Reproduction of Paper: Spike-driven Transformer

Project of AI3610 Brain-inspired Intelligence, 2023 Fall, SJTU

Xiangyuan Xue
521030910387

Shengmin Yang
521030910430

Yi Ai
521030910429

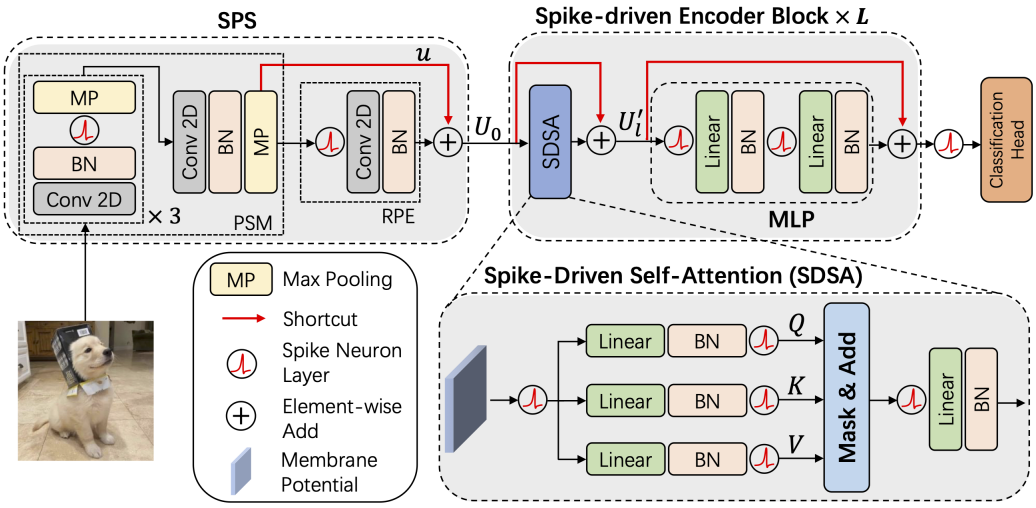

Figure 1. The overview of Spike-driven Transformer. Spike-driven Transformer is composed of 3 parts: spiking patch splitting, spike-driven encoder and classification head. The spiking patch splitting converts an image into a membrane potential embedding. The core of the spike-driven encoder is Spike-driven Self-Attention, which consists of only mask and sparse addition operations.

Abstract

Spiking Neural Networks (SNNs) are one of the most promising topics in the field of brain-inspired intelligence due to their unique spike-based event-driven paradigm. In this project, we successfully reproduce Spike-driven Transformer, verifying the effectiveness of the proposed method. We utilize the latest frameworks to simplify the original implementation, enhancing its readability and maintainability. In addition, we propose the method of transfer learning for training Spike-driven Transformer, which significantly reduces the training expenses. Our code is available at `https://github.com/Ark-ike/spike-driven-transformer`.

1. Introduction

Spiking Neural Networks (SNNs) are one of the most promising topics in the field of brain-inspired intelligence. They are biologically plausible and have the potential to achieve high energy efficiency. However, limited by the lack of effective training methods, SNNs cannot achieve the same performance as Artificial Neural Networks (ANNs) in most tasks. In recent years, ANNs have developed more network architectures with fancy performance, such as ResNet [5] and Transformer [9], while SNNs fail to keep up with the pace. Therefore, it is of great significance to introduce the powerful network architectures into SNNs and explore the potential of brain-inspired intelligence. There are many works devoted in this direction. For example, [2] first proposes an unsupervised method for training deep SNNs. [6] introduces the classical YOLO [8] architecture into SNNs and implements real-time object detection. [11] proposes threshold-dependent batch normalization to make it possible to train deep SNNs with back propagation. [3] follows the architecture of ResNet [5] and achieves the state-of-the-art performance on image classification tasks.

Many previous works have tried to introduce the Transformer architecture into SNNs, of which the most represen-

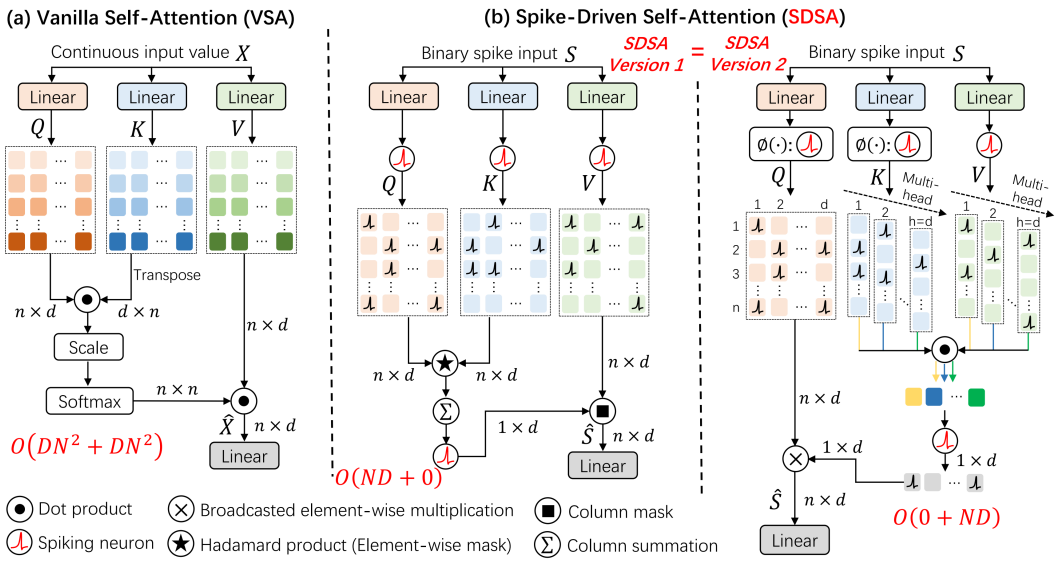

Figure 2. Comparison Vanilla Self-Attention (VSA) and Spike-Driven Self-Attention (SDSA)

tative is Spikformer [12], which proposes a spiking version of Transformer. However, Spikformer not only yields large integers in the output, requiring an additional scale multiplication for normalization to avoid gradient vanishing, but also fails to exploit the full energy-efficiency potential of the spike-driven paradigm combined with self-attention. To address these problems, Spike-driven Transformer [10] comes up with a novel attention mechanism, Spike-driven Self-Attention, which only consists of mask and sparse addition operations with essentially no energy consumption. In addition, the residual connections are rearranged so that all the spiking neurons in Spike-driven Transformer communicate via binary spikes. Spike-driven Transformer is claimed to achieve the state-of-the-art performance on multiple common datasets and be friendly to neuromorphic hardware.

In this project, we implement Spike-driven Transformer [10] and reproduce the results on several image classification tasks. We implement the network architecture and the training process by ourselves with the latest deep learning framework. In addtition, we propose some method innovations to improve the performance and efficiency of Spike-driven Transformer. We summarize our contributions in three main aspects:

- We implement Spike-driven Transformer [10] and reproduce the results on several image classification tasks, thus verifying the effectiveness of Spike-driven Transformer.
- We implement the network architecture and the training process by ourselves with the latest deep learning framework, which utilizes the newest features of the SpikingJelly framework [4], thus significantly simplifying the implementation of Spike-driven Transformer.
- We propose the method of transfer learning for Spike-driven Transformer, where the model is first pretrained on a large dataset and then finetuned on a small dataset, which can slightly improve the performance and significantly reduce the training time, thus solving the problem of large training expense of Spike-driven Transformer.

2. Method

The overview of Spike-driven Transformer is shown in Figure 1. Spike-driven Transformer includes 3 parts: Spiking Patch Splitting (SPS), Spike-driven Encoder Block (SEB) and Classification Head (CH). Spike-Driven Self-Attention (SDSA) is the core module in Spike-driven Encoder.

2.1. Spiking Patch Splitting

Spiking Patch Splitting converts a 2D image sequence $I \in \mathbb{R}^{T\times C\times H\times W}$ into a membrane potential embedding. We implement Spiking Patch Splitting as the paper [10] described. Given a 2D image sequence $I \in \mathbb{R}^{T\times C\times H\times W}$, the Patch Splitting Module (PSM) linearly projects and splits it into a sequence of N flattened spike patches s with D channels, where T, C, H, and W respectively denote the number of timesteps, the number of channels and the height and width of the image. Another convolutional layer is then used to generate Relative Position Embedding (RPE). Spiking Patch Splitting can be formulated as:

$$\begin{aligned} u &= \mathrm{PSM}\left(I\right), && u \in \mathbb{R}^{T\times N\times D} \\ s &= \mathcal{SN}(u), && s \in \mathbb{R}^{T\times N\times D} \\ \mathrm{RPE} &= \mathrm{BN}(\mathrm{Conv2d}(s)), && \mathrm{RPE} \in \mathbb{R}^{T\times N\times D} \\ U_0 &= u + \mathrm{RPE}, && U_0 \in \mathbb{R}^{T\times N\times D} \end{aligned} \tag{1}$$

where u and U_0 are the output membrane potential tensor of Patch Splitting Module and Spiking Patch Splitting respectively, and $\mathcal{SN}(\cdot)$ denotes the spike neuron layer.

2.2. Spike-Driven Self-Attention

The Vanilla Self-Attention (VSA) [9] can be formulated as:

$$\mathrm{VSA}(Q,K,V)=\mathrm{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{2}$$

where the attention weight is calculated by the dot product of the query and key and normalized by the `softmax` function, which is not suitable for SNNs. As shown in Figure 2, the paper proposes a novel attention mechanism, Spike-Driven Self-Attention (SDSA), which uses Hadamard product to replace matrix multiplication and uses matrix column-wise summation and spiking neuron layer to replace scaling and the `softmax` function. Given a spike input feature sequence $S \in \mathbb{R}^{T\times N\times D}$, float-point Q, K, and V in $\mathbb{R}^{T\times N\times D}$ are calculated by three learnable linear matrices respectively. Then a spike neuron layer $\mathcal{SN}(\cdot)$ converts them into spike tensors Q_S, K_S, and V_S. The Spike-Driven Self-Attention can be formulated as:

$$\mathrm{SDSA}(Q,K,V)=Q_S\otimes\mathcal{SN}\left(\mathrm{SUM_c}\left(K_S\otimes V_S\right)\right) \tag{3}$$

where $\otimes$ represents the Hadamard product and $\mathrm{SUM_c}(\cdot)$ represents the column-wise summation. The Hadamard product between spike tensors is equivalent to the mask operation. After the aforementioned improvements, the computational complexity of SDSA is $O(ND)$, which is linear with both N and D, much better than that of VSA, which is $O(N^2D)$.

2.3. Spike-driven Encoder Block

Spike-driven Encoder Block, inspired by the classical architecture in the Transformer [9], consists of a self-attention mechanism and a feed-forward neural network. The overall network architecture is composed of L such encoder blocks cascaded to create a deep structure, allowing the model to capture long-range dependencies in the input sequences. After passing through Spiking Patch Splitting, we feed U_0 into the Spike-driven Encoder Block. Residual connections [5] are also applied in both Spike-Driven Self-Attention and the feed-forward neural network. Spike-Driven Self-Attention provides an efficient approach to model the local-global information of images utilizing spike Q, K, and V without scaling and the `softmax` function. The Spike-driven Encoder Block can be formulated as:

$$\begin{aligned}
S_0 &= \mathcal{SN}(U_0), && S_0\in\mathbb{R}^{T\times N\times D}\\
U_l^{'} &= \mathrm{SDSA}(S_{l-1})+U_{l-1}, && U_l^{'}\in\mathbb{R}^{T\times N\times D}\\
S_l^{'} &= \mathcal{SN}(U_l^{'}), && S_l^{'}\in\mathbb{R}^{T\times N\times D}\\
S_l &= \mathcal{SN}(\mathrm{MLP}(S_l^{'})+U_l^{'}), && S_l\in\mathbb{R}^{T\times N\times D}
\end{aligned} \tag{4}$$

where $U_l^{'}$ and $S_l^{'}$ represent membrane potential and spike tensor generated by Spike-Driven Self-Attention at the l-th layer. As a supplement, the classification head utilizes a global average-pooling and a fully connected layer to output the prediction.

2.4. Simplified Implementation

The implementation provided in the paper [10] contains lots of redundant code and uses outdated frameworks, significantly lacking readability and maintainability, which brings lots of difficulties for our reproduction. In this project, we eliminate redundant parts and utilize the latest version of `spikingjelly`, which supports multi-step propagation, thus significantly improving the computing efficiency. In addition, we utilize `einops` for powerful and flexible tensor operations to simplify the code and enhance the readability, while maintaining the same network architecture as the original implementation. All the function blocks are implemented within $1/6$ code lines after simplification. A comparison of the number of code lines between our implementation and the original one is shown in Figure 3.

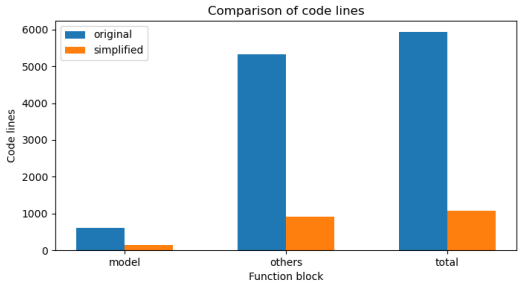

Figure 3. Comparison of code lines between our implementation and the original one. We implement all the function blocks within only $1/6$ code lines with the latest frameworks.

2.5. Transfer Learning

Transfer learning [13] is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. It is a popular approach in deep learning where pretrained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.

As the experiments progress, we notice that training models from scratch for each classification task requires a large amount of computing resources, which is hard for us to afford. We propose the method of transfer learning for Spike-driven Transformer, where the model is first pretrained on a large dataset and then finetuned on a small

dataset, which can slightly improve the performance and significantly reduce the training time, thus solving the problem of large training expense of Spike-driven Transformer. The method is proved to be useful in training Spike-driven Transformer, which will be described in detail in Section 3.

3. Experiments

We follow the paper to implement Spike-driven Transformer and conduct experiments on the common datasets. In addition, we innovatively introduce transfer learning where we pretrain the model on a large dataset and then finetune it on a small dataset. The details and results of the experiments are described as follows.

3.1. Experiment Setup

Dataset. ImageNet-1K is the most typical static image dataset, which is widely used in the field of image classification. It offers a large-scale natural image dataset of 1.28M training images and 50k test images, with a total of 1k categories. CIFAR-10 and CIFAR-100 are smaller datasets in image classification tasks that are often used for algorithm testing. The CIFAR-10 dataset consists of 60k images in 10 classes, with 6k images per class. The CIFAR-100 dataset has 60k images divided into 100 classes, each with 600 images. CIFAR-10-DVS is an event-based neuromorphic dataset converted from CIFAR-10 by scanning each image with repeated closed-loop motion in front of a Dynamic Vision Sensor (DVS). There are a total of 10k samples in CIFAR-10-DVS. DVS-128-Gesture is an event-based gesture recognition dataset. It records 1342 samples of 11 gestures.

We follow the paper to evaluate our model on both static datasets including CIFAR-10 and CIFAR-100 and neuromorphic datasets including CIFAR-10-DVS [7] and DVS-128-Gesture [1]. Considering the limited time and computing resources, we use the ImageNet-1k dataset only for a few rounds of pretraining instead of complete training.

Implementation details. We mainly use Spike-driven Transformer with 4 encoder blocks, 8 heads and 512 channels. The model is trained for 300 epochs on a single NVIDIA Tesla V100 GPU. We use the AdamW optimizer with a weight decay of 0.01 and a learning rate of 0.0005. The learning rate is warmed up for 10 epochs and then decayed by cosine annealing for 40 epochs. The batch size is 32 for static datasets and 16 for neuromorphic datasets. The number of timesteps is 4 for static datasets and 16 for neuromorphic datasets. Specifically, we employ simple data augmentation including random cropping, horizontal flipping and color jittering for static datasets.

3.2. Result Analysis

We conduct experiments on the CIFAR-10, CIFAR-100, CIFAR-10-DVS and DVS-128-Gesture datasets. The results are shown in Table 1. The training curves of the loss and accuracy are shown in Figure 4.

Table 1. Experiment results on CIFAR-10, CIFAR-100, CIFAR-10-DVS and DVS-128-Gesture. Spike-driven Transformer with 4 encoder blocks, 8 heads and 512 channels is employed.

Dataset	#Class	#Timestep	Accuracy
CIFAR-10	10	4	92.90%
CIFAR-100	100	4	72.44%
CIFAR-10-DVS	10	16	75.00%
DVS-128-Gesture	11	16	98.96%

We can see that the model achieves 92.90% accuracy on CIFAR-10 and 72.44% accuracy on CIFAR-100, which is comparable to the performance of ANNs. On the neuromorphic datasets, the model achieves 75.00% accuracy on CIFAR-10-DVS and 98.96% accuracy on DVS-128-Gesture, which significantly outperforms previous SNNs. Note that the metrics on all the datasets are slightly lower than those in the paper. This is because we only employ simple data augmentation and do not spend much time tuning the hyperparameters. This project is not intended to achieve the best performance with all kinds of tricks, but to verify the effectiveness of Spike-driven Transformer.

3.3. Transfer Learning

We also conduct experiments to verify the effectiveness of transfer learning in training Spike-driven Transformer. We first preprocess the ImageNet-1k dataset into 128×128 resolution to save the training time. Then we pretrain the model on it for 10 epochs and finetune it on the CIFAR-10 and CIFAR-100 datasets for 30 epochs respectively. Note that finetuning on the neuromorphic datasets is not feasible due to not only different input types and but also different data distributions. The results are shown in Table 2. The training curves are shown in Figure 5.

Table 2. Experiment results on CIFAR-10 and CIFAR-100. Spike-driven Transformer with 4 encoder blocks, 8 heads and 512 channels pretrained on the ImageNet-1k dataset is employed.

Dataset	Standard	Finetune
CIFAR-10	89.56%	**91.25%**
CIFAR-100	67.98%	**70.90%**

We can see that transfer learning brings 1.69% and 2.92% improvements on CIFAR-10 and CIFAR-100 respectively. With the help of transfer learning, the model can achieve satisfactory performance within several epochs of finetuning. In addition, the pretrained model shows smaller generalization error and faster convergence speed, which is

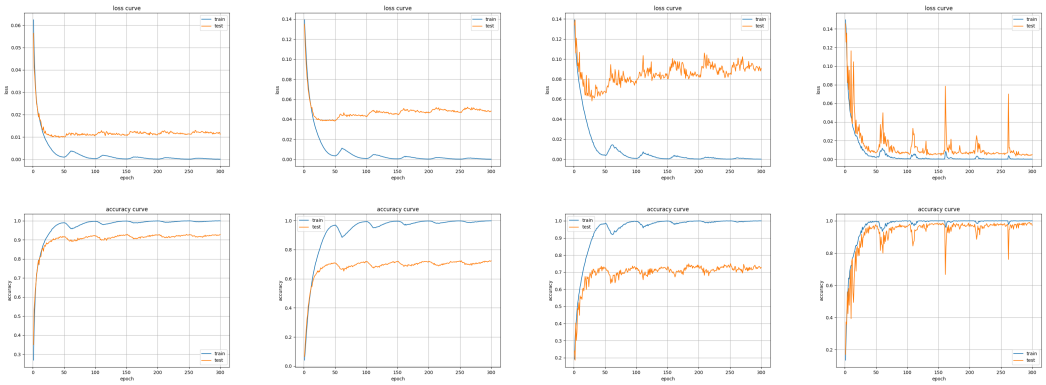

Figure 4. The training curves on the CIFAR-10, CIFAR-100, CIFAR-10-DVS and DVS-128-Gesture datasets. Spike-driven Transformer with 4 encoder blocks, 8 heads and 512 channels is employed. The model is trained for 300 epochs using the AdamW optimizer with a learning rate of 0.0005. The learning rate is warmed up for 10 epochs and then decayed by cosine annealing for 40 epochs.

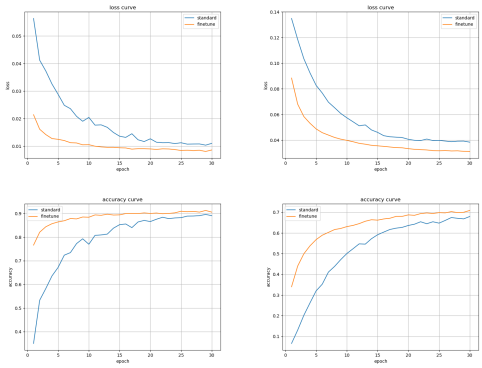

Figure 5. The training curves on the CIFAR-10, CIFAR-100 datasets. Spike-driven Transformer with 4 encoder blocks, 8 heads and 512 channels pretrained on the ImageNet-1k dataset is employed. The model is pretrained for 10 epochs using the AdamW optimizer with a learning rate of 0.001 and then finetuned for 30 epochs using the AdamW optimizer with a learning rate of 0.0001.

of vital importance for deep learning. We owe the success of transfer learning to the diversity of the ImageNet-1k dataset and the strong representation ability of Spike-driven Transformer.

Note that limited by training expenses, we only use the 4-layer model pretrained for 10 epochs. Previous works have proved that with a larger number of parameters, the model can benefit more from transfer learning. We believe that upstream pretraining and downstream finetuning will be a promising direction for future research and the standard paradigm for practical applications.

4. Conclusion

This project delineates the process of our reproduction of the paper Spike-driven Transformer [10]. In Section 1, we briefly outline the significance of the paper and the process of our reproduction along with some innovative aspects. In Section 2, we dive into the network architecture of the paper, analyzing the structures of each core components and their advantages over traditional methods. Furthermore, we highlight our innovations in this project, including the simplification of code implementation, as well as the introduction of transfer learning to improve training efficiency. Finally in Section 3, we present experimental results and corresponding analysis. In summary, we verify the correctness of the paper and prove the effectiveness of our innovative contributions.

A. Member Contribution

Xiangyuan Xue. Implement the project code, propose method innovations, analyze the experiment results and write corresponding part of the report.

Shengmin Yang. Collect the dataset, run part of the experiments and write corresponding part of the report.

Yi Ai. Participate in discussion and verify the code.

References

[1] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7243–7252, 2017. 4

[2] Peter U Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in computational neuroscience*, 9:99, 2015. 1

[3] Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. *Advances in Neural Information Processing Systems*, 34:21056–21069, 2021. 1

[4] 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. 2

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 3

[6] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: spiking neural network for energy-efficient object detection. In *Proceedings of the AAAI conference on artificial intelligence*, pages 11270–11277, 2020. 1

[7] 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. 4

[8] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1, 3

[10] Man Yao, Jiakui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer. *arXiv preprint arXiv:2307.01694*, 2023. 2, 3, 5

[11] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, pages 11062–11070, 2021. 1

[12] Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. *arXiv preprint arXiv:2209.15425*, 2022. 2

[13] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020. 3