SOFTMAX-FREE SPIKING TRANSFORMERS FOR LANGUAGE MODELING VIA WINNER-TAKE-ALL ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking Transformers, which combine the scalability of Transformers with the sparse, energy-efficient property of Spiking Neural Networks (SNNs), have achieved impressive results in neuromorphic and vision tasks and attracted increasing attention. Existing spiking language models often rely on softmax-based self-attention, whose exponentiation and normalization operations increase computational costs and complicate neuromorphic deployment. To address this issue, we introduce Winner-Take-All (WTA) mechanisms into spiking transformers and propose two novel softmax-free, spike-driven self-attention modules: WTA Spiking Self-Attention (WSSA) and Causal WTA Spiking Self-Attention (CWSSA). Based on them, we design WTA-based Encoder-only Spiking Transformer (WE-Spikingformer) for masked language modeling and WTA-based Decoder-only Spiking Transformer (WD-Spikingformer) for causal language modeling, systematically exploring softmax-free, spike-driven transformer architectures trained end-to-end for natural language processing tasks. Extensive experiments on 16 datasets spanning natural language understanding, question-answering tasks, and commonsense reasoning tasks validate the effectiveness of our approach and highlight the promise of spiking transformers for general language modeling and energy-efficient artificial intelligence.

1 INTRODUCTION

Spiking Neural Networks (SNNs), regarded as the third generation of neural networks (Maass, 1997), offer high biological plausibility and energy efficiency through their event-driven dynamics, making them promising alternatives to Artificial Neural Networks (ANNs) (Roy et al., 2019). By transmitting information through binary spikes, SNNs replace traditional high-power multiply-accumulate (MAC) operations with low-power accumulate (AC) operations, thereby achieving substantial energy savings.

Spiking transformers, which combine the architectural strengths of Transformers with the event-driven, sparse, and energy-efficient properties of spiking neural networks, have achieved great progress on both neuromorphic datasets and large-scale vision datasets (Zhou et al., 2023b; 2026; Yao et al., 2023a; 2024; Zhou et al., 2024; Yao et al., 2025; Wang et al., 2024) and have attracted significant attention. They achieve performance comparable to that of ANN counterparts while offering superior energy efficiency. Current directly trained spiking transformers have primarily targeted computer vision tasks (Zhou et al., 2023b; 2026; Yao et al., 2023a; 2024; Zhou et al., 2024; Yao et al., 2025) with encoder-only architectures. Spiking transformers with softmax-free self-attention perform well on visual tasks because image patches are locally correlated and highly redundant. Repeated low-level features, such as edges and textures, allow softmax-free attention to achieve competitive performance.

As interest in spiking language models grows, extending softmax-free spiking transformers from vision to language remains challenging. Language understanding requires capturing dependencies between tokens over long contexts, making it difficult to directly transfer attention mechanisms designed for locally correlated image patches. SpikeBERT (Lv et al., 2023) is an early attempt in this direction. As a softmax-free spiking transformer, SpikeBERT adapts Spikformer (Zhou et al., 2023b) for language tasks by proposing a two-stage knowledge distillation training method, and and achieves a 59.7% average score on the development sets of the General Language Understanding Evaluation

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(GLUE) benchmark (Wang et al., 2018). The substantial performance gap between SpikeBERT and BERT (Devlin et al., 2019) (19.9 percentage points) highlights the difficulty of directly applying spiking self-attention from a vision-based spiking transformer (spikformer) to language modeling. More recently, SpikeLM (Xing et al., 2024b) and SpikeLLM (Xing et al., 2024a) have achieved competitive results in language modeling; however, they do not fully adhere to a spike-driven design. SpikeLM retains non-spiking GeLU activations (Hendrycks & Gimpel, 2016) in the MLP blocks, while SpikeLLM preserves SiLU activations in its MLP blocks. More importantly, both SpikeLM and SpikeLLM are softmax-based spiking transformers. The exponentiation and division operations in softmax increase computational costs and complicate neuromorphic deployment (Zhou et al., 2023b; 2026).

To address these limitations, we explore softmax-free spike-driven transformers, tailor-made for language modeling. Firstly, we introduce a brain-inspired mechanism: Winner-Take-All (WTA) for spiking transformers to replace the softmax operation in language modeling. Winner-take-all mimics biological lateral inhibition to enforce sparsity and focus attention on the most relevant tokens, serving as an extremely sparse alternative to softmax, which makes it particularly suitable for spiking language modeling. By incorporating biologically inspired WTA mechanisms, we develop two softmax-free, spike-driven transformer architectures: a WTA-based Encoder-only Spiking Transformer for masked language modeling and a WTA-based Decoder-only Spiking Transformer for causal language modeling. Our main contributions are as follows:

1) Leveraging spike-driven and Winner-Take-All (WTA) biological mechanisms, we propose two spike-driven self-attention modules for language modeling: WTA Spiking Self-Attention (**WSSA**) and Causal WTA Spiking Self-Attention (**CWSSA**). Conventional self-attention relies on floating-point matrix multiplications and softmax normalization. Our attention mechanisms instead combine spike-based accumulation with WTA comparison and selection, avoiding the exponentiation and division operations required by softmax. This design preserves sparse attention outputs for spike-driven inference.

2) We develop a WTA-based Encoder-only Spiking Transformer (**WE-Spikingformer**) with WSSA for masked language modeling, and a WTA-based Decoder-only Spiking Transformer (**WD-Spikingformer**) with CWSSA for causal language modeling, systematically exploring direct-training-based softmax-free spike-driven transformers in language modeling.

3) We evaluate our models on 16 datasets covering natural language understanding, question-answering, and commonsense reasoning tasks. Extensive experiments demonstrate the promise of directly trained spiking transformers for general language modeling and energy-efficient artificial intelligence.

2 Related Work

SNNs are commonly constructed through ANN-to-SNN conversion (Liu et al., 2025; Cao et al., 2026; Wang et al., 2026) or direct training (Fang et al., 2022; Zhou et al., 2023b; 2026). We focus on direct training.

2.1 Spiking Transformers in Vision Tasks

Spiking transformers with encoder-only architectures have been widely adopted in visual tasks (Zhou et al., 2023b; 2026; Yao et al., 2023a; 2024; Zhou et al., 2024; Yao et al., 2025). Spikformer (Zhou et al., 2023b) introduced Spiking Self-Attention (SSA), which replaces softmax with sparse spike-form Query, Key, and Value, achieving 74.81% accuracy on ImageNet-1k with only four time steps—the first strong evidence of the potential of transformer-based SNNs. Building on it, Spikingformer (Zhou et al., 2026) employed a pre-activation shortcut to eliminate floating-point multiplications and reduce firing rates, further improving accuracy to 75.85%. Spike-driven Transformer (Yao et al., 2023a) proposed Spike-Driven Self-Attention (SDSA), which relies solely on masking and addition, reaching 77.07% on ImageNet-1k with significantly lower computational cost. Recently, hierarchical visual spiking transformers (Yao et al., 2024; Zhou et al., 2024; Yao et al., 2025) have achieved accuracy above 80% on ImageNet while maintaining high energy efficiency. MaxFormer Fang et al. (2025) and SpikePool Lee et al. (2025) show that spiking neurons suppress certain frequency components during information transmission. To compensate, they introduce max-

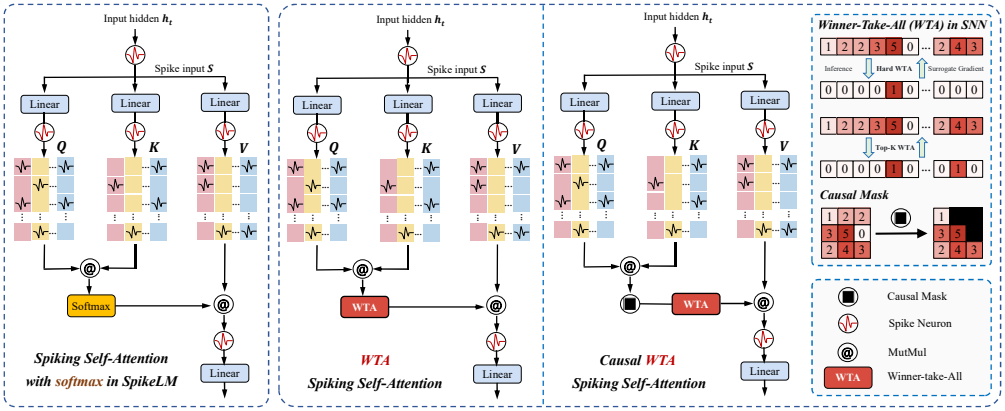


Figure 1: Overview of WTA Spiking Self-Attention (WSSA) and Causal WTA Spiking Self-Attention (CWSSA). The left shows the softmax-based spiking self-attention in SpikeLM (Xing et al., 2024b). The right shows our WSSA in WE-Spikingformer for masked language modeling and CWSSA in WD-Spikingformer for causal language modeling. The symbol "@" denotes matrix multiplication.

pooling into Spiking Transformers: MaxFormer uses it in patch embedding to recover high-frequency signals, while SpikePool replaces spike-based self-attention with max-pooling attention for selective band-pass filtering. The differences between the Hard WTA mechanism and max-pooling are shown in Appendix I. These methods are primarily designed for vision tasks and are hard to generalize to large-scale, data-driven scenarios such as language modeling. In contrast, our work targets language modeling and introduces Hard WTA directly on the attention map within the self-attention layer, acting as a one-hot encoding mechanism to produce extremely sparse yet scalable attention.

2.2 Spiking Transformers in Language Tasks

SpikeBERT (Lv et al., 2023) extends the softmax-free Spikformer (Zhou et al., 2023b) to language tasks and trains it via two-stage knowledge distillation from BERT during pretraining and task-specific fine-tuning. However, it achieves only a 59.7% average score on the GLUE development benchmark (Wang et al., 2018), nearly 20 percentage points below BERT, suggesting that directly transferring vision-oriented spiking self-attention to language yields suboptimal performance. SpikeGPT (Zhu et al., 2023) is an RWKV (Peng et al., 2023) architecture-based spiking language model and retains the exponential and division operations similar to the softmax operation. SpikeLM (Xing et al., 2024b) proposes a spike formulation with bi-directional ternary firing for language modeling and achieves promising results in language modeling; however, it largely preserves key components of vanilla attention, including floating-point matrix multiplications, softmax operations, and non-spiking activations (e.g., GeLU (Hendrycks & Gimpel, 2016)) in the MLP blocks. Similarly, SpikeLLM (Xing et al., 2024a) proposes generalized integrate-and-fire (GIF) neurons together with an optimal brain spiking framework for spiking transformers. However, SpikeLLM largely inherits from LLaMA (Touvron et al., 2023) and retains many nonlinear components from LLaMA, such as softmax operations in attention map computation, non-spiking SiLU activations in the MLP blocks, and rotary positional embeddings (Su et al., 2024) with non-spike-driven computations in the query and key projections. By contrast, our proposed WE-Spikingformer and WD-Spikingformer eliminate these non-spiking components to realize a spike-driven transformer architecture. In short, this work focuses on designing softmax-free, energy-efficient spike-driven transformers for language modeling.

3 Method

3.1 Spiking Neuron Model

In this section, to investigate more expressive yet efficient spiking neurons for language modeling, we also explore two variants of the Leaky Integrate-and-Fire (LIF) neuron model (Maass, 1997) for direct training in language modeling tasks: NI-LIF (Lei et al., 2025) and T-LIF (Xing et al., 2024b).

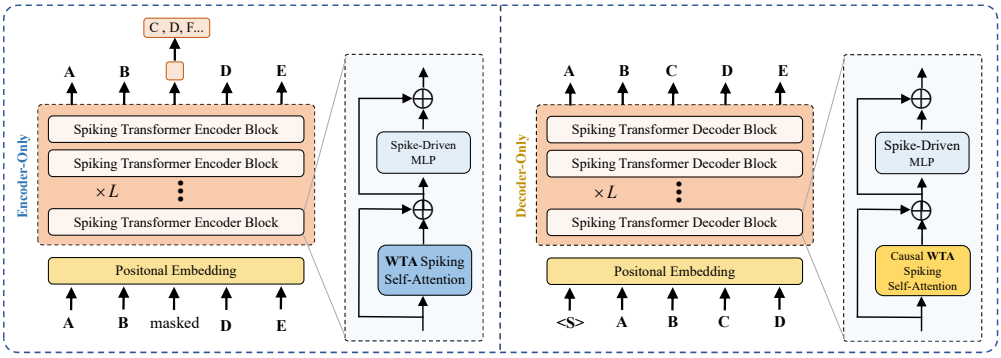


Figure 2: The overview of WE-Spikingformer and WD-Spikingformer. The left shows WE-Spikingformer (WTA-based Encoder-only Spiking Transformer) for spike-based masked language modeling. The right shows WD-Spikingformer (WTA-based Decoder-only Spiking Transformer) for spike-based causal language modeling.

Normalized Integer Leaky Integrate-and-Fire (NI-LIF) model. The NI-LIF model follows an integer-training, spike-inference paradigm (Luo et al., 2024), with dynamics given by:

$$U[t] = H[t-1] + X[t], \tag{1}$$

$$S[t] = \text{clip}(\text{round}(U[t]), 0, D)/D, \tag{2}$$

$$H[t] = \beta(U[t] - S[t] \times D), \tag{3}$$

Here, $\text{clip}(u, a, b)$ clips u to $[a, b]$, and D is the maximum quantized integer value. During inference on neuromorphic hardware, each integer time step is unfolded into D binary time steps. Thus, T_{int} integer time steps correspond to $T = T_{\text{int}}D$ binary time steps; for example, NI-LIF (1×4) corresponds to $T = 4$.

The dynamics of the T-LIF model are shown in Appendix A. In our experiments, we use NI-LIF neurons by default for faster spiking simulation, and we also compare the performance differences between the two neurons in Sec. 4.1.

3.2 Winner-Take-All Layer

In the visual system, multiple neurons are sensitive to different directions at the same location. The neurons with the strongest response (such as detecting vertical edges) will suppress other direction detectors, leaving only a dominant signal. The Winner-Take-All computation (Maass, 2000), inspired by this biological mechanism, is incorporated into spiking neural networks as a biologically plausible strategy for enforcing sparsity. It simulates the "lateral inhibition" phenomenon in biological nervous systems - that is, the strongest signal inhibits other competing signals, ultimately forming a single dominant response and ensuring that attention remains focused on the most relevant token. In this section, we primarily introduce multiple WTA mechanisms and their corresponding training methods.

Hard WTA. Hard WTA is a sparse neural activation mechanism. In a group of competing units, only the unit with the highest activation value is allowed to retain the original output, and the outputs of all other units are forced to zero. Given an input vector $\mathrm{A} = [a_1, \ldots, a_N]$, the output layer of WTA is $\mathrm{Y} = [y_1, \ldots, y_N]$. Hard WTA can be formulated as follows:

$$y_i = \begin{cases} a_i & \text{if } i = \arg\max_{j \in \{1,\ldots,N\}} a_j, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

In SNNs, the winning element needs to be binarized, that is, the winner is further defined as: $y_i = 1$, if $i = \arg\max_{j \in \{1,\ldots,N\}} a_j$. When the largest input is unique, binary Hard WTA is the zero-temperature limit of softmax. A detailed discussion of the relationship between Hard WTA and softmax is provided in Appendix E.

Top-K WTA. As a sparse neural activation mechanism, Top-K WTA selects the K units with the highest scores from a set of input signals, retains only their original values, and sets the outputs of all

other units to zero. Top-K WTA allows a limited number of "winners" to coexist, thus achieving a balance between sparsity and expressiveness. Top-K WTA can be formulated as follows:

$$y_i = \begin{cases} a_i & \text{if } a_i \in \text{Top-k}\left(a_1, \ldots, a_N\right), \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

In SNNs, the top-k winning elements are further binarized, that is: $y_i = 1$, if $a_i \in \text{Top-k}\left(a_1, \ldots, a_N\right)$.

Sparsemax. Sparsemax (Martins & Astudillo, 2016) is a continuous, almost-everywhere differentiable transformation that can automatically generate some zero values in the output, which can also be seen as an adaptive Top-K Winner-Take-All mechanism. That is, only a few significant elements are given non-zero probability, and the remaining elements are precisely truncated to zero, thereby achieving adaptive sparsity. The calculation process of Sparsemax can be described as follows:

$$y_i = \max\left\{a_i - \tau, 0\right\}, \quad i = 1, \ldots, N. \tag{6}$$

Eq. 6 needs to satisfy $\sum_i y_i = 1$, and τ is the adaptive threshold, whose calculation is shown in Appendix F.

Surrogate Gradient for the WTA Layer. Binary Hard WTA and Top-K WTA produce piecewise-constant outputs, with zero gradients almost everywhere and undefined gradients at selection boundaries. Therefore, we choose the surrogate gradient for Winner-Take-All during training, which does not affect the high energy efficiency of the model during inference. Specifically, we use the gradient of softmax to approximate the "gradient" of the Winner-Take-All layer, which is shown as follows:

$$\left.\frac{\partial y}{\partial a}\right|_{\text{backward}} := \text{softmax}'(a). \tag{7}$$

where $\text{softmax}'(.)$ is the derivative of the softmax function.

Hard WTA provides an extremely sparse alternative to softmax by selecting a single winner without exponential normalization. Inspired by lateral inhibition, its one-hot selection suppresses competing responses and supports spike-driven inference. During training, the softmax surrogate provides a gradient signal; during inference, the forward computation uses WTA selection without evaluating softmax. In our experiments, we use Hard WTA by default for its sparse, binary outputs.

3.3 WTA Spiking Self-Attention Mechanisms

In this part, we propose two self-attention mechanisms based on the Winner-Take-All layer for spiking language modeling: WTA Spiking Self-Attention (WSSA) and Causal WTA Spiking Self-Attention (CWSSA), shown in Figure 1.

CWSSA is a spike-driven self-attention for spike-based causal language modeling with Decoder-only Transformers. CWSSA can be formulated as follows:

$$\mathbf{X}' = \text{SN}(\mathbf{X}), \tag{8}$$

$$\mathbf{Q} = \text{SN}_Q(\text{Linear}_Q(\mathbf{X}')), \tag{9}$$

$$\mathbf{K} = \text{SN}_K(\text{Linear}_K(\mathbf{X}')), \tag{10}$$

$$\mathbf{V} = \text{SN}_V(\text{Linear}_V(\mathbf{X}')), \tag{11}$$

$$\text{A}_\text{w}(\mathbf{Q}, \mathbf{K}) = \text{WTA}(\mathbb{M}(\mathbf{Q}\mathbf{K}^\text{T} * s)), \tag{12}$$

$$\text{Out}(\text{A}_\text{w}, \mathbf{V}) = \text{Linear}(\text{SN}(\text{A}_\text{w}\,\mathbf{V})), \tag{13}$$

Here, s is the scaling factor used in Zhou et al. (2023b), A_w denotes the attention weights, SN denotes a spiking neuron layer, and $\mathbb{M}(\cdot)$ denotes causal masking.

WSSA is a spike-driven self-attention for spike-based masked language modeling with Encoder-only Transformers. The attention mechanism is similar to CWSSA, with the main difference that the causal mask in Eq. 13 is removed:

$$\text{A}_\text{w}(\mathbf{Q}, \mathbf{K}) = \text{WTA}((\mathbf{Q}\mathbf{K}^\text{T} * s)). \tag{14}$$

Compared to the softmax-based SSA in SpikeLM (Xing et al., 2024b), both WSSA and CWSSA are spike-driven and more energy-efficient, avoiding the costly exponential and division operations required by softmax.

Model	SD	T	MNLI_mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.(%)
$\text{BERT}_{\text{base}}$ (Devlin et al., 2019)	✗	–	83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
Q2BERT (Zhang et al., 2020)	✗	–	47.3	67.0	61.3	80.6	0.0	4.7	81.2	52.7	49.1
BiBERT (Qin et al., 2022)	✗	–	67.5	84.8	72.6	88.7	25.4	33.6	72.5	57.4	63.2
SpikeLM (Xing et al., 2024b)	✗	4	77.2	83.9	85.3	87.0	38.8	84.9	85.7	69.0	76.5
LIF-BERT (Xing et al., 2024b)	✓	4	35.2	0	50.5	50.9	0	0	81.2	52.7	34.6
PSN-BERT (Xing et al., 2024b)	✓	4	35.2	0	50.5	50.9	0	6.8	81.2	52.7	34.7
SpikeBERT (Lv et al., 2023)	✓	4	71.0	68.2	66.4	85.4	16.9	18.7	82.0	57.5	59.7
WE-Spikingformer (ours)	✓	4	73.4	83.8	78.3	85.6	23.7	56.0	76.4	48.4	65.7
WE-Spikingformer† (ours)	✓	4	70.1	85.1	77.5	89.0	27.9	42.8	81.6	55.9	**66.3**

Table 1: The results on the Natural Language Understanding task (GLUE datasets). "Avg.(%)" denotes "Mean GLUE score". "SD" denotes "Spike-Driven". The results of LIF-BERT, PSN-BERT, and SpikeLM are reported in Xing et al. (2024b). WE-Spikingformer uses NI-LIF neuron, while WE-Spikingformer† adopts the T-LIF neuron. Our model achieves state-of-the-art performance on the GLUE datasets among softmax-free spike-driven transformer models.

3.4 Language Modeling

Based on WSSA and CWSSA in Section 3.3, we build two spiking transformers for spike-based language modeling: WTA-based Spiking Encoder (WE-Spikingformer) and WTA-based Spiking Decoder (WD-Spikingformer) in this part.

Model Architecture. The overview of WE-Spikingformer and WD-Spikingformer is shown in Figure 2. The decoder block in WD-Spikingformer can be formulated as follows:

$$X_l' = \mathrm{CWSSA}\left(X_{l-1}\right) + X_{l-1}, \quad X_l' \in \mathbb{R}^{T\times N\times d}, l = 1\ldots L, \tag{15}$$

$$X_l = \mathrm{SMLP}\left(X_l'\right) + X_l', \quad X_l \in \mathbb{R}^{T\times N\times d}, l = 1\ldots L, \tag{16}$$

Here, T is the number of spike time steps, N is the sequence length, d is the hidden dimension, and L is the number of blocks. SMLP denotes a spike-driven multilayer perceptron. Unlike SpikeLM and SpikeLLM, which retain non-spiking activation functions such as GeLU or SiLU, our SMLP is fully spiking and implemented as a sequential stack of {SN-Linear-SN-Linear} layers. The encoder block in WE-Spikingformer is similar to the decoder one in WD-Spikingformer, with the main difference that CWSSA in Eq. 15 is replaced by WSSA in Eq. 17:

$$X_l' = \mathrm{WSSA}\left(X_{l-1}\right) + X_{l-1}, \quad X_l' \in \mathbb{R}^{T\times N\times d}, l = 1\ldots L. \tag{17}$$

Masked Language Modeling. Given a sequence of N tokens $\mathrm{x} = (x_1, x_2, \ldots, x_N)$, randomly select a portion of the locations $M \subseteq \{1, 2, \ldots, N\}$ for masking. The masked sequence is obtained as $\tilde{x} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_N)$. The goal of Masked Language Modeling (MLM) (Devlin et al., 2019) is to let the model predict the masked tokens based on the unmasked tokens, which can be formulated as follows:

$$p\left(x_M \mid \tilde{x}_{\setminus M}\right), \tag{18}$$

where $x_M = \{x_t : t \in M\}$ denotes the masked tokens, $\tilde{x}_{\setminus M}$ denotes the visible tokens that are retained. Thus, the loss function of MLM can be formulated as follows:

$$\mathcal{L}_{\mathrm{MLM}} = -\sum_{t\in M} \log p_\theta\left(x_t \mid \tilde{x}\right). \tag{19}$$

In our work, we apply WE-Spikingformer to masked language modeling for pretraining.

Causal Language Modeling. Causal Language Modeling (CLM) (Radford et al., 2018; Ouyang et al., 2022) is the most widely used language modeling method. Its core idea is autoregressive generation: when predicting the next word, only the context before the current position is allowed to be used, and future words cannot be accessed. For a sequence of N tokens $\mathrm{x} = (x_1, x_2, \ldots, x_N)$, the autoregressive factorization is:

$$p(x) = \prod_{t=1}^{N} p\left(x_t \mid x_{<t}\right), \tag{20}$$

where x_t represents the t-th token, $x_{<t} = (x_1, x_2, \ldots, x_{t-1})$ represents all previous tokens. CLM maximizes the sequence log-likelihood under this factorization. In network modeling, the causal

mask is used to ensure that the model only focuses on the previous tokens when calculating attention. The loss function of CLM can be formulated as follows:

$$\mathcal{L}_{\mathrm{CLM}} = -\sum_{t=1}^{N} \log p_\theta \left(x_t \mid x_{<t} \right). \tag{21}$$

We pretrain WD-Spikingformer using this causal language modeling objective.

4 EXPERIMENTS

In this section, we conduct experiments across 16 datasets spanning natural language understanding tasks, question-answering tasks, and commonsense reasoning tasks to evaluate our Winner-Take-All spiking transformer in language modeling.

Model	SD	E (mJ)	T	ARC-e	ARC-c	BoolQ	HeadQA	OBQA	Avg. (%)
SpikeLLM-7B (Xing et al., 2024a)	✗	-	4	31.3	23.6	53.8	-	-	-
Qwen-1.5B (Yang et al., 2024)	✗	3398.3	-	26.2	26.9	60.3	25.7	27.1	33.2
WD-Spikingformer-0.4B	✓	238.4	4	30.0	22.3	37.8	26.0	26.1	28.4

Table 2: The results on Question Answering Tasks (QAT). "Avg." denotes "Average Accuracy". "E (mJ)" means "Energy consumption (mJ)", with the calculation process detailed in Appendix C. "T" means time step. "SD" denotes "Spike-Driven".

Model	SD	E (mJ)	T	HellaSwag	PIQA	Winograd	Avg. (%)
SpikeLLM-7B Xing et al. (2024a)	✗	-	4	33.9	53.4	51.5	46.3
Qwen-1.5B (Yang et al., 2024)	✗	3398.3	-	26.5	53.7	52.8	44.3
WD-Spikingformer-0.4B	✓	238.4	4	25.9	53.4	50.2	43.2

Table 3: The results on Commonsense Reasoning Tasks (CRT). "Avg." denotes "Average Accuracy". "E (mJ)" means "Energy consumption (mJ)". "T" means time step. "SD" denotes "Spike-Driven".

4.1 NATURAL LANGUAGE UNDERSTANDING

We evaluate WE-Spikingformer on the standard GLUE (General Language Understanding Evaluation) benchmark (Wang et al., 2018), which is a widely adopted collection of datasets designed to evaluate and advance natural language understanding capabilities of machine learning models. We evaluate eight GLUE tasks for classification and regression, including single-sentence classification (CoLA, SST-2), pairwise sentence comparison (MRPC, QQP, RTE), sentence similarity (STS-B), and natural language inference (MNLI, QNLI).

We pretrain WE-Spikingformer and WE-Spikingformer† on Wikipedia-English (Devlin et al., 2019) using masked language modeling (Devlin et al., 2019) across 8 GPUs, and subsequently fine-tune them on the GLUE dev set. The ANN baselines include BERT (Devlin et al., 2019), Q2BERT (Zhang et al., 2020) and BiBERT (Qin et al., 2022). Q2BERT employs 2-bit weights and 8-bit activations. BiBERT uses 1-bit weights and activations and is trained using direction-matching distillation (DMD). WE-Spikingformer† (WE-Spikingformer with the T-LIF neuron) achieves an average GLUE score of 66.3%. Replacing T-LIF with the NI-LIF neuron yields comparable performance while significantly reducing pretraining time. These results demonstrate that both T-LIF and NI-LIF neurons are viable for language modeling. The experimental results are presented in Table 1. All models are maintained at the same parameter scale as $\mathrm{BERT}_{\mathtt{base}}$ (109M). As a spike-driven, softmax-free transformer, WE-Spikingformer† significantly outperforms other spike-based methods (LIF-BERT, PSN-BERT, and SpikeBERT). WE-Spikingformer† vs. SpikeBERT vs. LIF-BERT. Acc: 66.3% vs. 59.7% vs. 34.6%. In comparison, SpikeLM is a softmax-based spiking transformer and retains non-spiking activation GeLU (Hendrycks & Gimpel, 2016) in MLP blocks. Its higher score therefore comes from an architecture that does not satisfy the same softmax-free, fully spike-driven constraints.

4.2 Question Answering Tasks

For question answering, we use the FineWeb-Edu (Lozhkov et al., 2024) dataset, a high-quality subset of the FineWeb corpus curated for factual and educational content, and sample 0.1B tokens from it to pretrain WD-Spikingformer on 8 GPUs. We evaluate the model performance of WD-Spikingformer on a diverse set of Question-Answering Tasks (QAT), including ARC-e (Clark et al., 2018), ARC-c (Clark et al., 2018), BoolQ (Clark et al., 2019), HeadQA (Vilares & Gómez-Rodríguez, 2019), and OpenBookQA (OBQA) (Mihaylov et al., 2018). These tasks measure the generalization and reasoning abilities without task-specific finetuning.

The experimental results are shown in Table 2. On one hand, due to the unavailability of the open-source model file for SpikeLLM (Xing et al., 2024a) and the absence of reported energy consumption and results on HeadQA and OBQA, our comparative analysis is limited to the average accuracy on the ARC-e, ARC-c, and BoolQ benchmarks. With 17.5 times fewer parameters (0.4B vs. 7B), WD-Spikingformer-0.4B achieves 30.0% average accuracy across these three tasks, compared with 36.2% for the much larger SpikeLLM-7B under the (T=4, W2A8) configuration. On the other hand, WD-Spikingformer-0.4B reduces estimated inference energy by more than an order of magnitude (238.4 mJ vs. 3398.3 mJ, only 7% of the energy) with an accuracy gap of 4.8 percentage points (28.4% vs. 33.2%) against Qwen-1.5B, which adopts the same pretraining setup as WD-Spikingformer. These results highlight WD-Spikingformer's superior energy–accuracy trade-off.

4.3 Commonsense Reasoning Tasks

In this experiment, we evaluate the model performance of WD-Spikingformer on a diverse set of Commonsense Reasoning Tasks (CRT), including the HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), and Winograd (Sakaguchi et al., 2021) datasets. The pretraining process is the same as shown in Section 4.2.

The experimental results are shown in Table 3. WD-Spikingformer-0.4B achieves 43.2% average accuracy, compared with 44.3% for Qwen-1.5B and 46.3% for SpikeLLM-7B, while using only 0.4B parameters rather than 1.5B and 7B, respectively. These results highlight that WD-Spikingformer-0.4B delivers competitive performance despite its much smaller size.

Method	GLUE	QAT	CRT
Spiking Self-Attention	55.4	✗	✗
Softmax Attention	66.8	28.7	43.3
Hard WTA Attention	66.3	28.4	43.2

Table 4: Hard WTA ablation. Values are mean GLUE scores or average QAT/CRT accuracies (%) at $T = 4$. QAT and CRT use causal masking; ✗ denotes non-convergence during pretraining.

Method	T	Avg. (%)
Hard WTA	4	65.7
Top-K WTA	4	66.3
Sparsemax	4	66.6
Hard WTA	8	67.4
Hard WTA	4	65.7
Hard WTA	2	62.3

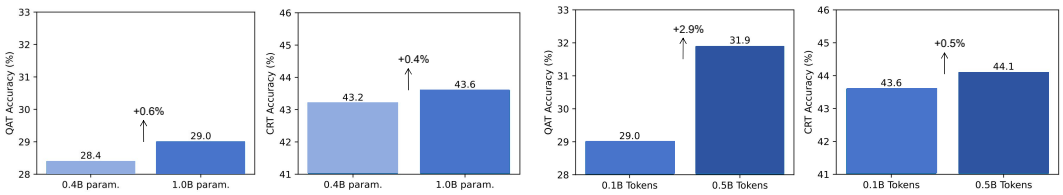
Table 5: WTA mechanisms and time steps on GLUE. Top-K WTA uses $K = 3$. T denotes time steps; Avg. denotes the mean GLUE score.

4.4 Ablation Study

We investigate Hard WTA, alternative WTA mechanisms, and the number of time steps across GLUE, QAT, and CRT.

Ablation study for Hard WTA. We compare Hard WTA with vision-oriented spiking self-attention (SSA) (Zhou et al., 2023b) and softmax-based attention (Xing et al., 2024b). On GLUE, SSA computes $\mathrm{Q}\,\mathrm{K}^{\mathrm{T}}\,\mathrm{V} * s$, while Softmax computes $\mathrm{Softmax}(\mathrm{Q}\,\mathrm{K}^{\mathrm{T}} * s)\,\mathrm{V}$. For QAT and CRT, all methods use causal masking: SSA computes $\mathbb{M}(\mathrm{Q}\,\mathrm{K}^{\mathrm{T}} * s)\,\mathrm{V}$, while Softmax and Hard WTA transform the masked scores before multiplication by V. Table 4 supports two observations.

1) Vision-oriented SSA gives suboptimal results on the evaluated language tasks. Hard WTA achieves a mean GLUE score of 66.3%, compared with 55.4% for SSA. On QAT and CRT, causal SSA fails to converge during pretraining, whereas Hard WTA achieves average accuracies of 28.4% and 43.2%.



(a) Increasing model parameters.

(b) Increasing pretraining tokens.

Figure 3: Scaling WD-Spikingformer on QAT and CRT. (a) Increasing model size from 0.4B to 1.0B parameters with 0.1B pretraining tokens. (b) Increasing pretraining data from 0.1B to 0.5B tokens for the 1.0B model.

2) Hard WTA performs close to Softmax across all three task groups: 66.3% versus 66.8% on GLUE, 28.4% versus 28.7% on QAT, and 43.2% versus 43.3% on CRT. Thus, replacing softmax with sparse, binary WTA selection yields small performance differences while avoiding softmax's exponentiation and normalization operations during inference. These comparisons support Hard WTA for both masked and causal language modeling.

Ablation study for WTA variants and time steps. Table 5 compares Hard WTA, Top-K WTA ($K = 3$), and Sparsemax on GLUE. Their mean scores are 65.7%, 66.3%, and 66.6%, respectively. Allowing more nonzero attention entries gives modest gains, while Hard WTA retains the sparsest, binary selection. Increasing Hard WTA's time steps from 4 to 8 improves the score from 65.7% to 67.4%; reducing them to 2 lowers it to 62.3%. The additional simulation steps therefore offer a trade-off between predictive performance and computational cost. Results for other values of K and corresponding CRT ablations appear in Appendix J.

4.5 Discussion

We examine how WD-Spikingformer scales with model size and pretraining data on QAT and CRT.

Parameter scaling. With 0.1B pretraining tokens, increasing the model size from 0.4B to 1.0B parameters raises average accuracy from 28.4% to 29.0% on QAT and from 43.2% to 43.6% on CRT (Figure 3a). These results show modest but consistent gains across both task groups.

Pretraining data scaling. For WD-Spikingformer-1.0B, increasing pretraining data from 0.1B to 0.5B tokens improves average accuracy from 29.0% to 31.9% on QAT and from 43.6% to 44.1% on CRT (Figure 3b), gains of 2.9 and 0.5 percentage points, respectively. The larger improvement on QAT suggests that the benefit of additional pretraining data varies across tasks.

Together, these results support the scalability of WD-Spikingformer within the evaluated range of model sizes and pretraining budgets. Detailed results are provided in Tables 6 and 7.

5 Conclusion

In this work, we explore softmax-free spiking transformers for language modeling by incorporating Winner-Take-All (WTA) mechanisms into spike-driven self-attention. We propose two novel attention modules, namely WTA Spiking Self-Attention (WSSA) and Causal WTA Spiking Self-Attention (CWSSA). Based on these modules, we design WE-Spikingformer for masked language modeling and WD-Spikingformer for causal language modeling. Our approach systematically extends directly trained spiking transformers from vision to language. Extensive experiments across 16 datasets spanning natural language understanding, question answering tasks, and commonsense reasoning tasks demonstrate the effectiveness of our models. These results underscore the potential of Spiking Transformers as a foundation for biologically inspired, energy-efficient, and general-purpose language modeling.

AI USE STATEMENT

AI tools were used solely for language polishing. The methods and experiments were developed and conducted by the authors without AI assistance. The authors take full responsibility for the content of this paper.

REFERENCES

Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Honglin Cao, Shuai Wang, Zijian Zhou, Ammar Belatreche, Wenjie Wei, Yu Liang, Yu Yang, Rui Xi, Malu Zhang, and Haizhou Li. Towards training-free and accurate ann-to-snn conversion via activation-aware redistribution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 1703–1711, 2026.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Shuai Dong, Junyi Yang, Xiaoqi Peng, Hongyang Shang, Ye Ke, Xiaofeng Yang, Hongjie Liu, and Arindam Basu. Topkima-former: Low-energy, low-latency inference for transformers using top-k in-memory adc. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 72(6):2509–2519, 2025. doi: 10.1109/TCSI.2025.3549060.

Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timoth Masquelier, and Yonghong Tian. Deep Residual Learning in Spiking Neural Networks. In *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 21056–21069, 2022.

Yuetong Fang, Deming Zhou, Ziqing Wang, Hongwei Ren, ZeCui Zeng, Lusong Li, Renjing Xu, et al. Spiking neural networks need high-frequency information. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

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

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

Donghyun Lee, Alex Sima, Yuhang Li, Panos Stinis, and Priyadarshini Panda. Spikepool: Event-driven spiking transformer with pooling attention. *arXiv preprint arXiv:2510.12102*, 2025.

Zhenxin Lei, Man Yao, Jiakui Hu, Xinhao Luo, Yanye Lu, Bo Xu, and Guoqi Li. Spike2former: Efficient spiking transformer for high-performance image segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 1364–1372, 2025.

Chang Liu, Jiangrong Shen, Xuming Ran, Mingkun Xu, Qi Xu, Yi Xu, and Gang Pan. Efficient ann-snn conversion with error compensation learning. *arXiv preprint arXiv:2506.01968*, 2025.

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. Fineweb-edu: the finest collection of educational content, 2024. URL `https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu`.

Xinhao Luo, Man Yao, Yuhong Chou, Bo Xu, and Guoqi Li. Integer-valued training and spike-driven inference spiking neural network for high-performance and energy-efficient object detection. In *European Conference on Computer Vision*, pp. 253–272. Springer, 2024.

Changze Lv, Tianlong Li, Jianhan Xu, Chenxi Gu, Zixuan Ling, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. Spikebert: A language spikformer learned from bert with knowledge distillation. *arXiv preprint arXiv:2308.15122*, 2023.

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

Wolfgang Maass. On the computational power of winner-take-all. *Neural computation*, 12(11): 2519–2535, 2000.

Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pp. 1614–1623. PMLR, 2016.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Priyadarshini Panda, Sai Aparna Aketi, and Kaushik Roy. Toward scalable, efficient, and accurate deep spiking neural networks with backward residual connections, stochastic softmax, and hybridization. *Frontiers in Neuroscience*, 14:653, 2020.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

Haotong Qin, Yifu Ding, Mingyuan Zhang, Qinghua Yan, Aishan Liu, Qingqing Dang, Ziwei Liu, and Xianglong Liu. Bibert: Accurate fully binarized bert. *arXiv preprint arXiv:2203.06390*, 2022.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

David Vilares and Carlos Gómez-Rodríguez. Head-qa: A healthcare dataset for complex reasoning. *arXiv preprint arXiv:1906.04701*, 2019.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Hongzhi Wang, Xiubo Liang, Tao Zhang, Yue Gu, and Weidong Geng. Pssd-transformer: powerful sparse spike-driven transformer for image semantic segmentation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 758–767, 2024.

Jingya Wang, Xin Deng, Wenjie Wei, Dehao Zhang, Shuai Wang, Qian Sun, Jieyuan Zhang, Hanwen Liu, Ning Xie, and Malu Zhang. Training-free ann-to-snn conversion for high-performance spiking transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 2128–2136, 2026.

Xingrun Xing, Boyan Gao, Zheng Zhang, David A Clifton, Shitao Xiao, Li Du, Guoqi Li, and Jiajun Zhang. Spikellm: Scaling up spiking neural network to large language models via saliency-based spiking. *arXiv preprint arXiv:2407.04752*, 2024a.

Xingrun Xing, Zheng Zhang, Ziyi Ni, Shitao Xiao, Yiming Ju, Siqi Fan, Yequan Wang, Jiajun Zhang, and Guoqi Li. Spikelm: Towards general spike-driven language modeling via elastic bi-spiking mechanisms. In *International Conference on Machine Learning*, pp. 54698–54714. PMLR, 2024b.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Man Yao, Jiakui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer. *Advances in neural information processing systems*, 36:64043–64058, 2023a.

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*, 2023b.

Man Yao, JiaKui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. *arXiv preprint arXiv:2404.03663*, 2024.

Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. Ternarybert: Distillation-aware ultra-low bit bert. *arXiv preprint arXiv:2009.12812*, 2020.

Chenlin Zhou, Liutao Yu, Zhaokun Zhou, Zhengyu Ma, Han Zhang, Huihui Zhou, and Yonghong Tian. Spikingformer: Spike-driven residual learning for transformer-based spiking neural network. *arXiv preprint arXiv:2304.11954*, 2023a.

Chenlin Zhou, Han Zhang, Zhaokun Zhou, Liutao Yu, Liwei Huang, Xiaopeng Fan, Li Yuan, Zhengyu Ma, Huihui Zhou, and Yonghong Tian. Qkformer: Hierarchical spiking transformer using qk attention. *Advances in Neural Information Processing Systems*, 37:13074–13098, 2024.

Chenlin Zhou, Liutao Yu, Zhaokun Zhou, Han Zhang, Jiaqi Wang, Huihui Zhou, Zhengyu Ma, and Yonghong Tian. Spikingformer: A key foundation model for spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 2236–2244, 2026.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng YAN, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=frE4fUwz_h.

Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K Eshraghian. Spikegpt: Generative pre-trained language model with spiking neural networks. *arXiv preprint arXiv:2302.13939*, 2023.

APPENDIX

Limitations. Our energy estimates are based on operation counts rather than measurements on neuromorphic hardware. We evaluate models up to 1.0B parameters and do not assess long-context generation or larger-scale deployment; these settings remain for future work.

A T-LIF

Ternary-spiking-based Leaky Integrate-and-Fire (T-LIF) model. Compared to the original LIF neuron, the ternary spiking in SpikeLM (Xing et al., 2024b) extends binary spikes $\{0, 1\}$ to ternary values $\{-\alpha, 0, \alpha\}$ based on membrane potential intensity. The dynamics of the T-LIF model are formulated as:

$$U[t] = H[t-1] + X[t], \tag{22}$$

$$S[t] = \begin{cases} -1 \cdot \alpha[t], & \text{if } U[t] < -\alpha[t], \\ 0 \cdot \alpha[t], & \text{if } U[t] \in (-\alpha[t], +\alpha[t]), \\ +1 \cdot \alpha[t], & \text{if } U[t] > +\alpha[t], \end{cases} \tag{23}$$

$$H[t] = V_{\text{reset}}\,|b| + (\beta U[t])(1 - |b|), \tag{24}$$

Here, $X[t]$ is the input current, $S[t] = b\alpha[t]$ with $b \in \{-1, 0, 1\}$, and $U[t]$ is the membrane potential before reset. If no spike is emitted, $H[t] = \beta U[t]$; otherwise, $H[t] = V_{\text{reset}}$, where $\beta < 1$ is the decay factor.

B DATASET INTRODUCTION

General Language Understanding Evaluation (GLUE). GLUE benchmark (Wang et al., 2018) is a widely adopted framework for assessing natural language understanding (NLU) models across a variety of tasks, including single-sentence classification, sentence-pair classification, and linguistic acceptability. In our experiments, we focus on eight representative datasets within GLUE: MNLI (Multi-Genre Natural Language Inference): Predict whether a hypothesis is entailed, contradicted, or neutral with respect to a given premise across multiple text genres. QQP (Quora Question Pairs): Detect whether two questions from Quora convey the same meaning. QNLI (Question Natural Language Inference): Reformulated from QA, determines whether a sentence contains the answer to a question. SST-2 (Stanford Sentiment Treebank): Perform binary sentiment classification on movie reviews. CoLA (Corpus of Linguistic Acceptability): Judge whether a sentence is grammatically acceptable. STS-B (Semantic Textual Similarity Benchmark): Measure sentence-level semantic similarity on a scale from 0 to 5. MRPC (Microsoft Research Paraphrase Corpus): Identify whether two sentences are paraphrases. RTE (Recognizing Textual Entailment): Decide whether a hypothesis can be inferred from a premise, based on multiple entailment datasets.

Question Answering Tasks (QAT). ARC-e (Clark et al., 2018) and ARC-c (AI2 Reasoning Challenge, easy and challenge subsets) assess scientific knowledge and reasoning skills, with ARC-e focusing on simpler multiple-choice science questions and ARC-c including more complex ones that require advanced reasoning. BoolQ (Boolean Questions) (Clark et al., 2019) is a yes/no question-answering dataset derived from natural queries, requiring models to determine the truthfulness of a statement given a supporting passage. HeadQA (Head Question Answering) (Vilares & Gómez-Rodríguez, 2019) is a multilingual medical question-answering benchmark composed of exams for healthcare professionals, testing domain-specific knowledge. OBQA (OpenBookQA) (Mihaylov et al., 2018) evaluates a model's ability to answer elementary science questions by combining provided core facts with external common knowledge.

Commonsense Reasoning Tasks (CRT). HellaSwag (Zellers et al., 2019) is a large-scale dataset for grounded commonsense inference, where models must select the most plausible continuation of a given context. PIQA (Physical Interaction Question Answering) (Bisk et al., 2020) focuses on physical commonsense reasoning, requiring models to choose the more feasible solution to everyday tasks. Winograd (Winograd Schema Challenge) (Sakaguchi et al., 2021) is a coreference resolution benchmark designed to test commonsense reasoning by requiring models to resolve pronoun references that cannot be disambiguated by syntax alone.

C Energy Consumption

SNNs replace traditional multiply-accumulate (MAC) operations with low-power accumulate (AC) operations. For ANNs, arithmetic energy can be estimated from MAC counts. For example, given a linear layer with input dimension m and output dimension n, its arithmetic energy can be estimated as:

$$E_{\text{Linear}}^{\text{Ann}} = m \times n \times E_{MAC}, \tag{25}$$

SNNs convert MAC-based matrix multiplications to pure accumulate operations (ACs). For SNNs, given the same example as the ANN case, the theoretical energy consumption of the linear layer can be calculated by:

$$E_{\text{Linear}}^{\text{Snn}} = m \times n \times E_{AC} \times fr \times T, \tag{26}$$

where fr is the layer firing rate and T is the number of simulation time steps, following prior work (Kundu et al., 2021; Zhou et al., 2023b;a; Panda et al., 2020; Yao et al., 2023b). We assume that the MAC and AC operations are implemented on the 45nm hardware (Horowitz, 2014), where $E_{MAC} = 4.6pJ$ and $E_{AC} = 0.9pJ$. We estimate inference energy for sequences of 512 tokens.

D Vision-oriented Spiking Self-Attention

Vision-oriented Spiking Self-Attention (Zhou et al., 2023b;a) uses sparse, spike-valued $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ for vision task modeling without softmax operation and floating-point matrix multiplication. The calculation process of vision-oriented spiking self-attention is formulated as follows:

$$\mathbf{X}' = \mathrm{SN}(\mathbf{X}), \tag{27}$$

$$\mathbf{Q} = \mathrm{SN}_Q(\mathrm{Linear}_Q(\mathbf{X}')), \tag{28}$$

$$\mathbf{K} = \mathrm{SN}_K(\mathrm{Linear}_K(\mathbf{X}')), \tag{29}$$

$$\mathbf{V} = \mathrm{SN}_V(\mathrm{Linear}_V(\mathbf{X}')), \tag{30}$$

$$\mathrm{SSA}'(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{SN}\left(\mathbf{Q}\mathbf{K}^{\mathrm{T}}\mathbf{V} * s\right), \tag{31}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathcal{R}^{T \times N \times D}$, the spike-form $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ are computed by learnable linear layers. s is a scaling factor. SN means spiking neuron layer. The calculation of SSA avoids floating-point multiplication, meeting the property of SNNs.

E Hard Winner-Take-All in SNN

Given an input vector $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, the softmax function is defined as:

$$\mathrm{softmax}\,(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{32}$$

Introducing a temperature $\tau > 0$ gives:

$$\mathrm{softmax}_\tau\,(x_i) = \frac{e^{x_i/\tau}}{\sum_{j=1}^{n} e^{x_j/\tau}} \tag{33}$$

At $\tau = 1$, this reduces to standard softmax.

Lemma E.1. *If* $\mathbf{x}$ *has a unique maximum, temperature-scaled softmax converges to binary Hard WTA as* $\tau \to 0^+$.

$$\lim_{\tau \to 0^+} \textit{softmax}_\tau(x_i) = \begin{cases} 1 & \textit{if } i = \arg\max_j x_j \\ 0 & \textit{otherwise} \end{cases}$$

Without loss of generality, assume that x_1 is the unique maximum, so $x_1 > x_j$ for all $j \neq 1$.

$$\begin{aligned}\operatorname{softmax}_\tau(x_1) &= \frac{e^{x_1/\tau}}{e^{x_1/\tau} + \sum_{j=2}^{n} e^{x_j/\tau}} \\ &= \frac{1}{1 + \sum_{j=2}^{n} e^{(x_j - x_1)/\tau}}.\end{aligned} \tag{34}$$

For all $j \neq 1$, $x_j - x_1 < 0$, as $\tau \to 0^+$:

$$e^{(x_j - x_1)/\tau} \to 0, \quad \forall j \neq 1 \tag{35}$$

Then:

$$\lim_{\tau \to 0^+} \operatorname{softmax}_\tau (x_1) = \frac{1}{1+0} = 1 \tag{36}$$

For each $i \neq 1$, similarly:

$$\lim_{\tau \to 0^+} \operatorname{softmax}_\tau (x_i) = \lim_{\tau \to 0^+} \frac{e^{(x_i - x_1)/\tau}}{1 + \sum_{j=2}^{n} e^{(x_j - x_1)/\tau}} = 0 \tag{37}$$

Thus, the limit is one-hot when the maximum is unique. If $m > 1$ entries share the maximum, softmax instead converges to a uniform distribution over those entries, with mass $1/m$ on each; selecting a single winner then requires a tie-breaking rule.

F SPARSEMAX

As shown in Martins & Astudillo (2016), Sparsemax is defined as:

$$\operatorname{sparsemax}(\boldsymbol{a}) := \underset{\boldsymbol{p} \in \Delta^{K-1}}{\operatorname{argmin}} \|\boldsymbol{p} - \boldsymbol{a}\|^2 \tag{38}$$

Sparsemax computes the Euclidean projection of the input vector $\boldsymbol{a}$ onto the probability simplex. This projection often lies on the boundary of the simplex, resulting in sparse outputs. The solution of Eq. 38 is shown in Eq. 6 ($y_i = \max\{a_i - \tau, 0\}, \quad i = 1, \ldots, N.$), where τ is the adaptive threshold. Furthermore, τ can be expressed as follows. Let $a_{(1)} \geq a_{(2)} \geq \ldots \geq a_{(K)}$ be the sorted coordinates of $\boldsymbol{a}$, and define $k(\boldsymbol{a}) := \max\left\{k \in [K] \mid 1 + k a_{(k)} > \sum_{j \leq k} a_{(j)}\right\}$. Then,

$$\tau(\boldsymbol{a}) = \frac{\left(\sum_{j \leq k(\boldsymbol{a})} a_{(j)}\right) - 1}{k(\boldsymbol{a})} = \frac{\left(\sum_{j \in S(\boldsymbol{a})} a_j\right) - 1}{|S(\boldsymbol{a})|}, \tag{39}$$

where $S(\boldsymbol{a}) := \{j \in [K] \mid \operatorname{sparsemax}_j(\boldsymbol{a}) > 0\}$ is the support of $\operatorname{sparsemax}(\boldsymbol{a})$. In essence, all we need for evaluating the Sparsemax transformation is to compute the threshold $\tau(\boldsymbol{a})$; all coordinates above this threshold (the ones in the set $S(\boldsymbol{a})$) will be shifted by this amount, and the others will be truncated to zero. Therefore, Sparsemax can also be seen as an adaptive threshold form for the Top-K Winner-Take-All.

G DETAILED RESULTS

Tables 6 and 7 provide detailed results for the scaling study in Section 4.5. In particular, when scaled up to 1.0B parameters, WD-Spikingformer further improves its accuracy, nearly matching that of Qwen-1.5B (43.6% vs. 44.3%). However, its estimated inference energy is only about 17% of that of the Qwen-1.5B (574.7 mJ vs. 3398.3 mJ).

H DETAILED RESULTS ON SPIKING NEURON COMPARISON

For WE-Spikingformer pretraining, the training time comparison on 8 × 4090 GPUs is shown in Table 8. NI-LIF significantly boosts training simulation speed compared to T-LIF, while maintaining comparable model performance.

Model	SD	E (mJ)	T	ARC-e	ARC-c	BoolQ	HeadQA	OBQA	Avg. (%)
SpikeLLM-7B (Xing et al., 2024a)	✗	-	4	31.3	23.6	53.8	-	-	-
Qwen-1.5B (Yang et al., 2024)	✗	3398.3	-	26.2	26.9	60.3	25.7	27.1	33.2
WD-Spikingformer-0.4B	✓	238.4	4	30.0	22.3	37.8	26.0	26.1	28.4
WD-Spikingformer-1.0B	✓	574.7	4	33.8	20.3	37.8	26.4	26.6	29.0
WD-Spikingformer-1.0B*	✓	574.7	4	41.9	20.4	42.1	26.7	28.2	31.9

Table 6: The results on Question Answering Tasks (QAT). "Avg." denotes "Average Accuracy". "E (mJ)" means "Energy consumption (mJ)". "T" means time step. "SD" denotes "Spike-Driven". WD-Spikingformer-1.0B* means WD-Spikingformer-1.0B with 0.5B tokens during pretraining.

Model	SD	E (mJ)	T	HellaSwag	PIQA	Winograd	Avg. (%)
SpikeLLM-7B Xing et al. (2024a)	✗	-	4	33.9	53.4	51.5	46.3
Qwen-1.5B (Yang et al., 2024)	✗	3398.3	-	26.5	53.7	52.8	44.3
WD-Spikingformer-0.4B	✓	238.4	4	25.9	53.4	50.2	43.2
WD-Spikingformer-1.0B	✓	574.7	4	26.0	55.8	49.1	43.6
WD-Spikingformer-1.0B*	✓	574.7	4	26.4	55.7	50.3	44.1

Table 7: The results on Commonsense Reasoning Tasks. "Avg." denotes "Average Accuracy". "E (mJ)" means "Energy consumption (mJ)". "T" means time step. "SD" denotes "Spike-Driven". WD-Spikingformer-1.0B* means WD-Spikingformer-1.0B with 0.5B tokens during pretraining.

Backbone	Neuron	T	Mean GLUE score (%)	Training time (hours)
WE-Spikingformer	NI-LIF	4	65.7	26
WE-Spikingformer	T-LIF	4	66.3	121

Table 8: Spiking neuron comparison for WE-Spikingformer pretraining.

I Differences between Hard WTA and Max Pooling

In this section, we highlight the key differences between the Hard WTA mechanism and max pooling:

Differences in information retention: max pooling retains the maximum value within a local receptive field. In contrast, Hard WTA assigns a value of 1 to the neuron with the highest response while suppressing all others to 0, yielding a binary and highly sparse representation. Here is a simple example illustrating the operation process: Input: [0.2, 0.8, 0.5, 0.1], max pooling: [0.8], Hard WTA: [0, 1, 0, 0].

Differences in dimensionality changes: With a downsampling stride, max pooling reduces the spatial resolution of the feature map (e.g., "28×28 $\rightarrow$ 14×14"). In contrast, Hard WTA preserves the original dimensionality, performing competitive selection without altering the feature size.

Scope of competition. Spatial max pooling operates over local neighborhoods within a feature map. In our attention modules, Hard WTA selects among key-token positions for each query token, with causal masking restricting the eligible keys in CWSSA.

Specifically, MaxFormer applies max pooling within the patch embedding module in vision tasks. Beyond downsampling, this operation enhances the representation of high-frequency components, facilitating more effective visual feature extraction. SpikePool incorporates max pooling within a {LIF–MaxPool–Conv–BN} block to replace the whole spiking self-attention mechanism, targeting small-scale event-based data. Our method uses a Winner-Take-All mechanism to replace the Softmax operation in self-attention, enabling sparse, spike-driven competition for spike-based language modeling.

J Additional Ablation Study

Surrogate gradient on GLUE. We compare Softmax and ReLU (Agarap, 2018) as surrogate gradients for Hard WTA in WE-Spikingformer at $T = 4$. As shown in Table 9, Softmax achieves an average GLUE score of 65.7%, compared with 40.4% for ReLU, a gain of 25.3 percentage points. This supports our choice of Softmax, motivated by the limiting relationship between Softmax and Hard WTA in Appendix E.

Surrogate gradient	MNLI_mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC_F1	RTE	Avg. (%)
Softmax	73.4	83.8	78.3	85.6	23.7	56.0	76.4	48.4	65.7
ReLU	31.8	63.2	50.5	50.9	0	4.3	75.0	47.3	40.4

Table 9: Surrogate gradients for Hard WTA in WE-Spikingformer on GLUE ($T = 4$). Avg. is the mean of the eight task scores.

Top-K WTA on GLUE. Table 10 extends the GLUE ablation in Table 5 by varying K in WE-Spikingformer at $T = 4$. Increasing K from 1 (Hard WTA) to 3 and 5 improves the average score from 65.7% to 66.3% and 66.5%, respectively, compared with 66.8% for Softmax. These modest gains come with reduced activation sparsity, motivating Hard WTA when sparsity is a priority. Top-K selection can use comparison and masking operations instead of the exponentiation and normalization required by Softmax (Dong et al., 2025); the table reports task performance rather than measured hardware energy savings.

Method	MNLI_mm	QQP	QNLI	SST-2	CoLA	STS-B	MRPC_F1	RTE	Avg. (%)
Softmax	72.5	84.7	76.0	87.2	24.4	54.5	79.7	55.6	66.8
Hard WTA ($K = 1$)	73.4	83.8	78.3	85.6	23.7	56.0	76.4	48.4	65.7
Top-K WTA ($K = 3$)	73.4	84.0	78.8	86.1	24.1	55.9	78.2	49.8	66.3
Top-K WTA ($K = 5$)	73.1	84.4	79.4	86.0	24.8	55.4	79.1	50.1	66.5

Table 10: Top-K WTA in WE-Spikingformer on GLUE ($T = 4$). Hard WTA is Top-K WTA with $K = 1$. Avg. is the mean of the eight task scores.

WTA mechanisms and time steps on CRT. Table 11 complements the GLUE ablation in Table 5 with results for WD-Spikingformer-0.4B on CRT. At $T = 4$, Hard WTA and Sparsemax both achieve 43.2% average accuracy, while Top-K WTA ($K = 3$) achieves 42.9%, showing no accuracy benefit over Hard WTA. For Hard WTA, increasing T from 2 to 4 and 8 improves accuracy from 40.1% to 43.2% and 43.7%, respectively, with a smaller gain beyond four time steps.

Method	T	Avg. (%)
Hard WTA	4	43.2
Top-K WTA	4	42.9
Sparsemax	4	43.2
Hard WTA	8	43.7
Hard WTA	4	43.2
Hard WTA	2	40.1

Table 11: WTA mechanisms and time steps for WD-Spikingformer-0.4B on CRT. All methods use causal masking; Top-K WTA uses $K = 3$. T denotes time steps; Avg. denotes average accuracy.