TCJA-SNN: Temporal-Channel Joint Attention for Spiking Neural Networks

Rui-Jie Zhu, *Graduate Student Member, IEEE*, Malu Zhang, *Member, IEEE*, Qihang Zhao, Haoyu Deng, Yule Duan, and Liang-Jian Deng, *Senior Member, IEEE*

Abstract*— Spiking neural networks (SNNs) are attracting widespread interest due to their biological plausibility, energy efficiency, and powerful spatiotemporal information representation ability. Given the critical role of attention mechanisms in enhancing neural network performance, the integration of SNNs and attention mechanisms exhibits tremendous potential to deliver energy-efficient and high-performance computing paradigms. In this article, we present a novel temporal-channel joint attention mechanism for SNNs, referred to as TCJA-SNN. The proposed TCJA-SNN framework can effectively assess the significance of spike sequence from both spatial and temporal dimensions. More specifically, our essential technical contribution lies on: 1) we employ the squeeze operation to compress the spike stream into an average matrix. Then, we leverage two local attention mechanisms based on efficient 1-D convolutions to facilitate comprehensive feature extraction at the temporal and channel levels independently and 2) we introduce the cross-convolutional fusion (CCF) layer as a novel approach to model the interdependencies between the temporal and channel scopes. This layer effectively breaks the independence of these two dimensions and enables the interaction between features. Experimental results demonstrate that the proposed TCJA-SNN outperforms the state-of-the-art (SOTA) on all standard static and neuromorphic datasets, including Fashion-MNIST, CIFAR10, CIFAR100, CIFAR10-DVS, N-Caltech 101, and DVS128 Gesture. Furthermore, we effectively apply the TCJA-SNN framework to image generation tasks by leveraging a variation autoencoder. To the best of our knowledge, this study is the first instance where the SNN-attention mechanism has been employed for high-level classification and low-level generation tasks. *Our implementation codes are available at https://github.com/ridgerchu/TCJA.

***Index Terms*— Attention mechanism, neuromorphic datasets, spatiotemporal information, spiking neural networks (SNNs).**

I. Introduction

SPIKING neural networks (SNNs) have emerged as a promising research area, offering lower energy consumption and superior robustness compared to conventional artificial neural networks (ANNs) [1], [2]. These characteristics make SNNs highly promising for temporal data processing and power-critical applications [1], [3]. In recent years, significant progress has been made by incorporating backpropagation into SNNs [3], [4], [5], [6], [7], [8], [9], [10], which allows the integration of various ANN modules into SNN architectures, including batch normalization blocks [11] and residual blocks [12]. By leveraging these ANN-based methods, it becomes possible to train large-scale SNNs while preserving the energy efficiency associated with SNN's binary spiking nature.

Despite significant progress, SNNs have yet to fully exploit the superior representational capability of deep learning, primarily due to their unique training mode, which struggles to model complex channel-temporal relationships effectively. To address this limitation, Zheng et al. [11] introduced a batch normalization method for the temporal dimension, overcoming issues of gradient vanishing and threshold-input balance. On the other hand, Wu et al. [13] proposed a method named NeuNorm to address the channel-wise challenges. NeuNorm includes an auxiliary neuron that adjusts the stimulus strength generated by the preceding layer, enhancing performance while mimicking the activity of the retina and nearby cells for added bio-plausibility. However, existing methods handle temporal and channel information separately, leading to limited joint information extraction. Given that SNNs reuse network parameters at each time step, there exists untapped potential for recalibration at both the temporal and channel dimensions. Especially, TA-SNN proposed by Yao et al. [14].

Previous studies in ANNs [15], [16] have often utilized the attention mechanism as a means to address the challenges posed by multidimensional dynamic problems. The attention mechanism, inspired by human cognitive processes, enables the selective focus on relevant information while disregarding irrelevant data. This approach has shown promise in the realm of SNNs and merits further exploration [14]. For instance, in the domain of neuroscience, an attention-based spike-timing-dependent plasticity (STDP) SNN was proposed by Bernert and Yvert [17] to solve the spike-sorting problem. Furthermore, Yao et al. [14] incorporated a channel attention block into the temporal-wise input of an SNN, as depicted in Fig. 1(a), enabling the assessment of frame significance during training and the exclusion of irrelevant frames during inference. Despite employing attention solely in the temporal dimension, this attention mechanism significantly improves the network's performance.

Manuscript received 16 July 2023; revised 28 December 2023 and 2 March 2024; accepted 12 March 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 12271083, Grant 62106038, and Grant 12171072; and in part by the National Key Research and Development Program of China under Grant 2020YFA0714001. *(Corresponding authors: Malu Zhang; Liang-Jian Deng.)*

Rui-Jie Zhu is with the University of California, Santa Cruz, Santa Cruz, CA USA, and also with the University of Electronic Science and Technology of China, Chengdu, China.

Malu Zhang, Haoyu Deng, Yule Duan, and Liang-Jian Deng are with the University of Electronic Science and Technology of China, Chengdu, China (e-mail: maluzhang@uestc.edu.cn; liangjian.deng@uestc.edu.cn).

Qihang Zhao is with Kuaishou Technology.

Digital Object Identifier 10.1109/TNNLS.2024.3377717

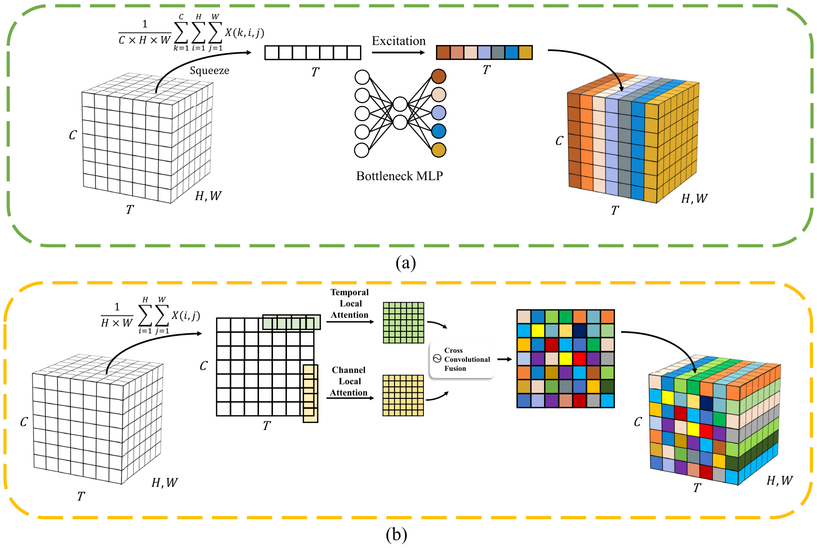
$\frac{1}{C\times H\times W}\sum_{k=1}^{C}\sum_{i=1}^{H}\sum_{j=1}^{W}X(k,i,j)$
Squeeze
T
Excitation
T
Bottleneck MLP
C
H,W
T
(a)
$\frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W}X(i,j)$
Temporal Local Attention
Channel Local Attention
Cross Convolutional Fusion
C
T
H,W
(b)

Fig. 1. How our TCJA differs from existing temporal-wise attention [14], which estimates the saliency of each time step by squeeze-and-excitation module. T denotes the time step, C denotes the channel, and H, W represents the spatial resolution. By utilizing two separate 1-D convolutional layers and the CCF operation, our TCJA establishes the association between the time step and the channel. (a) Temporal-wise attention. (b) TCJA.

In this article, we involve both temporal and channel attention mechanisms in SNNs, which is implemented by efficient 1-D convolution. Fig. 1(b) shows the whole structure, we argue that this cooperative mechanism can enhance the discrimination of the learned features and can make the temporal-channel learning in SNNs easier. The main contribution of this work can be summarized as follows.

1) We introduce a plug-and-play block into SNNs by considering the temporal and channel attentions cooperatively, which model temporal and channel information in the same phase, achieving better adaptability and bio-interpretability. To the best of our knowledge, this is the first attempt to incorporate the temporal-channel attention mechanism into the most extensively used model, leaky-integrate-and-fire (LIF)-based SNNs.
2) A cross-convolutional fusion (CCF) operation with a cross-receptive field is proposed to make use of the associated information. It not only uses the benefit of convolution to minimize parameters but also integrates features from both temporal and channel dimensions in an efficient fashion.
3) Experimental results show that the temporal-channel joint attention mechanism for SNN (TCJA-SNN) outperforms previous methods on static and neuromorphic datasets for classification tasks. It also performs well in generation tasks.

II. Related Works and Motivation

A. Training Techniques for SNNs

In recent years, the direct application of various ANN algorithms for training deep SNNs, including gradient-descent-based methods, has gained traction. However, the nondifferentiability of spikes poses a significant challenge. The Heaviside function, commonly used to trigger spikes, has a derivative that is zero everywhere except at the origin, rendering gradient-based learning infeasible. To overcome this obstacle, the commonly employed solutions are ANN-to-SNN [18], [19], [20] and the surrogate gradient descent method [21], [22], [23], [24], [25], [26], [27], [28].

During the forward pass, the Heaviside function is retained, while a surrogate function replaces it during the backward pass. One simple choice for the surrogate function is the Spike-Operator [29], which exhibits a gradient resembling a shifted ReLU function. In our work, we go beyond the conventional surrogate gradient method and introduce two additional surrogate functions: the ATan surrogate function and the triangle-like surrogate function designed by Fang et al. [30] and Bellec et al. [31]. These surrogates possess the capability to activate a specific range of samples, making them particularly suitable for the training of deep SNNs. By expanding the repertoire of surrogate functions, we aim to enhance the training process and improve the performance of deep SNNs.

B. Attention Mechanism in Convolutional Neural Networks

In the realm of ANNs, the squeeze and excitation (SE) block, introduced by Hu et al. [15], has proven to be a highly effective module for enhancing representation. The SE block can be seamlessly incorporated into a network, requiring only a minimal increase in parameters to recalibrate channel information. Employing squeezing and fully connecting operations allows the network to learn a trainable scale factor for each channel. This recalibration process significantly improves the discriminative power of individual channels. Recently, Yao et al. [14] extended the application of the SE block to SNNs by formulating a temporal-wise attention mechanism. This innovative approach enables SNNs to identify critical temporal frames of interest without being adversely affected by noise or interference. By incorporating temporal-wise attention, the proposed technique achieves state-of-the-art(SOTA) performance across various datasets. This accomplishment serves as compelling evidence for the immense potential of attention mechanisms within SNNs. The utilization of SE

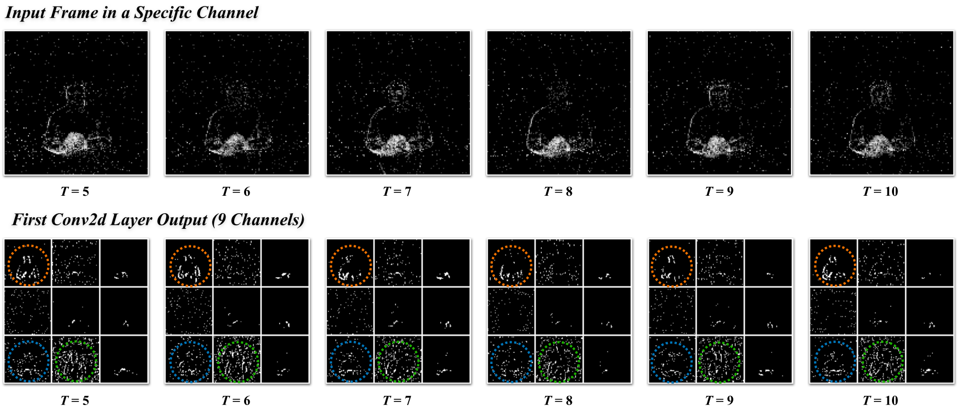
Input Frame in a Specific Channel
T = 5
T = 6
T = 7
T = 8
T = 9
T = 10
First Conv2d Layer Output (9 Channels)
T = 5
T = 6
T = 7
T = 8
T = 9
T = 10

Fig. 2. Correlation between proximity time steps and channels. The top row is the input frame selected from the DVS128 Gesture dataset. Each figure in the nine-pattern grid of the bottom row denotes a channel output from the first 2-D convolutional layer. It is clear that a significant correlation exists in channels with varying time steps, motivating us to merge the temporal and channel information.

blocks and the introduction of temporal-wise attention represent significant advancements in the field of SNNs. These techniques not only enhance the representation capability of SNNs, but also offer insights into effectively leveraging attention mechanisms for improved performance. In the following sections, we aim to explore and further leverage these attention mechanisms to improve the performance of SNNs and unlock their full potential in complex temporal data processing tasks.

C. Motivation

Based on the aforementioned analysis, the utilization of a temporal-wise attention mechanism in SNNs has exhibited substantial progress in effectively processing time-related data streams. Moreover, it has been observed in both biological neural networks [32] and ANNs [15] that recalibrating channel features within convolutional layers hold considerable potential for enhancing performance. Nevertheless, the existing SNNs-based works only process the data with either temporal or channel dimensions, thereby constraining the capacity for joint feature extraction. To illustrate the relationship between temporal steps and channel dimensions, we provide a visual representation. This is achieved by displaying the input frame alongside several adjacent channel outputs, which originate from the initial 2-D convolutional layer, as demonstrated in Fig. 2. As the circles indicate, a similar firing pattern can be distinguished from the surrounding time steps and channels. To fully use this associated information, we propose the temporal-channel joint attention (TCJA) module, a novel approach for modeling temporal and channel-wise frame correlations. Furthermore, considering the inevitable increases in the model parameters caused by the attention mechanism, we attempt to adopt the 1-D convolution operation to gain a reasonable tradeoff between model performance and parameters. Furthermore, existing SNN attention mechanisms primarily prioritize classification tasks, neglecting the needs of generation tasks. Our goal is to introduce an attention mechanism that can proficiently handle both classification and generation tasks, thereby establishing a universal attention mechanism for SNNs.

III. Methodology

A. Leaky Integrate and Fire Model

Various spiking neuron models have been proposed to simulate the functioning of biological neurons [33], [34], and among them, the LIF model [35] achieves a commendable balance between simplicity and biological plausibility. The membrane potential dynamics of an LIF neuron can be described as [13]

$$\tau \frac{dV(t)}{dt} = -(V(t) - V_{\text{reset}}) + I(t) \tag{1}$$

where τ denotes a time constant, $V(t)$ represents the membrane potential of the neuron at time t, and $I(t)$ represents the input from the presynaptic neurons. For better computational tractability, the LIF model can be described as an explicitly iterative version [1]

$$\begin{cases} \boldsymbol{V}_t^n = \boldsymbol{H}_{t-1}^n + \frac{1}{\tau}\left(\boldsymbol{I}_{t-1}^n - \left(\boldsymbol{H}_{t-1}^n - \boldsymbol{V}_{\text{reset}}\right)\right) \\ \boldsymbol{S}_t^n = \boldsymbol{\Theta}\left(\boldsymbol{V}_t^n - \boldsymbol{V}_{\text{threshold}}\right) \\ \boldsymbol{H}_t^n = \boldsymbol{V}_t^n \cdot \left(1 - \boldsymbol{S}_t^n\right) \end{cases} \tag{2}$$

$\boldsymbol{V}_t^n$ represents the membrane potential of neurons within the nth layer at time t. τ is a time constant, $\boldsymbol{S}$ is the spiking tensor with binary value, $\boldsymbol{I}$ denotes the input from the previous layer, $\boldsymbol{\Theta}(\cdot)$ denotes the Heaviside step function, and $\boldsymbol{H}$ represents the reset process after spiking.

As a mainstream neuron model, LIF-based SNN models can be trained directly using surrogate gradient methods [24] to attain SOTA performance [14], [30], [36]. Moreover, the LIF model is well-suited to common machine-learning frameworks because it allows forward and backward propagation along spatial and temporal dimensions. In our method, the parameters of the LIF model are set as follows: $\tau = 2$, $\boldsymbol{V}_{\text{reset}} = 0$, and $\boldsymbol{V}_{\text{threshold}} = 1$.

B. Temporal-Channel Joint Attention

As mentioned above, we contend that the frame at the current time step exhibits a significant correlation with its neighboring frames in both the channel and temporal dimensions. This correlation opens up the possibility of employing

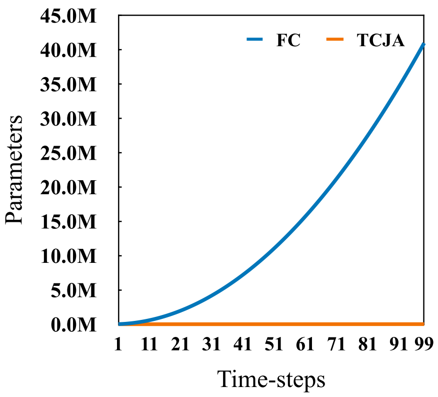

Fig. 3. Growth curve of parameters between the FC layer and the TCJA layer when channel size $C = 64$.

a mechanism to establish a connection between these two dimensions. Initially, we employed a fully connected (FC) layer to establish the correlation between the temporal and channel information, as it provides the most direct and prominent connection between these dimensions. However, as the number of channels and time steps increases, the number of parameters grows rapidly with a ratio of $T^2 \times C^2$, as illustrated in Fig. 3. Our subsequent attempt involved utilizing a 2-D convolutional layer for building this attention mechanism. Nevertheless, this approach encountered a limitation due to the fixed kernel size, which restricts the receptive field to a confined local area. In conventional CNNs, augmenting the number of layers can expand the receptive field [37], [38]. However, within the context of attention mechanisms, the feasibility of layer stacking, analogous to convolutional networks, is constrained, thereby limiting the receptive field when employing 2-D convolutions. For this reason, it is necessary to decrease the number of parameters while increasing the receptive field. In Section IV-G5, we provide a detailed theoretical analysis of the receptive field.

To effectively incorporate both temporal and channel attention dimensions while minimizing parameter usage and maximizing the receptive field, we present a novel attention mechanism termed TCJA. This attention mechanism is distinguished by its global cross-receptive field and its ability to achieve effective results with relatively fewer parameters, specifically $T^2 + C^2$. Fig. 4 shows the overall structure of the proposed TCJA, and we will introduce its key components in detail in the following. In Section III-B1, we utilize the squeezing operation on the input frame. Next, we introduce the temporal-wise local attention (TLA) mechanism and channel-wise local attention (CLA) mechanism in Sections III-B2 and III-B3, respectively. At last, we introduce the CCF mechanism to conjointly learn the information of temporal and channel in Section III-B4.

1) Average Matrix by Squeezing: To efficiently capture the temporal and channel correlations between frames, we first perform the squeeze step on the spatial feature map of the input frame stream $X \in \mathbb{R}^{T\times H\times W\times C}$, where C denotes the channel size, and T denotes the time step. The squeeze step calculates an average matrix $\mathcal{Z} \in \mathbb{R}^{C\times T}$ and each element $\mathcal{Z}_{(c,t)}$ of the average matrix $\mathcal{Z}$ as

$$\mathcal{Z}_{(c,t)} = \frac{1}{H \times W}\sum_{i=1}^{H}\sum_{j=1}^{W} X_{i,j}^{(c,t)} \tag{3}$$

where $X^{(c,t)}$ is the input frame of the cth channel at time step t.

2) Temporal-Wise Local Attention: Following the squeeze operation, we propose the TLA mechanism for establishing temporal-wise relationships among frames. We argue that the frame in a specific time step interacts substantially with the frames in its adjacent positions. Therefore, we adopt a 1-D convolution operation to model the local correspondence in the temporal dimension, as shown in Fig. 4. In detail, to capture the correlation of input frames at the temporal level, we perform C-channel 1-D convolution on each row of the average matrix $\mathcal{Z}$, and then accumulate the feature maps obtained by convolving different rows of the average matrix $\mathcal{Z}$. The whole TLA process can be described as

$$\mathcal{T}_{i,j} = \sum_{n=1}^{C}\sum_{m=0}^{K_T-1} W_{(n,i)}^{m}\mathcal{Z}_{(n,j+m)}. \tag{4}$$

Here, K_T ($K_T < T$) denotes the size of the convolution kernel, which indicates the number of time steps considered for the convolution operation. The parameter $W_{(n,i)}^{m}$ is a learnable parameter that represents the mth parameter of the ith channel when performing a 1-D convolution operation with C channels on the nth row of the input tensor $\mathcal{Z}$. $\mathcal{T} \in \mathbb{R}^{C\times T}$ is the attention score matrix after the TLA mechanism.

3) Channel-Wise Local Attention: As aforementioned, the frame-to-frame saliency score should not only take into account the temporal dimension but also take into consideration the information from adjacent frames along the channel dimension. To construct the correlation of different frames with their neighbors channel-wise, we propose the CLA mechanism. Similarly, as shown in Fig. 4, we perform T-channel 1-D convolution on each column of the matrix $\mathcal{Z}$, and then sum the convolution results of each row. This process can be described as

$$\mathcal{C}_{i,j} = \sum_{n=1}^{T}\sum_{m=0}^{K_C-1} E_{(n,j)}^{m}\mathcal{Z}_{(i+m,n)} \tag{5}$$

where K_C ($K_C < C$) represents the size of the convolution kernel, and $E_{(n,i)}^{m}$ is a learnable parameter, representing the mth parameter of the ith channel when performing T-channel 1-D convolution on the nth column of $\mathcal{Z}$. $\mathcal{C} \in \mathbb{R}^{C\times T}$ is the attention score matrix after CLA mechanism.

To maintain dimensional consistency between the input and output, a "same padding" technique is employed in both the TLA and CLA mechanisms. This padding strategy ensures that the output dimension matches the input dimension by adding an appropriate number of zeros to the input data. Specifically, this technique involves padding the input array with zeros on both sides, where the number of zeros added is determined based on the kernel size and the stride.

4) Cross-Convolutional Fusion: After TLA and CLA operations, we get the temporal (TLA matrix $\mathcal{T}$) and channel (CLA matrix $\mathcal{C}$) saliency scores of the input frame and its adjacent

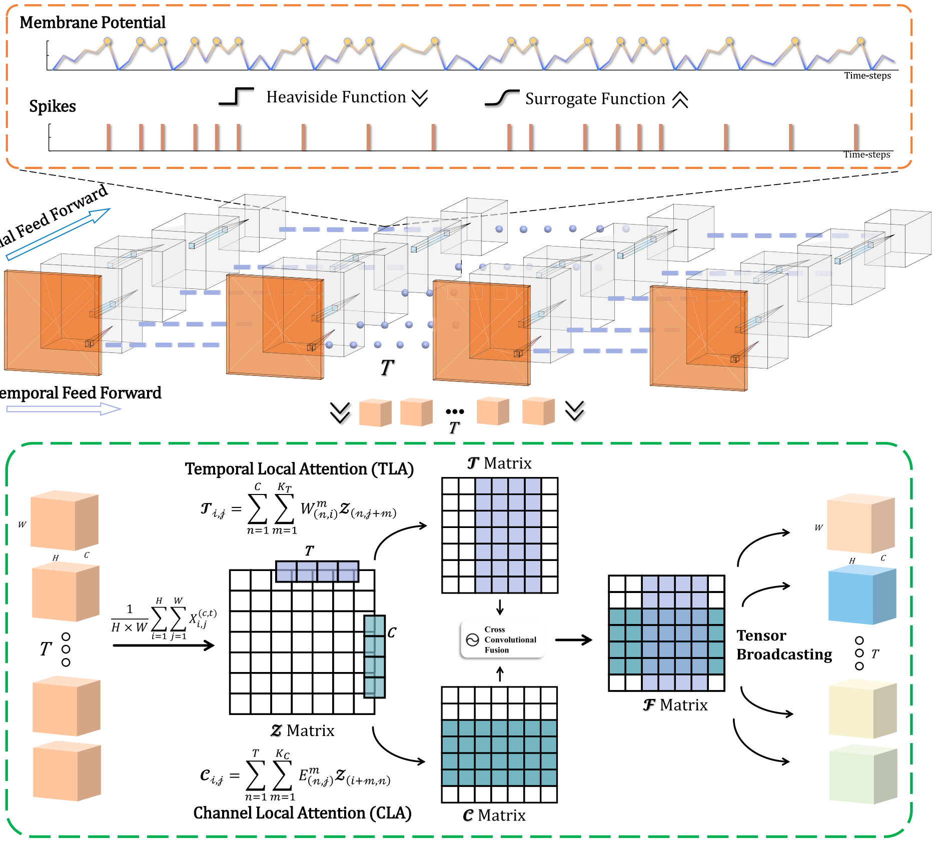
Membrane Potential
Time-steps
Spikes
Heaviside Function
Surrogate Function
Time-steps
Spatial Feed Forward
T
Temporal Feed Forward
T
Temporal Local Attention (TLA)
$\mathcal{T}_{i,j} = \sum_{n=1}^{C}\sum_{m=1}^{K_T} W_{(n,i)}^{m}\mathcal{Z}_{(n,j+m)}$
$\mathcal{T}$ Matrix
W
H
C
T
$\frac{1}{H \times W}\sum_{i=1}^{H}\sum_{j=1}^{W} X_{i,j}^{(c,t)}$
T
C
$\mathcal{Z}$ Matrix
Cross Convolutional Fusion
$\mathcal{F}$ Matrix
Tensor Broadcasting
T
W
H
C
$\mathcal{C}_{i,j} = \sum_{n=1}^{T}\sum_{m=1}^{K_C} E_{(n,j)}^{m}\mathcal{Z}_{(i+m,n)}$
Channel Local Attention (CLA)
$\mathcal{C}$ Matrix

Fig. 4. Framework of SNN with the TCJA module. In SNNs, information is transmitted in the form of spike sequences, encompassing both temporal and spatial dimensions. In temporal-wise, the spiking neuron with a threshold feed-forward in membrane potential ($\boldsymbol{V}$) and spike ($\boldsymbol{S}$) as the (2), and backpropagation with the surrogate function. Spatial-wise, data flows between layers as ANN. The TCJA module operates by initially compressing information along both temporal and spatial dimensions, then applying TLA and CLA to establish the relationship in both temporal and channel dimensions and blend them by CCF layer.

frames, respectively. Next, to learn the correlation between temporal and channel frames in tandem, we propose a cross-domain information fusion mechanism, that is, the CCF layer. The goal of CCF is to calculate a fusion information matrix $\mathcal{F}$, and $\mathcal{F}(\rangle, |)$ is used to measure the potential correlation between the ith channel of the jth input temporal frame and other frames. Specifically, the joint relationship between frames can be obtained by performing an element-wise multiplication of $\mathcal{T}$ and $\mathcal{C}$ as follows:

$$\begin{aligned}\mathcal{F}_{i,j} &= \boldsymbol{\sigma}\left(\mathcal{T}_{i,j} \cdot \mathcal{C}_{i,j}\right)\\ &= \boldsymbol{\sigma}\left(\sum_{n=1}^{C}\sum_{m=0}^{K_T-1} W_{(n,i)}^{m}\mathcal{Z}_{(n,j+m)} \cdot \sum_{n=1}^{T}\sum_{m=0}^{K_C-1} E_{(n,j)}^{m}\mathcal{Z}_{(i+m,n)}\right)\end{aligned} \tag{6}$$

where $\boldsymbol{\sigma}$ represents the Sigmoid function. Fig. 5 is provided to enhance the understanding of the entire computational process.

C. Training Framework

We integrate the TCJA module into the existing benchmark SNNs and propose the TCJA-SNN. Since the process of neuron firing is nondifferentiable, we utilize the derived ATan surrogate function $\sigma'(x) = (\alpha/(2(1 + ((\pi/2)\alpha x)^2)))$ and the derived triangle-like surrogate function $\epsilon'(x) = (1/\gamma^2)\max(0, \gamma - |x - 1|)$ for backpropagation, which is proposed by Fang et al. [30] and Bellec et al. [31], respectively. This latter function is particularly applied in the TCJA-TET-SNN, in alignment with the default surrogate function specification for temporal efficient training (TET)-based architectures. In our method, the spike mean-square-error (SMSE) [21], [30] is chosen as the loss function, which can be expressed as

$$\mathcal{L} = \frac{1}{T}\sum_{t=0}^{T-1}\mathcal{L}_t = \frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{E}\sum_{i=0}^{E-1}\left(s_{t,i} - g_{t,i}\right)^2 \tag{7}$$

where T denotes the simulation time step, E is the number of labels, s represents the network output, and g represents the one-hot encoded target label. We also employ the TET [36] loss, which can be represented as

$$\mathcal{L} = \frac{1}{T} \cdot \sum_{t=1}^{T}\mathcal{L}_{\text{CE}}\left[s(t), g(t)\right] \tag{8}$$

where T is the total simulation time, $\mathcal{L}_{\text{CE}}$ denotes the cross-entropy loss, s is the network output, and g represents the target label. The cross-entropy loss here can be represented by

$$\mathcal{L}_{\text{CE}}(p, y) = -\sum_{c=1}^{M} y_{o,c}\log\left(p_{o,c}\right) \tag{9}$$

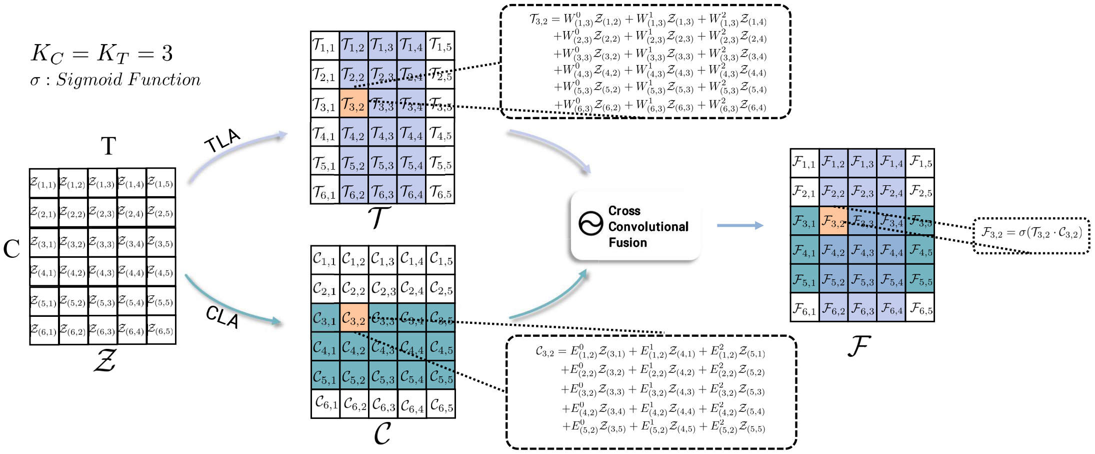
$K_C = K_T = 3$
σ : Sigmoid Function
T
C
TLA
CLA
$\mathcal{Z}$
$\mathcal{T}$
$\mathcal{C}$
$\mathcal{F}$
Cross Convolutional Fusion
$\mathcal{T}_{3,2} = W^0_{(1,3)}\mathcal{Z}_{(1,2)} + W^1_{(1,3)}\mathcal{Z}_{(1,3)} + W^2_{(1,3)}\mathcal{Z}_{(1,4)} + W^0_{(2,3)}\mathcal{Z}_{(2,2)} + W^1_{(2,3)}\mathcal{Z}_{(2,3)} + W^2_{(2,3)}\mathcal{Z}_{(2,4)} + W^0_{(3,3)}\mathcal{Z}_{(3,2)} + W^1_{(3,3)}\mathcal{Z}_{(3,3)} + W^2_{(3,3)}\mathcal{Z}_{(3,4)} + W^0_{(4,3)}\mathcal{Z}_{(4,2)} + W^1_{(4,3)}\mathcal{Z}_{(4,3)} + W^2_{(4,3)}\mathcal{Z}_{(4,4)} + W^0_{(5,3)}\mathcal{Z}_{(5,2)} + W^1_{(5,3)}\mathcal{Z}_{(5,3)} + W^2_{(5,3)}\mathcal{Z}_{(5,4)} + W^0_{(6,3)}\mathcal{Z}_{(6,2)} + W^1_{(6,3)}\mathcal{Z}_{(6,3)} + W^2_{(6,3)}\mathcal{Z}_{(6,4)}$
$\mathcal{C}_{3,2} = E^0_{(1,2)}\mathcal{Z}_{(3,1)} + E^1_{(1,2)}\mathcal{Z}_{(4,1)} + E^2_{(1,2)}\mathcal{Z}_{(5,1)} + E^0_{(2,2)}\mathcal{Z}_{(3,2)} + E^1_{(2,2)}\mathcal{Z}_{(4,2)} + E^2_{(2,2)}\mathcal{Z}_{(5,2)} + E^0_{(3,2)}\mathcal{Z}_{(3,3)} + E^1_{(3,2)}\mathcal{Z}_{(4,3)} + E^2_{(3,2)}\mathcal{Z}_{(5,3)} + E^0_{(4,2)}\mathcal{Z}_{(3,4)} + E^1_{(4,2)}\mathcal{Z}_{(4,4)} + E^2_{(4,2)}\mathcal{Z}_{(5,4)} + E^0_{(5,2)}\mathcal{Z}_{(3,5)} + E^1_{(5,2)}\mathcal{Z}_{(4,5)} + E^2_{(5,2)}\mathcal{Z}_{(5,5)}$
$\mathcal{F}_{3,2} = \sigma(\mathcal{T}_{3,2} \cdot \mathcal{C}_{3,2})$

Fig. 5. Illustration of the proposed TCJA. We give an average matrix $\mathcal{Z} \in \mathbb{R}^{6\times 5}$, and the goal of TCJA is to calculate a fusion matrix $\mathcal{F}$ integrating temporal and channel information. For instance, for a specific element in $\mathcal{F}$: $\mathcal{F}_{3,2}$, its calculation pipeline is as follows: 1) calculate $\mathcal{T}_{3,2}$ through TLA mechanism [(4)]; 2) utilize CLA mechanism [(5)] to calculate $\mathcal{C}_{3,2}$, and the calculation results are shown in the black dotted box in the figure; and 3) adopt CCF mechanism [(6)] to jointly learn temporal and channel information to obtain $\mathcal{F}_{3,2}$. In addition, we can also find that after the CCF mechanism, $\mathcal{F}_{3,2}$ integrates the information of the elements in the cross receptive field (colored areas in $\mathcal{F}$) as the anchor point, which indicates the *cross*-convolutional fusion.

where M is the number of classes, $y_{o,c}$ is a binary indicator (0 or 1) if class label c is the correct classification for observation o, and $p_{o,c}$ is the predicted probability of observation o being of class c.

To estimate the classification accuracy, we define the predicted label l_p as the index of the neuron with the highest firing rate $l_p = \max_i (1/T) \sum_{t=0}^{T-1} s_{t,i}$. Since the TCJA module simply utilizes the 1-D convolutional layer and Sigmoid function, it can be effortlessly introduced into the current network architecture as a plug-and-play module without adjusting to backpropagation.

IV. Experiments

We evaluate the classification performance of TCJA-SNN on both neuromorphic datasets (CIFAR10-DVS, N-Caltech 101, and DVS128 Gesture) and static datasets (Fashion-MNIST, CIFAR10, and CIFAR100). Note that all neuromorphic datasets are collected from the event sensor. To verify the effectiveness of the proposed method, we integrate the TCJA module into several architectures [30], [36] with competitive performance to see if the integrated architecture can generate significant improvement.

A. Dataset

1) Dataset Introduction: We have conducted experiments on both event-stream and static datasets for object classification. The summaries of the datasets involved in the experiment are listed below.

1) *CIFAR10-DVS:* The CIFAR10-DVS [39] dataset is an adapted event-driven version from the popular static dataset CIFAR10. This dataset converts 10 000 frame-based images of ten classes into event streams with the dynamic vision sensor. Since the CIFAR10-DVS dataset does not divide training and testing sets, we split the dataset into 9k training images and 1k test images and reduced the spatial resolution from 128 × 128 to 48 × 48 as [36], [40], [41], and [42].
2) *N-Caltech 101:* The N-Caltech 101 [43] dataset is also converted from the original version of Caltech 101 [44] with a slight change in object classes to avoid confusion. The N-Caltech 101 consists of 100 object classes plus one background class. We apply the 9:1 train-test split as CIFAR10-DVS.
3) *DVS128 Gesture:* The DVS128 Gesture [45] dataset is an event-stream dataset composed of 11 kinds of hand gestures from 29 subjects under three different illumination conditions, directly captured with the DVS128 camera. In this article, we employ all 11 gesture categories for classification.
4) *Fashion-MNIST:* The Fashion-MNIST [46] is a tiny but demanding static dataset designed to serve as a straight replacement for the original MNIST dataset for more complicated visual patterns. The Fashion-MNIST dataset contains 70 000 grayscale images of ten kinds of fashion products, all in a 28 × 28 size.
5) *CIFAR10/100:* The CIFAR10/100 dataset [47] consists of 60 000 32 × 32 images with three channels in 10/100 classes. There are 50 000 training images and 10 000 testing images.

2) Neuromorphic Dataset Preprocessing: We use the integrating approach to convert event stream to frame data, which is commonly used in SNNs [13], [14], [30], [48], to preprocess neuromorphic datasets. The coordinate of an event can be described as

$$E(x_i, y_i, p_i) \tag{10}$$

where x_i and y_i event's coordinate and p_i represents the event. To reduce computational consumption, we group events into T slices, where T is the network's time simulation step. A frame in the integrated frame data, denoted as $F(j)$, refers to the pixel value at position (p, x, y), represented as $F(j, p, x, y)$. It is obtained by integrating events indexed between j_l and j_r from the event stream, where j_l represents the initial timestamp for accumulation and j_r denotes the final

TABLE I

NETWORK ARCHITECTURE SETTING FOR EACH DATASET. xCy/MPy/APy DENOTES THE CONV2D/MAXPOOLING/AVGPOOLING LAYER WITH OUTPUT CHANNELS $= x$ AND KERNEL SIZE $= y$. nFC DENOTES THE FULLY CONNECTED LAYER WITH OUTPUT FEATURE $= n$, mDP IS THE SPIKING DROPOUT LAYER WITH DROPOUT RATIO m. THE VOTING LAYER IS A 1-D AVERAGE POOLING LAYER

Dataset	Network Architecture
DVS128 Gesture	128C3-LIF-MP2-128C3-LIF-MP2-128C3-LIF-MP2-128C3-LIF -MP2-128C3-LIF-MP2-0.5DP-512FC-LIF-0.5DP-100FC-LIF-Voting
CIFAR10-DVS	64C3-LIF-128C3-LIF-AP2-256C3-LIF-256C3-LIF-AP2-512C3-LIF -512C3-LIF-AP2-512C3-LIF-512C3-LIF-AP2-10FC-LIF
N-Caltech 101	64C3-LIF-MP2-128C3-LIF-MP2-256C3-LIF-MP2-256C3-LIF -MP2-512C3-LIF-0.8DP-1024FC-LIF-0.5DP-101FC-LIF
Fashion-MNIST	128C3-LIF-AP2-128C3-LIF-AP2-0.5DP-512FC-LIF-0.5DP-10FC-LIF
CIFAR10/100	64C7-LIF-64C3-LIF-64C3*2-LIF-128C3-LIF-128C3*2 -LIF-256C3-LIF-256C3*2-LIF-512C3-LIF-512C3*2-LIF-10/100FC

timestamp. The process can be described as

$$j_l = \left\lfloor \frac{N}{T} \right\rfloor \cdot j$$

$$j_r = \begin{cases} \left\lfloor \frac{N}{T} \right\rfloor \cdot (j+1), & \text{if } j < T-1 \\ N, & \text{if } j = T-1 \end{cases}$$

$$F(j,p,x,y) = \sum_{i=j_l}^{j_r-1} \mathcal{I}_{p,x,y}(p_i,x_i,y_i) \tag{11}$$

where $\lfloor \cdot \rfloor$ is the floor operation and $\mathcal{I}_{p,x,y}(p_i,x_i,y_i)$ is an indicator function and it equals 1 only when $(p,x,y) = (p_i,x_i,y_i)$. The function $F(j)$ is primarily designed to accumulate event data within a specified range. This accumulation is then segmented into frames, facilitating a format that is more conducive to the simulation of SNNs. This structured approach in framing the data not only enhances the compatibility with SNNs but also enables a more efficient analysis and processing of the event data, aligning it with the inherent temporal dynamics of SNNs.

3) Data Augmentation: To mitigate the apparent overfitting on the CIFAR10-DVS dataset, we adopt the neuromorphic data augmentation, which is also used in [36], [40], [41], and [42] for training the same dataset. We follow the same augmentation setting as [41]: we utilize horizontal Flipping and Mixup [49] in each frame, where the probability of Flipping is set to 0.5, and the Mixup interpolation factor is sampled from a beta distribution where $\alpha = 0.5, \beta = 0.5$. Then, we randomly select one augmentation among Rolling, Rotation, Cutout, and Shear, where random Rolling range is five pixels, the degree of Rotation is sampled from the uniform distribution where $\alpha = -15, \beta = 15$, the side length of Cutout is sampled from the uniform distribution where $\alpha = 1, \beta = 8$, and the shear degree is also sampled from the uniform distribution where $\alpha = -8, \beta = 8$.

B. Network Architecture

The architectures of networks corresponding to various datasets are enumerated in Table I. In the construction of each network, He et al. [50] initialization is methodically applied to both convolutional and FC layers. For the DVS128 dataset, we utilize the same network structure and hyperparameters as the [30] and add the TCJA module before the last two pooling layers. Dropout (DP) [51] rate is set to 0.5 in accordance with the original network. We added a 1-D average pooling voting layer in the last layer, which yielded a 10-D vector as the vote outcome. This is because the preprocess of DVS128 Gesture simulates a longer time step ($T = 20$), through such a voting layer the robustness of the network can be improved [13].

For the CIFAR10-DVS dataset, we adopt the VGG11-like architecture introduced in TET [36]. Due to the significant overfitting, we adopt the data augmentation as [36] and [41]. To maintain the same training settings as [36] for TCJA-TET-SNN, we use the triangle surrogate function, eliminate the last LIF layer, and replace the SMSE loss with TET loss. For TCJA-SNN, the TCJA module is added before the last two pooling layers, and for TCJA-TET-SNN, the TCJA module is included before the first pooling layer as the replacement of surrogate function and loss.

For the N-Caltech 101 dataset, we combine two architectures together and add the TCJA module before the last two pooling layers. We first reserve a pooling for each layer; then, with the network going deeper, spatial resolution is reduced with the increasing channel number. To relieve the evident overfitting, the ratio of the first dropout layer is increased to 0.8.

For the Fashion-MNIST dataset, we follow the network structure from [30]. Note that the first convolutional layer is a static encoding layer, transforming the static image into spikes.

For the CIFAR10/100 dataset, we employ the MS-ResNet architecture, as detailed in [52], to validate the effectiveness of the TCJA on deep residual neural networks. Specifically, we utilize the standard MS-ResNet-18 architecture for classifying the CIFAR datasets. The TCJA module is integrated at the bottom of each MS-ResNet block.

C. Network Implementation

We train and test our method on a workstation equipped with two Tesla P4 and two Tesla P10 GPUs. As the memory consumption, we use the Tesla P10 to train and test the CIFAR10-DVS dataset, N-Caltech 101 dataset, and DVS128 Gesture dataset and use the Tesla P4 to train and test the Fashion-MNIST and CIFAR10/100 dataset. In the various

TABLE II
HYPERPARAMETER SETTINGS OF TCJA-SNN

Hyperparameter	**CIFAR10-DVS**	**N-Caltech 101**	**DVS128**	**Fashion-MNIST**	**CIFAR10**	**CIFAR100**
Optimizer	Adam	Adam	Adam	Adam	SGD	SGD
Learning Rate	$1e-3$	$1e-3$	$1e-3$	$1e-3$	$1e-1$	$1e-1$
Batch Size	64	32	16	128	128	128
T	10	14	20	8	6/4	6/4
Automatic Mixed Precision	✗	✓	✓	✗	✗	✗
Training Epochs	1000	500	1000	1000	250	250

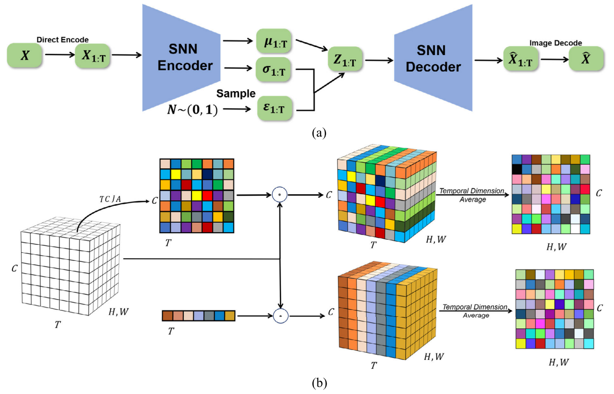

Fig. 6. Architecture of FSVAE and how TCJA applied on it. During training, input images are encoded into spiking inputs, obtaining features $\mu_{1:T}$, $\sigma_{1:T}$ after an SNN encoder. Latent encoding $z_{1:T}$ is randomly generated with a normal distribution. Finally, output images can be reconstructed through a symmetric SNN decoder. By making the best of the abundant temporal information of output spikes, our TCJA image decode performs better. (a) Workflow of FSVAE. (b) Comparison between different image decode.

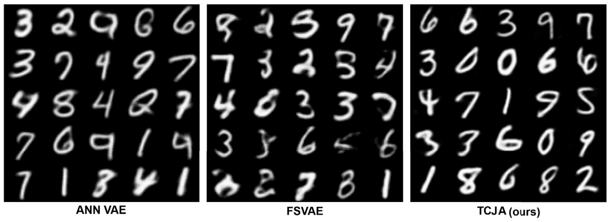

Fig. 7. Generated images of ANN VAE, FSVAE, and ours TCJA on the MNIST dataset.

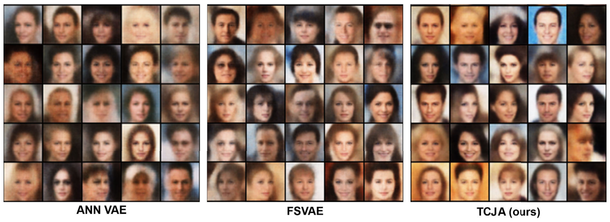

Fig. 8. Generated images of ANN VAE, FSVAE, and ours TCJA on the CelebA dataset.

datasets under consideration, the hyperparameters are detailed in Table II. The learning rate has been empirically set to 1×10^{-3} for each dataset when utilizing the Adam optimizer. Conversely, for the implementation involving ResNet with the SGD optimizer, a higher learning rate of 1×10^{-1} has been employed, as the SGD optimizer necessitates a more substantial learning rate and trains the same epochs as [30] except N-Caltech 101, which is not tested in [30]. We enable the automatic mixed precision in N-Caltech 101 and DVS128 Gesture for the excessive resolution (180 × 240 and 128 × 128). We strategically detach the reset process during backpropagation, a technique increasingly recognized for its effectiveness in optimizing SNNs. The detach operation decouples the reset operation from the computational graph. Such a detachment has been empirically validated to enhance network performance, offering a more efficient approach to managing the dynamics of SNNs [2], [21], [30]. This process ensures that the essential learning dynamics are retained while unnecessary computational complexities are minimized, thereby improving the overall efficacy of the network.

D. Comparison With Existing Classification SOTA Works

The performance of two TCJA-SNN variants is compared with some SOTA models in Tables III–V. We train and test two variants with SpikingJelly [53] package based on PyTorch [54] framework, resulting in enhanced performance across all tasks. Some studies [14], [21], [55] substitute binary spikes with floating-point (FP) spikes in whole or in part and retain the same temporal forward pipeline as SNN to obtain improved classification accuracy. Thus, we devise two variants to validate the efficiency of TCJA-SNN by utilizing the TET loss function. On CIFAR10-DVS, we obtain a 1.7% advantage over the prior method with binary spikes. On the N-Caltech 101 dataset, we achieved a classification accuracy of 82.5%, surpassing previous work by 1.6%. On DVS128, we get an accuracy of 99.0%, which is higher than TA-SNN [14] using three times fewer simulation time steps. Furthermore, by using a basic seven-layer CNN on the static dataset Fashion MNIST, our method can achieve the highest classification accuracy with the fewest simulation time steps. In the context of the CIFAR10 and CIFAR100 datasets, the implementation of TCJA demonstrates a significant improvement over the baseline models [52] that do not incorporate TCJA. Specifically, there is an enhancement of 2.08% and 3.83% in classification accuracy for CIFAR10 and CIFAR100, respectively. Additionally, our method surpasses the current SOTA models, as referenced in [56], by margins of 0.84% for CIFAR10 and 0.93% for CIFAR100. Overall, with binary spikes, TCJA-SNN simulates no more time steps while getting higher performance. Furthermore, our method can achieve higher classification accuracy by adopting the nonbinary spike technique.

TABLE III

COMPARISON BETWEEN THE PROPOSED METHODS AND EXISTING SOTA TECHNIQUES ON THREE MAINSTREAM NEUROMORPHIC DATASETS (BOLD: THE BEST)

Method	Binary Spikes	CIFAR10-DVS		N-Caltech 101		DVS128	
		T Step	Acc.	T Step	Acc.	T Step	Acc.
SLAYER [57]$^{\text{NeurIPS-2018}}$	✓	-	-	-	-	1600	93.4
HATS [58]$^{\text{CVPR-2018}}$	N/A	N/A	52.4	N/A	64.2	-	-
DART [59]$^{\text{TPAMI-2019}}$	N/A	N/A	65.8	N/A	66.8	-	-
NeuNorm [13]$^{\text{AAAI-2019}}$	✓	230-292	60.5	-	-	-	-
Rollout [48]$^{\text{Front. Neurosci-2020}}$	✓	48	66.8	-	-	240	97.2
DECOLLE [60]$^{\text{Front. Neurosci-2020}}$	✓	-	-	-	-	500	95.5
LIAF-Net [55]$^{\text{TNNLS-2021}}$	✗	10	70.4	-	-	60	97.6
tdBN [11]$^{\text{AAAI-2021}}$	✓	10	67.8	-	-	40	96.9
PLIF [30]$^{\text{ICCV-2021}}$	✓	20	74.8	-	-	20	97.6
TA-SNN [14]$^{\text{ICCV-2021}}$	✗	10	72.0	-	-	60	98.6
SEW-ResNet [21]$^{\text{NeurIPS-2021}}$	✓	16	74.4	-	-	16	97.9
Dspike [40]$^{\text{NeurIPS-2021}}$	✓	10	75.4*	-	-	-	-
SALT [61]$^{\text{Neural Netw-2021}}$	✓	20	67.1	20	55.0	-	-
TET [36]$^{\text{ICLR-2022}}$	✗	10	83.2*	-	-	-	-
DSR [42]$^{\text{CVPR-2022}}$	✓	10	77.3*	-	-	-	-
Event Transformer [62]	✗	N/A	71.2	N/A	78.9	-	-
STCA-SNN [63]$^{\text{Front. Neurosci-2023}}$	✓	10	81.6*	14	80.9	-	-
TCJA-SNN	✓	10	80.7*	14	78.5	20	**99.0**
TCJA-TET-SNN	✗	10	**83.3***	14	**82.5**	20	98.2

* With Data Augmentation.

TABLE IV

COMPARISON BETWEEN THE PROPOSED METHODS AND EXISTING SOTA TECHNIQUES ON STATIC CIFAR DATASETS (BOLD: THE BEST)

Methods	Architecture	CIFAR10		CIFAR100	
		T Step	Acc.	T Step	Acc.
ANN2SNN [64]$^{\text{AAAI-2023}}$	ResNet-18/ResNet-20	32	95.42	32	65.50
tdBN [11]$^{\text{AAAI-2021}}$	Spiking-ResNet-19	6	93.16	6	71.12
TET [36]$^{\text{ICLR-2022}}$	Spiking-ResNet-19	6	94.50	6	74.72
RecDis [65]$^{\text{CVPR-2022}}$	Spiking-ResNet-19	6	94.71	6	74.10
GLIF [56]$^{\text{NeurIPS-2022}}$	Spiking-ResNet-19	6	95.03	6	77.35
MS-ResNet [52]	MS-ResNet-18	6	93.79	6	74.45
TCJA-SNN	MS-ResNet-18	6	**95.87**	6	**78.28**
	MS-ResNet-18	4	95.60	4	77.72
ANN [52]	MS-ResNet-18	N/A	96.41	N/A	80.67

TABLE V

STATIC FASHION-MNIST ACCURACY

Method	Binary Spike	Time Step	Accuracy
ST-RSBP [66]$^{\text{NeurIPS-2019}}$	✓	400	90.1
LISNN [67]$^{\text{IJCAI-2020}}$	✓	20	92.1
PLIF [30]$^{\text{ICCV-2021}}$	✓	8	94.4
TCJA-SNN	✓	8	**94.8**
TCJA-TET-SNN	✗	8	94.6

E. Comparison With Existing Image Generation Works

In this experiment, we build a fully spiking variation autoencoder (FSVAE) for image generation with TCJA. Moreover, we replace the original image decoding way with our novel method by calculating the average output on the temporal dimension after the TCJA block. The workflow chart of this FSVAE with TCJA applied on image decoding is shown in Fig. 6. We apply log-likelihood evidence lower bound (ELBO) as the loss function

$$\text{ELBO} = \mathbb{E}_{q(z_{1:T}|\boldsymbol{x}_{1:T})}\big[\log p(\boldsymbol{x}_{1:T}|z_{1:T})\big] - \text{KL}\big[q(z_{1:T}|\boldsymbol{x}_{1:T})\big\|\, p(z_{1:T})\big] \tag{12}$$

where the first term is the reconstruction loss between the original input and the reconstructed one, which is the mean square error (MSE) in this model. The second term is the Kullback–Leibler (KL) divergence, representing the closeness of prior and posterior.

We employ the AdamW optimizer [68] for image-generating tasks, which trains 300 epochs at 0.001 learning rate and 0.001 weight decay. The batch size is set to 256. Moreover, the time step is set to 16. The performance of the TCJA image decode is compared with some SOTA models in Table VI. Because this method can make full use of the powerful temporal information, the inception score (IS) shows SOTA results compared to the original FSVAE [69] and the same structure ANN. Our TCJA image decoding outperforms better on all metrics for CIFAR10 datasets. Moreover, results

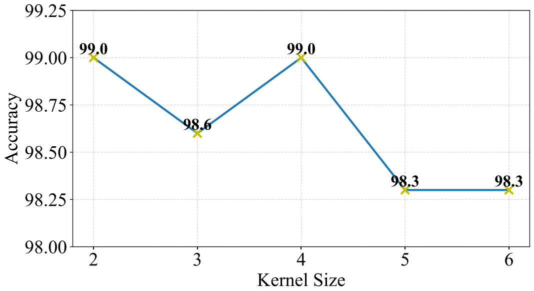

Fig. 9. Variation in test accuracy on DVS128 Gesture dataset as kernel size increases.

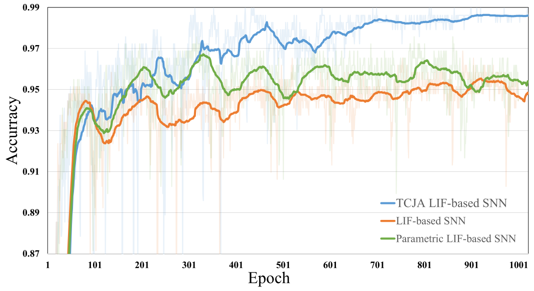

Fig. 10. Convergence of compared SNN methods on DVS128 Gesture.

on CelebA and MNIST are further visualized in Figs. 7 and 8, which demonstrates that our generated images are visually better than the previous method [69]. In addition to the primary evaluations, our model was also compared with other methods based on the Spiking VAEs, such as image decoding based on temporal attention (TAID) [70] and efficient spiking VAE (ESVAE) [71], which have been proposed recently. Despite these methods being specifically designed for low-level image reconstruction tasks, our approach remains competitive. It demonstrates robust performance, reflecting its adaptability and effectiveness in comparison to these specialized models.

TABLE VI

Comparison With Original SNN's Work on Image Generation for Each Dataset

Dataset	Method	IS ↑	FID ↓	FAD ↓
MNIST	ANN [69]$^{\text{AAAI-2022}}$	5.95	112.5	17.09
	FSVAE [69]$^{\text{AAAI-2022}}$	6.21	**97.06**	35.54
	ESVAE [71]	5.602	117.8	**10.99**
	This work	**6.45**	100.8	19.39
Fashion-MNIST	ANN [69]$^{\text{AAAI-2022}}$	4.25	123.7	18.08
	FSVAE [69]$^{\text{AAAI-2022}}$	4.55	**90.12**	15.75
	ESVAE [71]	**6.23**	125.3	**11.13**
	This work	5.61	93.41	12.46
CIFAR10	ANN [69]$^{\text{AAAI-2022}}$	2.59	229.6	196.9
	FSVAE [69]$^{\text{AAAI-2022}}$	2.94	175.5	133.9
	TAID [70]$^{\text{ICLR-2023}}$	3.53	171.1	120.5
	ESVAE [71]	**3.76**	**127.0**	**14.7**
	This work	3.73	170.1	100.4
CelebA	ANN [69]$^{\text{AAAI-2022}}$	3.23	92.53	156.9
	FSVAE [69]$^{\text{AAAI-2022}}$	3.69	101.6	112.9
	TAID [70]$^{\text{ICLR-2023}}$	**4.31**	99.5	105.3
	ESVAE [71]	3.87	**85.3**	**51.9**
	This work	3.84	100.6	119.9

TABLE VII

Accuracy of Different Blocks

Block	CIFAR10-DVS	N-Caltech 101	DVS128
TLA	79.7	78.3	97.9
CLA	80.5	78.4	98.6
TCJA	80.7	78.5	99.0

TABLE VIII

Test Accuracy on Three Datasets With Different CCF Operations

Type	**CIFAR10-DVS**	**N-Caltech 101**	**DVS128**
Addition	80.7	78.0	98.2
Multiplication	80.7	78.5	99.0

F. Ablation Study

To thoroughly examine the impact of the TLA and CLA modules, we conducted a series of ablation studies. The results, as presented in Table VII, indicate that the CLA module plays a crucial role in enhancing performance. This can be attributed to the fact that, in most SNN designs, the number of simulation time steps is significantly fewer than the number of channels. Consequently, the CLA module can extract additional relevant features compared to the TLA module. Furthermore, it is worth noting that the TCJA module consistently outperformed other models across all tested datasets. This outcome underscores the effectiveness of the CCF layer incorporated within the TCJA module, further reinforcing its potential for achieving superior performance.

G. Discussion

1) Kernel Size: We initially investigated the kernel size in the TCJA module. Intuitively, when the size of the kernel rises, the receptive field of the local attention mechanism will also expand, which may aid in enhancing the performance of TCJA-SNN. However, the experimental results in Fig. 9 overturn this conjecture. As the size of the kernel rises, the performance of the model waves. When the kernel size is more than 4, there is a perceptible decrease in overall performance. One reasonable explanation is that a frame mainly correlates with its nearby frames, and an excessively large receptive field may lead to undesired noise.

2) Multiplication Versus Addition: To verify the effectiveness of our proposed CCF mechanism, we devise a variant method that substitutes addition for multiplication of $\mathcal{T}_{i,j}$ and $\mathcal{C}_{i,j}$ in the (6). The results are shown in Table VIII.

As we observed, the addition operation achieves good performance, nevertheless, when compared to the multiplication operation, the final calculation result lacks the cross term, which prevents a robust construction of the correlation between frames; therefore, it is inferior.

3) Convergence: We also empirically demonstrate the convergence of our proposed method, as shown in Fig. 10. Specifically, Fig. 10 illustrates the performance trend of vanilla LIF-SNN, Parametric LIF-based SNN [30] without TCJA block and our proposed TCJA-SNN for 1000 epochs. As the

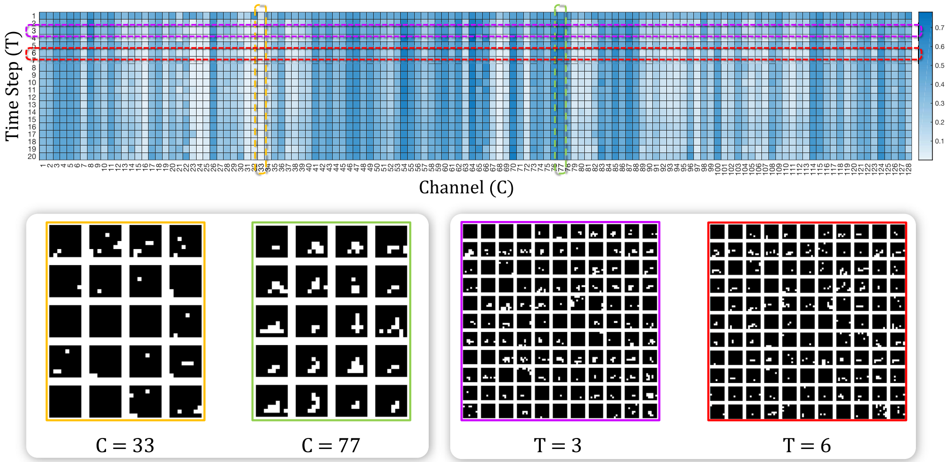

Fig. 11. Attention distribution between time step and channel. The top row is the weight from the first TCJA module in TCJA-SNN working with the DVS128 Gesture dataset. We select sparse and dense attention frames both temporal-wise ($T = 3, 6$) and channel-wise ($C = 33, 77$) in the bottom row.

training epoch increases, the performance trend of our proposed method becomes more stable and converges to a higher level. Moreover, the TCJA-SNN can achieve the SOTA performance when only training about 260 epochs, which demonstrates the efficacy of the proposed TCJA.

4) Complexity Analysis: Finally, we try to analyze the time and space complexity of TLA and CLA. For TLA, it can be concluded from the (4) that the time complexity of obtaining each element of $\mathcal{T}$ is $\mathcal{O}(CK)$ (K: Kernel size). Consequently, the time complexity of the whole TLA mechanism is $\mathcal{O}(TC^2K)$. Moreover, the space complexity is composed of the parameters and the memory occupied by variables. On the one hand, for the parameters, C-channel 1-D convolution is performed on each row of $\mathcal{Z}$, so the total amount of parameters required is $C * C * K$, on the other hand, for variables, in the whole process, we only need to maintain a matrix of dimension $C \times T$. In conclusion, the space complexity is $\mathcal{O}(C^2K + CT)$. Similarly, for CLA, the time complexity is $\mathcal{O}(T^2CK)$ and the space complexity is $\mathcal{O}(T^2K + CT)$.

5) Theoretical Analysis on Receptive Field: The global receptive field stands as a fundamental feature of our innovative TCJA approach. Our approach surpasses the limitations of dense layers by utilizing fewer parameters to achieve a comprehensive global receptive field. Moreover, it surpasses the capabilities of employing 2-D convolutions alone by effectively obtaining a larger receptive field. To provide a deeper comprehension of the salient aspects of our proposed method, we present the following theoretical analysis concerning the specific region where the network perceives and processes information throughout the training phase, known as the receptive field.

Lemma 1 (CCS of 1-D Convolution): For an input feature map $I \in \mathbb{R}^{C \times T}$, if the size of the 1-D convolution kernel is defined as k, then its cross-correlation scope (CCS) can be described as $P \in \mathbb{R}^{k \times T}$, where the T involves the information along the second dimension of I.

Lemma 2 (CCS of Two Orthogonal 1-D Convolution): For an input feature map $I \in \mathbb{R}^{C \times T}$, the dot multiplication of two orthogonal 1-D convolutions performed on I is equivalent to expanding the CCS into a cross shape, that is, its CCS can be described by two cross-overlaid matrices $P * Q$ (see, e.g., the colored area of $\mathcal{F}$ in Fig. 5), where $P \in \mathbb{R}^{k_1 \times T}$, $Q \in \mathbb{R}^{k_2 \times C}$, and k_1 and k_2 are the sizes of the two convolution kernels, respectively.

Referring to (6), Lemmas 1 and 2, we can obtain the following corollary.

Corollary 1: Based on the broad CCS obtained by TCJA, there exists information flow among $\mathcal{T}$ and $\mathcal{C}$, cooperatively considering the temporal and channel correlation, which is also clued in (6).

Recalling (4) and (5), through two 1-D convolutions along different dimensions, we construct two CCS in a vertical relationship, which are stored in $\mathcal{T}$ and $\mathcal{C}$. In particular, TCJA is to construct a CCS, which can perceive a larger area while realizing feature interaction in different directions. This cross-receptive field can abolish the limitations caused by the monotonic dimension, thus bringing performance improvements to the network. As a corollary, when the kernel sizes of the two dimensions are the same, we can obtain a square cross-shaped receptive field similar to that of conventional 2-D convolution, which is an effective scheme in 2-D convolution.

6) Attention Visualization: To make the attention mechanism easier to understand, we finally visualize the output of the first TCJA module in TCJA-SNN working with the DVS128 Gesture dataset, which can be seen in Fig. 11. Changes in attention weights are primarily accumulated among channels, verifying further the substantial role performed by the CLA in the TCJA module. To embody the attention weights, we extract some temporal-wise and channel-wise frames. The difference in firing patterns in the channel dimension is more significant than that in the temporal dimension.

7) Energy Consumption Analysis: Compared to the ANNs, SNNs consume less energy due to their sparser firing and poorer processing accuracy. Owing to the binary spikes, each operation in SNNs consists of a single FP addition. In ANNs, on the other hand, each operation computes a dot product as a multiply-accumulate (MAC) calculation consisting of one FP multiplication and one FP addition. Consequently, SNNs

TABLE IX

SPIKING RATE, FLOPS, AND SNN SINGLE OPERATION ENERGY COST OF EACH LAYER IN THE NETWORK FOR CLASSIFYING THE DVS128 DATASET, WHERE Conv$x$ DENOTES xTH 2-D CONVOLUTIONAL LAYER, A TTy DENOTES yTH TCJA MODULE, AND FCz REPRESENTS zTH FC LAYER OF THE NETWORK. NOTICE THAT THE FIRST 2-D CONVOLUTIONAL LAYER IS AN ENCODING LAYER TO TRANSFORM THE ANALOG INPUT INTO SPIKES

Layer	Encoding	Conv1	Conv2	Conv3	Conv4	Att1	Att2	FC1	FC2
Spiking Rate	-	2.27%	2.50%	4.74%	8.09%	-	-	27.80%	9.01%
FLOPS	37.74M	603.98M	151.00M	37.74M	9.44M	1.31M	1.31M	1.05M	0.06M
Energy Cost in SNN (Single Operation)	4.6pJ	0.9pJ	0.9pJ	0.9pJ	0.9pJ	4.6pJ	4.6pJ	0.9pJ	0.9pJ

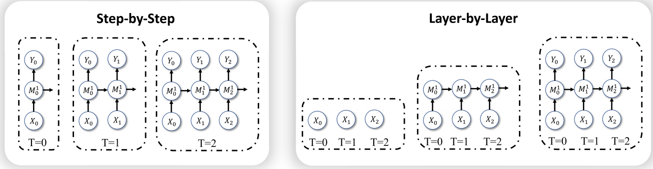

Fig. 12. Step-by-step propagation pattern and layer-by-layer propagation pattern. X_m denotes the input in the mth time step and M_j^i represents the ith middle layer in the jth time step. Y_n shows the output in the nth time step.

use less energy than ANNs in the same network design. This discrepancy can also be validated in 45-nm CMOS technology, where the energy cost of each SNN operation is 5.1× lower than that of each 32-bit ANN MAC operation (0.9 versus 4.6 pJ) [72], allowing us to examine the energy consumption of each network architecture.

We assess the energy consumption in the network for classifying the DVS128 dataset with both ANN and SNN. first, we assess the spiking rate, and floating-point operations per second (FLOPS) of each layer, the result is shown in Table IX. For ANNs, we can calculate the energy consumption by FLOPS × MAC energy cost; for SNNs, the energy cost should be quantified by FLOPS × SNN operation energy cost × spiking rate. The final power consumption calculation results are 1.90×10^{-3} J (TCJA-SNN) and 10.00×10^{-3} J (ANN), where our TCJA-SNN costs 5.26× lower energy consumption compared to its ANN version.

8) Propagation Pattern: The forward propagation process in SNNs encompasses both temporal and spatial domains. Intuitively, the computation graph for SNN forward propagation can be conceptualized as a sequential, step-by-step pattern. This pattern is depicted in Fig. 12. In this context, "step-by-step" refers to the process wherein the network's output at the initial time step is evaluated, along with updates to the hidden states of the spiking neurons. Following this, subsequent time steps are evaluated similarly, maintaining this sequential progression. Besides, the layer-by-layer pattern is also extensively used, which entails performing a spatial forward propagation procedure in which we calculate the output of the first layer at all time steps as the input of the second layer, then retrieve the output of the last layer at all time steps. Unlike parallel computing environments like GPU, where the layer-by-layer pattern is preferred, neuromorphic devices operate more like a step-by-step pattern. It can be proved that the output of the network in the two patterns is mathematically equivalent.

Although we train the network in TCJA using a layer-by-layer pattern, the convolutional structure of the network still benefits it when applied in a step-by-step manner. In terms of temporal attention, the TLA module only needs to compute a few adjacent time steps because of its convolutional nature, in contrast to mechanisms like SE that require full information at all time steps. Previous discussion reveals that TCJA reaches its peak performance when the convolutional kernel size is set to 2. Under this circumstance, TLA only needs to buffer one time step while propagating step-by-step.

V. CONCLUSION

In this article, we propose the TCJA mechanism, which innovatively recalibrates temporal and channel information in SNNs. Specifically, instead of utilizing a generic fully connected network, we use 1-D convolution to build the correlation between frames, reducing the computation and improving model performance. Moreover, we propose a CCF mechanism to realize joint feature interaction between temporal and channel information. Experiments verify the effectiveness of our method with SOTA results on four datasets, that is, CIFAR10-DVS (83.3%), N-Caltech101 (82.5%), DVS128 (99.0%), Fashion-MNIST (94.8%), CIFAR10 (95.9%), and CIFAR100 (78.3%). In addition to its outstanding performance in classification tasks, TCJA-SNN also exhibits a competitive performance in image generation tasks. To the best of our knowledge, this study represents the pioneering application of the SNN-attention mechanism to both high-level classification and low-level generation tasks. Remarkably, our approach has achieved SOTA performance in both domains, thus making a significant advancement in the field. However, the insertion of TCJA still resulted in a relatively sizable boost in the number of parameters. In future work, we believe that this method can easily be integrated into the neuromorphic chip for the hardware-friendly 1-D convolution operation and the binary spiking network structure.

ACKNOWLEDGMENT

The authors would like to express their gratitude to Xuerui Qiu for providing insightful comments on their work and for his efforts in exploring the application of TCJA for generation tasks.

REFERENCES

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

[2] E. Stromatias, D. Neil, M. Pfeiffer, F. Galluppi, S. B. Furber, and S.-C. Liu, "Robustness of spiking deep belief networks to noise and reduced bit precision of neuro-inspired hardware platforms," *Frontiers Neurosci.*, vol. 9, p. 222, Jul. 2015.

[3] M. Zhang et al., "Rectified linear postsynaptic potential function for backpropagation in deep spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 5, pp. 1947–1958, May 2022.

[4] S. M. Bohte, J. N. Kok, and J. A. L. Poutré, "SpikeProp: Backpropagation for networks of spiking neurons," in *Proc. ESANN*, Bruges, Belgium, 2000, pp. 419–424.

[5] J. Wu et al., "Progressive tandem learning for pattern recognition with deep spiking neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7824–7840, Nov. 2022.

[6] M. Dampfhoffer, T. Mesquida, A. Valentian, and L. Anghel, "Backpropagation-based learning techniques for deep spiking neural networks: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 7, 2023, doi: 10.1109/TNNLS.2023.3263008.

[7] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers Neurosci.*, vol. 10, p. 508, Nov. 2016.

[8] X. Luo, H. Qu, Y. Wang, Z. Yi, J. Zhang, and M. Zhang, "Supervised learning in multilayer spiking neural networks with spike temporal error backpropagation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 10141–10153, Dec. 2023.

[9] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *Int. J. Comput. Vis.*, vol. 113, no. 1, pp. 54–66, May 2015.

[10] T. Zhang, Q. Wang, and B. Xu, "Self-lateral propagation elevates synaptic modifications in spiking neural networks for the efficient spatial and temporal classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 30, 2023, doi: 10.1109/TNNLS.2023.3286458.

[11] H. Zheng, Y. Wu, L. Deng, Y. Hu, and G. Li, "Going deeper with directly-trained larger spiking neural networks," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, pp. 11062–11070.

[12] Y. Hu, H. Tang, and G. Pan, "Spiking deep residual networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 8, pp. 5200–5205, Aug. 2023.

[13] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi, "Direct training for spiking neural networks: Faster, larger, better," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 1311–1318.

[14] M. Yao et al., "Temporal-wise attention spiking neural networks for event streams classification," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10201–10210.

[15] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 2011–2023, Aug. 2020.

[16] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.

[17] M. Bernert and B. Yvert, "An attention-based spiking neural network for unsupervised spike-sorting," *Int. J. Neural Syst.*, vol. 29, no. 8, Oct. 2019, Art. no. 1850059.

[18] Z. Wang, Y. Zhang, S. Lian, X. Cui, R. Yan, and H. Tang, "Toward high-accuracy and low-latency spiking neural networks with two-stage optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Dec. 15, 2024, doi: 10.1109/TNNLS.2023.3337176.

[19] J. Wu, Y. Chua, M. Zhang, G. Li, H. Li, and K. C. Tan, "A tandem learning rule for effective training and rapid inference of deep spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 1, pp. 446–460, Jan. 2023.

[20] Q. Yang, M. Zhang, J. Wu, K. C. Tan, and H. Li, "LC-TTFS: Towards lossless network conversion for spiking neural networks with TTFS coding," *IEEE Trans. Cognit. Develop. Syst.*, early access, Nov. 20, 2024, doi: 10.1109/TCDS.2023.3334010.

[21] W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, "Deep residual learning in spiking neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 21056–21069.

[22] C. Jin, R.-J. Zhu, X. Wu, and L.-J. Deng, "SIT: A bionic and non-linear neuron for spiking neural network," 2022, *arXiv:2203.16117*.

[23] 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 Process. Mag.*, vol. 36, no. 6, pp. 51–63, Nov. 2019.

[24] 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.

[25] N. Rathi and K. Roy, "DIET-SNN: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 6, pp. 3174–3182, Jun. 2023.

[26] X. Xie, Y. Chua, G. Liu, M. Zhang, G. Luo, and H. Tang, "Event-driven spiking learning algorithm using aggregated labels," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Aug. 31, 2024, doi: 10.1109/TNNLS.2023.3306749.

[27] X. Qiu, R.-J. Zhu, Y. Chou, Z. Wang, L.-J. Deng, and G. Li, "Gated attention coding for training high-performance and efficient spiking neural networks," 2023, *arXiv:2308.06582*.

[28] X.-R. Qiu et al., "VTSNN: A virtual temporal spiking neural network," *Frontiers Neurosci.*, vol. 17, May 2023, Art. no. 1091097.

[29] J. K. Eshraghian et al., "Training spiking neural networks using lessons from deep learning," 2021, *arXiv:2109.12894*.

[30] 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 *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 2661–2671.

[31] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 31, 2018, pp. 1–11.

[32] V. Mante, V. Bonin, and M. Carandini, "Functional mechanisms shaping lateral geniculate responses to artificial and natural stimuli," *Neuron*, vol. 58, no. 4, pp. 625–638, May 2008.

[33] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003.

[34] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge, U.K.: Cambridge Univ. Press, 2002.

[35] L. Lapicque, "Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization," *J. Physiol Pathol Générale*, vol. 9, no. 9, pp. 620–635, 1907.

[36] S. Deng, Y. Li, S. Zhang, and S. Gu, "Temporal efficient training of spiking neural network via gradient re-weighting," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–17.

[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[38] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–9.

[39] 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.

[40] Y. Li, Y. Guo, S. Zhang, S. Deng, Y. Hai, and S. Gu, "Differentiable spike: Rethinking gradient-descent for training spiking neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 34, 2021, pp. 23426–23439.

[41] Y. Li, Y. Kim, H. Park, T. Geller, and P. Panda, "Neuromorphic data augmentation for training spiking neural networks," 2022, *arXiv:2203.06145*.

[42] Q. Meng, M. Xiao, S. Yan, Y. Wang, Z. Lin, and Z.-Q. Luo, "Training high-performance low-latency spiking neural networks by differentiation on spike representation," 2022, *arXiv:2205.00459*.

[43] 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.

[44] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop*, 2004, p. 178.

[45] 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. 7388–7397.

[46] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[47] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

[48] A. Kugele, T. Pfeil, M. Pfeiffer, and E. Chicca, "Efficient processing of spatio-temporal data streams with spiking neural networks," *Frontiers Neurosci.*, vol. 14, p. 439, May 2020.

[49] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–13.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1026–1034.

[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Sep. 2014.

[52] Y. Hu, Y. Wu, L. Deng, and G. Li, "Advancing deep residual learning by solving the crux of degradation in spiking neural networks," 2021, *arXiv:2201.07209*.

[53] W. Fang et al. (2020). *Spikingjelly*. Accessed: May 4, 2022. [Online]. Available: https://github.com/fangwei123456/spikingjelly

[54] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019, pp. 1–12.

[55] Z. Wu, H. Zhang, Y. Lin, G. Li, M. Wang, and Y. Tang, "LIAF-net: Leaky integrate and analog fire network for lightweight and efficient spatiotemporal information processing," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 11, pp. 6249–6262, Nov. 2022.

[56] X. Yao, F. Li, Z. Mo, and J. Cheng, "GLIF: A unified gated leaky integrate-and-fire neuron for spiking neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2022, pp. 1–12.

[57] S. B. Shrestha and G. Orchard, "SLAYER: Spike layer error reassignment in time," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 31, 2018, pp. 1–10.

[58] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "HATS: Histograms of averaged time surfaces for robust event-based object classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1731–1740.

[59] B. Ramesh, H. Yang, G. Orchard, N. A. L. Thi, S. Zhang, and C. Xiang, "DART: Distribution aware retinal transform for event-based cameras," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 11, pp. 2767–2780, Nov. 2019.

[60] J. Kaiser, H. Mostafa, and E. Neftci, "Synaptic plasticity dynamics for deep continuous local learning (DECOLLE)," *Frontiers Neurosci.*, vol. 14, p. 424, May 2020.

[61] Y. Kim and P. Panda, "Optimizing deeper spiking neural networks for dynamic vision sensing," *Neural Netw.*, vol. 144, pp. 686–698, Dec. 2021.

[62] Z. Li, M. Salman Asif, and Z. Ma, "Event transformer," 2022, *arXiv:2204.05172*.

[63] X. Wu et al., "STCA-SNN: Self-attention-based temporal-channel joint attention for spiking neural networks," *Frontiers Neurosci.*, vol. 17, Nov. 2023, Art. no. 1261543.

[64] Z. Hao, T. Bu, J. Ding, T. Huang, and Z. Yu, "Reducing ANN-SNN conversion error through residual membrane potential," in *Proc. AAAI Conf. Artif. Intell.*, Jun. 2023, pp. 11–21.

[65] Y. Guo et al., "RecDis-SNN: Rectifying membrane potential distribution for directly training spiking neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 326–335.

[66] W. Zhang and P. Li, "Spike-train level backpropagation for training deep recurrent spiking neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 32, 2019, pp. 1–12.

[67] X. Cheng, Y. Hao, J. Xu, and B. Xu, "LISNN: Improving spiking neural networks with lateral interactions for robust object recognition," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2020, pp. 1519–1525.

[68] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*.

[69] H. Kamata, Y. Mukuta, and T. Harada, "Fully spiking variational autoencoder," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 6, pp. 7059–7067.

[70] X. Qiu, Z. Luan, Z. Wang, and R. Zhu, "When spiking neural networks meet temporal attention image decoding and adaptive spiking neuron," in *Proc. ICLR*, K. Maughan, R. Liu, and T. F. Burns, Eds. Kigali, Rwanda, 2023, pp. 1–5. [Online]. Available: https://openreview.net/pdf?id=MuOFB0LQKcy

[71] Q. Zhan, X. Xie, G. Liu, and M. Zhang, "ESVAE: An efficient spiking variational autoencoder with reparameterizable Poisson spiking sampling," 2023, *arXiv:2310.14839*.

[72] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp. 10–14.