36] Y. Li, Y. Sun, X. He, *et al.*, "Parallel Spiking Unit for Efficient Training of Spiking Neural Networks," *2024 International Joint Conference on Neural Networks* (*IJCNN*), Yokohama, Japan, pp. 1-8, Jul. 2024.

[37] P. Sun, Y. Chua, P. Devos, and D. Botteldooren, "Learnable axonal delay in spiking neural networks improves spoken word recognition," *Frontiers in Neuroscience*, vol. 17, pp. 1-12, Nov. 2023.

[38] A. Bittar and P. N. Garner, "A surrogate gradient spiking baseline for speech command recognition," *Frontiers in Neuroscience*, vol. 16, pp. 1-18, Aug. 2022.

[39] E. Sadovsky, M.s Jakubec, and R. Jarina, "Speech command recognition based on convolutional spiking neural networks," In 2023 33rd International Conference Radioelektronika (RADIOELEKTRONIKA), Pardubice, Czech Republic, pp. 1–5, Apr. 2023.