One-step Spiking Transformer with a Linear Complexity

Xiaotian Song[1] , **Andy Song**[2] , **Rong Xiao**[1] and **Yanan Sun**[1*]
[1]College of Computer Science, Sichuan University
[2]School of Computing Technologies, RMIT University
songxt@stu.scu.edu.cn, andy.song@rmit.edu.au, xiaorong.scu@gmail.com, ysun@scu.edu.cn

Abstract

Spiking transformers have recently emerged as a robust alternative in deep learning. One focus of this field is the reduction of energy consumption, given that spiking transformers require lengthy simulation timesteps and complex floating-point attention mechanisms. In this paper, we propose a one-step approach that requires only one timestep and is of linear complexity. The proposed One-step Spiking Transformer (OST) incorporates a Time Domain Compression and Compensation (TDCC) component, which can significantly mitigate the spatio-temporal overhead of spiking transformers. Another novel component in OST is the Spiking Linear Transformation (SLT), designed to greatly reduce the number of floating-point multiply-and-accumulate operations. Experiments on both static and neuromorphic images show that OST can perform as well as or better than SOTA methods with just one timestep, even for more difficult tasks. For instance, comparing with Spikeformer, OST gains 1.59% in accuracy on ImageNet, yet 40.27% more efficient, and gains 0.7% on DVS128 Gesture. The supplementary materials and source code are available at https://github.com/songxt3/OST.

1 Introduction

Spiking Neural Network (SNN) is the third generation neural network inspired by the behavior of biological neurons [Maass, 1997]. While Artificial Neural Network (ANN) utilizes computationally intensive floating-point Multiply-and-Accumulate (MAC) operations, SNN significantly reduces energy consumption by employing event-driven binary spike singles [Roy *et al.*, 2019]. In practice, SNN is widely acknowledged as an energy-efficient alternative to ANN and has been extensively used in tasks such as image classification [Wu *et al.*, 2019] and object detection [Kim *et al.*, 2020]. On the other hand, Transformer [Vaswani *et al.*, 2017], initially designed for natural language processing, has become a prominent technique owing to its excellent performance. However, transformer's energy consumption is often substantial due to the high computational cost required for training and inference. For example, Switch Transformer [Fedus *et al.*, 2022] consumes 179,000 kWh of power, while the energy consumption for training GPT-3 [Brown *et al.*, 2020] is estimated to be 1,278,000 kWh [Patterson *et al.*, 2021]. To leverage the high performance of transformer and low energy use of SNN, great effort has been dedicated to incorporating energy-efficient spiking calculations into transformer models, known as spiking transformers [Mueller *et al.*, 2021; Yao *et al.*, 2021; Zhou *et al.*, 2023; Yao *et al.*, 2023b; Wang *et al.*, 2023; Yao *et al.*, 2023a].

Although spiking transformers have demonstrated great performance across various tasks, there is still room for improvement in both performance and energy consumption. Most spiking transformers still require a level of energy consumption too high to be suitable for deployment on mainstream neuromorphic hardware. The two contributing factors that often lead to a dramatic increase in energy consumption are (1) lengthy timesteps and (2) complex MAC operations. Firstly, spiking transformers require additional T timesteps of temporal information ($T > 3$) [Meng *et al.*, 2022]. Separate training is needed for every individual timestep, hence naturally resulting in $T\times$ spatio-temporal overhead compared to vanilla transformer. More importantly, deployment on neuromorphic hardware favors spiking transformers with fewer timesteps for simplicity and efficiency. Secondly, complex MAC operation [Zhou *et al.*, 2023] is an integral part of the self-attention mechanism in existing spiking transformers. That leads to a relatively high complexity of $O(L^2d)$, where L and d denote the sequence length and feature dimension of input, respectively. Hereby a load of floating-point matrix dot product operations are involved, defeating the original purpose of SNN design, i.e., minimizing computations.

Aiming to address the above issues, in this study, we present a one-step binarized spiking transformer, reducing the timesteps from multiple to only one. More specifically, we introduce Time Domain Compression and Compensation (TDCC) to mitigate the spatio-temporal overhead. The "*Compression*" here is to compress T timesteps of information into a single one, thus substantially reducing the computational cost. The subsequent "*Compensation*" is to compensate for the information loss in the time domain, caused by "*Compression*", which is not lossless. Combin-

[*] Corresponding author.

ing both, TDCC can fully leverage the spatio-temporal information and achieve pre-compression performance, but with a much lower cost. Furthermore, we introduce the Spiking Linear Transformation (SLT) to avoid the use of $O(L^2d)$ self-attention mechanism. Several variants of transformers, including Sparse Transformer [Child *et al.*, 2019], Linear Transformers [Katharopoulos *et al.*, 2020], and FLASH [Hua *et al.*, 2022], endeavor to achieve the same purpose. They however still rely on floating-point dot product, not ideal for SNNs. The proposed SLT, however, has two parts well suited for binary spike signals. The first is *SAF, Spiking Attention-Free*, which eliminates floating-point dot product computation for self-attention. The second is *SGFF, Spiking Gate Feed-Forward*, which can convert the floating-point element-wise product to a binary logical AND (&) operation.

The contributions of this study can be summarized as:

1. One-step Spiking Transformer (OST) is proposed that can achieve SOTA performance with low energy use.
2. Time Domain Compression and Compensation (TDCC) is introduced, which reduces timestep to only 1, significantly lowering transformer's spatio-temporal overhead.
3. Spiking Linear Transformation (SLT) is introduced, greatly reducing floating-point MAC operations.

2 Related Works

2.1 Vision Transformers (ViTs)

ViTs achieved remarkable performance in various computer vision tasks, such as image classification [Dosovitskiy *et al.*, 2020] and object detection [Carion *et al.*, 2020]. Despite their success, one of the main challenges faced by ViTs is the high computational complexity caused by the self-attention mechanism [Khan *et al.*, 2022]. The self-attention mechanism enables ViTs to selectively focus on different parts of the input image, effectively capturing global information. However, the calculation of dot products between each pair of input features becomes inevitable, resulting in a quadratic computational complexity, relative to the input sequence length.

Several recent methods have been proposed to address the above challenge. Sparse Transformer [Child *et al.*, 2019] used sparse attention mechanisms as a remedy, while the Synthesizer [Tay *et al.*, 2021] introduced approximate attention. AFT [Zhai *et al.*, 2021] utilized attention-free models for the same purpose. These methods still involve plenty of floating-point operations, not ideal for SNNs. Besides, they all need costly MAC operations, not helpful for energy saving.

2.2 Spiking Neural Networks (SNNs)

Training SNNs poses a significant challenge because their spike signals are non-differentiable, rendering stochastic gradient descent (SGD) ineffective. Two typical strategies are often used in practice for the training: (1) converting ANN to SNN (ANN-to-SNN) [Rueckauer *et al.*, 2017] and (2) direct training [Neftci *et al.*, 2019]. More specifically, the ANN-to-SNN strategy converts an ANN pre-trained with SGD to an SNN for a given task and often achieves competitive accuracy. For example, the works in [Mueller *et al.*, 2021] and [Wang *et al.*, 2023] can obtain a nearly equivalent accuracy compared to the vanilla ANN transformer models. However, ANN-to-SNN requires hundreds of timesteps to converge, causing significant inference latency and energy consumption. On the other hand, the direct training strategy uses surrogate gradients to replace discrete spikes with continuous functions, thereby bridging ANN transformers into spiking transformers across various tasks. Nevertheless, they still rely on multiple timesteps, which lead to higher energy consumption. Zhou *et al.* [Zhou *et al.*, 2023] proposed a spiking self-attention transformer using four timesteps, incurring nearly four times higher energy consumption compared to one timestep. In addition, Yao *et al.* [Yao *et al.*, 2023b] proposed attention spiking neural networks which do utilize one timestep. Despite significantly reduced energy consumption, the accuracy of their models suffers greatly with one timestep, unless the data are augmented. In comparison, this paper focuses on designing a spiking transformer with only one timestep while still maintaining high accuracy.

2.3 Leaky Integrate and Fire (LIF)

Spiking neurons transmit information in SNNs. They generate binary spike signals for transmission and are the fundamental part of SNNs. A key signal generation mechanism is LIF [Gerstner *et al.*, 2014], which is also used in this study. We follow the conventions in [Zheng *et al.*, 2021; Zhou *et al.*, 2023], using SpikingJelly [Fang *et al.*, 2023] to imply LIF. Their fire process is represented as Equation (1):

$$u_i^t = (1 - \frac{1}{\tau})u_i^{t-1} + \frac{1}{\tau}\sum_{j=1}^{n} w_{ij} o_j^{t-1}, \tag{1}$$

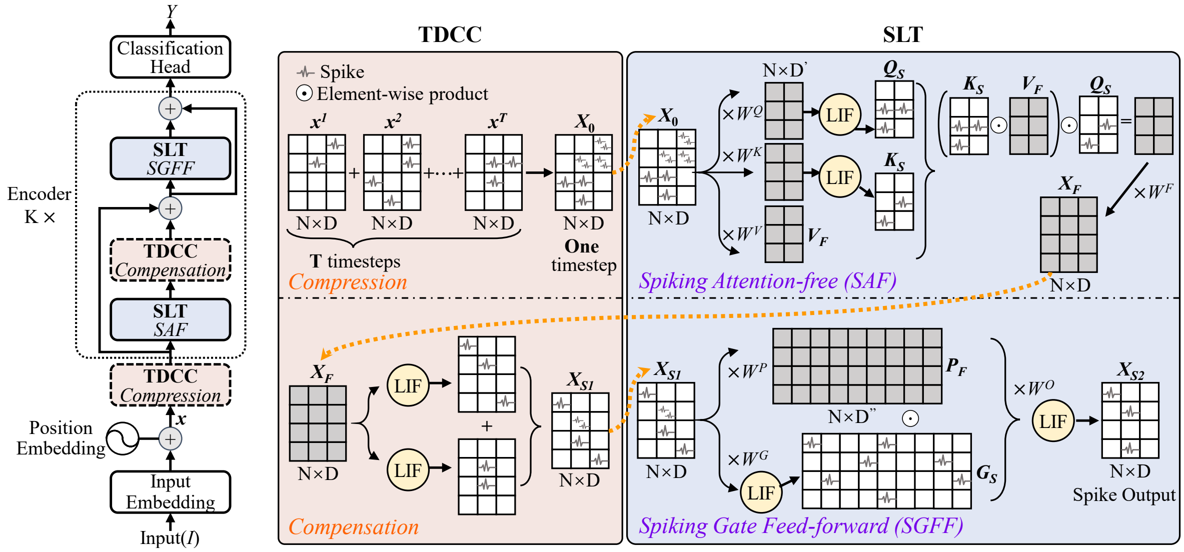
where u_i^t is the membrane potential of the i-th neuron at the t-th timestep; τ is the decay constant; w_{ij} is the connection weight between the i-th and the j-th neurons; and o_j^{t-1} is the spike signal generated by the j-th neuron at the $(t-1)$-th timestep. When u_i^t reaches the activation threshold, the spiking neuron fires. After the activation, the membrane potential is reset for the next cycle of accumulation.

3 Methodology

The proposed OST, the associated TDCC and SLT, and their sub-components are detailed in this section.

3.1 Overall Architecture

The overall architecture of OST is illustrated in Figure 1(a), where the proposed TDCC and SLT components are highlighted in different colors. The input of OST is a 2D image sequence $I \in \mathbb{R}^{T\times C\times H\times W}$, where T, C, H, W denote timestep, channel, height, and width, respectively. Then the input is embedded. I is transformed into $x \in \mathbb{R}^{T\times H\times W\times D}$. D represents the embedding dimension of OST. Note that OST uses Input Embedding (IE) and Position Embedding (PE) for embedding, following Spikformer. Subsequently, x is passed through the *Compression* module of TDCC, and is compressed from T timesteps to X_0, meaning one timestep. After that, X_0 is passed to the block of K encoders. Each encoder consists of three parts: *Spiking Attention-free (SAF)*



(a) The architecture of OST

(b) Implementation details of TDCC and SLT

Figure 1: The overview of One-step Spiking Transformer (OST), which contains a Time Domain Compression and Compensation (TDCC) component and a Spiking Linear Transformation (SLT) component. Spiking matrices with Spike flags are represented in a white background. Other matrices are floating-point, represented in gray.

of SLT, *Compensation* of TDCC, and *Spiking Gate Feed-forward (SGFF)* of SLT. The *SAF* is a linear transformer module that does not require a self-attention mechanism, while the *Compensation* module is to compensate for the information loss due to the compression. Moreover, the *SGFF* is a feed-forward module, more suitable for processing spike signals. The input of each encoder is passed into *SAF*, *Compensation*, and *SGFF* in turn to obtain the output X_K. We use skip connections before *SAF* and *SGFF* to avoid the vanishing gradient problem. Finally, X_K is fed into a fully connected layer, Classification Head (CH), to produce the prediction Y. The transformation process of the sequence I is shown below:

$$x = \mathrm{IE}(I) + \mathrm{PE}(I) \qquad I, x \in R_1, R_2 \tag{2}$$

$$X_0 = Compression(x) \qquad X_0 \in R_3 \tag{3}$$

$$X_{\mathcal{F}} = SAF(X_{k-1}) \qquad X_{\mathcal{F}} \in R_3 \tag{4}$$

$$X_k^{'} = Compensation(X_{\mathcal{F}}) + X_{k-1} \qquad X_{\mathcal{S}} \in R_3 \tag{5}$$

$$X_k = SGFF(X_k^{'}) + X_k^{'} \qquad X_k \in R_3 \tag{6}$$

$$Y = \mathrm{CH}(X_K), \tag{7}$$

where R_1, R_2, and R_3 are $\mathbb{R}^{T\times C\times H\times W}$, $\mathbb{R}^{T\times N\times D}$, and $\mathbb{R}^{1\times N\times D}$, respectively, and $k = 1 \cdots K$.

3.2 Time Domain Compression and Compensation (TDCC)

As shown in Figure 1(b), TDCC consists of two parts: *Compression* and *Compensation*.

Compression. It is to compress x in T timesteps to X_0, e.g. one timestep, as indicated by Equation (8):

$$X_0 = \sum_{t=1}^{T} x^t, \tag{8}$$

where x^t is the image sequence at the t-th timestep. The rationale behind *Compression* is that x contains T timesteps after embedding. If x is directly passed to the encoder, the transformation information needs to be computed at each timestep, resulting in significant spatio-temporal overhead.

In fact, the human brain also involves multiple timesteps while processing dynamic visual information [Rao and Ballard, 1999]. However, because of its remarkable capacity for information integration, it efficiently extracts crucial visual features with low latency and energy consumption. In particular, temporal integration employed by visual neurons plays a vital role in the processing of visual information [Wolff *et al.*, 2022]. Visual neurons receive inputs from the retina, as well as other regions of the visual cortex, and subsequently integrate and respond to these inputs. In this paper, we employ additive operations to create an analogy for such temporal integration, aiming to compress time-domain information. With the integration, each point in the matrix represents the intensity of the spike signals throughout all timesteps. However, compression naturally leads to information loss and subsequently performance deterioration (See Ablation Study, Section 4.3). Therefore, *Compensation* is introduced to regain more time-domain information.

Compensation. To accumulate time-domain information, we employ two LIF spiking neurons with different thresholds as illustrated in Equation (9):

$$X_{\mathcal{S}} = LIF_1(X_{\mathcal{F}}) + LIF_2(X_{\mathcal{F}}), \tag{9}$$

where LIF_1 and LIF_2 are two LIF spiking neurons with initial threshold values of 0.5 and 1, respectively. Furthermore, inspired by dynamic thresholds in SNNs [Hao *et al.*, 2020], the thresholds of LIF_1 and LIF_2 are set as learnable parameters. They update dynamically during training.

Neurons in the human brain are able to dynamically adjust their activation thresholds in response to changes in input information [Zhang and Linden, 2003]. We accumulate the spike signals at two different thresholds, to improve the diversity of time-domain information. To be more specifically, *Compensation* is to deal with possible information loss caused by compression. Its foundation is the rate coding of SNNs, converting information into spikes, making the number of spikes proportional to the information intensity [Kim *et al.*, 2022]. In this way, the information loss due to compression can be compensated, thus performance improved.

3.3 Spiking Linear Transformation (SLT)

As shown in Figure 1(b), OST also has two parts: *SAF*, Spiking Attention Free, and *SGFF*, Spiking Gate FeedForward.

SAF. It is of linear complexity and is a variant of the Attention-Free (AF) mechanism [Zhai *et al.*, 2021]. *SAF*'s spiking operations has the benefit of low energy consumption. More specifically, the vanilla AF linearly transforms the input $X_0 \in \mathbb{R}^{1\times N\times D}$ into $Q = X_0W^Q$, $K = X_0W^K$, and $V = X_0W^V$, where W^Q, W^K and $W^V \in \mathbb{R}^{D\times D'}$. D' is the hidden dimension of *SAF*. Subsequently, it performs the operations through Equations (10) and (11):

$$Q_{\mathcal{F}} = \sigma_q(Q), K_{\mathcal{F}} = softmax(K), V_{\mathcal{F}} = V, \tag{10}$$

$$\mathrm{AF}(Q_{\mathcal{F}}, K_{\mathcal{F}}, V_{\mathcal{F}}) = Q_{\mathcal{F}} \odot \sum_{d=1}^{D}(K_{\mathcal{F}} \odot V_{\mathcal{F}})_d, \tag{11}$$

where σ_q and $softmax$ are activation function Sigmoid and Softmax respectively; $\odot$ is the element-wise product.

AF uses element-wise operations, which completely gets rid of the need for dot product operations. Its complexity is $O(Ld)$ instead of $O(L^2d)$. The dot product operation violates the original design of SNN as it involves complex floating-point MAC operations. AF has the potential to replace the existing spiking dot attention. However, the vanilla AF could not be directly applied to SNN, since the fire value of the activation functions (σ_q and $softmax$) are floating-point, yet the process involves numerous floating-point exponent multiplication and division operations.

In comparison, our proposed *SAF* has much fewer floating-point operations, as shown in Figure 1(b). Following the same procedure as AF, we first obtain Q, K, and V by linear transformation from an input sequence X_0. After that, we convert Q and K into spiking sequences $Q_{\mathcal{S}}$ and $K_{\mathcal{S}}$ by spiking neurons. Finally, we employ the spiking sequences $Q_{\mathcal{S}}$, $K_{\mathcal{S}}$ along with the floating-point sequence $V_{\mathcal{F}}$ to execute the *SAF*, as in Equations (12) and (13):

$$Q_{\mathcal{S}} = LIF(Q), K_{\mathcal{S}} = LIF(K), V_{\mathcal{F}} = V, \tag{12}$$

$$SAF(Q_{\mathcal{S}}, K_{\mathcal{S}}, V_{\mathcal{F}}) = Q_{\mathcal{S}} \odot \sum_{d=1}^{D}(K_{\mathcal{S}} \odot V_{\mathcal{F}})_d, \tag{13}$$

where $Q_{\mathcal{S}}, K_{\mathcal{S}} \in \mathbb{R}^{1\times N\times D}$, and LIF is the spiking neuron. Since $V_{\mathcal{F}}$ is a floating-point matrix, the outcome of the element-wise product constructed in *SAF* is also a floating-point matrix. Subsequently, we feed this matrix into the *Compensation* module to obtain the spiking sequences output.

SGFF. It contains three linear transformations and two LIF spiking neurons, as in Equations (14) and (15):

$$G_{\mathcal{S}} = LIF(X_{\mathcal{S}}W^G), P_{\mathcal{F}} = X_{\mathcal{S}}W^P, \tag{14}$$

$$SGFF(G_{\mathcal{S}}, P_{\mathcal{F}}) = LIF((G_{\mathcal{S}} \odot P_{\mathcal{F}})W^O), \tag{15}$$

where $W^G, W^P \in \mathbb{R}^{D\times D''}$, and $W^O \in \mathbb{R}^{D''\times D}$. D'' is the hidden dimension of *SGFF*. For example, if the embedding dimension D is 128 and $D'' = 4D$, for an input sequence $X_{\mathcal{F}}$, the hidden dimension of $G_{\mathcal{S}}$ and $P_{\mathcal{F}}$ are expanded fourfold to 512 by linear transformations W^G and W^P. After computing the element-wise product between $G_{\mathcal{S}}$ and $P_{\mathcal{F}}$, the embedding dimension of the sequence is reversed back to 128 by the linear transformation W^O, allowing the sequence to pass to the next encoder.

The reason that we propose *SGFF* instead of using MLP in spiking transformers is to improve the performance. *SGFF* is a variant of gMLP [Liu *et al.*, 2021], with the additional incorporation of a gating mechanism. The gating mechanism enables the network to selectively learn and control the delivery of information. By controlling the open and closed states of the gate, gMLP can more effectively capture significant features and correlations in the input data, thus enhancing the representational ability of the model. However, the gate state in gMLP relies on an activation function like GELU, which involves a substantial number of floating-point computations. Furthermore, the element-wise product $\odot$ in gMLP is also calculated using floating-point multiplication, which is contrary to the intention of SNN. Consequently, we replace the GELU with the spiking neuron LIF to avoid excessive floating-point computations. Additionally, the *SGFF* mechanism downgrades the element-wise product $\odot$ to a binary logical AND (&) operation, which is certainly a more energy-efficient alternative to floating-point multiplication.

3.4 Theoretical Energy Consumption Analysis

This section analyzes the energy consumption of OST from a theoritical perspective. For the vanilla transformer, energy consumption is calculated as the product of the energy per floating-point operation (e.g., E_{MAC}, E_M) and the number of operations. In contrast, OST's energy consumption is determined by multiplying the energy per binary spike operation by the timestep, fire rate, and total number of operations. As shown in the first block (top two rows) of Table 2, all convolution (Conv) operations in the vanilla transformer are MAC operations. In comparison, only the first Conv in OST used to generate binary spikes is a MAC operation. The second block of Table 2 compares the self-attention of the vanilla transformer and AFT component of OST. The complexity of $f(Q, K, V)$ is linear, which is achieved by the element-wise product. Given that the elements in element-wise product are binary spikes (either 0 or 1), the operation can be executed as a mask operation, which requires no energy consumption [Yao *et al.*, 2023a]. Lastly, the third block (bottom three rows) of Table 2 addresses the MLP of the vanilla transformer and the SGFF component of OST. Unlike the vanilla transformer, OST incorporates an additional linear layer (Linear3) to implement a binary gating mechanism. A more detailed analysis can be found in **Supplementary S1**.

Methods	Architecture	Parameters (M)	OPs (G)	Power (mJ)	Timesteps	Accuracy (%)
Hybrid training [Rathi *et al.*, 2020]	ResNet-34	21.79	-	-	250	61.48
TET [Deng *et al.*, 2022]	Spiking-ResNet-34	21.79	-	-	6	64.79
	SEW-ResNet-34	21.79	-	-	4	68.00
Spiking ResNet [Hu *et al.*, 2021]	ResNet-34	21.79	65.28	59.30	350	71.61
	ResNet-50	25.56	78.29	70.93	350	72.75
STBP-tdBN [Zheng *et al.*, 2021]	Spiking-ResNet-34	21.79	6.50	6.39	6	63.72
SEW ResNet [Fang *et al.*, 2021a]	SEW-ResNet-34	21.79	3.88	4.01	4	67.0
	SEW-ResNet-50	25.56	4.83	4.89	4	67.78
	SEW-ResNet-101	44.55	9.30	8.91	4	68.76
	SEW-ResNet-152	60.19	13.72	12.89	4	**69.26**
Att MS ResNet [Yao *et al.*, 2023b]	Att-MS-ResNet-18	11.87	-	0.48	1	63.97
	Att-MS-ResNet-34	22.12	-	0.57	1	69.15
Vanilla Transformer	Transformer-8-512	29.68	8.33	38.34	N/A	**80.80**
SDT [Yao *et al.*, 2023a]	SDT-8-384	16.81	-	3.90	4	72.28
	SDT-8-512	29.68	-	4.50	4	74.57
Spikformer [Zhou *et al.*, 2023]	Spikformer-8-384	16.81	6.82	7.73	4	70.24
	Spikformer-8-512	20.31	11.09	11.58	4	73.38
OST (Ours)	OST-8-384	19.36	4.12	4.63	1^*	72.42
	OST-8-512	33.87	6.15	6.92	1^*	**74.97**

Table 1: Compare with SOTA SNNs on ImageNet. 'OPs (G)' denotes the synaptic operations [Merolla *et al.*, 2014] in SNN and floating-point operations in Vanilla Transformer. 'Power (mJ)' is the average theoretical energy for predicting one test image from ImageNet. * denotes the same initial timesteps as Spikformer. Transformer 'XXX-N-D' means an architecture with N encoder blocks and D embedding dimensions.

	Vanilla Trans.	OST (Ours)
Fisrt Conv	$E_{MAC} \cdot FL_C$	$E_{MAC} \cdot T \cdot R_C \cdot FL_C$
Other Conv	$E_{MAC} \cdot FL_C$	$E_{AC} \cdot T \cdot R_C \cdot FL_C$
Q, K, V	$E_{MAC} \cdot 3ND^2$	$E_{AC} \cdot R_1 \cdot 3ND^2$
$f(Q, K, V)$	$E_{MAC} \cdot 2N^2D$	$E_{AC} \cdot R_2 \cdot ND$
Scale	$E_M \cdot N^2$	-
Softmax	$E_{MAC} \cdot 2N^2$	-
Linear	$E_{MAC} \cdot FL_{L0}$	$E_{AC} \cdot R_{L0} \cdot FL_{L0}$
Linear1	$E_{MAC} \cdot FL_{L1}$	$E_{AC} \cdot R_{L1} \cdot FL_{L1}$
Linear2	$E_{MAC} \cdot FL_{L2}$	$E_{AC} \cdot R_{L2} \cdot FL_{L2}$
Linear3	-	$E_{AC} \cdot R_{L3} \cdot FL_{L3}$

Table 2: Energy evaluation of vanilla Transformer [Dosovitskiy *et al.*, 2020] and OST. FL_C and FL_L represent the FLOPs of the Conv and Linear models in the ANNs, respectively. R_C, R_1, R_2, R_L denote the spike firing rates in various spike matrices.

4 Experiments

OST is validated on both static image classification, involving ImageNet, CIFAR10, and CIFAR100, and neuromorphic image classification, using CIFAR10-DVS [Li *et al.*, 2017] and DVS128 Gesture [Amir *et al.*, 2017]. These experiments are detailed in Section 4.1 and Section 4.2 respectively, while the ablation study is presented in Section 4.3.

Component	Matrix	Average spike firing rate
SAF	$Q_{\mathcal{S}}$	0.59181
	$K_{\mathcal{S}}$	0.10228
	Output ($X_{\mathcal{S}1}$)	0.05317
SGFF	$G_{\mathcal{S}}$	0.06896
	Output ($X_{\mathcal{S}2}$)	0.06966

Table 3: Average spike firing rate of spiking tensors in OST-8-512.

4.1 Static Image Classification

ImageNet. Following Spikformer, OST training utilizes 224×224 images from ImageNet and Adam [Kingma and Ba, 2014]. The learning rate is initially set to $6e^{-5}$ and progressively reduced using a cosine decay. The batch size and the epochs are 16 and 310 respectively. Table 1 presents the results, where the methods for comparison are all the SOTA convolutional SNNs (top block), vanilla Transformer (middle row), and spiking Transformers (bottom block, where OST is). OST achieved a SOTA accuracy of 74.97%. In particular, OST narrows the accuracy gap compared to the vanilla Transformer while achieving a 5.5× reduction in energy consumption. OST reduces energy consumption by 46.35% and improves accuracy by 5.91% compared to SEW-ResNet-152. Furthermore, OST outperforms Spikformer by 1.56% in terms of accuracy despite having a comparable size. It also requires 4.94G fewer operations and consumes 4.66mJ less power when predicting a single image. Furthermore, we analyzed the average sparsity of different matrices of OST-8-512, as shown in Table 3, the output of SGFF exhibits high sparsity, indicating that the spike gating mechanism effectively selects critical information (Details in **Supplementary S2**).

CIFAR 10 & CIFAR 100. Also following Spikformer, the input image size here is 32×32 with a batch size of 128. The results are presented in Table 4. OST achieved the best performance on both CIFAR10 and CIFAR100 using one timestep. In particular, OST-4-256 and OST-2-384 achieved accuracy improvements of 1.19% and 0.67%, respectively, compared to Spikformer-4-256 and Spikeformer-2-384. In addition, OST-4-384 achieves the accuracy of 95.64% on CIFAR10, which is 1.1% and 0.35% higher than TET and Spikformer-4-384, respectively. On CIFAR100, OST-4-256 and OST-2-384 gain 1.09% and 0.52% compared to Spikformer-4-256 and Spikeformer-2-384, respectively, while OST-4-384 outperforms TET and Spikformer-4-384 by 4.31% and 0.95%,

Methods	**Architecture**	**Params (M)**	**Power (mJ)**	**Timesteps**	**CIFAR10 Acc.(%)**	**CIFAR100 Acc.(%)**
Hybrid training [Rathi *et al.*, 2020]	VGG-11	9.27	-	125	92.22	67.87
Diet-SNN [Rathi and Roy, 2021]	ResNet-20	10.91	-	10/5	92.54	64.07
ANN-to-SNN [Deng and Gu, 2021]	ResNet-20	10.91	-	32	93.30	68.40
TET [Deng *et al.*, 2022]	ResNet-19	12.63	-	4	**94.44**	**74.47**
STBP-tdBN [Zheng *et al.*, 2021]	ResNet-19	12.63	-	4	92.92	70.86
Vanilla Transformer	Transformer-4-384	9.32	4.25	N/A	**96.73**	**81.02**
SDT [Yao *et al.*, 2023a]	SDT-4-384	9.36	0.31	4	95.45	78.34
	SDT-2-512	9.76	0.42	4	95.6	78.4
Spikformer [Zhou *et al.*, 2023]	Spikformer-4-256	**4.15**	0.35	4	93.94	75.96
	Spikformer-2-384	5.76	0.59	4	94.80	76.95
	Spikformer-4-384	9.32	0.79	4	95.19	77.86
OST (Ours)	OST-4-256	5.15	**0.22**	1^*	95.13/95.12 ± 0.02	77.05/77.02 ± 0.03
	OST-2-384	7.62	0.42	1^*	95.47/95.44 ± 0.04	77.47/77.42 ± 0.06
	OST-4-384	11.37	0.46	1^*	95.64/95.63 ± 0.03	**78.76**/78.62 ± 0.14
	OST-2-512	10.29	0.72	1^*	**95.68**/95.66 ± 0.02	78.47/78.39 ± 0.07

Table 4: Compare with SOTA SNN methods on 'CIFAR10' and 'CIFAR100'. OST provides statistical results for five runs, separated by "/".

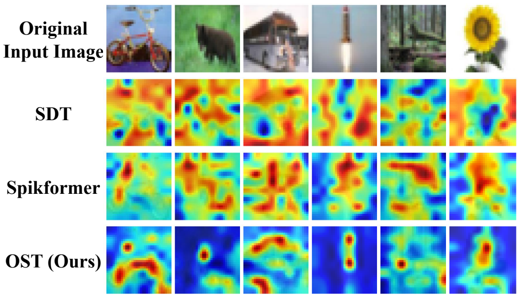


Figure 2: Visualization results of SDT-4-384, Spikformer-4-384, and OST-4-384 at the last block on CIFAR100.

respectively. Compared to the vanilla Transformer, OST further narrows the accuracy gap to 1.19% and 2.26% on CIFAR10 and CIFAR100, respectively. Such a gap is supposedly due to the information loss during the binarization of spike communication in OST, while Vanilla Transformer uses floating point communication, and its energy consumption is 4.25mJ, which is more than 9× higher than that of OST. We visualize the output of three SNNs, as in Figure 2. OST can provide more consistent coverage of the target object in comparison with Spikformer and SDT. Our results are statistically better as the P-values are all way below 0.05, reject the null hypothesis. More detailed experiments on different model scales with different parameters are in **Supplementary S3**.

4.2 Neuromorphic Image Classification

CIFAR10-DVS. This is an event-stream neuromorphic dataset containing 10,000 images, each of which was converted from CIFAR10 using a Dynamic Vision Sensor (DVS). The input image size is 128 × 128, with batch size 16. The learning rate is $1e^{-3}$ initially and reduces using a cosine decay. The initial timesteps here is not 4 as in Section 4.1, but 16 due to the increased task difficulty in classifying neuromorphic images. The training epoch is set to 106. The number of transformer encoder blocks N is set to 2, while the embedding dimension D is 256. We also use neuromorphic data augmentation, which is consistent as that in the literature. The results are presented in Table 5, which shows that OST, still with ONLY one timestep, outperforms others, including two SOTA SNNs, SDT (80.0%) and Spikeformer (80.9%).

Methods	**Spikes**	**Timesteps**	**Acc. (%)**
LIAF-Net [Wu *et al.*, 2021]	✗	10	70.4
TA-SNN [Yao *et al.*, 2021]	✗	10	72.0
Rollout [Kugele *et al.*, 2020]	✓	48	66.8
tdBN [Zheng *et al.*, 2021]	✓	10	67.8
PLIF [Fang *et al.*, 2021b]	✓	20	74.8
SEW-ResNet [Fang *et al.*, 2021a]	✓	16	74.4
Dspike [Li *et al.*, 2021]	✓	10	75.4^+
SALT [Kim and Panda, 2021]	✓	20	67.1
DSR [Meng *et al.*, 2022]	✓	10	77.3^+
SDT [Yao *et al.*, 2023a]	✓	16	80.0^+
Spikformer [Zhou *et al.*, 2023]	✓	16	80.9^+
OST (Ours)	✓	1^*	$\mathbf{81.2}^+/80.5 \pm 0.9$

Table 5: Compare with SOTA SNN methods on CIFAR10-DVS. $^+$ denotes using neuromorphic data augmentation [Li *et al.*, 2022].

DVS128 Gesture. This dataset contains 11 hand gesture categories from 29 individuals under 3 illumination conditions. The input size here is 128 × 128 and batch size 16. The initial learning rate is $1e^{-3}$ with a training epoch of 200. To minimize overfitting, we use two transformer encoder blocks with 256 embedding dimensions in the experiments. The results are presented in Table 6, where we achieved an accuracy of 99.0%. Compared to the TA-SNN (60 timesteps), we achieved the same accuracy but again, with ONLY one timestep. Moreover, to achieve a similar performance as OST, SDT and Spikformer need 16 timesteps instead of OST's 1 timestep. The cases of successful classification of three methods are analyzed in Figure 3. The analysis shows that OST places greater emphasis on the target, whereas SDT and Spkiformer exhibit a more dispersed pattern of spiking activations. More detailed experiments and analysis on this part of the study are presented in **Supplementary S4**.

4.3 Ablation Study

This ablation study of OST uses CIFAR100 to verify the effectiveness of TDCC and SLT. Note that all experiments follow the same setup as the above sections, if not specified. In addition, the impact of timesteps on spiking transformers, including SDT, Spikformer and our OST, are studied as well.

Methods	Spikes	Timesteps	Acc. (%)
LIAF-Net [Wu *et al.*, 2021]	✗	60	97.6
TA-SNN [Yao *et al.*, 2021]	✗	60	98.6
Rollout [Kugele *et al.*, 2020]	✓	240	97.2
DECOLLE [Kaiser *et al.*, 2020]	✓	500	95.5
tdBN [Zheng *et al.*, 2021]	✓	40	96.9
PLIF [Fang *et al.*, 2021b]	✓	20	97.6
SEW-ResNet [Fang *et al.*, 2021a]	✓	16	97.9
Att MS ResNet [Yao *et al.*, 2023b]	✓	20	98.2
SDT [Yao *et al.*, 2023a]	✓	16	**99.3**
Spikformer [Zhou *et al.*, 2023]	✓	16	98.3
OST (Ours)	✓	1*	99.0/98.4 ± 0.4

Table 6: Compare with SOTA SNN methods on DVS128 Gesture.

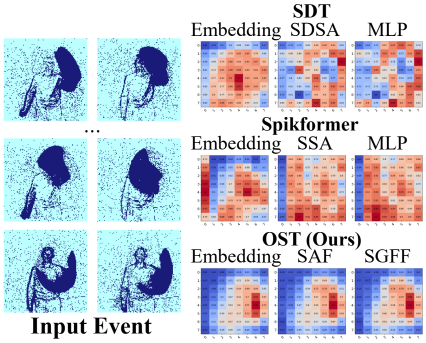


Figure 3: The heat map of SDT (with Spike-Driven Self-Attention (SDSA) and MLP), Spikformer (with Spiking Self Attention (SSA) and MLP), and OST (with *SAF* and *SGFF*) on DVS128 Gesture.

TDCC. To evaluate the impact of Compression and Compensation, we apply them on SDT (with SDT's SDSA and MLP), Spikformer (with Spikformer's SSA and MLP) and OST (with *SAF* and *SGFF*). The results are in Table 7, where '✓' means the presence of the method, yet '✗' is the absence. Looking at the first two rows of each method, *Compression* clearly reduces accuracies but lowers energy consumption as well. When *Compensation* is introduced (the third row of each method), accuracy bounces back with a slight increase in energy use, showing TDCC can achieve a good balance between accuracy and energy consumption.

SLT. The effectiveness study of SLT's two parts, *SAF* and *SGFF*, is presented in Table 8. Compared with the first row, where both *SAF* and *SGFF* are absent, the second row shows *SAF* improves in both accuracy and energy consumption. The third row shows *SGFF* brings even more improvement in accuracy but results in much higher energy as well. The results suggest that *SAF* can reduce power consumption, while *SGFF* can improve accuracy.

Initial Timesteps. The impact of the initial timesteps T is investigated on various tasks. In general, fewer timesteps would affect other spiking transformers quite negatively, especially on more difficult tasks like neuromorphic image classification. See details in **Supplementary S5, S6**.

Summary of Experiments. The experiments in the main paper and Supplementary have demonstrated that: (1) OST performs as well as or better than SOTA methods across static and neuromorphic datasets. (2) OST runs faster, especially noticeable when timesteps $T \geq 4$. The speed of OST is $2\times$ more than SOTA method with 8 timesteps. This is particularly important for tasks that may require more timesteps when using other methods. (3) OST performs better, particularly with small timesteps ($T < 4$). When $T = 2$, we achieved 5.6% higher accuracy compared to the SOTA methods. OST exhibits a very short inference time yet has high performance. (4) OST is less sensitive to the reduction of timesteps (even with only one timestep). It is crucial for deployment on mainstream neuromorphic hardware in practice.

Methods	TDCC		Acc.	Power
	Compression	*Compensation*	(%)	(mJ)
SDT	✗	✗	78.34	0.31
	✓	✗	77.26	**0.19**
	✓	✓	77.83	0.25
Spikformer	✗	✗	77.37	0.79
	✓	✗	76.73	0.44
	✓	✓	77.62	0.55
OST (Ours)	✗	✗	**78.85**	0.85
	✓	✗	77.68	0.42
	✓	✓	78.76	0.46

Table 7: Ablation studies of TDCC's *Compression* and *Compensation* on CIFAR-100. *Compensation* aims to enhance performance after *Compression*, hence no separate use of *Compensation*.

SLT		Acc.	Power
SAF	*SGFF*	(%)	(mJ)
✗	✗	77.37	0.79
✓	✗	78.48	**0.66**
✗	✓	**79.01**	1.12
✓	✓	78.85	0.85

Table 8: Ablation studies of *SAF* and *SGFF* on CIFAR-100.

5 Conclusion

This paper proposes One-step Spiking Transformer (OST), which only needs one timestep in the transformer block, and is of a linear complexity. To achieve this goal, two key components, Time Domain Compression and Compensation (TDCC) and Spiking Linear Tranfomration (SLT) are introduced. The former can compress information from multiple timesteps into one, reducing the spatio-temporal overhead while maintaining performance. The latter further reduces the floating-point MAC operations of the model. Its complexity is as low as linear. The efficacy of OST is verified on both static and neuromorphic image sets. OST can achieve SOTA performance without increasing timesteps even for more difficult tasks. The ablation studies show that TDCC and SLT are indeed effective in reducing energy consumption while improving accuracy. Hence, OST is a strong candidate in SNN.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant 62276175.

References

[Amir *et al.*, 2017] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, et al. A low power, fully event-based gesture recognition system. In *Proc. of CVPR*, 2017.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Proc. of NeurIPS*, 2020.

[Carion *et al.*, 2020] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proc. of ECCV*, 2020.

[Child *et al.*, 2019] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[Deng and Gu, 2021] Shikuang Deng and Shi Gu. Optimal conversion of conventional artificial neural networks to spiking neural networks. In *Proc. of ICLR*, 2021.

[Deng *et al.*, 2022] Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting. In *Proc. of ICLR*, 2022.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. of ICLR*, 2020.

[Fang *et al.*, 2021a] Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. *Proc. of NeurIPS*, 2021.

[Fang *et al.*, 2021b] Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proc. of ICCV*, 2021.

[Fang *et al.*, 2023] Wei Fang, Yanqi Chen, Jianhao Ding, Zhaofei Yu, Timothée Masquelier, Ding Chen, Liwei Huang, Huihui Zhou, Guoqi Li, and Yonghong Tian. Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, 9(40):eadi1480, 2023.

[Fedus *et al.*, 2022] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.

[Gerstner *et al.*, 2014] Wulfram Gerstner, Werner M Kistler, Richard Naud, and Liam Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[Hao *et al.*, 2020] Yunzhe Hao, Xuhui Huang, Meng Dong, and Bo Xu. A biologically plausible supervised learning method for spiking neural networks using the symmetric stdp rule. *Neural Networks*, 121:387–395, 2020.

[Hu *et al.*, 2021] Yangfan Hu, Huajin Tang, and Gang Pan. Spiking deep residual networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[Hua *et al.*, 2022] Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer quality in linear time. In *Proc. of ICML*, 2022.

[Kaiser *et al.*, 2020] Jacques Kaiser, Hesham Mostafa, and Emre Neftci. Synaptic plasticity dynamics for deep continuous local learning (decolle). *Frontiers in Neuroscience*, 14:424, 2020.

[Katharopoulos *et al.*, 2020] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proc. of ICML*, 2020.

[Khan *et al.*, 2022] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM Computing Surveys (CSUR)*, 54(10s):1–41, 2022.

[Kim and Panda, 2021] Youngeun Kim and Priyadarshini Panda. Optimizing deeper spiking neural networks for dynamic vision sensing. *Neural Networks*, 144:686–698, 2021.

[Kim *et al.*, 2020] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: spiking neural network for energy-efficient object detection. In *Proc. of AAAI*, 2020.

[Kim *et al.*, 2022] Youngeun Kim, Hyoungseob Park, Abhishek Moitra, Abhiroop Bhattacharjee, Yeshwanth Venkatesha, and Priyadarshini Panda. Rate coding or direct coding: Which one is better for accurate, robust, and energy-efficient spiking neural networks? In *Proc. of ICASSP*. IEEE, 2022.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kugele *et al.*, 2020] Alexander Kugele, Thomas Pfeil, Michael Pfeiffer, and Elisabetta Chicca. Efficient processing of spatio-temporal data streams with spiking neural networks. *Frontiers in Neuroscience*, 14:439, 2020.

[Li *et al.*, 2017] Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in Neuroscience*, 11:309, 2017.

[Li *et al.*, 2021] Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Proc. of NeurIPS*, 2021.

[Li *et al.*, 2022] Yuhang Li, Youngeun Kim, Hyoungseob Park, Tamar Geller, and Priyadarshini Panda. Neuromorphic data augmentation for training spiking neural networks. In *Proc. of ECCV*, 2022.

[Liu *et al.*, 2021] Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. *Proc. of NeurIPS*, 2021.

[Maass, 1997] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997.

[Meng *et al.*, 2022] Qingyan Meng, Mingqing Xiao, Shen Yan, Yisen Wang, Zhouchen Lin, and Zhi-Quan Luo. Training high-performance low-latency spiking neural networks by differentiation on spike representation. In *Proc. of CVPR*, 2022.

[Merolla *et al.*, 2014] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

[Mueller *et al.*, 2021] Etienne Mueller, Viktor Studenyak, Daniel Auge, and Alois Knoll. Spiking transformer networks: A rate coded approach for processing sequential data. In *Proc. of ICSAI*, 2021.

[Neftci *et al.*, 2019] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. 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.

[Patterson *et al.*, 2021] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

[Rao and Ballard, 1999] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999.

[Rathi and Roy, 2021] Nitin Rathi and Kaushik Roy. Diet-snn: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[Rathi *et al.*, 2020] Nitin Rathi, Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation. In *Proc. of ICLR*, 2020.

[Roy *et al.*, 2019] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

[Rueckauer *et al.*, 2017] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017.

[Tay *et al.*, 2021] Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. Synthesizer: Rethinking self-attention for transformer models. In *Proc. of ICML*, 2021.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Proc. of NeurIPS*, 2017.

[Wang *et al.*, 2023] Ziqing Wang, Yuetong Fang, Jiahang Cao, Qiang Zhang, Zhongrui Wang, and Renjing Xu. Masked spiking transformer. In *Proc. of ICCV*, 2023.

[Wolff *et al.*, 2022] Annemarie Wolff, Nareg Berberian, Mehrshad Golesorkhi, Javier Gomez-Pilar, Federico Zilio, and Georg Northoff. Intrinsic neural timescales: temporal integration and segregation. *Trends in Cognitive Sciences*, 26(2):159–173, 2022.

[Wu *et al.*, 2019] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. Direct training for spiking neural networks: Faster, larger, better. In *Proc. of AAAI*, 2019.

[Wu *et al.*, 2021] Zhenzhi Wu, Hehui Zhang, Yihan Lin, Guoqi Li, Meng Wang, and Ye Tang. Liaf-net: Leaky integrate and analog fire network for lightweight and efficient spatiotemporal information processing. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11):6249–6262, 2021.

[Yao *et al.*, 2021] Man Yao, Huanhuan Gao, Guangshe Zhao, Dingheng Wang, Yihan Lin, Zhaoxu Yang, and Guoqi Li. Temporal-wise attention spiking neural networks for event streams classification. In *Proc. of ICCV*, 2021.

[Yao *et al.*, 2023a] Man Yao, JiaKui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo XU, and Guoqi Li. Spike-driven transformer. In *Proc. of NeurIPS*, 2023.

[Yao *et al.*, 2023b] Man Yao, Guangshe Zhao, Hengyu Zhang, Yifan Hu, Lei Deng, Yonghong Tian, Bo Xu, and Guoqi Li. Attention spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[Zhai *et al.*, 2021] Shuangfei Zhai, Walter Talbott, Nitish Srivastava, Chen Huang, Hanlin Goh, Ruixiang Zhang, and Josh Susskind. An attention free transformer. *arXiv preprint arXiv:2105.14103*, 2021.

[Zhang and Linden, 2003] Wei Zhang and David J Linden. The other side of the engram: experience-driven changes in neuronal intrinsic excitability. *Nature Reviews Neuroscience*, 4(11):885–900, 2003.

[Zheng *et al.*, 2021] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proc. of AAAI*, 2021.

[Zhou *et al.*, 2023] Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng YAN, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. In *Proc. of ICLR*, 2023.