Abstract

The combination of Spiking Neural Networks (SNNs) and Transformers has attracted significant attention due to their potential for high energy efficiency and high-performance nature. However, existing works on this topic typically rely on direct training, which can lead to suboptimal performance. To address this issue, we propose to leverage the benefits of the ANN-to-SNN conversion method to combine SNNs and Transformers, resulting in significantly improved performance over existing state-of-the-art SNN models. Furthermore, inspired by the quantal synaptic failures observed in the nervous system, which reduce the number of spikes transmitted across synapses, we introduce a novel Masked Spiking Transformer (MST) framework. This incorporates a Random Spike Masking (RSM) method to prune redundant spikes and reduce energy consumption without sacrificing performance. Our experimental results demonstrate that the proposed MST model achieves a significant reduction of 26.8% in power consumption when the masking ratio is 75% while maintaining the same level of performance as the unmasked model. The code is available at: https://github.com/bic-L/Masked-Spiking-Transformer.

1. Introduction

Spiking neural networks (SNNs), considered as the next generation neural networks [30], are brain-inspired neural networks based on the dynamic characteristics of biological neurons [31, 17]. SNNs have attracted significant attention due to their unique properties in handling sparse data, which can yield great energy efficiency benefits on neuromorphic hardware. Due to their specialties, they have been widely utilized in various fields, such as classification [32, 18], object detection [3] and tracking [51], etc. Nevertheless, SNNs currently can hardly realize a comparable performance to that of artificial neural networks (ANNs), especially for complex tasks such as ImageNet [40].

In order to improve the performance of SNNs, various training methods have been proposed, broadly categorized as the direct training method and the ANN-to-SNN conversion method. Direct training methods leverage a continuous relaxation of the non-smooth spiking mechanism to enable backpropagation with a surrogate gradient function for handling non-differentiability [36], but this can lead to unstable gradient propagation and relatively low accuracy.
compared to leading ANNs [38]. Alternatively, ANN-to-SNN conversion methods convert pre-trained ANNs into SNNs for better performance while requiring more time steps, with increased power consumption to reduce conversion errors [46, 25, 2, 7]. Our focus is on implementing the ANN-to-SNN conversion method to narrow the performance gap between leading ANNs and SNNs, but the required long time steps pose challenges in reducing energy consumption. Therefore, identifying strategies to decrease power consumption while maintaining excellent performance is crucial.

The biological nervous system offers valuable insights for addressing the challenges of implementing high-performance Spiking Transformers using the ANN-to-SNN conversion method. The quantal synaptic failure theory suggests that missing information during neuronal signal transmission may not impact the computational information transmitted to a postsynaptic neuron under certain conditions, but can reduce energy consumption and heat production [23]. Likewise, in the ANN-to-SNN conversion process, missing spikes can possibly be compensated for by leveraging the correlations between signals in the space and time domains during the information propagation over multiple time steps. In addition, neural network models possess lots of redundant connections: prior works reveal that the redundancy in the self-attention module of Transformers can be pruned without significantly impacting performance [34, 49]. Therefore, eliminating redundant information during the transmission of neuronal signals can possibly reduce overall energy consumption in the Spiking Transformer model while preserving high performance.

In our work, we propose a Masked Spiking Transformer (MST), which incorporates a Random Spike Masking (RSM) method designed specifically for SNNs. The RSM method randomly selects only a subset of input spikes, significantly reducing the number of spikes involved in the computation process. We evaluate the MST model on both static and neuromorphic datasets, demonstrating its superiority over existing SNN models. Our experiments show that the RSM method can reduce energy consumption on the self-attention module and the MLP module in Transformer, enabling the SNNs to take advantage of energy efficiency and high performance. Furthermore, the proposed RSM method is not limited to Transformer, but can be extended to other backbones such as ResNet and VGG, highlighting its potential as a general technique to improve SNN efficiency. Our results demonstrate the potential of this approach to provide a new direction for developing high-performance and energy-efficient SNN models.

The main contributions of this paper can be summarized as follows:

- We propose a Masked Spiking Transformer (MST) using the ANN-to-SNN conversion method. To the best of our knowledge, it is the first exploration of applying the self-attention mechanism fully in SNNs utilizing the ANN-to-SNN conversion method.

- The MST model is evaluated on both static and neuromorphic datasets, and the results show that it outperforms state-of-the-art (SOTA) SNNs on all datasets. In specific, the top-1 accuracy of the MST model is 1.21%, 7.3%, and 3.7% higher than the current SOTA SNN model on the CIFAR-10, CIFAR-100, and ImageNet datasets, respectively.

- We design a Random Spike Masking (RSM) method for SNNs trained with the ANN-to-SNN conversion method to prune the redundant spikes during inference and save energy consumption.

- Extensive experiments show that our proposed RSM is a versatile and general method that can be utilized in other spike-based deep networks, such as ResNet and VGG SNN model variants.

2. Related Work

**Spiking Neural Networks** SNNs have gained popularity in the field of brain-inspired intelligence due to their compatibility with neuromorphic hardware and biological properties. With the increasing interest in larger-scale and higher-performance SNNs, recent research has focused on developing novel training algorithms and architectures.
Zheng et al. proposed a threshold-dependent batch normalization (tdBN) method based on spatiotemporal backpropagation to train a large-scale SNN model with 50 layers [56]. Besides, Fang et al. proposed the SEW ResNet architecture for residual learning in deep SNNs to overcome the gradient vanishing problem [10]. Later, they introduced a training algorithm that learns the threshold of each spiking neuron to improve the performance of SNNs [11]. However, these methods mainly discuss the SNN models that are dominated by convolutional layers, such as the SNN variants of VGG [42] and ResNet [15]. Despite their improvements, the performance of these methods still struggles to match the performance of ANNs [41].

Transformer Transformer [47] was first introduced in Natural Language Processing (NLP) and quickly gained popularity for its remarkable capabilities in capturing long-range dependencies. Its success in NLP has inspired researchers to explore its potential in computer vision. Vision Transformer (ViT) [8] was the first attempt to apply the Transformer to image classification. ViT has achieved impressive results on various computer vision benchmarks, demonstrating the effectiveness of the self-attention mechanism in image understanding. Following the success of ViT, a series of works [29, 14] proposed improvements to the original ViT architecture. Motivated by the success of Transformers and its variations, this paper proposes a new architecture for SNNs that leverages the capacities of the Transformer and the energy efficiency of SNNs.

Spiking Transformer The combination of the Transformer and SNNs can achieve better performance, which has been discussed in prior studies, including STNet [54] and Spike-T [55]. These models utilized separate branches of SNNs and Transformers for feature extraction, leading to the inability to run independently on neuromorphic hardware and failing to exploit the energy efficiency benefits of SNNs fully. In addition, Mueller et al. [35] proposed a Spiking Transformer using the ANN-to-SNN conversion method, but they did not implement the self-attention module in SNNs. The recently proposed Spikformer [57] directly trained the Transformer in SNNs, but still struggles to achieve comparable performance to leading ANNs. To address these limitations, we apply the self-attention mechanism fully in SNNs by utilizing the ANN-to-SNN conversion method and propose the RSM method to improve both the performance and energy efficiency of the Spiking Transformer. Our model offers a new direction for developing high-performance SNNs using the ANN-to-SNN conversion method.

3. Methods

3.1. Spiking Neuron Model

For ANNs, the input $a^{l-1}$ to layer $l$ is mapped to the output $a^l$ by a linear transformation matrix $W^l$ and a nonlinear activation function $f(\cdot)$, that is ($l = 1, 2, 3, \cdots, L$):

$$a^l = f \left( W^l a^{l-1} \right)$$  

(1)

where $f(\cdot)$ is often set as the ReLU activation function.

In SNNs, the Integrate-and-Fire (IF) spiking neuron model is commonly used in ANN-to-SNN conversion [25, 2, 7]. The dynamics of the IF model are described by:

$$v^l(t) = v^l(t - 1) + W^l \theta^{l-1} s^{l-1}(t) - \theta^l s^l(t)$$  

(2)

where $v^l(t)$ denotes the membrane potential of neurons in layer $l$ at time step $t$, which corresponds to the linear transformation matrix $W^l$, the threshold $\theta^l$, and the binary output spikes of neurons in the previous layer $l - 1$, denoted as $s^{l-1}(t)$. The $s^l(t)$ is defined as follows:

$$s^l(t) = H(\mu^l(t) - \theta^l)$$  

(3)

where $\mu^l(t) = v^l(t - 1) + W^l \theta^{l-1} s^{l-1}(t)$ denotes the membrane potential of neurons before the trigger of a spike at time step $t$, $H(\cdot)$ denotes the Heaviside step function. The neurons generate output spikes whenever the membrane potential $\mu^l(t)$ exceeds the threshold value $\theta^l$, and the membrane potential is reset by subtracting the threshold value to reduce information loss [41].

3.2. ANN-to-SNN conversion

To achieve the ANN-SNN conversion, a relationship is established between the rectified linear unit (ReLU) activation of analog neurons in ANNs and the firing rate or postsynaptic potential of spiking neurons in SNNs. This is obtained by summing Eq. 2 from time step 1 to $T$ dividing $T$ on both sides, resulting in the following equation:

$$\frac{v^l(T) - v^l(0)}{T} = \frac{\sum_{t=1}^{T} W^l \theta^{l-1} s^{l-1}(t)}{T} - \frac{\sum_{t=1}^{T} \theta^l s^l(t)}{T}$$  

(4)

The linear relationship between $\phi^l(T)$ and $\phi^{l-1}(T)$ is established by defining $\phi^l(T) = \frac{\sum_{t=1}^{T} \theta^l s^l(t)}{T}$ as the average postsynaptic potential:

$$\phi^l(T) = W^l \phi^{l-1}(T) - \frac{v^l(T) - v^l(0)}{T}$$  

(5)

The equivalence between Eq. 1 and 5 holds only as $T$ goes to infinity, resulting in a conversion error. To address this issue, we replace the ReLU activation function with the quantization clip-floor-shift (QCFS) [2] function in the ANNs.
3.3. Model Architecture

An overview of the MST is depicted in Fig. 2, where the Swin Transformer [29] is adopted as the backbone network. To convert the original network into a fully-spiking manner, we incorporate the QCFS activation functions after each linear or regularization layer during the training phase, which are replaced with Integrate-and-Fire (IF) neurons in the inference process, resulting in more efficient computation.

The entire computation process in the spiking self-attention module can be formulated as:

\[ Q_{spk}[t] = \text{IF}(X[t] \ast W_q) \]
\[ K_{spk}[t] = \text{IF}(X[t] \ast W_k) \]

(6)

where \( Q_{spk}, K_{spk} \) denote the spike matrices of the query and key at \( t \) time step, \( \text{IF}(\cdot) \) is the IF neuron function, \( W_q, W_k \) denote the corresponding weight matrices. Attention score is defined as:

\[ A_{spk}[t] = \text{IF}(\frac{Q_{spk}[t] \ast K_{spk}^T[t]}{\sqrt{d}}) \]

(7)

where \( A_{spk} \) represents the spike matrix of the attention score calculated by the dot product of the query spike matrix and the key spike matrix, and \( d \) is a scaling factor equal to the feature dimension of a given attention head.

The calculation of LN and BN can be expressed by:

\[ y = \frac{x - \text{E}[x]}{\sqrt{\text{Var}[x] + \epsilon}} \cdot \gamma + \beta \]

(8)

where \( x \) denotes the input tensor to be normalized, \( \text{E}[x] \) and \( \text{Var}[x] \) represent the mean and variance of \( x \), \( \epsilon \) is a small constant added to the variance for numerical stability, \( \gamma \) and \( \beta \) are trainable scaling and bias parameters respectively, and \( y \) is the normalized output of the layer.

In LN, the mean and variance are calculated across the features of each sample in a batch. Therefore, each sample in the batch has its normalization parameters. On the other hand, BN calculates the mean and variance across all samples in a batch for each feature, which means the normalization parameters are shared across all samples in a batch.

Normalization is important for ensuring the feasibility of ANN-to-SNN conversion. As illustrated in Fig. 3, different normalization approach results in vastly different membrane potential in the inference process. Normalizing along the channel dimension (LN) causes a distribution mismatch between ANN and SNN, which leads to performance degradation, while normalizing along the batch dimension (BN) preserves a similar result. Replace all LN layers with BN layers is a straightforward approach to normalize the post-activation distribution of ANNs during the conversion process, but for large datasets like ImageNet, there are convergence issues. We resolve this problem by simply adding a BN layer after each linear layer in the MLP module, inspired by [53]. Mathematically, the first layer of the modified MLP module is formulated as follows:

\[ \text{MLP}(x) = \text{IF}(\text{Linear}(\text{BN}(x))) \]

(9)

where \( x \) represents the input tensor, Linear denotes the linear layer, BN represents the batch normalization layer. Notably, the MLP module consists of two linear layers, each followed by a BN layer and an IF neuron function.

3.4. Random Spike Masking Method

Implementing the Transformer in SNNs using the ANN-to-SNN conversion method presents a challenge due to the high power consumption demand. To address this issue, we propose the Random Spike Masking (RSM) method for reducing redundant spikes during inference.

Traditional ANN pruning methods remove weights with low magnitudes, which are believed to have little impact on the final output of the model [58]. In contrast, our RSM method prunes input spikes randomly according to the masking ratio, where each spike (\( s = 1 \)) has the probability of turning into a failure state (\( s = 0 \)) that can be implemented using binary mask matrices.

It is important to note that our proposed RSM method for spike pruning in SNNs differs from Dropout [44] used in ANNs. Dropout is a regularization technique that randomly masks some neurons during training, but all neurons are used for inference, and the output needs to be scaled by the retained probability used during training. This means that Dropout cannot directly reduce the number of spikes in the network. Unlike Dropout, the RSM method reduces the computational cost of SNNs by randomly pruning input spikes in both training and inference, explicitly removing redundancy in input spikes while obtaining excellent performance.

The RSM method, illustrated in Fig. 2(c-e), randomly generates a binary mask with the same shape as the input spike matrix based on the masking ratio. This determines...
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Method</th>
<th>Architecture</th>
<th># Param (M)</th>
<th>Time Steps</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR-10</td>
<td>ANN [29]</td>
<td>Direct Training</td>
<td>Swin-T (BN)</td>
<td>27.6</td>
<td>/</td>
<td>98.14</td>
</tr>
<tr>
<td></td>
<td>Diet-SNN [39]</td>
<td>Direct Training</td>
<td>VGG-16</td>
<td>39.9</td>
<td>10</td>
<td>93.44</td>
</tr>
<tr>
<td></td>
<td>tdBN [56]</td>
<td>Direct Training</td>
<td>ResNet-19</td>
<td>12.6</td>
<td>4</td>
<td>92.92</td>
</tr>
<tr>
<td></td>
<td>Spikformer [57]</td>
<td>Direct Training</td>
<td>Spikformer-4-384</td>
<td>9.3</td>
<td>4</td>
<td>95.51</td>
</tr>
<tr>
<td></td>
<td>RMP [13]</td>
<td>ANN-to-SNN</td>
<td>VGG-16</td>
<td>39.9</td>
<td>2048</td>
<td>93.63</td>
</tr>
<tr>
<td></td>
<td>QCFS [2]</td>
<td>ANN-to-SNN</td>
<td>RestNet-18</td>
<td>11.7</td>
<td>512</td>
<td>96.06</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>ANN-to-SNN</td>
<td>Swin-T (BN)</td>
<td>27.6</td>
<td>64</td>
<td>96.32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Method</th>
<th>Architecture</th>
<th># Param (M)</th>
<th>Time Steps</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR-100</td>
<td>ANN [29]</td>
<td>Direct Training</td>
<td>Swin-T (BN)</td>
<td>27.6</td>
<td>/</td>
<td>88.72</td>
</tr>
<tr>
<td></td>
<td>Diet-SNN [39]</td>
<td>Direct Training</td>
<td>VGG-16</td>
<td>39.9</td>
<td>5</td>
<td>69.67</td>
</tr>
<tr>
<td></td>
<td>tdBN [56]</td>
<td>Direct Training</td>
<td>ResNet-19</td>
<td>12.6</td>
<td>4</td>
<td>70.86</td>
</tr>
<tr>
<td></td>
<td>TET [6]</td>
<td>Direct Training</td>
<td>ResNet-19</td>
<td>12.6</td>
<td>6</td>
<td>74.72</td>
</tr>
<tr>
<td></td>
<td>DSR [32]</td>
<td>Direct Training</td>
<td>PreActResNet-18</td>
<td>11.2</td>
<td>20</td>
<td>78.50</td>
</tr>
<tr>
<td></td>
<td>Spikformer [57]</td>
<td>Direct Training</td>
<td>Spikformer-4-384</td>
<td>9.3</td>
<td>4</td>
<td>78.21</td>
</tr>
<tr>
<td></td>
<td>RMP [13]</td>
<td>ANN-to-SNN</td>
<td>VGG-16</td>
<td>39.9</td>
<td>2048</td>
<td>70.93</td>
</tr>
<tr>
<td></td>
<td>QCFS [2]</td>
<td>ANN-to-SNN</td>
<td>RestNet-18</td>
<td>11.7</td>
<td>512</td>
<td>79.61</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>ANN-to-SNN</td>
<td>Swin-T (BN)</td>
<td>27.6</td>
<td>64</td>
<td>85.40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Method</th>
<th>Architecture</th>
<th># Param (M)</th>
<th>Time Steps</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageNet</td>
<td>ANN [29]</td>
<td>Direct Training</td>
<td>Swin-T (BN)</td>
<td>28.5</td>
<td>/</td>
<td>80.51</td>
</tr>
<tr>
<td></td>
<td>Diet-SNN [39]</td>
<td>Direct Training</td>
<td>VGG-16</td>
<td>39.9</td>
<td>5</td>
<td>69.00</td>
</tr>
<tr>
<td></td>
<td>tdBN [56]</td>
<td>Direct Training</td>
<td>SEW-ResNet-34</td>
<td>21.8</td>
<td>4</td>
<td>67.04</td>
</tr>
<tr>
<td></td>
<td>TET [6]</td>
<td>Direct Training</td>
<td>SEW-ResNet-34</td>
<td>21.8</td>
<td>4</td>
<td>68.00</td>
</tr>
<tr>
<td></td>
<td>Spikformer [57]</td>
<td>Direct Training</td>
<td>Spikformer-8-768</td>
<td>66.3</td>
<td>4</td>
<td>74.81</td>
</tr>
<tr>
<td></td>
<td>RMP [13]</td>
<td>ANN-to-SNN</td>
<td>VGG-16</td>
<td>39.9</td>
<td>2048</td>
<td>73.09</td>
</tr>
<tr>
<td></td>
<td>QCFS [2]</td>
<td>ANN-to-SNN</td>
<td>RestNet-18</td>
<td>11.7</td>
<td>1024</td>
<td>74.32</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>ANN-to-SNN</td>
<td>Swin-T (BN)</td>
<td>28.5</td>
<td>128</td>
<td>77.88</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>256</td>
<td>78.37</td>
</tr>
</tbody>
</table>

Table 1. Performance comparison between the proposed model and the SOTA models on different static datasets, where Swin-T (BN) refers to the self-implemented ANN baseline replacing LN with BN [29].

which spikes are transmitted for calculation in subsequent neurons, enabling high matrix sparsity. With a masking ratio of 50%, for example, each spike has a 50% chance of being masked, reducing the number of spikes and power consumption. Hence, combining both the SNN-based self-attention mechanism with the RSM approach promises a balance between performance and energy efficiency.

4. Experiments

We conduct extensive experiments on both static datasets including CIFAR-10 [22], CIFAR-100 [21], and ImageNet datasets [5], and neuromorphic datasets including CIFAR10-DVS [24], N-Caltech101 [37], N-Cars [43], ActionRecognition [33], and ASL-DVS [1] datasets, to validate the effectiveness of the MST model. In addition, we evaluate the effect of the RSM approach on accuracy and energy efficiency using the SpikingJelly framework [9]. More details of the training can be found in the supplementary.

4.1. Performance on Static Datasets

Tab. 1 presents a comprehensive comparison of the MST model with the current SOTA SNN models on the CIFAR-10/100 and ImageNet datasets. The results show that the
The proposed MST model outperforms all other models in terms of top-1 accuracy on all three datasets.

Direct training models, such as Diet-SNN [39], tdBN [56], TET [6], and DSR [32], require lower time steps, but their top-1 accuracy is relatively low, compared to that of the MST model, in which tdBN [56] is 10% less accurate on the CIFAR-100 and ImageNet datasets.

Conversely, models employing the ANN-to-SNN conversion method typically necessitate a greater number of time steps to reach optimal performance, including RMP [13], RNL [7], and QCFS [2]. The MST model also employs this method but requires only 64 time steps to achieve superior performance, which is far fewer than RMP [13]. This suggests that the MST model can achieve the highest accuracy in a reasonable number of time steps.

Compared to transformer-based SNN models such as Spikformer [57], the MST model achieves higher top-1 accuracy on all three datasets. Specifically, the top-1 accuracy of the MST model is 1.8%, 8.7%, and 3.7% higher than Spikformer on the CIFAR-10, CIFAR-100, and ImageNet datasets, respectively. Moreover, the MST model has significantly fewer parameters compared to the 8-layer Spikformer [57] model.

### 4.2. Performance on Neuromorphic Datasets

We showcase the suitability of the MST model for processing event-based data by evaluating its performance on neuromorphic datasets. Fig. 4 shows the framework for the MST model to process the neuromorphic datasets. We utilize a frame-based representation for preprocessing, where the event streams are transformed into a sequence of high-rate frames. Each event in the stream consists of four dimensions, including two spatial coordinates (x, y), timestamp, and polarity. The frames are integrated into an input tensor of size \( (n \times C, H, W) \), where \( n \) is the number of frames, \( C \) is the number of original channels (which equals two, representing polarity), and \( H \) and \( W \) represent the height and width of the input, respectively. To align the input dimension with the model, an additional reduction layer is added to the first layer of the entire model, which reduces the channel dimension to 3. This enables us to leverage pre-training weights from the ImageNet dataset, accelerating the training convergence. Data augmentation for SNNs [27] is also applied to improve the accuracy. The ensuing experimental results demonstrate that the proposed fine-tuning approach for ANN-to-SNN conversion achieves remarkably high accuracy.

The experimental results presented in Tab. 2 demonstrate the effectiveness of the proposed MST model in processing neuromorphic datasets. We compare the MST model with several SOTA SNN models on five popular neuromorphic datasets, including CIFAR10-DVS, N-Caltech101, N-Cars, Action Recognition, and ASL-DVS dataset. The experimental results show that the MST model achieves the highest top-1 accuracy on all datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>Time Steps</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR10-DVS</td>
<td>Swin-T (BN)</td>
<td>/</td>
<td>88.98</td>
</tr>
<tr>
<td></td>
<td>TA-SNN [52]</td>
<td>10</td>
<td>72.00</td>
</tr>
<tr>
<td></td>
<td>PLIF [11]</td>
<td>20</td>
<td>74.80</td>
</tr>
<tr>
<td></td>
<td>Dspkie [26]</td>
<td>10</td>
<td>75.40</td>
</tr>
<tr>
<td></td>
<td>DSR [32]</td>
<td>10</td>
<td>77.30</td>
</tr>
<tr>
<td></td>
<td>TET [6]</td>
<td>10</td>
<td>83.17</td>
</tr>
<tr>
<td></td>
<td>NDA [27]</td>
<td>10</td>
<td>81.70</td>
</tr>
<tr>
<td></td>
<td>Spikformer [57]</td>
<td>10</td>
<td>80.90</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>128</td>
<td><strong>86.60</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>256</td>
<td><strong>87.20</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>512</td>
<td><strong>88.12</strong></td>
</tr>
<tr>
<td>N-CALTECH101</td>
<td>Swin-T (BN)</td>
<td>/</td>
<td>92.00</td>
</tr>
<tr>
<td></td>
<td>SALT [19]</td>
<td>20</td>
<td>55.00</td>
</tr>
<tr>
<td></td>
<td>NDA [27]</td>
<td>10</td>
<td>83.70</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>64</td>
<td><strong>84.71</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td><strong>89.42</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>256</td>
<td><strong>91.38</strong></td>
</tr>
<tr>
<td>N-CARS</td>
<td>Swin-T (BN)</td>
<td>/</td>
<td>97.14</td>
</tr>
<tr>
<td></td>
<td>CarSNN [48]</td>
<td>10</td>
<td>86.00</td>
</tr>
<tr>
<td></td>
<td>NDA [27]</td>
<td>10</td>
<td>91.90</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>32</td>
<td><strong>94.67</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>64</td>
<td><strong>96.58</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td><strong>97.28</strong></td>
</tr>
<tr>
<td>Action Recognition</td>
<td>Swin-T (BN)</td>
<td>/</td>
<td>90.14</td>
</tr>
<tr>
<td></td>
<td>STCA [12]</td>
<td>10</td>
<td>71.20</td>
</tr>
<tr>
<td></td>
<td>Mb-SNN [28]</td>
<td>10</td>
<td>78.10</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>64</td>
<td><strong>84.76</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td><strong>86.92</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>256</td>
<td><strong>88.21</strong></td>
</tr>
<tr>
<td>ASL-DVS</td>
<td>Swin-T (BN)</td>
<td>/</td>
<td>99.90</td>
</tr>
<tr>
<td></td>
<td>Meta-SNN [45]</td>
<td>100</td>
<td>96.04</td>
</tr>
<tr>
<td></td>
<td>MST (ours)</td>
<td>64</td>
<td><strong>98.04</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td><strong>98.51</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>256</td>
<td><strong>99.10</strong></td>
</tr>
</tbody>
</table>

Table 2. Performance comparison between the proposed model and the SOTA models on different neuromorphic datasets.
constructed by converting the static datasets into event data by using event-based cameras. Tab. 2 shows that the MST model outperforms other SOTA SNN models significantly, achieving improvements of 4.95%, 7.68%, and 5.38% on CIFAR10-DVS, N-Caltech101, and N-Cars, respectively. The Action Recognition dataset consists of a series of human actions captured by event-based cameras. By adopting the data preprocessing method of [20], our MST model achieves a top-1 accuracy of 88.21%, which is far higher than other models. Additionally, the ASL-DVS dataset is a large 24-class dataset of gestures and the experimental results demonstrate that our MST model outperforms other SOTA models by 3.06%.

These results highlight the effectiveness of our proposed MST model in processing various types of neuromorphic datasets.

4.3. Effectiveness of the Random Spike Masking Method

The ANN-to-SNN conversion method typically involves a large number of time steps, leading to high power consumption. To address this issue, we propose the RSM method for spike pruning, and we demonstrate its effectiveness in the following section.

Drawing inspiration from knowledge distillation [16], we use the model without masked as a teacher model, and the model after masked as a student model. By fine-tuning the student model, the masked model achieves high performance comparable to the original model without masking in terms of accuracy. This finding highlights the efficacy of the knowledge distillation technique in training masked models without compromising their performance. Consequently, we employ the fine-tuning approach as the training method for our subsequent comparative analysis.

We evaluate the RSM method in two critical modules in the Transformer: the self-attention (SA) module and the Multi-layer Perceptron (MLP) module. As shown in Fig. 2(d-e), we apply the RSM method to the Query, Key, and Attention matrices in the SA module, as well as the output spike matrix of the first fully connected layer within each block of the MLP module.

Fig. 5(a) and (b) depict the variation in accuracy for different masking ratios on the CIFAR-10 and CIFAR-100 datasets, respectively. The experimental results demonstrate that the accuracy decreases with the increasing masking ratio for both the SA and MLP modules, but their sensitivity to masking ratio changes differs. Specifically, the accuracy of the SA module remains stable over a certain range of masking ratios. In contrast, the accuracy of the MLP module declines more sharply and is more sensitive to masking ratio changes. Consistent with prior research [34, 49], the redundancy in the SA module enables the MST to withstand the losses caused by missing input spikes without significantly affecting the overall performance of the SNN.

In addition to the variation in sensitivity to changes in the masking ratio between different modules, our experiments also demonstrate that the sensitivity can vary depending on the dataset used. The CIFAR-10 dataset is relatively less sensitive to changes in the masking ratio, with only a 0.1% decrease in accuracy when the masking ratio reaches 80%. In contrast, the CIFAR-100 dataset is more sensitive to changes in the masking ratio, experiencing a more substantial accuracy decline as the masking ratio increases.

We also investigate the potential of the RSM method as a general approach for reducing power consumption in the context of ANN-to-SNN conversion. By applying the RSM
Table 3. Comparison of power consumption and accuracy between models with a Masking ratio of 0%, 50%, and 75% on the CIFAR-10 dataset, respectively. (· · ·) in the table denotes the power consumption/accuracy compared to the unmasked model (with 0% random ratio).

<table>
<thead>
<tr>
<th>Model</th>
<th>Random Ratio</th>
<th>P (α Watts)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST</td>
<td>0%</td>
<td>3.9G (×1)</td>
<td>97.27 (+0)</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>3.2G (×0.82)</td>
<td>97.25 (-0.02)</td>
</tr>
<tr>
<td></td>
<td>75%</td>
<td>2.9G (×0.74)</td>
<td>97.29 (+0.02)</td>
</tr>
<tr>
<td>ResNet-18</td>
<td>0%</td>
<td>58.2M (×1)</td>
<td>96.48 (+0)</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>40.7M (×0.70)</td>
<td>92.88 (-3.60)</td>
</tr>
<tr>
<td></td>
<td>75%</td>
<td>34.1M (×0.58)</td>
<td>82.68 (-13.80)</td>
</tr>
<tr>
<td>VGG-16</td>
<td>0%</td>
<td>24.4M (×1)</td>
<td>95.46 (+0)</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>18.9M (×0.77)</td>
<td>89.56 (-5.90)</td>
</tr>
<tr>
<td></td>
<td>75%</td>
<td>16.7M (×0.68)</td>
<td>79.09 (-16.37)</td>
</tr>
</tbody>
</table>

5. Discussion

In this paper, we proposed the Masked Spiking Transformer (MST) with the Random Spike Masking (RSM) method. By pruning input spikes, the proposed RSM method effectively reduces power consumption while maintaining the performance of the model within a certain range. Though our experiments highlight the superiority of the MST model over state-of-the-art SNN models, our model still has limitations that need addressing. A key constraint is the relatively long time steps needed by the ANN-to-SNN model can reduce power consumption by 26.8% without any loss in performance when the masking ratio reaches 75%. This finding suggests that there exist substantial redundant spikes in the SA module that can be pruned for better energy efficiency. As for ResNet-18 and VGG-16, the RSM method also yields significant power reduction. For instance, ResNet-18 can reduce power consumption by 30.1% with a moderate accuracy loss of 3.6%. However, excessive masking leads to a significant drop in accuracy.

In summary, our results demonstrate that the proposed RSM method is effective when applied to the SA module in the Transformer, as well as other SNN models. By eliminating redundant spikes, the RSM method reduces power consumption while preserving performance, making it a promising approach for energy-efficient ANN-to-SNN conversion.

To visualize the impact of different masking ratios on attention maps, we compared the models with 0%, 50%, and 75% masking ratios using the Spike Activation Maps (SAM) method. The results, shown in Fig. 7, demonstrate that the models with different masking ratios concentrate on similar areas of the object at the same time step, with the red parts outlining the object. Comparison with the ANN model using the ScoreCAM method [50] reveals that both models focus on similar key information. These results suggest that the proposed RSM method preserves the regions of interest within the model, contributing to accuracy preservation.
conversion method, limiting the suitability of the proposed model for real-time applications with strict timing demands.

Furthermore, even though the RSM method reduces the number of spikes and energy consumption, the experiment in this article merely investigates its applicability of ANN-to-SNN conversion, thus exhibiting relatively high energy consumption compared to direct-trained SNN models, which take fewer time steps. Consequently, future research could apply the RSM method to direct training methods to optimize energy consumption further and make SNNs more practical for real-world applications. Additionally, adopting different masking ratios across layers may achieve a better balance between performance and energy efficiency.

6. Conclusion

In this work, we propose a Masked Spiking Transformer (MST) framework that combines the energy efficiency of SNNs with the high-performance self-attention mechanism of Transformers using the ANN-to-SNN method. Additionally, we introduce a Random Spike Masking (RSM) method to prune input spikes, thus reducing power consumption. The experimental results demonstrate that the MST model outperforms current SOTA SNN models on both static and neuromorphic datasets. Furthermore, the proposed RSM method shows significant power reduction while maintaining performance in different modules of the Transformer and other SNN models. Our work opens up new possibilities for developing high-performance SNN models, paving the way for future research in this area.

7. Acknowledgement

This work is supported by the Guangzhou-HKUST(GZ) Joint Funding Program (Grant No. 2023A03J082), HK RGC (Grant Nos. 27206321, 17205922, 17212923), NSFC (Grant No. 62122004), National Key R&D Program of China (Grant No. SQ2022YFB360159). This research is also partially supported by ACCESS – AI Chip Center for Emerging Smart Systems, sponsored by Innovation and Technology Fund (ITF), Hong Kong SAR.

References


