Converting Artificial Neural Networks to Ultra-Low-Latency Spiking Neural Networks for Action Recognition

Hong You, Xian Zhong, *Member, IEEE*, Wenxuan Liu, Qi Wei, Wenxin Huang, *Member, IEEE*, Zhaofei Yu, *Member, IEEE*, Tiejun Huang, *Senior Member, IEEE*

***Abstract*—Spiking neural networks (SNNs) have garnered significant attention for their potential in ultra-low-power event-driven neuromorphic hardware implementations. One effective strategy for obtaining SNNs involves the conversion of artificial neural networks (ANNs) to SNNs. However, existing research on ANN-SNN conversion has predominantly focused on image classification task, leaving the exploration of action recognition task limited. In this paper, we investigate the performance degradation of SNNs on action recognition task. Through in-depth analysis, we propose a framework called Scalable Dual Threshold Mapping (SDM) that effectively overcomes three types of conversion errors. By effectively mitigating these conversion errors, we are able to reduce the time required for the spike firing rate of SNNs to align with the activation values of ANNs. Consequently, our method enables the generation of accurate and ultra-low-latency SNNs. We conduct extensive evaluations on multiple action recognition datasets, including UCF-101 and HMDB-51. Through rigorous experiments and analysis, we demonstrate the effectiveness of our approach. Notably, SDM achieves a remarkable Top-1 accuracy of 92.94% on UCF-101 while requiring ultra-low latency (4 time-steps), highlighting its high performance with reduced computational requirements. The code for our SDM framework will be available at https://github.com/githuberyh/SDM.**

***Index Terms*—Action Recognition, Spiking Neural Networks, ANN-SNN Conversion, Ultra-Low-Latency, Dual Threshold.**

I. Introduction

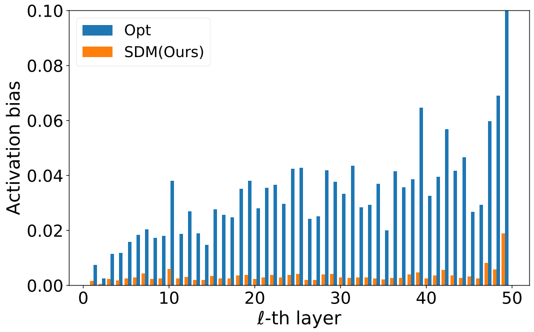


Fig. 1. Comparison of our proposed SDM with Opt [11], a strong baseline for ANN-SNN conversion, using a 50-layer ResNet3D architecture. Our evaluation is performed on UCF-101, and we measure the expectation of activation bias in each layer. Notably, when the inference time-steps are set to 16, our method demonstrates significantly lower activation bias.

DEEP artificial neural networks (ANNs) have demonstrated remarkable achievements across various domains of artificial intelligence, including computer vision [1]–[3] and natural language processing [4]. Nevertheless, the considerable power consumption associated with ANNs significantly limits their scalability in edge computing and power-constrained applications. To mitigate the computational power requirements, researchers have turned to spiking neural networks (SNNs) [5] as a potential solution. SNNs, inspired by energy-efficient neurons that process and transmit information using discrete spikes, are commonly referred to as third-generation ANNs [6]. Due to their event-driven computation and sparse-asynchronous spiking nature [7], SNNs have demonstrated superior energy efficiency compared to ANNs when implemented on neuromorphic chips, e.g., TrueNorth [8], Loihi [9], and Tianjic [10]. However, effectively training high-performance SNNs remains challenging due to their intricate temporal dynamics and discrete spike transmission patterns.

To achieve high-performance SNNs, two main strategies are commonly employed: direct training of SNNs and ANN-SNN conversion. In the realm of direct training of SNN [12]–[15], a surrogate gradient is leveraged to address the non-differentiable property of the binary activation function during SNN training. However, this strategy often demands a great deal of graphics processing unit (GPU) memory and suffers from unsatisfactory performance on large-scale datasets. In

Manuscript received August 7, 2023; revised January 16, 2024; accepted March 5, 2024. This work was supported in part by the National Natural Science Foundation of China under Grants 62271361 and 62301213. The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University. (*Corresponding authors: Xian Zhong and Zhaofei Yu.*)

Hong You and Wenxuan Liu are with the Hubei Key Lab of Transportation Internet of Things, School of Computer Science and Artificial Intelligence, Wuhan University of Technology, Wuhan, 430070, China (e-mail: hyou@whut.edu.cn; lwxfight@whut.edu.cn).

Xian Zhong is with the Hubei Key Lab of Transportation Internet of Things, School of Computer Science and Artificial Intelligence, Wuhan University of Technology, Wuhan, 430070, China and also with the Rapid-Rich Object Search Lab, School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore (e-mail: zhongx@whut.edu.cn).

Qi Wei is with the School of Computer Science and Engineering, Nanyang Technological University, 639798, Singapore. (e-mail: 1998v7@gmail.com).

Wenxin Huang is with the School of Computer Science and Information Engineering, Hubei University, Wuhan, 430062, China (e-mail: wenxinhuang_wh@163.com).

Zhaofei Yu is with the Institute for Artifical Intelligence, Peking University, Beijing, 100091, China (e-mail: yuzf12@pku.edu.cn)

Tiejun Huang is with the School of Computer Science and the Institute for Artifical Intelligence, Peking University, Beijing, 100091, China (e-mail: tjhuang@pku.edu.cn)

contrast, ANN-SNN conversion [16]–[18] involves training an initial ANN and subsequently converting it into an SNN by replacing rectified linear units (ReLU) with integrate-and-fire (IF) neurons. Existing ANN-SNN conversion methods have demonstrated significant success in image classification, specifically in achieving comparable performance between the converted SNN and the original ANN at low time-steps. However, when it comes to action recognition task involving more intricate video inputs, deeper network models, and increased network dimensions, the challenge of aligning SNN spike firing rate with ANN activation values intensifies. Moreover, employing an ANN as a feature extraction network for action recognition results in a substantial computational overhead. These factors have motivated us to investigate an ANN-SNN conversion method for training SNNs that are well-suited for action recognition.

We experimentally investigate the challenges underlying ANN-SNN conversion in action recognition task from the perspective of *activation bias*, which represents the disparity between the activation value in ANNs and the spiking firing rate in SNNs. To be specific, we conduct an experimental analysis and employ quantitative visualization techniques to assess the expectation of activation bias in each layer of the network. The results, presented in Fig. 1, clearly indicate that the existing ANN-SNN conversion method incurs a large activation bias in each layer leading to unsatisfying performance and thus cannot be directly applied to this particular task. In practice, the activation bias can be categorized into three types of conversion errors [19], which encourages us to solve all types of conversion errors in a united framework.

This paper introduces a novel and efficient training framework called Scalable Dual threshold Mapping (SDM) for ANN-SNN conversion. To mitigate both clipping error and unevenness error, we first propose the Dual Threshold Mapping (DTM) module. This module incorporates two thresholds that effectively reduce the excessive release of positive membrane potential, thereby alleviating the unevenness error. Then, these thresholds are obtained through a mapping function applied to a pre-trained ANN. This mapping function ensures a transition between the activation of ANNs and the firing rate of SNNs, effectively eliminating any potential clipping error. Furthermore, we present the Scalable Threshold in Burst neurons (STB) module, which aims to mitigate the quantization error while simultaneously addressing the reintroduction of clipping error. By effectively reducing the three types of conversion errors, the firing rate of SNNs requires shorter time-steps to align with the activation value of ANNs, thereby reducing the inference latency of SNNs. To the best of our knowledge, this is the first deep SNN for action recognition that achieves comparable results to those of ANNs on non-trivial datasets.

Our contributions can be succinctly summarized fourfold:

- As far as we know, we are the first to employ ANN-SNN conversion method to train deep SNN for action recognition task. We propose an ANN-SNN conversion framework called Scalable Dual threshold Mapping (SDM), which simultaneously reduces three types of conversion errors.
- Dual Threshold Mapping (DTM) module adaptively learns the dual threshold for spiking neurons, effectively resolving the output discrepancies arising from the uneven time distribution of heterogeneous input spikes.
- Scalable Threshold in Burst neurons (STB) module reduces the quantization resolution due to the discrete nature of spikes during the numerical mapping of ANN's activation value and SNN's spike firing rate, thereby minimizing the difference in ANN and SNN output.
- To evaluate the efficacy of our proposed method, we conduct experiments on two widely recognized benchmark datasets for action recognition, UCF-101 and HMDB-51, for performance assessment. The experimental results validate the effectiveness of the proposed method, which can achieve competitive accuracy with ultra-low latency.

II. Related Work

A. ANN-SNN Conversion

The exploration of ANN-SNN conversion was initiated by [20]. Subsequently, Diehl et al. [21] successfully converted a three-layer convolutional neural network (CNN) structure, devoid of bias, to an SNN using weight normalization techniques. Following [21], [22] further refined the weight normalization approach by the P^{th} maximum output and integrating the batch normalization (BN) layer with the convolution layer. Sengupta et al. [23] introduced Spike-Norm as achieving threshold balancing from an SNN perspective, recognizing the equivalence of weight normalization to threshold balancing. Kim et al. [24] proposed Spiking-YOLO, in an attempt to apply channel-wise weight normalization to object detection. Wang et al. [16] proposed a weight-threshold balance conversion method to obtain SNNs with binary weights for object recognition. Han et al. [25] and Zhang et al. [26] recommended the utilization of a soft reset mechanism to reduce errors caused by resetting. To minimize conversion errors at low inference latency, several studies [17], [19], [27] derived the sources of conversion errors. Deng et al. [27] and Li et al. [19] addressed conversion errors through shift weighting, bias, membrane potential, and parameter calibration, respectively. Bu et al. [17] introduced a quantization clip-floor-shift activation function to replace ReLU, resulting in high-performance converted SNNs with ultra-low latency (4 time-steps). Liu et al. [18] proposed a temporal separation scheme that divides neural calculations into an accumulation phase and a generation phase, effectively mitigating the discrepancy between the activation values of the source ANN and the generated spike train of the target SNN. Furthermore, Li et al. [28] and Park et al. [29] demonstrated that the introduction of burst mechanisms allows multiple spikes to be transmitted in a single step. Lastly, Wang et al. [30] introduced signed neurons with a memory function to eliminate errors arising from asynchronous spikes.

B. Video Action Recognition

Researchers have extended 2D CNNs to 3D structures to effectively capture both spatial and temporal context information in videos crucial for action recognition. The pioneering work of C3D [31] introduced deep 3D CNNs for learning spatio-temporal features. However, the performance of C3D on standard benchmarks was found to be unsatisfactory.

I3D [32] inflated the ImageNet pre-trained 2D kernels to 3D kernels for capturing spatio-temporal information. To better represent motion patterns, I3D was intended as a dual-stream network that extracts both optical streams and RGB features. Recognizing that most conventional 3D networks only exploit local correlation within input channels, STCNet [33] inserted the STC block into 3D ResNet architecture, which aimed to model the spatial-temporal correlation between different channels of 3D CNN, thereby improving the performance of 3D networks. Another notable approach, SlowFast [34] divided the network into slow and fast paths based on the video sampling rate, where the slow path was responsible for capturing actual spatial semantics and the fast path focused on capturing motion information at an acceptable temporal resolution. By incorporating these two paths, SlowFast achieved improved performance in action recognition task. In addition, some models [35], [36] based on Vision Transformer (ViT) have also achieved satisfactory results.

III. Preliminaries

A. Conversion from ANNs to SNNs

1) Neuron Model for ANNs: Given a typical ANN which contains L full-connected (FC) layers, the output of the l^{th} convolutional layer can be formulated as:

$$\boldsymbol{o}^l = g\left(\boldsymbol{w}^l \boldsymbol{o}^{l-1}\right), \tag{1}$$

where l satisfies $1 \leq l \leq L$ and $\boldsymbol{w}^l$ represent the weight in the l^{th} layer. $g(\cdot)$ denotes the ReLU activation function.

2) Neuron Model for SNNs: We obey the convention in previous works [20], [21] and discuss the Integrate-and-Fire (IF) model for SNNs. Assuming we have the output value of the $(l-1)^{th}$ layer and formulate it as $\boldsymbol{x}^{l-1}(t)$, then we input this value into IF neurons in the l^{th} layer and get its temporal potential as follows:

$$\begin{aligned}
\boldsymbol{m}^l(t) &= \boldsymbol{v}^l(t-1) + \boldsymbol{w}^l \boldsymbol{x}^{l-1}(t), \\
\boldsymbol{s}^l(t) &= G\left(\boldsymbol{m}^l(t) - \theta^l\right), \\
\boldsymbol{v}^l(t) &= \boldsymbol{m}^l(t) - \boldsymbol{s}^l(t)\theta^l,
\end{aligned} \tag{2}$$

where represent the charging, discharging and resetting process, $\boldsymbol{m}^l(t)$ and $\boldsymbol{v}^l(t)$ denote the membrane potential before and after the trigger of a spike at the t^{th} time-step. Meanwhile, a firing threshold θ^l exists in the l^{th} layer IF model. $G(\cdot)$ is the Heaviside step function. $\boldsymbol{s}^l(t)$ denotes the spikes output of neurons in layer l at time-step t. For every element $m_i^l(t)$ in $\boldsymbol{m}^l(t)$ and $m_i^l(t) > \theta^l$, the neuron will fire a spike and update the membrane potential $v_i^l(t)$. To avoid information loss, we use a soft reset mechanism to update $\boldsymbol{v}^l(t)$ instead of directly resetting it to 0, i.e., the potential $\boldsymbol{v}^l(t)$ minus the threshold θ^l when a spike is fired.

$$\boldsymbol{x}^l(t) = \boldsymbol{s}^l(t)\theta^l. \tag{3}$$

Following previous work, we suppose that the postsynaptic neuron in the l^{th} layer receives unweighted postsynaptic potential θ^l if the presynaptic neuron in the $(l-1)^{th}$ layer fires a spike.

3) ANN-SNN Conversion: The key to ANN-SNN conversion is to map the activation values of the analog neurons in ANNs to the average firing rates of the spiking neurons in SNNs. We combine Eq. (2) to obtain the following equation:

$$\boldsymbol{v}^l(t) - \boldsymbol{v}^l(t-1) = \boldsymbol{w}^l \boldsymbol{x}^{l-1}(t) - \boldsymbol{s}^l(t)\theta^l. \tag{4}$$

The above equation expresses the relation between the potential at time t and time $t-1$. Expanding Eq. (4) by T time-steps and adding it up, the following equation is introduced:

$$\frac{\boldsymbol{v}^l(T) - \boldsymbol{v}^l(0)}{T} = \frac{\boldsymbol{w}^l \sum_{t=1}^{T} \boldsymbol{x}^{l-1}(t)}{T} - \frac{\sum_{t=1}^{T} \boldsymbol{s}^l(t)\theta^l}{T}. \tag{5}$$

The average post-synaptic potential of neurons in the l^{th} layer at time T is represented by the equation $\varphi^l(T) = \frac{\boldsymbol{w}^l \sum_{t=1}^{T} \boldsymbol{x}^{l-1}(t)}{T}$. Based on this, we can derive the relation between $\varphi^l(T)$ and $\varphi^{l-1}(T)$ at time T:

$$\varphi^l(T) = \boldsymbol{w}^l \varphi^{l-1}(T) - \frac{\boldsymbol{v}^l(T) - \boldsymbol{v}^l(0)}{T}. \tag{6}$$

Since $\varphi^l(T) \geq 0$, Eq. (1) differs from Eq. (6) in the $\frac{\boldsymbol{v}^l(T) - \boldsymbol{v}^l(0)}{T}$ term. If $\boldsymbol{v}^l(0)$ is equal to 0, we can reduce the difference between these two equations by decreasing $\frac{\boldsymbol{v}^l(T)}{T}$, thus reducing the conversion error. So when the simulation time is long enough, the conversion error approaches zero, but the high simulation time hinders the practical application of SNNs.

B. Three Types of Errors in ANN-SNN Conversion

ANN-SNN conversion error comes from activation bias. As analyzed in [17], the errors that existed in ANN-SNN conversion process are divided into three types: clipping error, quantization error and unevenness error.

Clipping Error, which is caused by different ranges of activation values per layer in ANNs and SNNs. Rueckauer et al. [37] summarized that 99% of the activation values in ANNs is in the interval of [0, $\frac{o^l_{\max}}{3}$], where $o^l_{\max}$ represents the maximum activation value of the l-layer in ANNs, thus setting the θ^l to 99% of the maximum activation value. In this way, if we set λ^l to be the threshold of l-layer in ANNs, activation values in the interval of [λ^l, $o^l_{\max}$] in ANNs are represented by the same value θ^l in SNNs, resulting in clipping error.

Quantization Error, which arises from mapping continuous ANN activation value to discrete SNN spike firing rate. The output values $\varphi(t)$ of SNNs are discrete, leading to the quantization resolution $\frac{\theta}{T}$. As shown in Fig. 2(a), $\boldsymbol{o}^l \in [\frac{k\lambda^l}{T}, \frac{(2k+1)\lambda^l}{2T})$ mapping to $\varphi(T)$ will be rounded down to $\frac{k\lambda^l}{T}$ and $\boldsymbol{o}^l \in [\frac{(2k+1)\lambda^l}{2T}, \frac{k\lambda^l}{T})(k = 0, 1, \cdots, T)$ mapping to $\varphi(T)$ will be raised to $\frac{k\lambda^l}{T}$, triggering a quantization error.

Unevenness Error, which derives from the uneven distribution of the input spikes arrival times over the total time-step. The discussion of the previous two types of error is based on the assumption that the input spikes arrive at a uniform time. However, the spikes received at the deep layer are often uneven to some degree, leading to differences in the output of that layer and further resulting in unevenness errors. As illustrated in Figs. 2(b), (c), and (d), three scenarios show that uneven inputs lead to more or fewer spikes in output compared

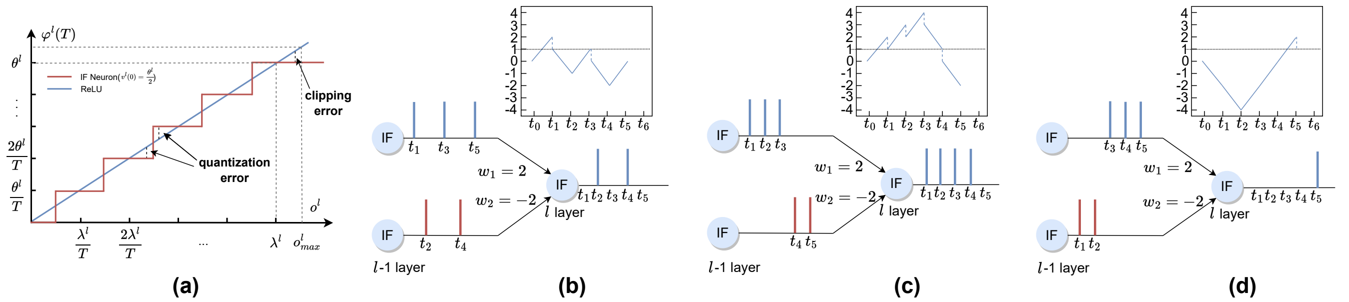


Fig. 2. Three distinct types of errors during the process of ANN-SNN conversion phases: (a) Clipping error and Quantization error, (b), (c), and (d) Unevenness error.

to the ideal situation. Ideally, as shown in Fig. 2(b), we set $\boldsymbol{o}^{l-1} = \varphi^{l-1}(T) = [0.6, 0.4], \boldsymbol{w}^l = [2, -2], \theta = 1, T = 5$, and calculate $\varphi^l(T) = \frac{\sum_{t=1}^{T} \boldsymbol{s}^l(t)\theta^l}{T} = \frac{2}{5}$ through the internal potential of the IF neuron. However, in the case of Fig. 2(c) and (d), $\varphi^l(T) = \frac{4}{5}$ and $\varphi^l(T) = \frac{1}{5}$ are calculated according to the output of the spike train.

Limitation of Existing Conversions. In previous studies, these three errors can be reduced by replacing the activation function in ANNs. Specifically, Opt [11] used the learnable upper bound and the optimal initial membrane potential to eliminate the clipping error and decrease the quantization error to a certain extent. The quantized activation function is proposed to replace the ReLU activation function in ANNs, thereby reducing the quantization error in the conversion [17], [38]. SRP [39] proposed an optimal strategy based on residual membrane potential to eliminate unevenness error. Nonetheless, it remains a challenge to simultaneously address the three types of errors on the action recognition task. For action recognition tasks, a pre-trained model is usually required to fine-tune the network. However, using the pre-trained model with ReLU to fine-tune the network with the quantization function works poorly. Current conversion methods with quantization functions are not suitable for particular task. Consequently, this paper is intended to solve the three errors in ANN-SNN conversion and achieve high performance at extremely low latency.

IV. Proposed Method

A. Overview

We illustrate our proposed method in terms of reducing three types of error. We propose a Dual Threshold Mapping (DTM) module, which consists of two inseparable parts, the acquisition of the threshold and the setting of the dual threshold. This module can simultaneously reduce clipping and unevenness errors. Then, we design a Scalable Threshold in Burst neurons (STB) to reduce the quantization error and eliminate the clipping error generated again. Eventually, we exhibit the total process of training, conversion, and inference. The overall framework is illustrated in Fig. 3.

B. Dual Threshold Mapping

To solve the clipping error and unevenness error simultaneously, we design a simple yet effective module called DTM, including two operations.

Firstly, to mitigate the clipping error, we replace the ReLU in 3D ANNs with a clip mapping function containing a learnable threshold λ. To be specific, given an input value a, the output of clip function $\hat{a}$ can be written as:

$$\hat{a} = \mathrm{clip}(a, 0, \lambda) = \begin{cases} a, & \text{if } 0 \le a \le \lambda, \\ \lambda, & \text{elif } a > \lambda, \\ 0, & \text{else.} \end{cases} \tag{7}$$

In practice, this learnable threshold λ is trained on 3D ANNs and can be directly mapped into the corresponding layer of SNNs. This function can avoid clipping error since the upper threshold λ in 3D ANNs is equal to the firing threshold θ in SNNs, where the gap between these two values disappears.

Secondly, we are motivated by the observation that the positive membrane potential is always released excessively, while the release of negative membrane potential is always ignored. Further, this issue causes the different outputs of spike trains when received inputs contain the same number of spikes but different time sequences. Therefore, to mitigate the unevenness error, we propose an operation embedding dual threshold. In contrast with previous methods which only contain a firing threshold θ, we introduce an extra threshold τ. In the l^{th} layer of SNN, given the j^{th} neuron, the output of our dual threshold operation can be represented as:

$$s_j^l(t) = \begin{cases} \theta_j^l, & \text{if } v_j^l(t) > \theta_j^l, \\ \tau_j^l, & \text{elif } v_j^l(t) < \tau_j^l, \\ 0, & \text{else.} \end{cases} \tag{8}$$

To be brief, we omit the subscript. Due to the property of mapping operation, θ in Eq. (8) can be replaced by λ in Eq. (7). Besides, to reduce complex cross-validation of other hyper-parameter τ, we obey the previous convention in [30], set $\tau = -\theta$ and introduce an accumulated potential $M(t)$. Specifically, $M(t)$ is the accumulation of the input potentials at t time-steps, which as a judgment condition avoids the excessive release of negative membrane potentials caused by the introduction of dual threshold so that neurons can simulate the processing of negative values by ReLU. Thus, Eq. (8) can be written as:

$$s_j^l(t) = \begin{cases} \lambda^l, & \text{if } v_j^l(t) > \lambda^l, \\ -\lambda^l, & \text{elif } v_j^l(t) < -\lambda^l \textit{ and } M_j^l(t) > 0, \\ 0, & \text{else.} \end{cases} \tag{9}$$

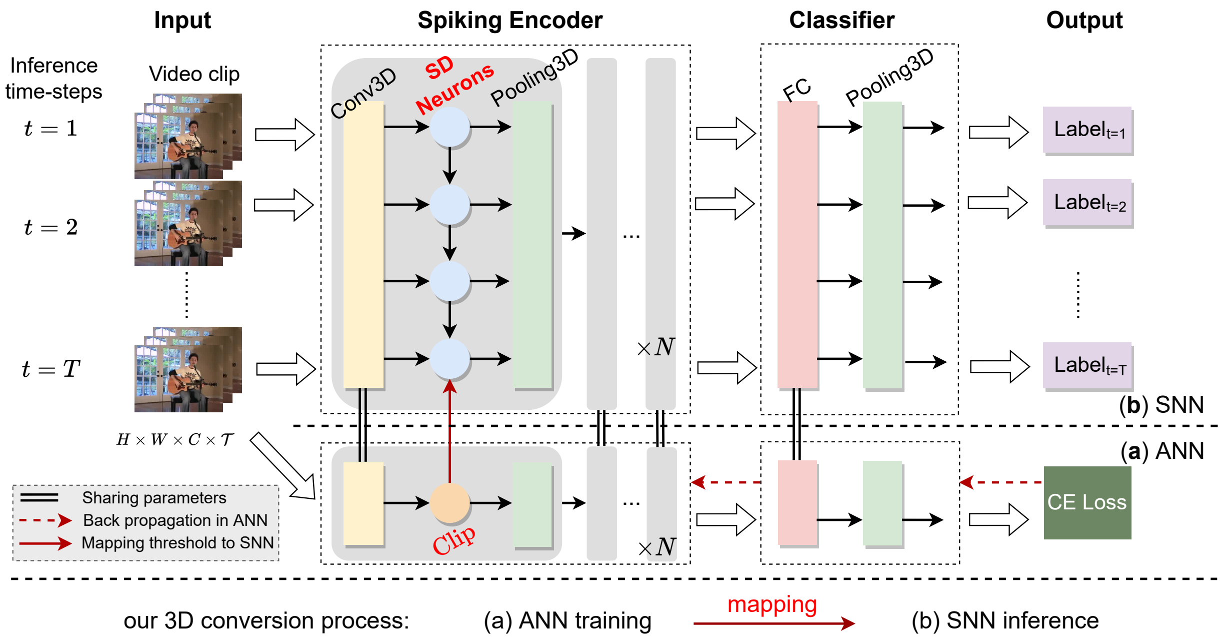


Fig. 3. Flowchart of the proposed SDM. It mainly includes two parts: clip mapping function (Clip) and scalable dual threshold neuron (SD Neurons). N indicates that $N \times$ {Conv3D-Spiking Neurons-Pooling3D} connected sequentially exist in the Spiking Encoder. The key point in our proposed SDM is the threshold mapping operation and constructed SD Neurons.

As shown in Fig. 2(c), we exhibit an example of excessive output and consider that the negative membrane potential at the last step ought to fire negative spikes, leading to mitigation of the excessive release of positive spikes. If the above dual threshold setting is adopted, two negative spikes will be generated at t_5 time-step and t_6 time-step to offset the positive spikes, so that only two positive spikes are received by the next layer, which is an error-free propagation process. In Fig. 2(d), since the cumulative potential $M_j^l(t_1)$ and $M_j^l(t_2)$ is less than 0, the negative spike is not released, thus avoiding the impact of negative membrane potential release on the whole process. Besides, two positive spikes will be fired at t_5 time-step and t_6 time-step and the output is correct as in Fig. 2(b). To summarize, our proposed dual threshold can decrease unevenness error.

C. Scalable Threshold in Burst Neurons

We introduce a novel aspect to analyze the quantization error and further propose a scalable threshold strategy to alleviate this error, which is based on a burst mechanism.

Quantization error is related to the quantization resolution, where this error increases with a growing quantization resolution. Thus, we attempt to restrict quantization resolution. Formally, the quantization resolution can be represented as $\frac{\theta^l}{T}$, which is decided by T and θ^l. Intuitively, both increasing the value of T and decreasing the value of θ^l can achieve the restriction of the quantization resolution. However, increasing the value of T suffers from high inference time, encouraging us to reduce the value of θ^l.

For this, we propose a scalable threshold to decrease θ^l. Given a scalable ratio α, we multiply this value on the firing threshold in each layer of SNN. The modified firing threshold in the l^{th} layer can be written as $\theta^l \leftarrow \alpha\theta^l$, where α satisfies $\alpha \in [0, 1]$. However, in practice, this scaling operation would reintroduce clipping error which has been eliminated in our proposed clip mapping function (Eq. (7)). In Fig. 4, it is evident that scaling the threshold for the scalable dual (SD) neuron (yellow line) effectively mitigates the quantization error in comparison to the integrate-and-fire (IF) neuron (red line). However, this improvement is accompanied by the reoccurrence of the clipping error. Specifically, we classify the neurons in the $(l-1)^{th}$ layer into two sets: $\mathbb{R}_1 = \left\{j \mid \boldsymbol{m}_j^{l-1} <= \alpha\theta\right\}$, representing neurons in the $(l-1)^{th}$ layer with a membrane potential below the threshold at time t, and $\mathbb{R}_2 = \left\{j \mid \boldsymbol{m}_j^{l-1} > \alpha\theta\right\}$, representing neurons in the $(l-1)^{th}$ layer at time t with a membrane potential exceeding the threshold. The weighted input potential of the i^{th} neuron in layer l at time t is expressed as:

$$
\begin{aligned}
\sum_{t=0}^{T} \boldsymbol{I}_i^l(t) &= \sum_{t=0}^{T} \sum \boldsymbol{w}_{ij} \boldsymbol{x}_j^{l-1}(t) \\
&= \sum_{j \in \mathbb{R}_1} \boldsymbol{w}_{ij} \sum_{t=0}^{T} \boldsymbol{x}_j^{l-1}(t) + \sum_{j \in \mathbb{R}_2} \boldsymbol{w}_{ij} \sum_{t=0}^{T} \boldsymbol{x}_j^{l-1}(t) \\
&= \sum_{j \in \mathbb{R}_1} \boldsymbol{w}_{ij} \sum_{t=0}^{T} \boldsymbol{x}_j^{l-1}(t) + \sum_{j \in \mathbb{R}_2} \boldsymbol{w}_{ij} T \alpha \theta^{l-1}.
\end{aligned}
\tag{10}
$$

The above equation reveals that the output potential of neurons belonging to $\mathbb{R}_2$ is constrained to the threshold potential. Consequently, any residual potential exceeding the threshold is not transmitted, leading to the occurrence of errors. To solve this issue, we introduce a burst mechanism that fires multiple spikes in one time-step. Burst neurons can adaptively adjust the capacity of the transmitted information, thus reducing the residual membrane potential in the hidden layer. The key point

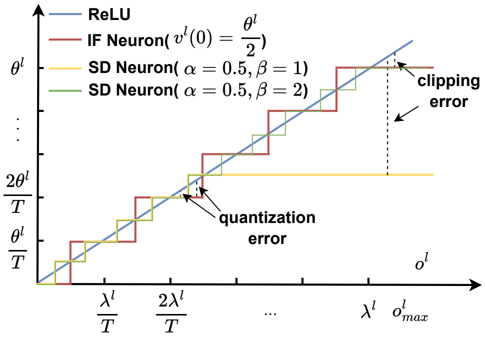


Fig. 4. Comparison of errors in the conversion process of different spiking neurons and hyper-parameter settings.

in the burst neuron is a maximum burst spikes number β. Compared to the typical IF neuron model in SNNs (Eq. (2)), this neuron resorts to β to prevent interference with the next simulation step, which is formulated as:

$$\begin{aligned}\boldsymbol{s}^l(t)&=\sum_{b=0}^{\beta}G\left(\boldsymbol{m}^l\left(t\right)-\alpha\theta^l\right),\\ \boldsymbol{x}^l(t)&=\sum_{b=0}^{\beta}G\left(\boldsymbol{m}^l\left(t\right)-\alpha\theta^l\right)\alpha\theta^l.\end{aligned}\tag{11}$$

Through the combination of Eqs. (10) and (11), it becomes evident that parameter β guarantees the release of residual potential, thereby reducing errors in information transmission.

By integrating the scalable threshold and the burst mechanism, we achieve the aim of mitigating quantization error as well as avoiding the reintroduction of clipping error.

D. Training and Inference

We design DTM and STB to solve three types of conversion errors existed in ANN-SNN conversion. In practice, these two modules can be integrated into a Scalable Dual threshold Neuron (SD Neuron) and a mapping operation with a clip function. In Fig. 3, we exhibit these two parts in our training framework which includes ANN training, ANN-SNN conversion, and SNN inference.

1) ANN Training: Given an action recognition task, we have a training set $\mathcal{D}=\{(z_i,y_i)\}_{i=0}^{n}$, where z_i denotes the i^{th} training video clip, $z\in\mathbb{R}^{H\times W\times C\times\mathcal{T}}$, and $\mathcal{T}$ represents the time dimension of the video clip. Suppose a 3D ANN M_{ANN}, the output of z_i can be represented as $M_{\text{ANN}}(z_i)$. Note that this phase requires learning a threshold λ for ANN-SNN conversion. Therefore, we first replace the ReLU function in each layer with a clip function that contains a learnable threshold λ. Then, given a loss function $\ell(\cdot)$, the training loss in each training batch can be written as:

$$\mathcal{L}=\frac{1}{m}\sum_{i=1}^{m}\ell\left(M_{\text{ANN}}\left(z_i;\boldsymbol{w}\right),y_i\right),\tag{12}$$

where m denotes the size of the mini-batch. By stochastic gradient descent (SGD) [40], we update the parameters of ANN until convergence.

Algorithm 1 Pseudo-code for SDM Algorithm.

```
Require: Training set $\mathcal{D}=\{(z_i,y_i)\}_{i=1}^{n}$, dynamic threshold $\lambda$, scaling
    factor $\alpha$, learning rate $\epsilon$, mini-batch size $m$.
Require: ANN model $M_{\text{ANN}}(\boldsymbol{w})$ with $L$ layers.
Ensure: $M_{\text{SNN}}(\hat{\boldsymbol{w}})$.
 1: for $l=1$ to $L$ do
 2:     Replace ReLU function with clip$(a,0,\lambda)$.                ▷ Eq. (7)
 3:     Replace MaxPooling layer with AvgPooling layer.
 4: end for
 5: Initialize $\boldsymbol{w}$ with a pre-trained model on KINETICS.
 6: while $e<$ MaxIters do
 7:     Sample a mini-batch $\{(z_i,y_i)\}_{i=0}^{m}$ from $\mathcal{D}$.
 8:     for $l=1$ to $L$ do
 9:         $a^l=\text{clip}\left(\boldsymbol{w}^l a^{l-1},0,\lambda\right)$        // reinitialize $a$ in clip function
10:     end for
11:     Compute training loss $\mathcal{L}$ for $\{(z_i,y_i)\}_{i=0}^{m}$.        ▷ Eq. (12)
12:     for $l=1$ to $L$ do
13:         $\boldsymbol{w}^l\leftarrow\boldsymbol{w}^l-\epsilon\frac{\partial\mathcal{L}}{\partial\boldsymbol{w}^l}$        // updating ANN parameters $\boldsymbol{w}$
14:         $\lambda^l\leftarrow\lambda^l-\epsilon\frac{\partial\mathcal{L}}{\partial\lambda^l}$        // updating $\lambda$ in clip function
15:     end for
16: end while
17: for $l=1$ to $L$ do
18:     $M_{\text{SNN}}.\hat{\boldsymbol{w}}^l\leftarrow M_{\text{ANN}}.\boldsymbol{w}^l$
19:     $M_{\text{SNN}}.\theta^l\leftarrow M_{\text{ANN}}.\lambda^l\cdot\alpha$        ▷ Eq. (9)
20:     $M_{\text{SNN}}.\tau^l\leftarrow -M_{\text{ANN}}.\lambda^l\cdot\alpha$
21:     $M_{\text{SNN}}.v^l(0)\leftarrow M_{\text{SNN}}.\theta^l/2$
22: end for
23: return $M_{\text{SNN}}$
```

2) ANN-SNN Conversion: In the above phase, we learn a 3D ANN with L clip functions, where L is the number of layers of 3D ANN. In the conversion phase, ANN and SNN share weights and map the thresholds learned in ANN to the dual threshold of the SD neurons. Formally, this phase contains four steps. 1) We load the learned weight in ANN to corresponding layers in SNN. 2) We replace the clip functions in ANN with SD neurons, and the process involves the threshold transmission. Specifically, we multiply λ^l in the clip function with a scalable factor α and then transmit this value into θ^l in SNN. 3) We assign another threshold τ^l in SNN by the opposite of the threshold θ^l in SNN. 4) We set the initial membrane potential of SNN to half of the threshold θ^l. Previous research [11] has demonstrated that this specific configuration minimizes the expectation of the squared conversion error.

3) SNN Inference: Given a video clip z_i, we repeatedly input z_i with T times into the converted SNN network, where T represents the preset number of inference time-steps. Specifically, in the spiking encoder, image inputs are encoded into a spike train of length T, whereas video inputs are encoded into a spike train of length T for a specified number of frames. Then the spiking encoder extracts feature embeddings of z_i and the classifier gives corresponding prediction labels.

Our proposed SDM training pseudo-code is shown in Algorithm 1, which includes two stages, one is the stage of ANN training (Lines 1–15), and the other is the stage of ANN-SNN conversion (Lines 16–22).

V. Experimental Results

A. Datasets and Experimental Setup

1) Datasets: To assess the efficacy of our proposed ANN-SNN conversion approach, we focus on action recognition as the targeted task and evaluate its performance on two widely adopted benchmark datasets: UCF-101 and HMDB-51.

TABLE I
COMPARISON PERFORMANCES OF TOP-1 ACCURACY (%) ON TWO BENCHMARK DATASETS WITH THE STATE-OF-THE-ART METHODS AND VARYING NUMBER OF INFERENCE TIME-STEPS T. † INDICATES REPRODUCED METHODS. THE BEST-PERFORMING RESULTS ARE HIGHLIGHTED IN BOLD.

Dataset	Backbone	Method	Venue	ANN	T = 4	T = 8	T = 16	T = 32	T = 64	T = 128	T = 256
UCF-101	SlowFast	RTS [27] †	ICLR '21	94.76	0.93	0.90	0.90	3.57	47.63	86.15	91.78
		Burst [28] †	IJCAI '22	94.97	0.95	0.93	1.06	9.65	71.66	90.09	93.13
		SNM [30] †	IJCAI '22	94.97	1.06	0.90	1.40	44.81	92.15	94.40	**94.87**
		Opt [11] †	AAAI '22	94.76	1.69	1.85	3.17	16.39	73.62	89.82	92.57
		SDM (β = 2)	Ours	94.76	15.12	72.35	92.31	93.68	94.37	94.66	94.74
		SDM (β = 5)	Ours	94.76	**92.94**	**93.50**	**93.95**	**94.48**	**94.55**	**94.69**	94.76
	SlowOnly	RTS [27] †	ICLR '21	92.75	0.98	0.98	0.90	1.61	43.03	85.49	90.85
		Burst [28] †	IJCAI '22	93.15	0.98	0.98	0.90	3.62	58.05	87.89	91.59
		SNM [30] †	IJCAI '22	93.15	0.85	0.85	0.85	57.44	90.99	92.49	**92.97**
		Opt [11] †	AAAI '22	92.75	2.38	3.09	6.77	34.89	78.59	89.45	91.78
		SDM (β = 2)	Ours	92.75	20.62	78.11	91.36	92.36	92.69	92.70	92.70
		SDM (β = 5)	Ours	92.75	**90.77**	**92.47**	**92.60**	**92.73**	**92.78**	**92.78**	92.76
HMDB-51	SlowFast	RTS [27] †	ICLR '21	72.22	1.96	1.96	1.96	1.96	24.44	63.40	69.35
		Burst [28] †	IJCAI '22	72.54	1.96	1.96	2.09	6.47	48.50	65.69	70.33
		SNM [30] †	IJCAI '22	72.54	1.96	1.96	1.96	32.75	68.95	71.90	72.09
		Opt [11] †	AAAI '22	72.22	2.75	2.55	3.66	9.22	39.61	62.03	68.56
		SDM (β = 2)	Ours	72.22	9.87	43.14	67.45	71.05	71.90	**72.29**	**72.42**
		SDM (β = 5)	Ours	72.22	**67.71**	**70.46**	**71.57**	**72.16**	**72.29**	**72.29**	72.16
	SlowOnly	RTS [27] †	ICLR '21	65.16	1.96	1.96	2.42	3.59	21.37	52.16	61.50
		Burst [28] †	IJCAI '22	67.71	1.63	1.96	1.96	4.71	38.37	60.46	65.62
		SNM [30] †	IJCAI '22	67.71	1.76	1.96	2.55	35.95	64.51	**66.60**	**66.73**
		Opt [11] †	AAAI '22	65.16	3.20	3.53	5.23	18.89	39.87	56.80	61.83
		SDM (β = 2)	Ours	65.16	15.82	48.24	63.40	64.38	65.03	64.90	65.10
		SDM (β = 5)	Ours	65.16	**62.68**	**64.97**	**64.71**	**64.77**	**65.10**	65.10	65.16

Furthermore, we conduct additional experiments on CIFAR-10 and CIFAR-100 datasets for image classification tasks to validate the generalizability of our method.

UCF-101 [41] is a realistic video dataset that comprises a collection of 13,320 videos sourced from YouTube. Each video typically represents a distinct action category, and the dataset exhibits a wide range of variations in action acquisition, including changes in appearance, attitude, object proportions, and other factors.

HMDB-51 [42] consists of 6,849 videos encompassing 51 action categories. The videos in this dataset are primarily extracted from movie clips. To ensure consistency and remove any interference caused by camera motion, the video clips have been carefully aligned using standard stitching techniques.

CIFAR-10[1] comprises a training set with 50,000 images and a testing set with 10,000 images, encompassing ten distinct classes. Similarly, CIFAR-100 follows the same sample size and division method as CIFAR-10. However, it includes 100 fine categories that are further grouped into 20 rough categories.

By utilizing these benchmark datasets, we aim to comprehensively evaluate the performance of our proposed ANN-SNN conversion method for action recognition task. Furthermore, we verify the generalizability of our method by applying it to the task of image classification.

2) Training Setup: We adhere to the established evaluation protocols provided by the aforementioned datasets and employ standard training/testing splits for both UCF-101 and HMDB-51. In line with the data format used in [34], we resize the videos to have a short edge size of 256 using FFmpeg. For our backbone network, we utilize a 50-layer dual-stream ResNet3D [43] architecture. Initially, we pre-train this model on KINETICS400 [44] dataset. Then, we fine-tune the model using the clip function on both datasets. We employ stochastic gradient descent (SGD) [40] as the optimizer with a momentum value of 0.9 and weight decay set to 1e-5. The initial learning rate is set to 1e-3 and is decayed by 0.1 every 20 epochs. The training settings for CIFAR-10 and CIFAR-100 experiments are consistent with QCFS [17] settings. All our code is implemented using PyTorch 1.9.0, and the deep model is trained on Tesla V100 GPUs with a batch size of 16. These specifications ensure efficient and effective training of our proposed approach on the chosen datasets.

B. Comparison Results

To validate the efficacy of our proposed method for ANN-SNN conversion, we conduct extensive experiments focusing on action recognition task. Two different 3D ANNs are utilized, and the inference time-steps T are varied within the set $\{4, 8, 16, 32, 64, 128, 256\}$. The results of our experiments are presented in Table I. It is evident that our method consistently outperforms the baseline Opt [11], across all tested cases. Particularly noteworthy is the significant improvement brought about by our proposed method at lower time-steps. For instance, when $T = 8$ and experiments are conducted on SlowFast [34] backbone architecture, SDM achieves a remarkable increase of 70.50% in Top-1 accuracy on UCF-101 and 40.59% in Top-1 accuracy on HMDB-51. Furthermore, by setting β = 5, the

[1] https://www.cs.toronto.edu/ kriz/cifar.html

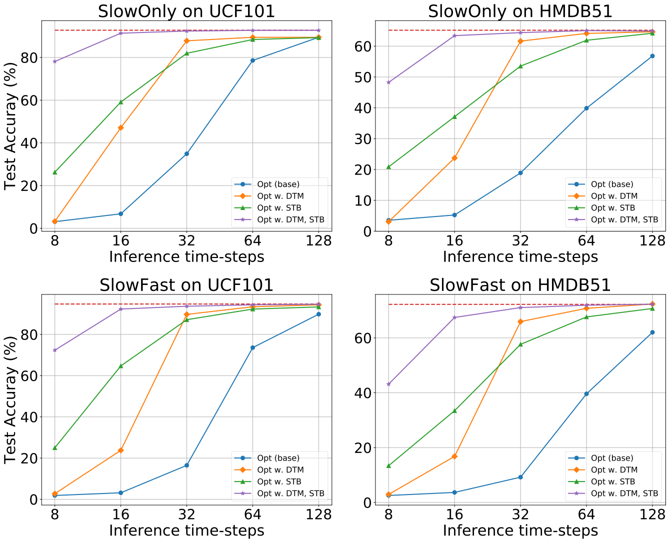


Fig. 5. Ablation studies with a fixed value of β set to 2 in our proposed framework. The experiments are conducted on SlowOnly and SlowFast. The red dashed line is utilized to represent ANN inference Top-1 accuracy in the testing set.

converted SlowFast model achieve a substantial increase of 91.25% in Top-1 accuracy on UCF-101, while operating at an ultra-low latency of only 4 time-steps. Our proposed method, SDM, exhibits distinct advantages over other competitive approaches in terms of performance at low time-steps. By effectively mitigating the three types of errors associated with ANN-SNN conversion, SDM ensures high accuracy in the converted SNN. These findings underline the effectiveness and superiority of our proposed method in achieving accurate and efficient ANN-SNN conversion.

C. Ablation Study

1) Effectiveness of Each Component: To assess the effectiveness of our proposed method, we conduct ablation experiments on UCF-101 and HMDB-51, employing two 3D ANNs: SlowFast and SlowOnly [34]. Our proposed training framework consists of two key components: the Dual Threshold Mapping (DTM) module and the Scalable Threshold in Burst neurons (STB) module. Notably, we choose Opt [11] as the baseline method and initialize the optimal membrane potential for the process from ANN-SNN conversion.

The results of the ablation studies, as depicted in Fig. 5, highlight two key observations. 1) When compared to the baseline method Opt, each component proposed in our framework consistently improves performance across different numbers of inference time-steps. On UCF-101, the addition of DTM module into Opt, the converted SNN takes only 32 inference time-steps, which achieves over 70% Top-1 accuracy improvement on SlowFast network and more than 50% accuracy improvements on SlowOnly network. Similarly, the incorporation of STB module into Opt yields remarkable performance gains. With just 32 inference time-steps, the converted SNN achieves substantial improvements in Top-1 accuracy, surpassing 70% on SlowFast network and exceeding 40% on SlowOnly network. Furthermore, utilizing both DTM and STB modules lead to better performance at all inference steps compared to using a single module. On HMDB-51, the performance trends of the two modules align with those observed on UCF-101. These results substantiate the effectiveness of DTM and STB modules, particularly under lower inference time-steps. 2) Our proposed method achieves superior performance with only 16 inference time-steps compared to Opt, which employs 128 time-steps. Importantly, our method rarely compromises the performance of the original ANN. Specifically, when inferring with 16 time-steps, our proposed SDM achieves 92.31% accuracy on UCF-101, while the original ANN achieves a Top-1 accuracy of 94.76%. This indicates that our proposed method achieves comparable performance to the original ANN with significantly fewer time-steps, demonstrating the efficiency and effectiveness of our approach.

TABLE II
COMPARISON PERFORMANCES OF TOP-1 ACCURACY (%) WITH DIFFERENT VARIANTS OF OUR METHOD (β = 2) ON UCF-101, INCLUDING MAPPING FUNCTION (**M**), DUAL THRESHOLD (**D**), BURST MECHANISM (**B**), AND SCALING OPERATION (**S**).

M	D	B	S	T = 16	T = 32	T = 64
●	○	○	○	3.15	16.47	73.59
●	●	○	○	23.74	89.72	93.42
●	○	●	○	3.86	26.51	80.94
●	○	○	●	15.01	31.32	42.00
●	●	●	○	76.61	92.73	93.55
●	●	○	●	14.99	31.35	41.98
●	○	●	●	64.71	87.15	92.31
●	●	●	●	**92.31**	**93.68**	**94.37**

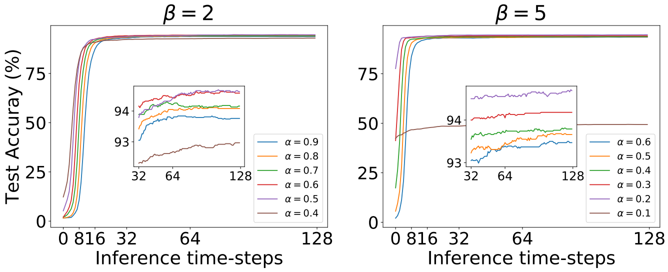


Fig. 6. Sensitivity analysis with various hyper-parameter α on UCF-101.

In Table II, we present a more detailed analysis of the ablation results for our two modules, specifically focusing on relatively low time-steps. Notably, we decompose our modules into more specific variants to gain further insights into their contributions. We observe that the MDS variant is not as effective as the MD variant. This can be attributed to the fact that while the use of the scaling threshold in MDS helps reduce quantization error, the reintroduction of a clipping error has a more substantial impact on the overall results.

2) Hyper-Parameter Selection: In our training framework, fine-tuning the performance involves adjusting two key hyper-parameters: a scaling factor denoted as α, and a maximum burst spike number denoted as β. The choice of β depends on the specific implementation of the neuromorphic chip, and in line with the convention proposed in [28], we set β to be 2 and 5 for our experiments. Fig. 6 illustrates the results of our experiments with varying values of α in the range of [0.4, 0.9]

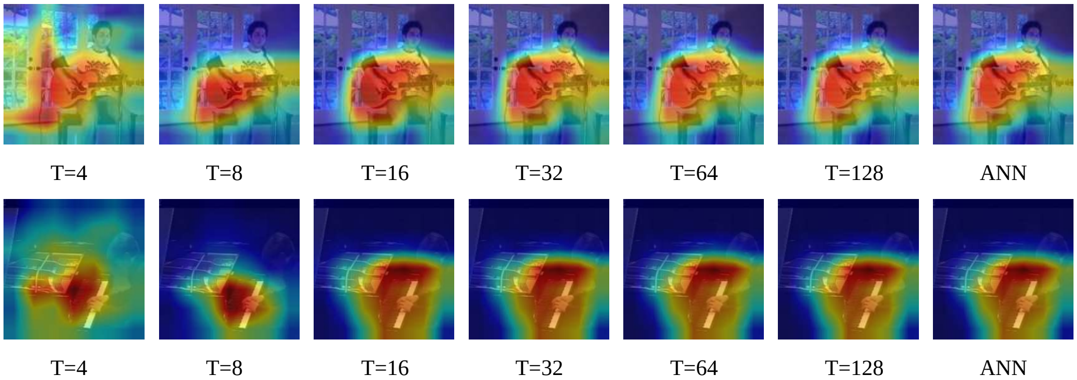


Fig. 7. Comparison of the attention heatmaps generated by SNNs with different inference time-steps to those of ANNs.

TABLE III
COMPARISON PERFORMANCES OF TOP-1 ACCURACY (%) AND ENERGY CONSUMPTION ON UCF-101 WITH VARIOUS ANN METHODS. # OP_{ANN} AND # OP_{SNN} REPRESENT THE NUMBER OF FLOATING POINT OPERATIONS AND THE NUMBER OF BINARY SPIKE OPERATIONS.

Method	Top-1 ↑	# OP_{ANN} ↓	# OP_{SNN} ↓	Energy ↓
C3D [45]	83.27	38.62G	-	0.178J
I3D [46]	93.76	33.35G	-	0.153J
TimeSformer [47]	94.52	100.96G	-	0.464J
SlowOnly [34]	93.47	21.00G	-	0.097J
SlowFast [34]	**94.87**	27.90G	-	0.128J
SlowOnly w SDM	90.77	1.92G	23.89G	**0.030J**
SlowFast w SDM	92.94	11.60G	49.50G	0.098J

for β = 2. Notably, as the value of α decreases, the test accuracy increases at lower step sizes, but decreases at relatively higher step sizes. Specifically, when α is set to 0.4, unsatisfactory results are observed across low and relatively high step sizes. Conversely, the best performance is achieved when α is set to 0.5, as evidenced by the highest accuracy during the decline of α. In a separate experimental setup where β is set to 5, we consistently observe that an α value of 0.2 always yields the highest accuracy across all inference time-steps.

D. Attention Visualization

We randomly select a frame from two video clips as an example to visualize attention mechanism. Fig. 7 presents the distribution of the attention heatmap generated by SNNs using our proposed method, considering different inference time-steps. Upon examination, we observe that as the inference time-steps T increase, the attention distribution of SNN becomes more similar to that of ANN. Notably, our method demonstrates remarkable similarity to ANN's attention distribution at lower inference time-steps, such as T = 8. For instance, at T = 4, the attention heatmap of SNN predominantly focuses on the man playing the guitar, albeit with somewhat shifted in some of the hot zones. However, at T = 8, SNN's attention aligns closely with ANN's attention, with a strong emphasis on the guitar. This indicates that our method achieves a desirable performance, where SNN's attention closely resembles that of ANN at lower inference time-steps.

E. Energy Efficiency Analysis

1) Intuitive Level: Due to the nature of SNNs, where neurons consume power only when they generate spikes, SNNs inherently possess characteristics of low power consumption. This attribute makes SNNs well-suited for hardware deployment, as the power consumption is directly related to the spiking activity of the neurons.

2) Experimental Level: We conduct a comparative analysis of the power consumption between SlowOnly and SlowFast [34] after converting them using SDM, in comparison to existing 3D networks. To further investigate the power consumption on neuromorphic chips, we compute the energy cost per operation for both ANNs and SNNs using 45nm CMOS technology. To ensure a fair comparison, we adopt the convention proposed by [30], [48], where the energy cost for a 32-bit ANN MAC operation is defined as 4.6pJ, which is 5.1 times higher than the energy cost for an SNN addition operation. Then, the number of synaptic operations is calculated as the first layer of convolution plus the layer i spike firing rate multiplied by the layer $i+1$ convolution, where $i \in \{1, \cdots, L-1\}$. The power consumption ratio of ANN and SNN can be summarized as:

$$\text{Energy}_{S/A} = \frac{c\gamma + d\zeta}{b\gamma}, \quad \text{where } d = \sum_{l=1}^{L-1} R^l \text{OP}^{l+1}, \tag{13}$$

where γ represents the power consumption for MAC calculations (4.6pJ), ζ represents the power consumption for additive calculations (0.9pJ), b represents the number of MAC operations in ANN, c represents the number of MAC operations in SNN, and d represents the number of addition operations in SNN. To analyze the power consumption comparison, we introduce the power consumption ratio of ANN and SNN as defined in Eq. (13). In addition, R^l and OP^{l+1} refer to the spike firing rate of the l^{th} layer and operands for 3D convolution of the $(l+1)^{th}$ layer. Moreover, we employ direct encoding [49] where the first convolutional layer performs floating-point operations as the encoding layer.

By following the procedure outlined in Eq. (13), we perform detailed calculations to determine the power consumption of SNNs at T = 4 for both SlowOnly and SlowFast. Table III presents the comparison results of accuracy and power consumption for our converted SNNs, along with several

TABLE IV
COMPARISON PERFORMANCES OF TOP-1 ACCURACY (%) ON CIFAR-10 AND CIFAR-100 WITH THE STATE-OF-THE-ART METHODS AND VARYING NUMBER OF INFERENCE TIME-STEPS T.

Dataset	Backbone	Method	Venue	ANN	T = 2	T = 4	T = 8	T = 16	T = 32	T = 64	T = 128
CIFAR-10	VGG-16	RTS [27]	ICLR '21	95.72	-	-	-	-	76.24	90.64	-
		SNNC-AP [19]	ICML '21	95.72	-	-	-	-	93.71	95.14	-
		RNL [50]	IJCAI '21	92.90	-	-	-	57.90	85.40	91.15	92.51
		SNM [30]	IJCAI '22	94.09	-	-	-	-	93.43	94.07	94.07
		Opt [11]	AAAI '22	94.57	-	-	90.96	93.38	94.20	94.45	94.50
		QCFS [17]	ICLR '22	95.52	91.18	93.96	94.95	95.40	95.54	95.55	-
		SRP [39]	AAAI '23	95.52	94.47	95.32	**95.52**	95.44	95.42	95.40	-
		SDM (β = 2)	Ours	95.57	93.52	94.83	95.41	95.50	**95.55**	95.54	**95.58**
		SDM (β = 5)	Ours	95.57	**95.36**	**95.46**	**95.52**	**95.54**	**95.55**	**95.56**	95.57
	ResNet-18	RTS [27]	ICLR '21	95.46	-	-	-	-	84.06	92.48	-
		SNNC-AP [19]	ICML '21	95.46	-	-	-	-	94.78	95.30	-
		RNL [50]	IJCAI '21	93.84	-	-	-	47.63	83.95	91.96	93.27
		SNM [30]	IJCAI '22	95.39	-	-	-	-	94.03	94.03	95.19
		Opt [11]	AAAI '22	96.04	-	-	75.44	90.43	94.82	95.92	96.08
		QCFS [17]	ICLR '22	96.04	75.44	90.43	94.82	95.92	96.08	96.06	-
		SRP [39]	AAAI '23	95.64	95.06	95.25	95.60	95.55	95.55	95.58	-
		SDM (β = 2)	Ours	96.09	93.15	95.14	95.93	**96.09**	**96.11**	**96.08**	**96.09**
		SDM (β = 5)	Ours	96.09	**95.98**	**96.01**	**96.06**	96.07	96.08	**96.08**	96.06
CIFAR-100	VGG-16	RTS [27]	ICLR '21	77.89	-	-	-	-	7.64	21.84	-
		SNNC-AP [19]	ICML '21	77.89	-	-	-	-	73.55	76.64	-
		SNM [30]	IJCAI '22	74.13	-	-	-	-	71.80	73.69	73.95
		Opt [11]	AAAI '22	76.31	-	-	60.49	70.72	74.82	75.97	**76.25**
		QCFS [17]	ICLR '22	76.28	63.79	69.62	73.96	76.24	**77.01**	**77.10**	-
		SRP [39]	AAAI '23	76.28	74.31	**75.42**	**76.25**	**76.42**	76.45	76.37	-
		SDM (β = 2)	Ours	75.26	67.07	72.31	74.78	75.17	75.31	75.33	75.27
		SDM (β = 5)	Ours	75.26	**74.65**	75.01	75.26	75.26	75.31	75.31	75.28
	ResNet-18	RTS [27]	ICLR '21	77.16	-	-	-	-	51.27	70.12	-
		SNNC-AP [19]	ICML '21	77.16	-	-	-	-	76.32	77.29	-
		SNM [30]	IJCAI '22	78.26	-	-	-	-	74.48	77.59	**77.97**
		SDM (β = 2)	Ours	77.63	64.53	73.43	77.10	**77.89**	**77.71**	**77.73**	77.66
		SDM (β = 5)	Ours	77.63	**76.97**	**77.63**	**77.66**	77.62	77.64	77.64	77.64
	ResNet-20	Opt [11]	AAAI '22	70.43	-	-	23.09	52.34	67.18	69.96	**70.51**
		QCFS [17]	ICLR '22	69.94	19.96	34.14	55.37	67.33	69.82	**70.49**	-
		SRP [39]	AAAI '23	69.94	53.96	59.34	62.94	64.71	65.50	65.82	-
		SDM (β = 2)	Ours	69.83	22.08	48.28	66.49	69.32	69.65	69.79	69.81
		SDM (β = 5)	Ours	69.83	**66.04**	**69.38**	**69.64**	**69.71**	**69.87**	69.79	69.82

competitive 3D ANNs, where the power consumption is computed based on the energy consumption per video. Notably, the converted SlowOnly achieves a Top-1 accuracy of 90.77%, while only requiring an energy cost of 0.03J, significantly lower than the original SlowOnly's energy consumption of 0.097J. Moreover, our spiking SlowOnly model demonstrates an energy consumption equivalent to just 30.93% of that of ANN counterpart. These results clearly illustrate the high inference performance and energy efficiency achieved by our proposed framework. The comparison between accuracy and power consumption highlights the efficacy of our converted SNN models in achieving superior performance while operating in a highly energy-efficient manner.

F. Generalization Experiment

To assess the generalizability of our ANN-SNN conversion method, we employ SDM framework in an image classification task. Table IV presents a comparison between our method and state-of-the-art ANN-SNN conversion methods on CIFAR-10 and CIFAR-100. For CIFAR-10, regardless of whether the model is VGG-16 [51] or ResNet-18 [52], SDM surpasses previous conversion methods across all time-steps. Notably, SDM achieves an accuracy of 95.98% on ResNet-18 at T = 2, which closely approximates the accuracy of ANNs. On CIFAR-100, SDM outperforms comparative approaches at low and even ultra-low time-steps when converting ResNet-18 and ResNet-20 models. Specifically, on ResNet-18, SDM achieves an accuracy of 77.63% at T = 4, which is equivalent to the performance of ANNs. Although the performance of our ANN on VGG-16 is not satisfactory, SNN converted using SDM exhibits higher accuracy compared to the comparison method at T = 2. These results validate the effectiveness of our method, not only in action recognition tasks but also in image classification tasks, showcasing its broad applicability.

VI. CONCLUSION

In this paper, we introduce a novel training framework for ANN-SNN conversion. To the best of our knowledge, we are the first to employ ANN-SNN conversion method to train deep SNN for action recognition task. Specifically, DTM module is designed to mitigate the clipping and unevenness errors that arise during the conversion process. By carefully mapping the activation thresholds, we are able to reduce the impact of these errors and achieve improved conversion results. However, we also discover a trade-off between the quantization error and clipping error when adjusting the thresholds. To overcome this challenge, we propose an STB module, which introduces a scalable threshold strategy specifically for burst neurons. This strategy effectively reduces the quantization error while preventing the reintroduction of clipping error. Through extensive experiments, we demonstrate that our converted SNNs can achieve comparable accuracy to ANNs while operating at ultra-low latency in complex action recognition task using 3D networks. Furthermore, our converted SNNs exhibit significant advantages in terms of power consumption when compared to traditional ANN methods. We achieve accurate and energy-efficient SNN models, paving the way for the application of SNNs in more complex tasks requiring spatio-temporal information processing.

References

[1] Q. Wei, H. Sun, X. Lu, and Y. Yin, "Self-filtering: A noise-aware sample selection for label noise with confidence penalization," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 516–532.

[2] W. Liu, X. Zhong, Z. Zhou, K. Jiang, Z. Wang, and C. Lin, "Dual-recommendation disentanglement network for view fuzz in action recognition," *IEEE Trans. Image Process.*, vol. 32, pp. 2719–2733, 2023.

[3] Q. Wei, L. Feng, H. Sun, R. Wang, C. Guo, and Y. Yin, "Fine-grained classification with noisy labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 11 651–11 660.

[4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Adv. Neural Inf. Process. Syst.*, 2020.

[5] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.

[6] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[7] K. Roy, A. R. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nat.*, vol. 575, pp. 607–617, 2019.

[8] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. V. Arthur, P. Merolla, N. Imam, Y. Y. Nakamura, P. Datta, G. Nam, B. Taba, M. P. Beakes, B. Brezzo, J. B. Kuang, R. Manohar, W. P. Risk, B. L. Jackson, and D. S. Modha, "TrueNorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1537–1557, 2015.

[9] M. Davies, N. Srinivasa, T. Lin, G. N. Chinya, Y. Cao, S. H. Choday, G. D. Dimou, P. Joshi, N. Imam, S. Jain, Y. Liao, C. Lin, A. Lines, R. Liu, D. Mathaikutty, S. McCoy, A. Paul, J. Tse, G. Venkataramanan, Y. Weng, A. Wild, Y. Yang, and H. Wang, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[10] J. Pei, L. Deng, S. Song, M. Zhao, Y. Zhang, S. Wu, G. Wang, Z. Zou, Z. Wu, W. He, F. Chen, N. Deng, S. Wu, Y. Wang, Y. Wu, Z. Yang, C. Ma, G. Li, W. Han, H. Li, H. Wu, R. Zhao, Y. Xie, and L. Shi, "Towards artificial general intelligence with hybrid tianjic chip architecture," *Nat.*, vol. 572, pp. 106–111, 2019.

[11] T. Bu, J. Ding, Z. Yu, and T. Huang, "Optimized potential initialization for low-latency spiking neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 11–20.

[12] 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, 2019.

[13] X. Lin, T. Hu, and X. Wang, "One-pass online learning based on gradient descent for multilayer spiking neural networks," *IEEE Trans. Cogn. Dev. Syst.*, vol. 15, no. 1, pp. 16–31, 2023.

[14] Q. Meng, M. Xiao, S. Yan, Y. Wang, Z. Lin, and Z. Luo, "Training high-performance low-latency spiking neural networks by differentiation on spike representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognition*, 2022, pp. 12 434–12 443.

[15] X. Zhu, B. Zhao, D. Ma, and H. Tang, "An efficient learning algorithm for direct training deep spiking neural networks," *IEEE Trans. Cogn. Dev. Syst.*, vol. 14, no. 3, pp. 847–856, 2022.

[16] Y. Wang, Y. Xu, R. Yan, and H. Tang, "Deep spiking neural networks with binary weights for object recognition," *IEEE Trans. Cogn. Dev. Syst.*, vol. 13, no. 3, pp. 514–523, 2021.

[17] T. Bu, W. Fang, J. Ding, P. Dai, Z. Yu, and T. Huang, "Optimal ANN-SNN conversion for high-accuracy and ultra-low-latency spiking neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2022.

[18] F. Liu, W. Zhao, Y. Chen, Z. Wang, and L. Jiang, "SpikeConverter: An efficient conversion framework zipping the gap between artificial neural networks and spiking neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 1692–1701.

[19] Y. Li, S. Deng, X. Dong, R. Gong, and S. Gu, "A free lunch from ANN: Towards efficient, accurate spiking neural networks calibration," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 6316–6325.

[20] 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, 2015.

[21] P. U. Diehl, D. Neil, J. Binas, M. Cook, S. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *Proc. IEEE Int. Joint Conf. Neural Networks*, 2015, pp. 1–8.

[22] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Front. Neurosci.*, vol. 11, p. 682, 2017.

[23] A. Sengupta, Y. Ye, R. Y. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: VGG and residual architectures," *Front. Neurosci.*, vol. 13, p. 95, 2018.

[24] S. J. Kim, S. Park, B. Na, and S. Yoon, "Spiking-YOLO: Spiking neural network for energy-efficient object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11 270–11 277.

[25] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13 555–13 564.

[26] A. Zhang, Y. Gao, Y. Niu, X. Li, and Q. Chen, "Intrinsic plasticity for online unsupervised learning based on soft-reset spiking neuron model," *IEEE Trans. Cogn. Dev. Syst.*, vol. 15, no. 2, pp. 337–347, 2023.

[27] S. Deng and S. Gu, "Optimal conversion of conventional artificial neural networks to spiking neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2021.

[28] Y. Li and Y. Zeng, "Efficient and accurate conversion of spiking neural network with burst spikes," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 2485–2491.

[29] S. Park, S. J. Kim, H. Choe, and S. Yoon, "Fast and efficient information transmission with burst spikes in deep spiking neural networks," in *Proc. ACM Annu. Design Autom. Conf.*, 2019, p. 53.

[30] Y. Wang, M. Zhang, Y. Chen, and H. Qu, "Signed neuron with memory: Towards simple, accurate and high-efficient ANN-SNN conversion," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 2501–2508.

[31] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.

[32] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognition*, 2017, pp. 4724–4733.

[33] A. Diba, M. Fayyaz, V. Sharma, M. M. Arzani, R. Yousefzadeh, J. Gall, and L. Van Gool, "Spatio-temporal channel correlation networks for action classification," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 299–315.

[34] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6201–6210.

[35] Z. Tong, Y. Song, J. Wang, and L. Wang, "VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training," in *Adv. Neural Inf. Process. Syst.*, 2022.

[36] C. Feichtenhofer, H. Fan, Y. Li, and K. He, "Masked autoencoders as spatiotemporal learners," in *Adv. Neural Inf. Process. Syst.*, 2022.

[37] B. Rueckauer, I. Lungu, Y. Hu, and M. Pfeiffer, "Theory and tools for the conversion of analog to spiking convolutional neural networks," *arXiv:1612.04052*, 2016.

[38] Z. Yan, J. Zhou, and W. Wong, "Near lossless transfer learning for spiking neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 10 577–10 584.

[39] 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.*, 2023, pp. 11–21.

[40] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Stat.*, pp. 400–407, 1951.

[41] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *arXiv:1212.0402*, 2012.

[42] H. Kuehne, H. Jhuang, E. Garrote, T. A. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2011, pp. 2556–2563.

[43] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6546–6555.

[44] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The kinetics human action video dataset," *arXiv:1705.06950*, 2017.

[45] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.

[46] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4724–4733.

[47] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?" in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 813–824.

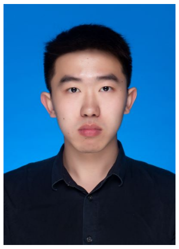
[48] I. Garg, S. S. Chowdhury, and K. Roy, "DCT-SNN: Using DCT to distribute spatial information over time for low-latency spiking neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 4651–4660.

[49] Y. Kim, H. Park, A. Moitra, A. Bhattacharjee, Y. Venkatesha, and P. Panda, "Rate coding or direct coding: Which one is better for accurate, robust, and energy-efficient spiking neural networks?" in *Proc. IEEE Int. Conf. Acoustics Speech Signal Process.*, 2022, pp. 71–75.

[50] J. Ding, Z. Yu, Y. Tian, and T. Huang, "Optimal ANN-SNN conversion for fast and accurate inference in deep spiking neural networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 2328–2336.

[51] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognition*, 2016, pp. 770–778.

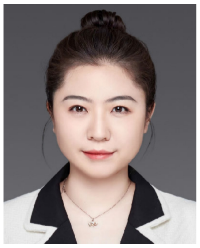


Hong You received the Bachelor's degree from Wuhan University of Science and Technology in 2020. He is now pursuing the Master's degree at Wuhan University of Technology. His current research interests include learning algorithms in spiking neural networks and action recognition.

Xian Zhong (Member, IEEE) received Bachelor's degree in computer science from Wuhan University in 2007, and Ph.D. degree in computer science from Huazhong University of Science and Technology in 2013. He is currently a Professor with the School of Computer Science and Artificial Intelligence, Wuhan University of Technology. He served as a Visiting Scholar with the Peking University from 2021 to 2022. And he is currently a Visiting Scholar with the ROSE@EEE, Nanyang Technological University. His research interests include cross-modality analytics, image processing, action recognition, and neuromorphic computing.

Wenxuan Liu received the M.S. degree in computer science and technology from the Wuhan University of Technology, Wuhan, in 2019, where she is currently pursuing the Ph.D. degree in computer science. Her research interests include action recognition and multimedia content analysis.

Qi Wei received Bachelor's degree from Wuhan University of Science and Technology in 2020 and Master's degree from Shandong University in 2023. Now he is a Research Assistant in Nanyang Technological University. His research interests include machine learning and data mining.



Wenxin Huang (Member, IEEE) received the Bachelor's degree from EVTEK, Helsinki, in 2010, the Master's degree from Aalto University, Helsinki, in 2012, and the Ph.D. degree in computer science from Wuhan University, Wuhan, in 2020. She is currently a Lecturer with Hubei University and a Research Officer with the Centre for Frontier AI Research, Agency for Science, Technology and Research (A*STAR). Her research interests include multimedia content analysis and retrieval, large-scale multimedia data mining, and artificial intelligence.

Zhaofei Yu (Member, IEEE) received the BS degree from the Hong Shen Honors School, College of Optoelectronic Engineering, Chongqing University, Chongqing, China, in 2012, and the Ph.D. degree from the Automation Department, Tsinghua University, Beijing, China, in 2017. He is currently an Assistant Professor with the Institute for Artificial Intelligence, Peking University, Beijing. His current research interests include artificial intelligence, brain-inspired computing, and computational neuroscience.

Tiejun Huang (Senior Member, IEEE) received the Bachelor's and Master's degrees in computer science from the Wuhan University of Technology, Wuhan, China, in 1992 and 1995, respectively, and the Ph.D. degree in pattern recognition and intelligent system from the Huazhong (Central China) University of Science and Technology, Wuhan, in 1998. He is currently a Professor with the School of Computer Science, Peking University, Beijing, China. His research interests include video coding, image understanding, digital right management, and digital library. He has authored or coauthored over 100 peer-reviewed papers and three books. He is a member of the Board of Director for Digital Media Project, the Advisory Board of the IEEE Computing Society, and the Board of the Chinese Institute of Electronics.