SpikCommander: A High-performance Spiking Transformer with Multi-view Learning for Efficient Speech Command Recognition

Jiaqi Wang[1,2], Liutao Yu[2], Xiongri Shen[1], Sihang Guo[1], Chenlin Zhou[3,2], Leilei Zhao[1], Yi Zhong[1,4], Zhiguo Zhang[1*], Zhengyu Ma[2*]

[1]Harbin Institute of Technology, Shenzhen, China
[2]Pengcheng Laboratory, China
[3]Peking University, China
[4]Great Bay University, China
mhwjq1998@gmail.com, zhiguozhang@hit.edu.cn, mazhy@pcl.ac.cn

Abstract

Spiking neural networks (SNNs) offer a promising path toward energy-efficient speech command recognition (SCR) by leveraging their event-driven processing paradigm. However, existing SNN-based SCR methods often struggle to capture rich temporal dependencies and contextual information from speech due to limited temporal modeling and binary spike-based representations. To address these challenges, we first introduce the **m**ulti-view **s**piking **t**emporal-**a**ware **s**elf-**a**ttention (**MSTASA**) module, which combines effective spiking temporal-aware attention with a multi-view learning framework to model complementary temporal dependencies in speech commands. Building on MSTASA, we further propose **SpikCommander**, a fully spike-driven transformer architecture that integrates MSTASA with a **s**piking **c**ontextual **r**efinement MLP (**SCR-MLP**) to jointly enhance temporal context modeling and channel-wise feature integration. We evaluate our method on three benchmark datasets: the Spiking Heidelberg Dataset (SHD), the Spiking Speech Commands (SSC), and the Google Speech Commands V2 (GSC). Extensive experiments demonstrate that SpikCommander consistently outperforms state-of-the-art (SOTA) SNN approaches with fewer parameters under comparable time steps, highlighting its effectiveness and efficiency for robust speech command recognition.

Code — https://github.com/JackieWang9811/SCommander

Introduction

Recognized as the third generation of neural networks (Maass 1997), spiking neural networks (SNNs) effectively mimic the spiking behavior of biological neural circuits through binary information communication (Guo et al. 2024). This brain-inspired attribute enables SNNs to achieve high computational efficiency with their event-driven processing paradigm and significantly reduce energy consumption through spike-based accumulation (AC) operations. These advantages become especially prominent when SNNs are deployed on neuromorphic hardware platforms such as Tianjic (Pei et al. 2019), Loihi (Davies et al. 2021), and TrueNorth (Akopyan et al. 2015), positioning them as a compelling and energy-efficient alternative to conventional artificial neural networks (ANNs) (Fang et al. 2023).

Recent studies (Wang et al. 2025; Shen et al. 2024; Hammouamri, Khalfaoui-Hassani, and Masquelier 2024; Wang et al. 2024; He et al. 2024) have demonstrated that SNN models can effectively tackle speech command recognition (SCR) tasks (also known as keyword spotting) by leveraging temporal-embedded information through their spatio-temporal encoding mechanisms. These characteristics make SNNs promising candidates for crucial auditory front-end applications, where the goal is to efficiently detect predefined spoken commands in dynamic, real-world settings. Despite these merits, current SNN-based approaches for speech command recognition still tend to underperform compared to ANNs (Gong, Chung, and Glass 2021; Schöne et al. 2024), largely due to the challenges posed by their binary and sparse spike-based representations, which hinder the effectiveness of conventional operations over continuous-valued features for capturing complex speech patterns.

To address the aforementioned challenges of applying SNNs to speech command recognition, we structure our work around two key parts. First, although recent spiking attention mechanisms have achieved notable success in vision (Zhou et al. 2023b; Yao et al. 2023; Hua et al. 2025) and language modeling (Xing et al. 2024a,b; Zhang et al. 2024b), they remain underexplored for SCR tasks. To bridge this gap, we introduce a spiking temporal-aware self-attention (**STASA**), an efficient module with linear complexity designed to model essential temporal dependencies in speech sequences. We further extend it into a multi-view learning framework (**MSTASA**) that captures complementary temporal cues via three paths: a sliding-window STASA branch for local context, a long-range STASA branch for global context, and a convolutional path (termed V-branch) that operates on the value stream to inject shift-invariant, position-aware patterns. This unified design facilitates comprehensive temporal modeling. Second, based on MSTASA, we present **SpikCommander**, a compact spiking transformer architecture that integrates MSTASA with a spiking contextual refinement MLP (**SCR-MLP**). SCR-MLP adopts a channel-mixing MLP structure with selective contextual refinement to enhance both context modeling and inter-

*Corresponding authors.

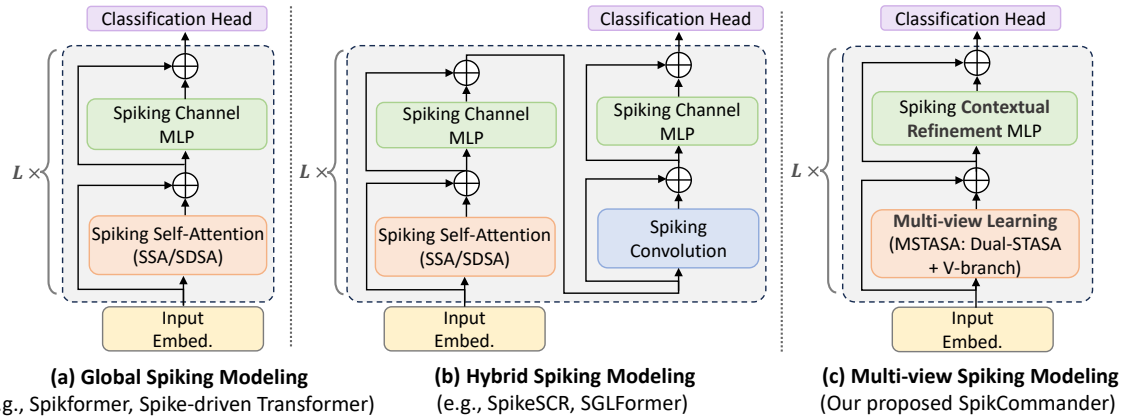


Figure 1: Illustration of three spiking transformer block variants with different modeling strategies: **(a).** Global spiking self-attention with channel MLP; **(b).** Hybrid spiking self-attention and convolution with channel MLP; **(c).** Multi-view temporal-aware self-attention with spiking contextual refinement MLP (our **SpikCommander**).

channel interactions. Together, SpikCommander produces expressive spike-based representations for SCR tasks.

We evaluate our SpikCommander on three speech command datasets: the Spiking Heidelberg Digits (SHD) and Spiking Speech Commands (SSC), both of which are spiking command datasets (Cramer et al. 2020), as well as the Google Speech Commands V2 (GSC) (Warden 2018). Our experimental results demonstrate that SpikCommander outperforms current SOTA SNN methods across these datasets with fewer parameters under the same time step settings.

Our main contributions are summarized as follows:

- **Multi-view learning:** We introduce the MSTASA, which leverages effective spiking temporal-aware self-attention mechanism combined with a multi-view learning framework to capture richer and complementary temporal dependencies in speech commands.
- **SpikCommander:** We present the SpikCommander architecture that integrates MSTASA with a novel spiking contextual refinement MLP (SCR-MLP), jointly enhancing temporal context modeling and channel-wise feature integration in a fully spike-driven transformer.
- **Performance:** Extensive experimental results on three benchmark datasets (SHD, SSC and GSC) demonstrate that SpikCommander surpasses existing SOTA SNN approaches under the same time step settings.

Related Works

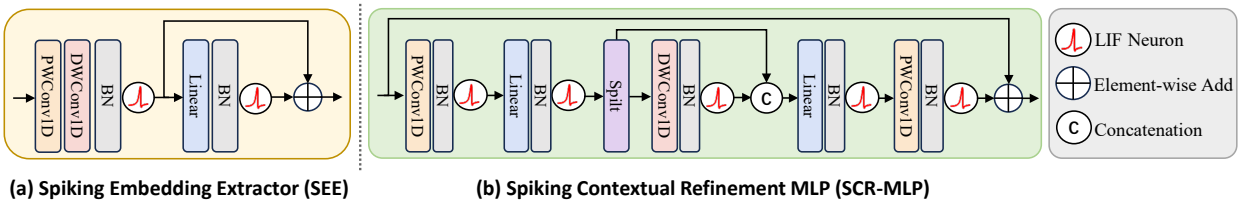
Spiking Transformer Design. Recent efforts have sought to extend Transformer architectures (Vaswani et al. 2017; Dosovitskiy et al. 2020) to the SNNs. As illustrated in Fig.1, existing approaches broadly fall into two categories based on their architecture. Fig.1(a) illustrates models that adopt spiking self-attention mechanisms for global context modeling, combined with channel-wise MLPs for feature mixing. Such as Spikformer (Zhou et al. 2023b), which introduces spiking self-attention (SSA), and Spike-driven Transformer (SDT) (Yao et al. 2023, 2024), with a variant called spike-driven self-attention (SDSA). Fig.1(b) shows hybrid designs integrating spiking attention with convolutional operations to separate global and local processing pathways, as shown in SpikeSCR (Wang et al. 2025) and SGLFormer (Zhang et al. 2024a). In contrast, our proposed SpikCommander (see Fig. 1(c)) introduces two key innovations: i) a unified multi-view learning module (MSTASA) that leverages sliding-window STASA, long-range STASA, and a V-branch to jointly model global and local dependencies; ii) a spiking contextual refinement MLP (SCR-MLP) that explicitly refines contextual and channel-wise features. These innovations enable comprehensive feature modeling, clearly distinguishing SpikCommander from prior designs.

SNNs for Speech Command Recognition. Recent research can be broadly grouped into four directions. First, innovative spiking neuron designs have been proposed to improve the learning of temporal dynamics, such as d-cAdLIF (Deckers et al. 2024) and SE-adLIF (Baronig et al. 2025). The second focuses on developing delay-learning based methods that introduce learnable delay mechanisms to enrich temporal representations, as seen in DL-SNN (Sun et al. 2023) and DCLS-Delays (Hammouamri, Khalfaoui-Hassani, and Masquelier 2024). Third, spiking-based memory modules have been employed to better capture sequential dependencies, including DH-SNN (Zheng et al. 2024) and Spiking LMUFormer (Liu et al. 2024). Finally, recent work involves incorporating spiking attention mechanisms for SCR, such as TIM (Shen et al. 2024), SpikeSCR (Wang et al. 2025), and PfA-SNN (Sun et al. 2025). Despite recent progress, the inherent binary and sparse nature of spikes still limits effective feature extraction from speech commands. Our SpikCommander addresses this with a more expressive and energy-efficient architecture.

Methods

Spiking Neuron

Spiking neurons are a fundamental component of SNNs, offering bio-plausible abstractions of neuronal dynamics (Ma



(a) Spiking Embedding Extractor (SEE) (b) Spiking Contextual Refinement MLP (SCR-MLP)

Figure 2: Two key modules of the SpikCommander architecture. **(a).** Spiking embedding extractor (SEE) encodes speech inputs into spiking embeddings for subsequent attention processing; **(b).** Spiking contextual refinement MLP (SCR-MLP) integrates spike-aware channel mixing and selective contextual refinement to enhance both spatial and temporal feature learning.

et al. 2025). We adopt the Leaky Integrate-and-Fire (LIF) neuron model, which has been widely validated for its effectiveness in spiking architectures (Roy, Jaiswal, and Panda 2019). The dynamics of the LIF neuron can be described as:

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

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

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

where τ is the membrane time constant, $X[t]$ is the input current at time step t, V_{reset} is the reset potential, and V_{th} is the firing threshold. Eq. (1) describes the membrane potential update by integrating incoming currents with a leak term determined by τ. Eq. (2) defines spike generation via the Heaviside step function $\Theta(\cdot)$, which outputs "1" when the membrane potential exceeds V_{th}, indicating a spike. Eq. (3) models the hard-reset mechanism: if a spike is generated ($S[t] = 1$), the membrane potential is reset to V_{reset}; otherwise, it retains its value from the previous time step. This formulation allows the neuron to maintain temporal information across time steps when no spike is fired.

SpikCommander Architecture

Spiking Embedding Extractor (SEE). The SEE module acts as the initial effective spiking embedding extractor, transforming input speech sequences into structured spiking representations. Given the input $\mathbf{X} \in \mathbb{R}^{T\times B\times N}$, where T is the number of time steps, B is the batch size, and N is the number of input neurons or frequency bins. As shown in Fig. 2(a), SEE is designed with depthwise-separable convolutions to extract local features, and a residual-connected linear transformation path to improve channel projection, which can be formulated as:

$$\mathbf{X}' = \mathcal{SN}(\mathrm{BN}(\mathrm{DConv}(\mathrm{PConv}(\mathbf{X})))) \in \mathbb{R}^{T\times B\times D}, \quad (4)$$

$$\mathbf{X}'' = \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathbf{X}'))) + \mathbf{X}' \in \mathbb{R}^{T\times B\times D}, \quad (5)$$

where $\mathcal{SN}$ denotes the spiking neuron, and D is the hidden feature dimension. PConv and DConv refer to pointwise 1D (kernel = 1) and depthwise 1D (kernel = 7) convolutions, respectively, which enable efficient extraction of channel and temporal features. BN denotes batch normalization, and Linear is a fully connected transformation. This compact design effectively generates informative spiking embeddings for subsequent attention-based processing.

Spiking Temporal-aware Self-Attention (STASA). As illustrated in Fig. 3(b), we first describe the internal mechanism of the proposed STASA. Given the spike-based input $\mathbf{X_S} \in \mathbb{R}^{T\times B\times D}$, the spiking query, key and value representations are computed as:

$$Q_S = \mathbf{W_Q}(\mathbf{X_S}),\ K_S = \mathbf{W_K}(\mathbf{X_S}), V_S = \mathbf{W_V}(\mathbf{X_S}), \quad (6)$$

where $\mathbf{W}$ represents the learnable weight matrix implemented via a {PConv-BN-SN} block. Q_S, K_S, and V_S share the same shape as $\mathbf{X_S}$ ($\in \mathbb{R}^{T\times B\times D}$). To exclude contributions from zero-padded time steps introduced for sequence length standardization, we apply a temporal mask to Q_S and K_S, yielding the masked representations $Q'_S = Mask(Q_S)$ and $K'_S = Mask(K_S)$, as shown in Fig. 7 of Appendix C. We then permute the spiking query and key representations and aggregate information across time by summing over the temporal dimension, resulting in $\hat{Q}_S = \sum_{t=1}^{T} Q'_S[:,t,:] \in \mathbb{R}^{B\times 1\times D}$ and $\hat{K}_S = \sum_{t=1}^{T} K'_S[:,t,:] \in \mathbb{R}^{B\times 1\times D}$, and compute the attention weight S_{attn} as:

$$S_{attn} = \beta * (\hat{Q}_S + \hat{K}_S) \in \mathbb{R}^{B\times 1\times D}, \quad (7)$$

where β is a scaling factor that mitigates gradient vanishing from large integer accumulations, then S_{attn} is re-permuted to match the temporal-first format. Finally, the spiking attention map (M_{attn}) is generated by passing the S_{attn} through a spiking neuron and broadcasting along the temporal dimension to the spiking value representation $V_S \in \mathbb{R}^{T\times B\times D}$ via Hadamard product ($\odot$):

$$M_{attn} = \mathcal{SN}(S_{attn}) \odot V_S \in \mathbb{R}^{T\times B\times D}. \quad (8)$$

STASA readily extends to **multi-head** attention by splitting features dimension (D) and applying temporal-aware summation per head. Details of the sliding-window variant of STASA are provided in Appendix C. Compared with the classic spiking self-attention (SSA) (Zhou et al. 2023b) that relies on the matrix multiplication $Q_S K_S^\top$ with $\mathcal{O}(N^2D)$ complexity, where N denotes the number of tokens, which corresponds to the number of time steps T in SCR tasks. STASA reduces the computational cost to linear complexity $\mathcal{O}(ND)$, making it more efficient for long time sequences.

Multi-view STASA (MSTASA). Fig. 3(a) shows the architecture of the proposed MSTASA module, designed to comprehensively capture temporal dependencies leveraging spiking representations via three complementary branches with **shared** spiking query, key and value representations.

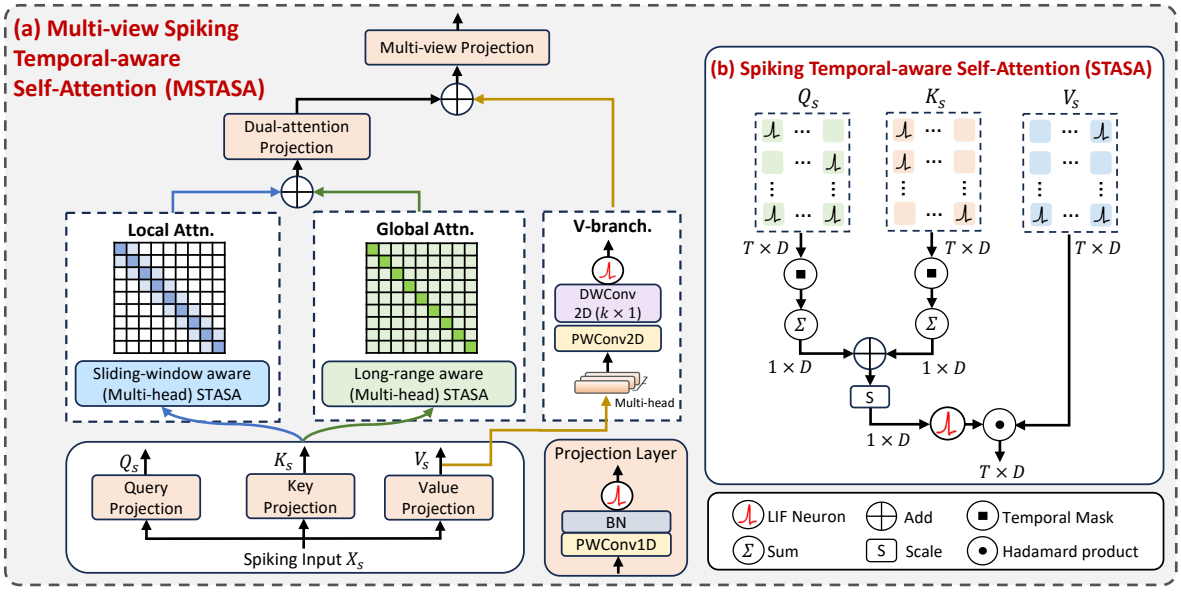


Figure 3: Illustration of the multi-view spiking temporal-aware self-attention (MSTASA). **(a).** Architecture combining local sliding-window STASA, long-range STASA, and a complementary convolutional V-branch; **(b).** Internal mechanism of STASA.

Branch1: The *sliding-window aware (SWA-STASA)* models local temporal dependencies by restricting attention to a fixed-size sliding window of $(2w+1)$ time steps. To ensure appropriate attention field coverage across different time scales, the w is dynamically adjusted with T. SWA aims to incorporate only nearby context at each time step and leverages spiking query and key interactions with spiking activations to adaptively emphasize relevant local dynamics.

Branch2: The *long-range aware (LRA-STASA)* captures global temporal dependencies by attending over the entire sequence, enabling learning of long-range contextual relationships essential for high-level temporal understanding.

Branch3: The *V-branch* complements the attention pathways with a convolutional perspective, applying depthwise (kernel = 9×1) and pointwise 2D convolutions over multi-head value representations to capture shift-invariant temporal patterns. This design injects precise positional cues, thereby enhancing multi-view diversity and improving the model's capacity to capture complex temporal patterns.

In MSTASA, the outputs of the two STASA branches are first fused via a dual-attention projection block, aligning their learned temporal dependencies. This fused attention output is then combined with the V-branch, followed by a multi-view projection block that produces the output:

$$\mathbf{X}' = \mathbf{W_M}((\mathbf{W_D}(B_1(\mathbf{X}) + B_2(\mathbf{X})) + B_3(\mathbf{X}))) \in \mathbb{R}^{T\times B\times D}, \tag{9}$$

where $\mathbf{W_D}$ and $\mathbf{W_M}$ are learnable weight matrices implemented via $\{\text{PConv-BN-SN}\}$ projection blocks, and $B_i(\mathbf{X})$ denotes the output of the i-th branch.

Spiking Contextual Refinement MLP (SCR-MLP). As shown in Fig. 2(b), SCR-MLP adopts a spike-aware channel-mixing MLP architecture with selective temporal context refinement, enabling sparse and energy-efficient modeling of both spatial and temporal features. It consists of three stages: **(i) Pre-projection**: The input $\mathbf{X} \in \mathbb{R}^{T\times B\times D}$ is first processed through the following operations:

$$\mathbf{X}' = \text{LinBlock}(\text{PCBlock}(\mathbf{X})) \in \mathbb{R}^{T\times B\times \alpha D}. \tag{10}$$

Here, $\text{PCBlock} = \{\text{PConv-BN-SN}\}$ and $\text{LinBlock} = \{\text{Linear-BN-SN}\}$ serve as spike-aware modules for lightweight channel mixing and feature expansion through an inverted bottleneck structure. The expansion ratio α controls the hidden dimensionality and is set to 4 by default. **(ii) Selective contextual refinement:** The output $\mathbf{X}'$ is then evenly split along the channel dimension into two parts, $\mathbf{H}_1$ and $\mathbf{H}_2$, where $\mathbf{H}_1, \mathbf{H}_2 \in \mathbb{R}^{T\times B\times \frac{\alpha D}{2}}$. And then $\mathbf{H}_1$ is processed with $\text{DCBlock} = \{\text{DConv}_{\text{k=31}}\text{-BN-SN}\}$ to capture local temporal context, and the two branches are later concatenated: $[\mathbf{H}_1, \mathbf{H}_2] = Split(\mathbf{X}')$, $\mathbf{H}_1' = \text{DCBlock}(\mathbf{H}_1) \in \mathbb{R}^{T\times B\times \frac{\alpha D}{2}}$, $\mathbf{X}'' = Concat(\mathbf{H}_1', \mathbf{H}_2) \in \mathbb{R}^{T\times B\times \alpha D}$. **(iii) Post-projection**: The merged features $\mathbf{X}''$ are transformed and projected back to the original dimension D :

$$\mathbf{X}''' = \text{PCBlock}(\text{LinBlock}(\mathbf{X}'')) \in \mathbb{R}^{T\times B\times D}. \tag{11}$$

The above architecture achieves efficient spatial and temporal feature modeling with minimal computational overhead.

Training Strategy

Note that our model is trained end-to-end from scratch using backpropagation-through-time (BPTT) with surrogate gradients (Zhou et al. 2023a). Table 7 in Appendix B provides detailed settings of the optimizer and learning rate. The classification head encodes per-time-step outputs using a softmax over the spikes $s_i[t]$ at each time step t:

$$out_i[t] = softmax(s_i[t]) = \frac{e^{s_i[t]}}{\sum_{j=1}^{Y} e^{s_j[t]}}, \tag{12}$$

Dataset	Model	Param (M)	Time Steps	Acc (%)
SHD	SDT (*1L*) (Yao et al. 2023)	1.77	100	89.61†
	Spikformer (*1L*) (Zhou et al. 2023b)	1.77	100	90.10†
	DH-SNN (*2L*) (Zheng et al. 2024)	0.05	1000	92.10
	d-cAdLIF (*2L*) (Deckers et al. 2024)	0.08	100	94.85
	DCLS-Delays (*2L*) (Hammouamri, Khalfaoui-Hassani, and Masquelier 2024)	0.20	100	95.07
	SpikeSCR (*1L*) (Wang et al. 2025)	0.26	100	95.60
	SE-adLIF (*2L*) (Baronig et al. 2025)	0.45	250	95.81
	Event-SSM$^{\mathbf{A}}$ (Schöne et al. 2024)	0.40	—	95.90
	Pfa-SNN (Sun et al. 2025)	0.20	100	96.26
	SpikCommander (*1L-8-128*)	**0.19**	**100**	**96.41**
SSC	SDT (*2L*) (Yao et al. 2023)	2.57	100	79.82†
	DCLS-Delays (*2L*) (Hammouamri, Khalfaoui-Hassani, and Masquelier 2024)	1.40	100	80.16
	Spikformer (*2L*) (Zhou et al. 2023b)	2.57	100	80.18†
	Pfa-SNN (Sun et al. 2025)	0.71	100	80.18
	d-cAdLIF (*2L*) (Deckers et al. 2024)	0.70	100	80.23
	SE-adLIF (*2L*) (Baronig et al. 2025)	1.60	250	80.44
	DCLS-Delays (*3L*) (Hammouamri, Khalfaoui-Hassani, and Masquelier 2024)	2.50	100	80.69
	DH-SNN (*3L*) (Zheng et al. 2024)	0.35	1000	82.46
	SpikeSCR (*1L*) (Wang et al. 2025)	1.71	100	82.54
	SpikeSCR (*2L*) (Wang et al. 2025)	3.30	100	82.79
	SpikCommander (*1L-16-256*)	**1.12**	**100**	**83.26**
	SpikCommander (*2L-16-256*)	**2.13**	**100**	**83.49**
GSC	Spikformer (*2L*) (Zhou et al. 2023b)	2.57	100	91.86†
	SDT (*2L*) (Yao et al. 2023)	2.57	100	91.88†
	DH-SNN (*3L*) (Zheng et al. 2024)	0.11	1000	94.05
	DCLS-Delays (*2L*) (Hammouamri, Khalfaoui-Hassani, and Masquelier 2024)	1.40	100	95.00
	DCLS-Delays (*3L*) (Hammouamri, Khalfaoui-Hassani, and Masquelier 2024)	2.50	100	95.35
	SpikeSCR (*1L*) (Wang et al. 2025)	1.71	100	95.46
	SpikeSCR (*2L*) (Wang et al. 2025)	3.30	100	95.60
	d-cAdLIF (*2L*) (Deckers et al. 2024)	0.61	100	95.69
	Spiking LMUFormer (Liu et al. 2024)	1.69	—	96.12
	LMUFormer$^{\mathbf{A}}$ (Liu et al. 2024)	1.62	—	96.53
	SpikCommander (*1L-16-256*)	**1.12**	**100**	**96.71**
	SpikCommander (*2L-16-256*)	**2.13**	**100**	**96.92**

Table 1: Comparison of model performances with prior SNN works on three different datasets, SHD, SSC, and GSC. † indicates our reproduced performance. The notation of (*nL-m-d*) in the table specifies the model architecture, where *n* represents the number of blocks, *m* represents the number of attention heads, and *d* represents the hidden size. $^{\mathbf{A}}$ indicates ANN model.

where Y is the number of classes. The final prediction is obtained by summing over time steps $\hat{y}_i = \sum_{t=1}^{T} out_i[t]$, where $\hat{y}_i$ denotes the accumulated score for class i over all time steps. We employ the standard cross-entropy loss $\mathcal{L}_{CE}$ over the accumulated scores $\hat{y}_i$ and ground-truth labels.

Experiments

Datasets and Experimental Setup

We evaluate our models on two spiking datasets, SHD and SSC (Cramer et al. 2020), as well as the non-spiking counterpart of SSC, GSC (Warden 2018). SHD contains 10k recordings across 20 classes (digits zero to nine in English and German). SSC and GSC are larger datasets, each comprising 100k recordings with 35 distinct speech command classes. Table 6 in Appendix B summarizes detailed dataset statistics. We follow the same preprocessing pipeline as in (Wang et al. 2025; Hammouamri, Khalfaoui-Hassani, and Masquelier 2024) for all three datasets. For SHD and SSC, the original 700 input neurons are first reduced to 140 by applying spatio-temporal binning over every five neurons. To standardize input lengths, all samples are zero-padded to a fixed number of time steps. Specifically, the total number of time steps T is computed as $T = 1000/\Delta t$, where 1000 ms is the total duration of each spiking sample, and different temporal resolutions are discretized using fixed-duration windows of Δt ms. Considering $\Delta t \in \{1, 4, 10\}$ yields $T \in \{1000, 250, 100\}$, respectively. For GSC, all audio waveforms are first downsampled from 16 kHz to 8 kHz. We then compute Mel spectrograms with 140 frequency bins, matching the input dimensionality used for SSC. To approximately match the temporal resolutions used in SSC, we use a fixed window size $l = 256$ (corresponding to 32 ms) and vary the hop length h to control the number of time steps. The time steps T are calculated as $T = (8000 - l)/h + 1$, yielding $T \in \{1000, 250, 100\}$ for $h \in \{8, 32, 80\}$. For consistency with prior work, we focus on 100 time steps, with the sliding window radius w in SpikCommander set to 20.

To enhance robustness and generalization, we apply augmentation tailored to speech commands, focusing on two modalities: Mel spectrograms and spike trains. For Mel spectrograms, we adopt SpecAugment (Park et al. 2019), applying frequency and time masking to improve robustness to variations in time-frequency patterns. For spike trains, we utilize an augmentation strategy (Wang et al. 2025) with two operations: drop-by-time and drop-by-neuron, randomly re-

Dataset	Model	Param (M)	Sampling Rate (kHz)	Acc (%)	FLOPs (G)	SOPs (G)	Energy (mJ)
GSC	KWT-3[A] (Berg, O'Connor, and Cruz 2021)	5.36	16	98.54	1.053	—	4.84
	AST[A] (Gong, Chung, and Glass 2021)	86.93	16	98.11	25.67	—	118.08
	KW-MLP[A] (Morshed and Ahsan 2021)	0.42	16	97.56	0.045	—	0.207
	SpikeSCR (*1L-16-256*) (Wang et al. 2025)	1.63	8	95.56	0.011	0.019	0.067
	SpikeSCR (*2L-16-256*) (Wang et al. 2025)	3.15	8	95.60	0.011	0.049	0.094
	Spiking LMUFormer (Liu et al. 2024)	1.69	16	96.12	0.007	0.031	0.059
	SpikCommander (*1L-16-256*)	**1.12**	**8**	**96.71**	**0.005**	**0.008**	**0.028**
	SpikCommander (*2L-16-256*)	**2.13**	**8**	**96.92**	**0.005**	**0.020**	**0.042**

Table 2: Efficiency on non-spiking SCR task (GSC dataset) under model size, sampling rate, accuracy, floating-point operations (FLOPs), theoretical synaptic operations (SOPs), and estimated energy consumption. [A] indicates ANN model.

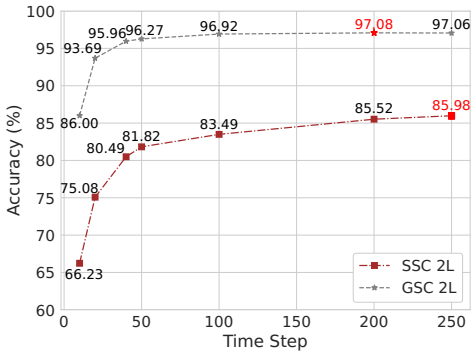


Figure 4: Long-term learning performance of 2-block SpikCommander on SSC and GSC under varying time steps.

moving events along the temporal and neuronal dimensions to simulate realistic noise. The effects of these augmentations on the GSC and SSC datasets are illustrated in Fig. 5 and Fig. 6, respectively, with detailed implementation parameters provided in Appendix A.

Main Results

Comprehensive performance. We compare our method with recent SNN models across three datasets in terms of accuracy and model size. As shown in Table 1, SpikCommander consistently outperforms prior methods and establishes a new SOTA. Notably, it achieves superior performance using only standard LIF neurons, surpassing models employing more sophisticated neuron dynamics like SE-adLIF (Baronig et al. 2025) and d-cAdLIF (Deckers et al. 2024) under the same time steps. For example, on SHD, SpikCommander reaches **96.41%** accuracy with just 0.19M parameters, surpassing SE-adLIF's 95.81% with 0.45M. Moreover, delay learning-based models like DCLS-Delays (Hammouamri, Khalfaoui-Hassani, and Masquelier 2024) show strong temporal modeling capability, yet SpikCommander attains higher accuracy with fewer parameters at the same time steps. On SSC, it outperforms the 3-block DCLS-Delays (2.50M) by 2.57% using only 1.12M. Furthermore, when compared to spiking memory modules like DH-SNN (Zheng et al. 2024), SpikCommander also exhibits clear advantages: although DH-SNN achieves decent accuracy with a minimal parameter count, it relies on significantly longer sequences (e.g., 1000 time steps), whereas SpikCommander achieves higher accuracy with only 100 time steps (e.g., **96.71%** vs. 94.05% on GSC), making it more efficient for latency-sensitive scenarios. Finally, while Spikformer (Zhou et al. 2023b) and SDT (Yao et al. 2023) struggle with effective temporal modeling, recent SpikeSCR (Wang et al. 2025) provides competitive baselines; SpikCommander still achieves higher accuracy with fewer parameters. On SSC, it improves accuracy by 0.70% (**83.49%** vs. 82.79%) over SpikeSCR while reducing parameters by 1.18M (2.13M vs. 3.30M); and on GSC, it achieves 1.32% (**96.92%** vs. 95.60%) higher accuracy with the same parameter reduction. The accuracy drop of replacing STASA with SSA or SDSA and the five-seed analysis presented in Appendix D further validate the effectiveness and stability of SpikCommander.

We further evaluate the **long-term learning capability** of SpikCommander by varying the number of time steps from T=10 to T=250 across three datasets, with dynamically adjusted window radius w, while keeping the model size. Here, we focus on the 2-block architecture, given its superior performance and more prominent trends, while 1-block results are presented in Fig. 8 of Appendix D. Fig. 4 illustrates accuracy trends on SSC and GSC with increasing time step. On SSC, SpikCommander consistently benefits from longer input durations: accuracy improves from **83.49%** at T=100 to **85.52%** at T=200, and reaches **85.98%** at T=250, reflecting strong scalability and robustness in long-term modeling. On GSC, which uses Mel spectrogram inputs, the model already achieves **96.27%** at T=50, outperforming recent SOTA SNNs. As T increases, performance further improves, reaching **97.08%** at T=200 and **97.06%** at T=250. To our knowledge, this marks the first SNN model to surpass the 97% threshold on GSC. These results demonstrate SpikCommander's strong scalability and temporal modeling capacity across diverse modalities of speech commands, making it a promising backbone for challenging SCR tasks.

Efficiency on non-spiking SCR task. As shown in Table 2, we compare SpikCommander with a range of SOTA ANN and SNN models on the GSC dataset, in terms of accuracy, model size, computational operations (FLOPs/SOPs), and theoretical energy consumption. Details of the theoretical energy estimation are provided in Appendix E. Compared to ANN-based models such as AST (Gong, Chung, and Glass 2021) (86.93M) and KWT-3 (Berg, O'Connor, and Cruz 2021) (5.36M), SpikCommander achieves comparable accuracy (96.92%, under 100 time steps) while reducing energy consumption by orders of magnitude (e.g.,

Dataset	Method	Param (M)	SOPs (G)	Energy (mJ)
SSC	Spikformer (*2L*)	2.57	0.3169	0.2853
	SDT (*2L*)	2.57	0.3084	0.2776
	SpikeSCR (*2L*)	3.30	0.0348	0.0314
	SpikCommander (*2L*)	**2.13**	**0.0173**	**0.0180**
SHD	Spikformer (*1L*)	1.77	0.2054	0.1849
	SDT (*1L*)	1.77	0.1847	0.1663
	SpikeSCR (*1L*)	0.26	0.0056	0.0051
	SpikCommander (*1L*)	**0.19**	**0.0034**	**0.0039**

Table 3: Efficiency on spiking SCR tasks in terms of model parameters, theoretical SOPs, and energy consumption.

0.042 mJ vs. 4.84 mJ for KWT-3, 0.207 mJ for KW-MLP (Morshed and Ahsan 2021)). Notably, our model maintains competitive performance even at a lower sampling rate (8 kHz vs. 16 kHz), leading to reduced signal processing cost and improved efficiency. Among SNN baselines, SpikCommander also demonstrates clear advantages. Compared to Spiking LMUFormer, which attains 96.12% accuracy with 0.059 mJ energy consumption, our 2-block SpikCommander achieves +0.80% higher accuracy while consuming 28.8% less energy. Compared to SpikeSCR (2L), which achieves 95.60% accuracy with 0.094 mJ energy consumption, our 2-block SpikCommander achieves higher accuracy (+1.32%) with less than half the energy consumption (0.042 mJ).

Efficiency on spiking SCR tasks. We further report a quantitative comparison of model size, synaptic operation counts (SOPs), and theoretical energy consumption on the SSC and SHD datasets under 100 time steps, as shown in Table 3. While Spikformer and SDT present moderate parameters, both models are re-implemented following the 2D spatial computation setup described in (Shen et al. 2024), where 1D sequences are reshaped into pseudo-images and processed with spatiotemporal operations via nearest-neighbor interpolation. This leads to substantial redundant computation, resulting in significantly higher SOPs (e.g., 0.3084G on SSC). In contrast, both SpikeSCR and SpikCommander operate directly on sequential data. SpikeSCR adopts a deeper hybrid design to capture local and global dependencies, which increases its computational overhead (e.g., 0.0348G SOPs and 3.30M parameters on SSC). Notably, SpikCommander consistently outperforms SpikeSCR, achieving 42.7% lower energy consumption (0.0180 mJ vs. 0.0314 mJ), 50.3% fewer SOPs (0.0173G vs. 0.0348G), and a 35.5% smaller model size (2.13M vs. 3.30M). On SHD, SpikCommander again demonstrates superior efficiency, reducing energy consumption by 23.5%, SOPs by 39.3%, and model size by 26.9% compared to SpikeSCR. These collectively demonstrate SpikCommander's superior trade-off between performance, parameter efficiency, and low energy consumption, underscoring its potential for low-power speech processing on neuromorphic hardware.

Detailed ablation studies are conducted on the large-scale, two-modality datasets SSC and GSC using different block configurations, as summarized in Table 4. First, removing data augmentation (DA) reduces accuracy (0.62% on SSC, 0.43% on GSC), validating its role in enhancing generalization across noisy or variable inputs. Next, excluding the V-branch from MSTASA reduces accuracy by 0.49% (SSC) and 0.28% (GSC), indicating that it complements attention-based branches by capturing shift-invariant patterns. Subsequently, removing the sliding-window aware STASA branch further degrades performance (0.35% on SSC and 0.33% on GSC), underscoring the value of localized temporal attention for fine-grained modeling. We also observe a significant accuracy drop when replacing the SCR-MLP with a standard MLP (2.16% on SSC and 1.00% on GSC), despite a reduction in parameters, highlighting the importance of spike-aware temporal modeling and structured channel interactions. Finally, substituting a lightweight SEE module with Conv1D projection yields appreciable decline (0.5% and 1.97% for SSC and GSC, respectively), demonstrating the effectiveness of our dedicated spiking embedding extractor in preserving informative temporal features. Sequential module removal reduces parameters by 20% (e.g., from 2127K to 1694K on SSC), yet the resulting performance deterioration confirms the necessity of each proposed component in achieving robust SCR tasks. Appendix D also analyzes the impact of the temporal mask and the sliding window radius w in SWA-STASA on spiking datasets, with w=20 offering a good trade-off between local context and temporal dynamics.

Dataset	Model	Param (K)	Acc (%)
SSC	SpikCommander (*2L-16-256*)	2127	83.49
	w/o DA	2127	82.87
	MSTASA w/o V-branch	1994	82.38
	MSTASA w/o SWA-STASA	1994	82.03
	SCR-MLP $\rightarrow$ MLP	1694	79.87
	SEE $\rightarrow$ Conv1D Projection	1699	79.37
GSC	SpikCommander (*1L-16-256*)	1120	96.71
	w/o DA	1120	96.28
	MSTASA w/o V-branch	1053	96.00
	MSTASA w/o SWA-STASA	1053	95.67
	SCR-MLP $\rightarrow$ MLP	904	94.67
	SEE $\rightarrow$ Conv1D Projection	908	92.70

Table 4: Detailed sequential ablation studies of SpikCommander on GSC and SSC datasets under 100 time steps.

Conclusion

We introduce SpikCommander, a novel spike-driven transformer architecture designed for efficient speech command recognition. At its core, MSTASA leverages a multi-view learning framework that models temporal dependencies via three complementary branches, effectively addressing the limitations of existing SNN-based SCR methods. Furthermore, SpikCommander combines the MSTASA module and the spiking contextual refinement MLP, jointly enhancing temporal contextual modeling and channel-wise feature integration. Extensive evaluations across SHD, SSC, and GSC datasets demonstrate that SpikCommander consistently surpasses prior SNN methods in terms of accuracy, parameter and energy consumption under comparable time step settings. These results highlight the potential of SpikCommander as a high-performance and energy-efficient neuromorphic solution in resource-constrained environments.

Acknowledgments

This work is supported by the National Science and Technology Innovation 2030 Major Project (No. 2025ZD0215501), the National Natural Science Foundation of China (No. 82272114) and the Shenzhen Science and Technology Program (No. ZDSYS20230626091203008).

References

Akopyan, F.; Sawada, J.; Cassidy, A.; Alvarez-Icaza, R.; Arthur, J.; Merolla, P.; Imam, N.; Nakamura, Y.; Datta, P.; Nam, G.-J.; et al. 2015. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE transactions on computer-aided design of integrated circuits and systems*, 34(10): 1537–1557.

Baronig, M.; Ferrand, R.; Sabathiel, S.; and Legenstein, R. 2025. Advancing spatio-temporal processing through adaptation in spiking neural networks. *Nature Communications*, 16(1): 5776.

Berg, A.; O'Connor, M.; and Cruz, M. T. 2021. Keyword transformer: A self-attention model for keyword spotting. *arXiv preprint arXiv:2104.00769*.

Cramer, B.; Stradmann, Y.; Schemmel, J.; and Zenke, F. 2020. The heidelberg spiking data sets for the systematic evaluation of spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7): 2744–2757.

Davies, M.; Wild, A.; Orchard, G.; Sandamirskaya, Y.; Guerra, G. A. F.; Joshi, P.; Plank, P.; and Risbud, S. R. 2021. Advancing neuromorphic computing with loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5): 911–934.

Deckers, L.; Van Damme, L.; Van Leekwijck, W.; Tsang, I. J.; and Latré, S. 2024. Co-learning synaptic delays, weights and adaptation in spiking neural networks. *Frontiers in Neuroscience*, 18: 1360300.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

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

Gong, Y.; Chung, Y.-A.; and Glass, J. 2021. Ast: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*.

Guo, W.; Sun, Y.; Xu, Y.; Qiao, Z.; Yang, Y.; and Xiong, H. 2024. SpGesture: Source-free domain-adaptive sEMG-based gesture recognition with Jaccard attentive spiking neural network. *arXiv preprint arXiv:2405.14398*.

Hammouamri, I.; Khalfaoui-Hassani, I.; and Masquelier, T. 2024. Learning delays in spiking neural networks using dilated convolutions with learnable spacings. In *International Conference on Learning Representations*.

He, X.; Li, Y.; Zhao, D.; Kong, Q.; and Zeng, Y. 2024. MSAT: Biologically inspired multistage adaptive threshold for conversion of spiking neural networks. *Neural Computing and Applications*, 36(15): 8531–8547.

Hua, W.; Zhou, C.; Wu, J.; Chua, Y.; and Shu, Y. 2025. MSVIT: Improving spiking vision Transformer using multi-scale attention fusion. *arXiv preprint arXiv:2505.14719*.

Liu, Z.; Datta, G.; Li, A.; and Beerel, P. A. 2024. Lmuformer: low complexity yet powerful spiking model with legendre memory units. *arXiv preprint arXiv:2402.04882*.

Ma, C.; Chen, X.; Li, Y.; Yang, Q.; Wu, Y.; Li, G.; Pan, G.; Tang, H.; Tan, K. C.; and Wu, J. 2025. Spiking neural networks for temporal processing: Status quo and future prospects. *arXiv preprint arXiv:2502.09449*.

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

Morshed, M. M.; and Ahsan, A. O. 2021. Attention-free keyword spotting. *arXiv preprint arXiv:2110.07749*.

Park, D. S.; Chan, W.; Zhang, Y.; Chiu, C.-C.; Zoph, B.; Cubuk, E. D.; and Le, Q. V. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*.

Pei, J.; Deng, L.; Song, S.; Zhao, M.; Zhang, Y.; Wu, S.; Wang, G.; Zou, Z.; Wu, Z.; He, W.; et al. 2019. Towards artificial general intelligence with hybrid Tianjic chip architecture. *Nature*, 572(7767): 106–111.

Roy, K.; Jaiswal, A.; and Panda, P. 2019. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784): 607–617.

Schöne, M.; Sushma, N. M.; Zhuge, J.; Mayr, C.; Subramoney, A.; and Kappel, D. 2024. Scalable event-by-event processing of neuromorphic sensory signals with deep state-space models. In *2024 International Conference on Neuromorphic Systems (ICONS)*, 124–131. IEEE.

Shen, S.; Zhao, D.; Shen, G.; and Zeng, Y. 2024. TIM: An efficient temporal interaction module for spiking Transformer. *arXiv preprint arXiv:2401.11687*.

Sun, P.; Chua, Y.; Devos, P.; and Botteldooren, D. 2023. Learnable axonal delay in spiking neural networks improves spoken word recognition. *Frontiers in Neuroscience*, 17: 1275944.

Sun, P.; Wu, J.; Devos, P.; and Botteldooren, D. 2025. Towards parameter-free attentional spiking neural networks. *Neural Networks*, 185: 107154.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Wang, J.; Yu, L.; Huang, L.; Zhou, C.; Zhang, H.; Song, Z.; Liu, H.; Zhang, M.; Ma, Z.; and Zhang, Z. 2025. Efficient Speech Command Recognition Leveraging Spiking Neural Networks and Progressive Time-scaled Curriculum Distillation. *Neural Networks*, 108253.

Wang, S.; Zhang, D.; Shi, K.; Wang, Y.; Wei, W.; Wu, J.; and Zhang, M. 2024. Global-local convolution with spiking

neural networks for energy-efficient keyword spotting. In *Proc. Interspeech 2024*, 4523–4527.

Warden, P. 2018. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*.

Xing, X.; Gao, B.; Zhang, Z.; Clifton, D. A.; Xiao, S.; Du, L.; Li, G.; and Zhang, J. 2024a. SpikeLLM: Scaling up spiking neural network to large language models via saliency-based spiking. *arXiv preprint arXiv:2407.04752*.

Xing, X.; Zhang, Z.; Ni, Z.; Xiao, S.; Ju, Y.; Fan, S.; Wang, Y.; Zhang, J.; and Li, G. 2024b. Spikelm: Towards general spike-driven language modeling via elastic bi-spiking mechanisms. *arXiv preprint arXiv:2406.03287*.

Yao, M.; Hu, J.; Hu, T.; Xu, Y.; Zhou, Z.; Tian, Y.; XU, B.; and Li, G. 2024. Spike-driven Transformer V2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. In *International Conference on Learning Representations*.

Yao, M.; Hu, J.; Zhou, Z.; Yuan, L.; Tian, Y.; Xu, B.; and Li, G. 2023. Spike-driven Transformer. In *Advances in Neural Information Processing Systems*, volume 36.

Zhang, H.; Zhou, C.; Yu, L.; Huang, L.; Ma, Z.; Fan, X.; Zhou, H.; and Tian, Y. 2024a. SGLFormer: Spiking global-local-fusion transformer with high performance. *Frontiers in Neuroscience*, 18: 1371290.

Zhang, J.; Shen, J.; Wang, Z.; Guo, Q.; Yan, R.; Pan, G.; and Tang, H. 2024b. SpikingMiniLM: Energy-efficient spiking transformer for natural language understanding. *Science China Information Sciences*, 67(10): 200406.

Zheng, H.; Zheng, Z.; Hu, R.; Xiao, B.; Wu, Y.; Yu, F.; Liu, X.; Li, G.; and Deng, L. 2024. Temporal dendritic heterogeneity incorporated with spiking neural networks for learning multi-timescale dynamics. *Nature Communications*, 15(1): 277.

Zhou, C.; Yu, L.; Zhou, Z.; Zhang, H.; Ma, Z.; Zhou, H.; and Tian, Y. 2023a. Spikingformer: Spike-driven Residual Learning for Transformer-based Spiking Neural Network. *arXiv preprint arXiv:2304.11954*.

Zhou, Z.; Zhu, Y.; He, C.; Wang, Y.; YAN, S.; Tian, Y.; and Yuan, L. 2023b. Spikformer: When spiking neural network meets Transformer. In *International Conference on Learning Representations*.