SVFormer: A Direct Training Spiking Transformer for Efficient Video Action Recognition

Liutao Yu[1], Liwei Huang[1,2], Chenlin Zhou[1], Han Zhang[1,3], Zhengyu Ma[1*], Huihui Zhou[1*], and Yonghong Tian[1,2]

[1] AI Department, Peng Cheng Laboratory, Shenzhen, China
{yult, zhoucl, mazhy, zhouhh}@pcl.ac.cn
[2] National Key Laboratory for Multimedia Information Processing, School of Computer Science, Peking University, Beijing, China
huanglw20@stu.pku.edu.cn, yhtian@pku.edu.cn
[3] Faculty of Computing, Harbin Institute of Technology, Harbin, China
23B303002@stu.hit.edu.cn

Abstract. Video action recognition (VAR) plays crucial roles in various domains such as surveillance, healthcare, and industrial automation, making it highly significant for the society. Consequently, it has long been a research spot in the computer vision field. As artificial neural networks (ANNs) are flourishing, convolution neural networks (CNNs), including 2D-CNNs and 3D-CNNs, as well as variants of the vision transformer (ViT), have shown impressive performance on VAR. However, they usually demand huge computational cost due to the large data volume and heavy information redundancy introduced by the temporal dimension. To address this challenge, some researchers have turned to brain-inspired spiking neural networks (SNNs), such as recurrent SNNs and ANN-converted SNNs, leveraging their inherent temporal dynamics and energy efficiency. Yet, current SNNs for VAR also encounter limitations, such as nontrivial input preprocessing, intricate network construction/training, and the need for repetitive processing of the same video clip, hindering their practical deployment. In this study, we innovatively propose the directly trained SVFormer (Spiking Video transFormer) for VAR. SVFormer integrates local feature extraction, global self-attention, and the intrinsic dynamics, sparsity, and spike-driven nature of SNNs, to efficiently and effectively extract spatio-temporal features. We evaluate SVFormer on two RGB datasets (UCF101, NTU-RGBD60) and one neuromorphic dataset (DVS128-Gesture), demonstrating comparable performance to the mainstream models in a more efficient way. Notably, SVFormer achieves a top-1 accuracy of 84.03% with ultra-low power consumption (21 mJ/video) on UCF101, which is state-of-the-art among directly trained deep SNNs, showcasing significant advantages over prior models.

Keywords: Video action recognition · Spiking transformer · SVFormer · Direct training · Energy efficiency.

---

[*] Corresponding author

1 Introduction

Video is becoming a prevalent and indispensable medium to convey information in daily life, which captures movements, actions, and events over time. Video action recognition (VAR) is an important aspect of video understanding, focusing on automatically identifying actions or activities from videos. This capability is invaluable for various applications, including surveillance, healthcare, entertainment, sports, education, industrial automation, and beyond. Nevertheless, video processing presents greater challenges compared to images, given the necessity to model temporal dynamics within large data volume and heavy information redundancy introduced by the temporal dimension.

As artificial neural networks (ANNs) have shown great success in various computer vision tasks, a lot of studies based on convolutional neural networks (CNNs) [9] or vision transformers (ViTs) [46] have emerged for VAR in recent years. In the early stages, CNNs are the mainstream approaches to VAR, including 2D-CNNs [56,68,13] and 3D-CNNs [52,7,22]. The decomposition of 3D-CNNs [42,60,53], as well as the combination of 2D-CNNs and 3D-CNNs [17], are adopted to reduce the computation cost. However, CNNs struggle to learn long-range dependency between patches, due to the inductive bias of convolution. To overcome this challenge, network models based on ViTs are becoming popular and showing good performance for VAR in recent years [46], including TimeSformer [3], ViViT [2], MViT [12], video swin transformer [35], and so on. To further improve the performance, methods combining convolution and self-attention [36,33,34], or using self-supervised pretraining [51,55,59] are proposed. Nevertheless, the huge computation cost of current ANNs for VAR still limits their practical deployment, especially in power-constrained situations.

The brain-inspired spiking neural networks (SNNs) have garnered significant attention for their potential in temporal information processing and energy efficiency [44], thus are ideal candidates for processing videos. Previous studies show that SNNs exhibit good performance on various tasks, such as object recognition/detection/tracking, robotics control and so on [61,20,70]. In recent years, recurrent SNNs (RSNNs) [41,8] and ANN-converted SNNs [67,65] have been applied to VAR. However, they also encounter limitations in practical application: RSNNs usually need nontrivial input preprocessing and network construction/training, as well as long simulation steps [41,8]; ANN-converted SNNs are based on well-trained ANNs, and need to process the same video clip several times to accomplish the task [67,65].

In this study, we innovatively propose the directly trained SVFormer (Spiking Video transFormer) for VAR, aiming to address the challenges outlined above. SVFormer processes a video clip frame-by-frame without the need for complex input processing, and can be trained end-to-end through the surrogate gradient method, enabling straightforward incremental learning and facilitating practical deployment. SVFormer integrates local feature extraction, global self-attention, and the intrinsic dynamics, sparsity, and spike-driven nature of SNNs, to efficiently and effectively extract spatio-temporal features. Besides, we incorporate parametric LIF neuron [16], a local-global-fusion operation and a novel time-dependent batch normalization into SVFormer, contributing to its good performance. We first evaluate SVFormer on the classical UCF101 dataset, and achieve a top-1 accuracy of 84.03% with ultra-low power consumption (21 mJ/video), which is state-of-the-art among directly trained deep SNNs, showing

significant advantages over previous models. To validate its generalizability, SVFormer is then evaluated on a larger RGB datasets (NTU-RGBD60) and a neuromorphic dataset (DVS128-Gesture), both demonstrating comparable performance to the mainstream models in a more efficient way. These results showcase that the directly trained SVFormer is an effective and efficient model for VAR.

2 Related work

2.1 ANNs for VAR

In the early stages, CNNs including 2D-CNNs and 3D-CNNs, are the mainstream approaches to VAR [9]. Some studies applied two-stream models to process RGB frames and optical flows in two separate CNNs, with a late fusion operation in deeper layers [27,10,18,19]. Another line of approach adopts 2D-CNNs to extract frame-level features, and then models temporal information of the input sequence in various ways, such as the consensus module in TSN [56], the bag of features in TRN [68], the pointwise convolutions across frames in TAM [13], and so on. Moreover, to better model the spatio-temporal features, many variants of 3D-CNNs have been introduced, such as C3D [52], I3D [7] and ResNet3D [22]. To mitigate the high computational burden brought by 3D convolutions, some studies tried to decompose the 3D convolution into 2D spatial convolution and 1D temporal convolution, such as P3D [42], S3D [60] and R(2+1)D [53]; or to use a combination of 2D-CNNs and 3D-CNNs, such as SlowFast [17]. However, CNNs mainly extract local features due to limited receptive fields, ignoring long-range dependency across patches.

Recently, ViTs outperform CNNs in many visual tasks due to their enhanced ability to capture long-range dependencies [11,21]. Some researchers proposed different variants based on ViTs for VAR [46]. The classical TimeSformer enables spatio-temporal feature extraction directly from a sequence of frame-level patches, and explores different space-time self-attention schemes [3]. ViViT is a pure-transformer based model, which factorizes the spatial- and temporal-dimensions of the input to handle the long sequences of tokens [2]. To reduce the computation cost, Fan et al. introduced a multiscale pyramid of features and a pooling self-attention mechanism in MViT [12]; and Liu et al. proposed an inductive bias of locality with shifted window-based self-attention in a video swin transformer [35]. Besides, the combination of convolution and self-attention is also adopted for efficiently learning video representations [36,33,34]. Furthermore, assisted with large-scale self-supervised pretraining, great improvements have been achieved for VAR, such as in VideoMAE [51,55] and MaskFeat [59]. Although current models have realized high classification accuracy for VAR, the excessive data volume and heavy information redundancy introduced by the temporal dimension limit their practical deployment in power-constrained situations.

2.2 SNNs for VAR

Considering the intrinsic dynamics and energy efficiency of SNNs, they are ideal candidates for VAR and have been attempted in some recent work [41,8,57,67,65].

Panda and Srinivasa developed a Driven-Autonomous based reservoir recurrent SNN to recognize video actions from limited training examples, which is not trivial to train, demonstrating a top-1 accuracy of 81.3% on the UCF101 dataset with pre-extracted multi-scan spike sequences of 300 time steps as input to the model [41]. Further, Chakraborty and Mukhopadhyay presented a heterogeneous recurrent SNN (HRSNN) with unsupervised learning for VAR on several RGB datasets (KTH, 94.32%; UCF101, 77.53%) and one neuromorphic dataset (DVS128-Gesture, 96.54%) [8], using a similar input preprocessing method as [41] and a nontrivial initialization method. Wang et al. progressively trained a two-stream hybrid network (TSRNN) consisting of CNN, RNN, and a novel spiking module, where spiking signals correct the memory of RNN, achieving competitive performance on UCF101 (94.4%) and HMDB51 (69.9%) with both RGB frames and optical flows as input [57]. Zhang et al. constructed a two-stream deep recurrent SNN model through a hybrid ANN-to-SNN conversion method combining channel-wise normalization and tandem learning, obtaining an accuracy of 88.46% on the UCF101 dataset with 200 time steps [67]. Recently, You et al. proposed an improved ANN-to-SNN conversion framework (scalable dual threshold mapping) to mitigate three types of conversion errors (unevenness error, clipping error and quantization error), and obtained ultra-low-latency SNNs based on the SlowFast model backbone, achieving high accuracy on UCF101 (92.94%) and HMDB51 (67.71%) with carefully selected hyperparameters [65]. We observe that current SNNs for VAR have limitations for incremental training and practical application, such as complicated model construction/conversion, nontrivial input preprocessing, and long simulation time steps. Therefore, we directly train a novel spiking transformer for efficient and effective VAR in this paper, which is more suitable for practical deployment.

3   Preliminary and Methodology

In this section, we introduce the proposed SVFormer in detail. Firstly, we briefly introduce the basics of SNNs. Then, we describe the overview of SVFormer and explain the key components such as patch embedding (PE) module, local feature extractor (LFE), global self-attention (GSA) module, local pathway (LP), classification head (CH) and so on. Moreover, we introduce the method to calculate theoretical energy consumption for model inference.

3.1   Spiking neural networks

The brain-inspired SNNs are regarded as the third generation of neural networks [37], which innately exhibit temporal dynamics and communicate through binary spikes/events, attracting considerable attention and achieving significant advancements in recent years. SNNs offer powerful computation capability in virtue of their biological plausibility, temporal processing property with intrinsic dynamics, and energy efficiency with event-driven nature, thus considered as promising alternatives to conventional ANNs [44]. Specifically, the communication through sparse binary spikes allow SNNs to adopt low-power accumulate (AC) operations in convolution or linear layers, in place
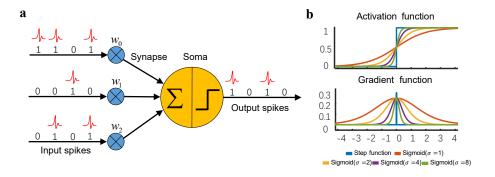
Fig. 1. (a) The scheme of a spiking neuron, of which the input and output are both binary spikes. (b) Sigmoid function approximates the Heaviside activation function of a spiking neuron, and the derivative of it can be utilized to calculate gradients during backpropagation.

of power-hungry multiply-and-accumulate (MAC) operations when implemented on neuromorphic hardware, leading to high energy efficiency [40,30,29,64].

There are mainly two methods to obtain well-performing deep SNNs: ANN-to-SNN conversion and direct training through surrogate gradients. In ANN-to-SNN conversion, a pretrained ANN is converted to an SNN by replacing the ReLU activation layers with

The basic units of SNNs are spiking neurons, like ReLUs in ANNs. Leaky Integrate-and-Fire (LIF) neuron model (Fig. 1a) is one of the most commonly adopted neuron models in SNNs [15,71,63,69,66]. The dynamics of a LIF neuron are described as:

$$H[t] = V[t-1] + \frac{1}{\tau}\left(X[t] - (V[t-1] - V_{reset})\right), \tag{1}$$

$$S[t] = \Theta\left(H[t] - V_{th}\right), \tag{2}$$

$$V[t] = H[t]\left(1 - S[t]\right) + V_{reset}S[t], \tag{3}$$

where τ is the membrane time constant, $X[t]$ is the input current at time step t, V_{reset} is the reset potential, V_{th} is the firing threshold. Eq. (1) describes the update of membrane potential. Eq. (2) describes the spike generation process, where $\Theta(v)$ is the Heaviside step function: if $H[t] \geq V_{th}$ then $\Theta(v) = 1$, meaning a spike is generated; otherwise $\Theta(v) = 0$. $S[t]$ represents whether a neuron fires a spike at time step t. Eq. (3) describes the resetting process of membrane potential, where $H[t]$ and $V[t]$ represent the membrane potential before and after the evaluation of spike generation at time step t, respectively. To improve the temporal representation ability of spiking neurons, trainable parameters are incorporated. For example, inspired by heterogeneous neurons in the brain, Fang et al. proposed Parametric LIF (PLIF) neuron [16] by using trainable membrane time constant as follows:

$$H[t] = V[t-1] + k\left(a\right)\left(X[t] - (V[t-1] - V_{reset})\right), \tag{4}$$

where $k\left(a\right) = \frac{1}{1+exp(-a)} \in (0,1)$, $\tau = \frac{1}{k(a)}$, and a is the trainable parameter.

spiking neurons and using scaling operations like weight normalization and threshold balancing [6,25,45,5,38,58]. To mitigate conversion error, the converted SNNs usually suffer from long simulation time steps, which causes high computational cost in practice. In direct training, SNNs are unfolded over the temporal dimension like RNNs and trained with backpropagation through time [32,48,39]. Due to the non-differentiability of the spike generation process, the surrogate gradient method is employed for backpropagation [39,31,15,16]. Specifically, the forward propagation utilizes Heaviside step function to generate spikes (Eq. (2)), which can be approximated by differentiable functions like sigmoid and arctan functions, and the derivative of them are adopted for gradient calculation during backpropagation. Fig. 1b illustrates the application of sigmoid function to calculate back-propagated gradients. Direct training with surrogate gradients achieves good performance with few time steps, especially on image classification tasks, thus greatly promotes the development of deep SNNs [15,16,71,63,20,62,70].

3.2   The proposed SVFormer

Overview The overall framework of our proposed SVFormer is shown in Fig. 2a. Inspired from the biological brains, efficient and effective multiscale hierarchical modular structures have been widely adopted in DNNs, showing great potential for various tasks [28,26,33,62]. Thus, the backbone of SVFormer is strategically structured in a hierarchical manner, comprising four stages and one local pathway, if not specified. Each of the first two stages consists of one patch embedding (PE) module and several local feature extractors (LFEs), while each of the last two stages consists of one PE module and several global self-attention (GSA) modules. The number of LFEs or GSAs in each stage is indicated as S_i, and the number of channels (or embedding dimension) for each stage is represented as C_i. If not specified, $S = [1, 1, 3, 1]$, meaning that there are one LFE in the first two stages, three GSAs in the third stage, and one GSA in the last stage; and $C = [128, 256, 384, 512]$, meaning that the embedding dimensions for four stages are 128, 256, 384, and 512, respectively. Given a batch of video sequences $I \in \mathbb{R}^{B \times T \times 3 \times H_{in} \times W_{in}}$, T is the length of one video sequence (usually a temporal downsampling of the original video sample), B is the batch size, and H_{in}/W_{in} is the height/width of the video frame. Firstly, we reshape the input into $I \in \mathbb{R}^{T \times B \times 3 \times H_{in} \times W_{in}}$, and feed one video frame into SVFormer in each time step, where T is also the total simulation time steps of SVFormer. After the backbone extracts spatio-temporal information from the video input, the classification head (CH) makes a decision about the action category of the input.

Patch embedding module The PE modules play roles for feature extraction, channel dimension expansion, and patch embedding, raising channel dimension of the feature map, while downsampling the feature map in spatial dimension. The first PE module consists of a convolution (Conv) layer and a batch-normalization (BN) layer, to extract features and spatially downsample the input frame before encoding them into spikes. Other PE modules consist of a spiking neuron (SN) layer, a Conv layer, and a BN layer. For the sake of brevity, the combination of a Conv layer and a BN layer is termed as ConvBN. The computation in the last three PE modules can be expressed as
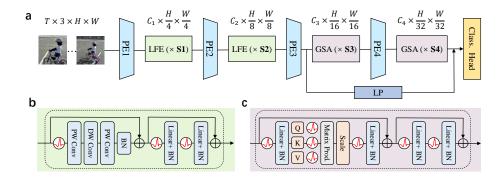
Fig. 2. (a) The overall structure of the hierarchical SVFormer, which includes four stages and one local pathway in default. (b) Structure of one local feature extractor (LFE). (c) Strcuture of one global self-attention (GSA) module.

$X_{PEout} = ConvBN(SN(X_{PEin}))$, and the SN operation is omitted in the first PE module as previously described.

Local feature extractor As shown in Fig. 2b, one LFE is composed of three cascaded Conv layers and a multi-layer perception (MLP) module, designed for extracting local spatial features. Both parts adopt residual learning with membrane shortcut [62] to avoid gradient vanishing and to improve the model performance. To reduce the number of parameters, we adopt the style of PWConv-DWConv-PWConv like the MobileNet block [24], where PWConv (DWConv) represents pointwise (depthwise) convolution. PWConv is responsible for fusing information across the channel dimension, while DWConv with a relatively large kernel size (5×5 if not specified) for extracting spatial information in each channel. Besides, there is an SN layer ahead of the three Conv layers to transform the feature map into spikes. The MLP module consists of two SN-Linear-BN motifs as usual, which first expands the channel dimension by a ratio ($r = 2$ if not specified) and then reduces it back to the original size, improving the model's representation capability. The combination of a Linear layer and a BN layer is termed as LinearBN. The computation in a LFE can be expressed as: $X = X_{LFEin} + BN(PWConv(DWConv(PWConv(SN(X_{LFEin})))))$, and $X_{LFEout} = X + MLP(X)$, where $MLP(X) = LinearBN(SN(LinearBN(SN(X))))$.

Global self-attention module As shown in Fig. 2c, a GSA module is composed of one spiking self-attention (SSA) module and one MLP module. Here, we adopt the SSA module which has been proven effective and efficient in [71], and use the same MLP module as that in a LFE. The computation in an SSA module is formulated as follows:

$$A = SN_A(LinearBN_A(SN(X_{SSAin})), \quad A \in \{Q, K, V\}, \quad X_{SSAin} \in \mathbb{R}^{T \times B \times N \times C}, \quad (5)$$

$$X_{SSAout} = X_{SSAin} + LinearBN(SN(QK^T V * s)), \quad X_{SSAout} \in \mathbb{R}^{T \times B \times N \times C}. \quad (6)$$

In an SSA module, an SN layer first transforms the input into spikes, then three parallel Linear-BN-SN motifs generate the spike-form Q, K, and V tensor, based on which the self-attention score $QK^TV * s$ is calculated through energy-efficient AC operations, where $s = 1/\sqrt{64} = 0.125$ is a predefined scaling factor. The self-attention score is then transformed by a Linear-BN-SN motif before being added to X_{SSAin}. In the above two equations, N is the number of tokens, which is the multiplication between the height H and width W of the current feature map, i.e. $N = H * W$. Taken together, the computation in a GSA module can be expressed as: $X_{GSAout} = X_{SSAout} + MLP(X_{SSAout})$, where X_{SSAout} is calculated through Eq. (5) and (6).

Local pathway To better utilize the features of different scales, we add a local pathway (LP) after the third PE module, and the output of the local pathway will be fused with the output of the last GSA module, as shown in Fig. 2a. The local pathway consists of cascaded SN-DWConv-PWConv-BN layers to further extract local spatial feature in an economical way regarding parameters, and outputs a tensor with the same shape as the output of the last GSA module, which will then be concatenated along the channel dimension before sent into the classification head. The computation of the local pathway can be expressed as $LP(X) = BN(PWConv(DWConv(SN(X))))$, where X means the input of the local pathway.

Classification head To make the most of the extracted spatio-temporal features from the backbone, we adopt a classification head (CH) similar to that in [66], which makes a learnable weighted sum of the feature map across temporal and spatial dimension, rather than simply averaging across them. Specifically, for the fused input $X_{CHin} \in \mathbb{R}^{T \times B \times 2C_4 \times H_{out} \times W_{out}}$, we first use an SN layer to convert the feature map into spikes, and reshape it into $X_{CHinS} \in \mathbb{R}^{B \times 2C_4 \times T \times H_{out} \times W_{out}}$. Then, we operate X_{CHinS} with a depthwise 3D convolution layer and a BN layer, i.e. $X_{STF} = 3DConvBN(X_{CHinS})$, which only slightly increases the number of parameters. Further, $X_{STF} \in \mathbb{R}^{B \times 2C_4 \times 1 \times 1 \times 1}$ will be squeezed and operated with a linear layer to generate the classification results. Taken together, the computation in the classification head can be formulated as $Y = CH(X_{CHin}) = Linear(3DConvBN(SN(X_{CHin})))$, where the reshape and squeeze operations are omitted for brevity.

Time-dependent batch normalization For the sake of efficiency, a BN layer in an SNN is usually executed in a parallel way, because the calculation is not time-dependent. The common approach is to reshape the input $X \in \mathbb{R}^{T \times B \times C \times H \times W}$ into $X \in \mathbb{R}^{TB \times C \times H \times W}$, and adopt $nn.BatchNorm2d$ to implement batch normalization [71,69,66]. Obviously, this approach utilizes the feature maps of all time steps equally and simultaneously, which is unreasonable because one cannot use future information for calculation in the current time step. Therefore, we proposed the novel time-dependent batch normalization (TDBN) method here. Specifically, we reshape the input into $X \in \mathbb{R}^{T \times BC \times H \times W}$, and adopt $nn.BatchNorm2d$ to implement batch normalization, which treats each time step as a channel and thus independently utilizes information from different time steps.

Overall computation process Based on the above contents, the computation of SV-Former can be summarized as follows.

$$X_{PE_i out} = PE_i(X_{PE_i in}), \quad i \in \{1, 2, 3, 4\} \tag{7}$$

$$X_{LFE_{i,j} out} = LFE_{i,j}(X_{LFE_{i,j} in}), \quad i \in \{1, 2\}, \quad j \in \{1, .., S_i\} \tag{8}$$

$$X_{GSA_{i,j} out} = GSA_{i,j}(X_{GSA_{i,j} in}), \quad i \in \{3, 4\}, \quad j \in \{1, .., S_i\} \tag{9}$$

$$X_{CHin} = Concate(X_{GSA_{4,S_4} out}, LP(X_{PE_3 out})) \tag{10}$$

$$Y = CH(X_{CHin}), \quad Y \in \mathbb{R}^{B \times \#cls} \tag{11}$$

In the above equations, i indicates the stage index, H and W are the height and width of the intermediate feature maps, and $\#cls$ is the number of categories. PE, LFE, GSA, LP and CH represent the computation of patch embedding module, local feature extractor, global self-attention module, local pathway and classification head, respectively. The input shapes of PE, LFE, GSA, LP and CH are $X_{PE_i in} \in \mathbb{R}^{T \times B \times C_{i-1} \times H \times W}$, $X_{LFE_{i,j} in} \in \mathbb{R}^{T \times B \times C_i \times H \times W}$, $X_{GSA_{i,j} in} \in \mathbb{R}^{T \times B \times C_i \times H \times W}$, $X_{PE_3 out} \in \mathbb{R}^{T \times B \times C_3 \times H \times W}$, and $X_{CHin} \in \mathbb{R}^{T \times B \times 2C_4 \times H_{out} \times W_{out}}$, respectively. The input to the first PE module is the reshaped video frames, i.e. $X_{PE_1 in} = I \in \mathbb{R}^{T \times B \times 3 \times H_{in} \times W_{in}}$. Besides, it should be noted that both the LFE and GSA module do not change the shape of the feature map.

3.3 Theoretical calculation of energy consumption

The theoretical energy consumption of an SNN is usually calculated through multiplication between the number of MAC/AC operations and the energy consumption of each operation on predefined hardware [40,71,63,66]. The number of synaptic operations (SOPs) are calculated as follows:

$$SOP^l = fr^{l-1} \times FLOP^l \tag{12}$$

where fr^{l-1} is the firing rate of spiking neuron layer $l - 1$. $FLOP^l$ refers to the number of floating-point MAC operations (FLOPs) of layer l, and SOP^l is the number of spike-based AC operations (SOPs). Assuming the MAC and AC operations are performed on the 45nm hardware [23], i.e. $E_{MAC} = 4.6pJ$ and $E_{AC} = 0.9pJ$, the energy consumption of SVFormer can be calculated as follows:

$$E_{SVFormer} = E_{AC} \times \left(\sum_{i=2}^{N} SOP_{Conv/LN}^i + \sum_{j=1}^{M} SOP_{SSA}^j \right) + E_{MAC} \times \left(FLOP_{Conv}^1 \right). \tag{13}$$

$FLOP_{Conv}^1$ represents the FLOPs of the first layer before encoding input frames into spikes, $SOP_{Conv/LN}$ represents the SOPs of a convolution or linear layer, and SOP_{SSA} represents the SOPs of an SSA module. N is the total number of convolution layers and linear layers, and M is the number of SSA modules. During model inference, several cascaded linear operation layers such as convolution, linear and BN layers, can be fused into one single linear operation layer [66,54], still enjoying the AC-type operations with a spike-form input tensor.

4   Experimental Results

In this section, we evaluate the performance of SVFormer on two RGB video datasets (UCF101 [50] and NTU-RGBD60 [47]), as well as a neuromorphic dataset (DVS128-Gesture [1]). We directly train the proposed SVFormer from scratch based on the surrogate gradient method using SpikingJelly [14], which is a popular deep learning framework for building and training SNNs. We compare the performance and energy consumption of SVFormer with existing SNNs or conventional ANNs to demonstrate its effectiveness and efficiency. Besides, we conduct ablation studies to show the effects of some network modules or simulation setups.

4.1   Datasets and Experimental Setup

UCF101 includes 101 action classes with a total of 13,320 videos, which were collected from YouTube [50]. These YouTube videos are recorded in unconstrained environments with cluttered background, camera motion, various illumination conditions, and beyond. These videos have a frame rate of 25 fps and a spatial resolution of 320×240. The length of each video sample ranges from 1.06s to 71.04s, of which the average is 7.21s. In this study, we adopt the first official training-testing split.

NTU-RGBD60 contains 60 kinds of actions with a total of 56,880 samples collected with three different camera angles, which include depth, 3D skeleton, RGB and infrared sequences [47]. This study only adopts RGB videos for action recognition. Both cross-subject (C-Subject) and cross-view (C-View) splits are evaluated in this study. The C-Subject split divides the training set and testing set according to the person ID, while the C-View split divides the samples by the camera ID, as in [47].

DVS128-Gesture contains 11 different classes of gestures collected from 29 individuals under 3 different illumination conditions, which is a neuromorphic dataset collected by dynamic vision sensors [1]. The spatial resolution of DVS128-Gesture is 128×128. We first integrate the stream of events into a sequence of frames as the model input, as usually done [16,71,63].

For both RGB video datasets, the input size of the network is 224×224 in both training and testing phase. Without specification, the batch size is 64, distributed across 4 Nvidia V100 GPUs. The number of training epochs is 600, with a warming-up-then-cosine-decay schedule of learning rate, of which the base value is set empirically to 0.006. For the DVS128-Gesture dataset, the input size of the network is 128×128. We applied a 3-stage model for this small dataset. The batch size is set to 16, and the number of training epochs is 600. The learning rate is set empirically to 0.005 and decayed with a cosine schedule. For all three datasets, AdamW is applied as the optimizer. Moreover, the implementation is based on the SlowFast repository [17] and Uniformer repository [33], and common data augmentation methods like crop, flip, and random erase are applied in the training phase.

4.2   Comparison Results

UCF101 is a popular dataset in the VAR field, which is also commonly applied when using SNNs for VAR. Hence, we evaluate the proposed SVFormer on UCF101

and compare the performance to previous SNNs. The results are listed in Tab. 1. The SVFormer-base ($S = [1, 1, 3, 1]$, $C = [128, 256, 384, 512]$) is the default network introduced in Section 3.2, achieving a top-1 accuracy of 84.03% on the UCF101 dataset, which is state-of-the-art among directly trained deep SNNs for VAR. And we evaluate four modifications of the base model: the shallower SVFormer-ss ($S = [1, 1, 2, 1]$, $C = [128, 256, 384, 512]$), the thinner SVFormer-st ($S = [1, 1, 3, 1]$, $C = [64, 128, 256, 512]$), the deeper SVFormer-dp ($S = [1, 2, 4, 2]$, $C = [128, 256, 384, 512]$) and the wider SVFormer-wd ($S = [1, 1, 3, 1]$, $C = [128, 256, 512, 768]$). All these modifications demonstrate lower accuracy compared to the base model. Three listed recurrent SNNs need nontrivial input preprocessing and 300 simulation time steps to perform the task [41,8], which is unsuitable for practical deployment. The recent ANN-converted SNN, SlowFast-SDM-cv [65], only needs four simulation time steps to achieve a comparable accuracy (92.94%) to its ANN compartment, which is critically dependent on the cautious choice of hyperparameters in the conversion process. Besides, it requires a well-trained ANN as basis and need to repeatedly process the same video clip for four times, thus not friendly for incremental training and not economic for deployment in practice. Further, we directly train two recently published well-performing deep SNNs, SGLFormer [66] and Meta-SpikeFormer [62], on the UCF101 dataset, and the results are inferior to the SVFormer-base model.

To validate the generalizability of the proposed SVFormer, we evaluate SVFormer on a large RGB dataset (NTU-RGBD60) and a neuromorphic dataset (DVS128-Gesture). To the best of our knowledge, SVFormer is the first SNN model that has ever been assessed on the NTU-RGBD60 dataset. Hence, we compare it to two recently published well-performing ANNs tested on RGB frames [49,43]. The results in Tab. 1 show that SVFormer's accuracy is slightly lower, which is acceptable for the substantial savings of energy consumption. For DVS128-Gesture, We applied the SVFormer-3stg model with one local stage and two global stages, where $S = [1, 2, 1]$ and $C = [64, 128, 256]$. At the same time, we exclude the local pathway for it. The accuracy of the SVFormer-3stg model is 97.92%, which is comparable to mainstream SNNs but with significantly fewer parameters (Tab. 1).

As described in Section 3.1, a main strength of SNNs is energy efficiency, which can be attributed to the sparsity of spikes and the spike-driven communication in SNNs. In this section, we calculate the average theoretical energy consumption of the SVFormer-base model in inference for one video clip from UCF101, with the method described in Section 3.3. Firstly, we calculate the number of MAC operations (FLOPs) of convolution and linear layers; then, we count the average firing rate of each spiking neuron layer, and convert corresponding MAC operations (FLOPs) to AC operations (SOPs) by Eq. (12); finally, we calculate the theoretical energy consumption by Eq. (13). The average firing rate of each spiking neuron layer is shown in Fig. 3 (a, b), which indicates the sparsity of spikes during model inference. The FLOPs of the SVFormer-base model's ANN counterpart is 229.163G if executed for 16 repetitions (as 16 time steps in the SNN version), of which the energy consumption is 1054.148mJ, where all the spiking neurons are replaced by ReLUs. According to the above method, the remaining FLOPs of the SVFormer-base model is 0.700G, and the SOPs is 20.760G, thus the energy consumption is 21.904mJ. Obviously, the energy cost of SVFormer-base is much lower than its ANN

Table 1. Comparison results of top-1 accuracy on UCF101, DVS128-Gesture and NTU-RGBD60. † indicates our implementations. The best-performing results are highlighted in bold. SVFormer-base is SOTA of directly trained deep SNNs on UCF101.

Dataset	Model	Param (M)	Time Steps	Top-1 Acc (%)
UCF101	RSNN-reservoir-DA [41]	40.40	300	81.30
	RSNN-HeNHeS-STDP [8]	-	300	77.53
	RSNN-HeNB-BP [8]	-	300	84.32
	RSNN2s-tandem-cvt [67]	-	200	88.46
	SlowFast-SDM-cvt [65]	-	4	**92.94**
	SGLFormer-8-384 [66] †	11.76	16	74.70
	Meta-SpikeFormer [62] †	13.81	16	83.66
	SVFormer-base	13.80	16	**84.03**
	SVFormer-ss	12.53	16	83.61
	SVFormer-st	8.77	16	80.15
	SVFormer-dp	17.72	16	80.25
	SVFormer-wd	24.07	16	81.23
DVS128-Gesture	ConvNet-PLIF [16]	-	20	97.57
	RSNN-HeNHeS-STDP [8]	-	100	96.54
	RSNN-HeNB-BP [8]	-	100	98.12
	Spikformer-2-256 [71]	2.57	16	98.30
	Spikingformer-2-256 [69]	2.57	16	98.30
	SD-Transformer-2-256 [63]	2.57	16	**99.30**
	SGLFormer-3-256 [66]	2.17	16	98.60
	SVFormer-3stg	1.88	16	97.92
NTU-RGBD60	DVANet (ANN) [49]	-	-	93.40(CS)/**98.20**(CV)
	π-ViT (ANN) [43]	-	-	**94.00(CS)**/97.90(CV)
	SVFormer-base	13.76	16	88.12(CS)/94.68(CV)

Table 2. Performance of SVFormer-base on the UCF101 dateset under different noise conditions.

Noise condition	Null	Gaussian noise (a)			Salt-and-pepper noise (P)		
	-	0.1	0.5	1	0.1	0.2	0.3
Top1-acc (%)	84.03	82.26	77.13	64.76	75.36	66.32	55.17

counterpart, validating the energy efficiency of SNNs. Further, the ratio of the energy cost of SVFormer-base to that of its ANN counterpart is 1.99% (21:1054), which is much lower than that of SlowFast-SDM-cv (98:128=76.56%) [65], showing advantages of directly trained SNNs compared to ANN-converted ones.

Moreover, to test the robustness of the model, we evaluate the trained SVFormer-base model with noisy frames from the UCF101 dataset. Here, we applied two commonly adopted noise types for images, i.e. Gaussian noise and salt-and-pepper noise. For the Gaussian type, we add noise with zero mean and different level of standard deviation $\sigma = a * \sigma_{ori}$ to the original frame, where σ_{ori} indicates the *std* of the original frame. For the salt-and-pepper noise, we randomly transform a pixel into the highest or lowest value of the current frame with a predefined probability P. Fig. 4 demonstrates the effects of different level of noise with an exemplar frame. The results in Tab. 2 demonstrate that SVFormer exhibits resilience to a moderate level of noise, sustaining commendable performance. However, when confronted with excessive noise that markedly degrades frame quality, there is a substantial decline in the model's performance.
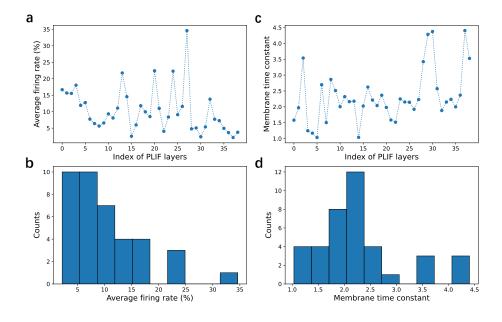

Fig. 3. Average firing rates (a, b) during model inference on the UCF101 dataset, and learned membrane time constants (c, d), for all PLIF layers, showing the trend of both variables as the network goes deeper (a, c), and the corresponding histograms (b, d).
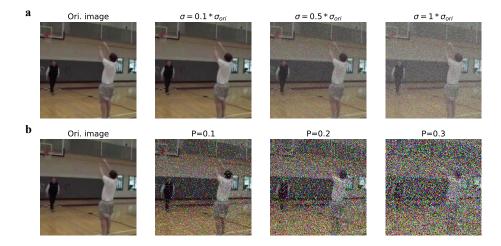
a

b

Fig. 4. (a) An exemplar frame with different level of Gaussian noise ($N(0, \sigma)$). (b) An exemplar frame with different level of salt-and-pepper noise, where P means the sprinkling probability.

4.3   Ablation studies

In this subsection, we conduct some ablation studies based on the UCF101 dataset to show the effects of some network modules or simulation setups. The results are listed in Tab. 3, where batch size are adjusted according to the model size. First, classification accuracy of a SVFormer without the local pathway is 83.29%, which is slightly lower than the original 84.03%, showing that the local pathway improves the model's representational power for VAR. Second, classification accuracy of a SVFormer with LIF neurons is only 80.89%, demonstrating that learnable membrane time constants of spiking neurons (PLIF) enhance the model's temporal processing capability. As shown in Fig. 3 (c, d), the learned membrane time constants are variable, helping to better represent temporal information, which is also consistent with heterogeneous neurons observed in biological brains. Besides, the membrane time constants of the deeper layers are on average larger than those of the shallower layers, implying that deeper layers integrate spatio-temporal information for longer duration. Third, without using time-dependent batch normalization to utilize information from different time steps independently, the model' classification accuracy decrease from 84.03% to 80.65%, indicating the effectiveness of the proposed time-dependent BN. Fourth, we apply different number of frames (i.e. T=8 or T=24) as network input to test the effects of input video length, where T is the number of frames sampled from the original video sample. And the simulation duration of the SNN model is adjusted to 8 or 24 steps accordingly. Both 8-frame and 24-frame inputs exhibit lower accuracies compared to the original 16-frame input, indicating the necessity of finding a suitable input length to balance accuracy and computational cost.

Additionally, when utilizing SNNs to process time sequences such as videos, addressing the alignment of temporal resolution becomes a critical concern that warrants attention. In this work, we mainly adopt the frame-by-frame approach, i.e. the input

Table 3. Ablation studies on the UCF101 dateset.

Architecture	Param (M)	Batch size	Top-1 Acc (%)
SVFormer Base (T=16)	13.80	64	84.03
Base - Local pathway	12.32	64	83.29
Base + (PLIF $\rightarrow$ LIF)	13.80	64	80.89
Base + (BN $\rightarrow$ TDBN)	13.35	64	80.65
Base + (T=8)	12.76	64	81.26
Base + (T=24)	14.84	40	81.81
Base + 3D clip input (16/4)	12.69	56	74.20
Base + 3D clip input (20/4)	12.89	40	74.09

length (number of frames) is equal to the simulation duration (number of time steps) of the SNN model, which processes one frame per time step. Here, we also test the clip-by-clip approach, where the SNN model processes a clip of frames in each time step. Firstly, we need to modify the 2D-Conv and 2D-BN layers in the model into 3D ones accordingly, to process a 3D video clip. Then, we separate a sampled video sequence with 16 or 20 frames into 4 parts uniformly, meaning that the model runs 4 time steps and processes one clip with 4 or 5 frames in each time step. Finally, we train the model from scratch and test its performance. The results in Tab. 3 demonstrate that the frame-by-frame approach is superior to the clip-by-clip approach for the UCF101 dataset.

5   Conclusion

In this paper, we propose the directly trained SVFormer, which integrates local feature extraction, global self-attention, and the intrinsic dynamics, sparsity, and spike-driven nature of SNNs, effectively and efficiently learning spatio-temporal representation for VAR. We evaluate SVFormer on two RGB datasets (UCF101, NTU-RGBD60) and a neuromorphic dataset (DVS128-Gesture). The experimental results demonstrate that SVFormer achieves comparable performance to mainstream models for VAR tasks in a more efficient way. Specifically, SVFormer achieves state-of-the-art (top-1 accuracy of 84.03%) among directly trained deep SNNs on UCF101, with ultra-low power consumption (21 mJ/video). These results verify SVFormer's strong representation capability, showing that the multiscale spatio-temporal feature-extraction characteristic endows it with great potential as a backbone for diverse video tasks when equipped with properly designed task heads. However, there are certain limitations associated with this pioneering endeavor. On the one hand, the recognition accuracy lags behind the state-of-the-art of traditional ANNs. One possible solution is to try large-scale self-supervised pretraining for SNNs, which has already shown great success for ANNs [51,55,59]. On the other hand, although a video clip is processed frame-by-frame in SVFormer, the length of the clip is predefined for the ease of network implementation, meaning that the model needs to process all frames before making a decision. One can try to achieve speed-accuracy

tradeoff in the model as a biological brain [4], i.e. the model may stop at any step if a decision is made based on predefined criteria, which is more flexible and efficient.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Amir, A., Taba, B., Berg, D., Melano, T., McKinstry, J., Di Nolfo, C., Nayak, T., Andreopoulos, A., Garreau, G., Mendoza, M., Kusnitz, J., Debole, M., Esser, S., Delbruck, T., Flickner, M., Modha, D.: A low power, fully event-based gesture recognition system. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7388–7397 (2017)
2. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6836–6846 (2021)
3. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: Proceedings of the 38th International Conference on Machine Learning. vol. 139, pp. 813–824 (2021)
4. Bogacz, R.: Speed-accuracy tradeoff. In: Encyclopedia of Computational Neuroscience, pp. 3225–3228. Springer (2022)
5. Bu, T., Fang, W., Ding, J., Dai, P., Yu, Z., Huang, T.: Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks. In: International Conference on Learning Representations (2022)
6. Cao, Y., Chen, Y., Khosla, D.: Spiking deep convolutional neural networks for energy-efficient object recognition. International Journal of Computer Vision **113**(1), 54–66 (2015)
7. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6299–6308 (2017)
8. Chakraborty, B., Mukhopadhyay, S.: Heterogeneous recurrent spiking neural network for spatio-temporal classification. Frontiers in Neuroscience **17**, 994517 (2023)
9. Chen, C.F.R., Panda, R., Ramakrishnan, K., Feris, R., Cohn, J., Oliva, A., Fan, Q.: Deep analysis of cnn-based spatio-temporal representations for action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6165–6175 (2021)
10. Chéron, G., Laptev, I., Schmid, C.: P-cnn: Pose-based cnn features for action recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3218–3226 (2015)
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021)
12. Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6824–6835 (2021)
13. Fan, Q., Chen, C.F.R., Kuehne, H., Pistoia, M., Cox, D.: More is less: Learning efficient video representations by temporal aggregation modules. In: Advances in Neural Information Processing Systems. vol. 32, p. 2264–2273 (2019)

14. Fang, W., Chen, Y., Ding, J., Yu, Z., Masquelier, T., Chen, D., Huang, L., Zhou, H., Li, G., Tian, Y.: Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. Science Advances **9**(40), eadi1480 (2023)

15. Fang, W., Yu, Z., Chen, Y., Huang, T., Masquelier, T., Tian, Y.: Deep residual learning in spiking neural networks. In: Advances in Neural Information Processing Systems. vol. 34, pp. 21056–21069 (2021)

16. Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., Tian, Y.: Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2661–2671 (2021)

17. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6202–6211 (2019)

18. Feichtenhofer, C., Pinz, A., Wildes, R.P.: Spatiotemporal residual networks for video action recognition. In: Advances in Neural Information Processing Systems. vol. 29, pp. 3468–3476 (2016)

19. Feichtenhofer, C., Pinz, A., Wildes, R.P.: Spatiotemporal multiplier networks for video action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4768–4777 (2017)

20. Guo, Y., Huang, X., Ma, Z.: Direct learning-based deep spiking neural networks: A review. Frontiers in Neuroscience **17**, 1209795 (2023)

21. Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y., et al.: A survey on vision transformer. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(1), 87–110 (2022)

22. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6546–6555 (2018)

23. Horowitz, M.: 1.1 computing's energy problem (and what we can do about it). In: 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers. pp. 10–14. IEEE (2014)

24. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)

25. Hunsberger, E., Eliasmith, C.: Spiking deep networks with lif neurons. arXiv preprint arXiv:1510.08829 (2015)

26. Jiao, L., Gao, J., Liu, X., Liu, F., Yang, S., Hou, B.: Multiscale representation learning for image classification: A survey. IEEE Transactions on Artificial Intelligence **4**(1), 23–43 (2021)

27. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1725–1732 (2014)

28. Kruger, N., Janssen, P., Kalkan, S., Lappe, M., Leonardis, A., Piater, J., Rodriguez-Sanchez, A.J., Wiskott, L.: Deep hierarchies in the primate visual cortex: What can we learn for computer vision? IEEE transactions on pattern analysis and machine intelligence **35**(8), 1847–1871 (2012)

29. Kundu, S., Datta, G., Pedram, M., Beerel, P.A.: Spike-thrift: Towards energy-efficient deep spiking neural networks by limiting spiking activity via attention-guided compression. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3953–3962 (2021)

30. Kundu, S., Pedram, M., Beerel, P.A.: Hire-snn: Harnessing the inherent robustness of energy-efficient deep spiking neural networks by training with crafted input noise. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5209–5218 (2021)

31. Lee, C., Sarwar, S.S., Panda, P., Srinivasan, G., Roy, K.: Enabling spike-based backpropagation for training deep neural network architectures. Frontiers in Neuroscience **14**, 119 (2020)
32. Lee, J.H., Delbruck, T., Pfeiffer, M.: Training deep spiking neural networks using backpropagation. Frontiers in Neuroscience **10**, 508 (2016)
33. Li, K., Wang, Y., Gao, P., Song, G., Liu, Y., Li, H., Qiao, Y.: Uniformer: Unified transformer for efficient spatiotemporal representation learning. In: International Conference on Learning Representations (2022)
34. Li, K., Wang, Y., Zhang, J., Gao, P., Song, G., Liu, Y., Li, H., Qiao, Y.: Uniformer: Unifying convolution and self-attention for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (2023)
35. Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3202–3211 (2022)
36. Liu, Z., Luo, S., Li, W., Lu, J., Wu, Y., Sun, S., Li, C., Yang, L.: Convtransformer: A convolutional transformer network for video frame synthesis. arXiv preprint arXiv:2011.10185 (2020)
37. Maass, W.: Networks of spiking neurons: The third generation of neural network models. Neural Networks **10**(9), 1659–1671 (1997)
38. Meng, Q., Xiao, M., Yan, S., Wang, Y., Lin, Z., Luo, Z.Q.: Training high-performance low-latency spiking neural networks by differentiation on spike representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12444–12453 (2022)
39. Neftci, E.O., Mostafa, H., Zenke, F.: Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. IEEE Signal Processing Magazine **36**(6), 51–63 (2019)
40. Panda, P., Aketi, S.A., Roy, K.: Toward scalable, efficient, and accurate deep spiking neural networks with backward residual connections, stochastic softmax, and hybridization. Frontiers in Neuroscience **14**, 653 (2020)
41. Panda, P., Srinivasa, N.: Learning to recognize actions from limited training examples using a recurrent spiking neural model. Frontiers in Neuroscience **12**, 126 (2018)
42. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5533–5541 (2017)
43. Reilly, D., Das, S.: Just add π! Pose induced video transformers for understanding activities of daily living. arXiv preprint arXiv:2311.18840 (2023)
44. Roy, K., Jaiswal, A., Panda, P.: Towards spike-based machine intelligence with neuromorphic computing. Nature **575**(7784), 607–617 (2019)
45. Rueckauer, B., Lungu, I.A., Hu, Y., Pfeiffer, M., Liu, S.C.: Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. Frontiers in Neuroscience **11**, 682 (2017)
46. Selva, J., Johansen, A.S., Escalera, S., Nasrollahi, K., Moeslund, T.B., Clapés, A.: Video transformers: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence (2023)
47. Shahroudy, A., Liu, J., Ng, T.T., Wang, G.: NTU RGB+D: A large scale dataset for 3d human activity analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1010–1019 (2016)
48. Shrestha, S.B., Orchard, G.: Slayer: Spike layer error reassignment in time. In: Advances in Neural Information Processing Systems. vol. 31, pp. 1419–1428 (2018)
49. Siddiqui, N., Tirupattur, P., Shah, M.: Dvanet: Disentangling view and action features for multi-view action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 4873–4881 (2024)

50. Soomro, K., Zamir, A.R., Shah, M.: UCF101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
51. Tong, Z., Song, Y., Wang, J., Wang, L.: Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In: Advances in Neural Information Processing Systems. vol. 35, pp. 10078–10093 (2022)
52. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4489–4497 (2015)
53. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6450–6459 (2018)
54. Waeijen, L., Sioutas, S., Peemen, M., Lindwer, M., Corporaal, H.: Convfusion: A model for layer fusion in convolutional neural networks. IEEE Access **9**, 168245–168267 (2021)
55. Wang, L., Huang, B., Zhao, Z., Tong, Z., He, Y., Wang, Y., Wang, Y., Qiao, Y.: Videomae v2: Scaling video masked autoencoders with dual masking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14549–14560 (2023)
56. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: European Conference on Computer Vision. pp. 20–36. Springer (2016)
57. Wang, W., Hao, S., Wei, Y., Xiao, S., Feng, J., Sebe, N.: Temporal spiking recurrent neural network for action recognition. IEEE Access **7**, 117165–117175 (2019)
58. Wang, Y., Zhang, M., Chen, Y., Qu, H.: Signed neuron with memory: Towards simple, accurate and high-efficient ann-snn conversion. In: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence. pp. 2501–2508 (2022)
59. Wei, C., Fan, H., Xie, S., Wu, C.Y., Yuille, A., Feichtenhofer, C.: Masked feature prediction for self-supervised visual pre-training. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14668–14678 (2022)
60. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: Proceedings of the European Conference on Computer Vision. pp. 305–321 (2018)
61. Yamazaki, K., Vo-Ho, V.K., Bulsara, D., Le, N.: Spiking neural networks and their applications: A review. Brain Sciences **12**(7), 863 (2022)
62. Yao, M., Hu, J., Hu, T., Xu, Y., Zhou, Z., Tian, Y., XU, B., Li, G.: Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. In: International Conference on Learning Representations (2024)
63. Yao, M., Hu, J., Zhou, Z., Yuan, L., Tian, Y., Xu, B., Li, G.: Spike-driven transformer. In: Advances in Neural Information Processing Systems. vol. 36 (2023)
64. Yin, B., Corradi, F., Bohté, S.M.: Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. Nature Machine Intelligence **3**(10), 905–913 (2021)
65. You, H., Zhong, X., Liu, W., Wei, Q., Huang, W., Yu, Z., Huang, T.: Converting artificial neural networks to ultra-low-latency spiking neural networks for action recognition. IEEE Transactions on Cognitive and Developmental Systems (2024)
66. Zhang, H., Zhou, C., Yu, L., Huang, L., Ma, Z., Fan, X., Zhou, H., Tian, Y.: Sglformer: Spiking global-local-fusion transformer with high performance. Frontiers in Neuroscience **18**, 1371290 (2024)
67. Zhang, J., Wang, J., Di, X., Pu, S.: High-accuracy and energy-efficient action recognition with deep spiking neural network. In: International Conference on Neural Information Processing (ICONIP). pp. 279–292. Springer (2022)
68. Zhou, B., Andonian, A., Oliva, A., Torralba, A.: Temporal relational reasoning in videos. In: Proceedings of the European Conference on Computer Vision. pp. 803–818 (2018)

69. Zhou, C., Yu, L., Zhou, Z., Zhang, H., Ma, Z., Zhou, H., Tian, Y.: Spikingformer: Spike-driven residual learning for transformer-based spiking neural network. arXiv preprint arXiv:2304.11954 (2023)

70. Zhou, C., Zhang, H., Yu, L., Ye, Y., Zhou, Z., Huang, L., Ma, Z., Fan, X., Zhou, H., Tian, Y.: Direct training high-performance deep spiking neural networks: A review of theories and methods. arXiv preprint arXiv:2405.04289 (2024)

71. Zhou, Z., Zhu, Y., He, C., Wang, Y., YAN, S., Tian, Y., Yuan, L.: Spikformer: When spiking neural network meets transformer. In: International Conference on Learning Representations (2023)