Spiking Neural Networks Need High-Frequency Information

Yuetong Fang[1], **Deming Zhou**[1], **Ziqing Wang**[4], **Hongwei Ren**[1],
Zecui Zeng[3], **Lusong Li**[3], **Shibo Zhou**[2,*], **Renjing Xu**[1*]

[1]The Hong Kong University of Science and Technology (Guangzhou)
[2]Brain Mind Innovation INC [3]JD Explore Academy [4]Northwestern University

yfang870@connect.hkust-gz.edu.cn,
bob@brain-mind.com.cn, renjingxu@hkust-gz.edu.cn

Abstract

Spiking Neural Networks promise brain-inspired and energy-efficient computation by transmitting information through binary (0/1) spikes. Yet, their performance still lags behind that of artificial neural networks, often assumed to result from information loss caused by sparse and binary activations. In this work, we challenge this long-standing assumption and reveal a previously overlooked frequency bias: **spiking neurons inherently suppress high-frequency components and preferentially propagate low-frequency information**. This frequency-domain imbalance, we argue, is the root cause of degraded feature representation in SNNs. Empirically, on Spiking Transformers, adopting Avg-Pooling (low-pass) for token mixing lowers performance to 76.73% on Cifar-100, whereas replacing it with Max-Pool (high-pass) pushes the top-1 accuracy to 79.12%. Accordingly, we introduce **Max-Former** that restores high-frequency signals through two frequency-enhancing operators: (1) extra Max-Pool in patch embedding, and (2) Depth-Wise Convolution in place of self-attention. Notably, **Max-Former** attains 82.39% top-1 accuracy on ImageNet using only 63.99M parameters, surpassing Spikformer (74.81%, 66.34M) by +7.58%. Extending our insight beyond transformers, our **Max-ResNet-18** achieves state-of-the-art performance on convolution-based benchmarks: 97.17% on CIFAR-10 and 83.06% on CIFAR-100. We hope this simple yet effective solution inspires future research to explore the distinctive nature of spiking neural networks. Code is available: https://github.com/bic-L/MaxFormer.
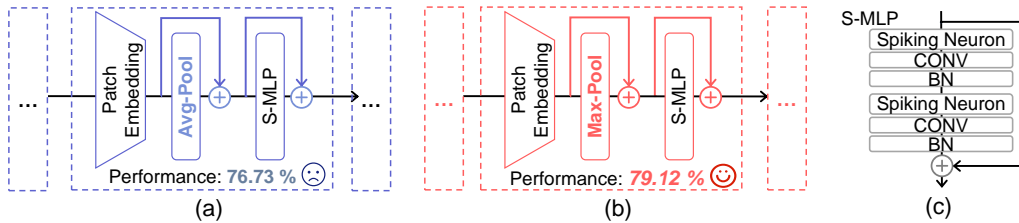
Figure 1: Spiking Transformer architectures: (a) Avg-Pool vs. (b) Max-Pool for token mixing, with (c) detailed implementation of the Spiking MLP (S-MLP) block. In mainstream (non-spiking) Vision Transformer research, Avg-Pool that captures global low-frequency patterns is a more common token mixing strategy than Max-Pool (high-pass) [1, 2]. Surprisingly, in Spiking Transformers, replacing Avg-Pool with Max-Pool yields a +2.39% improvement on Cifar-100.
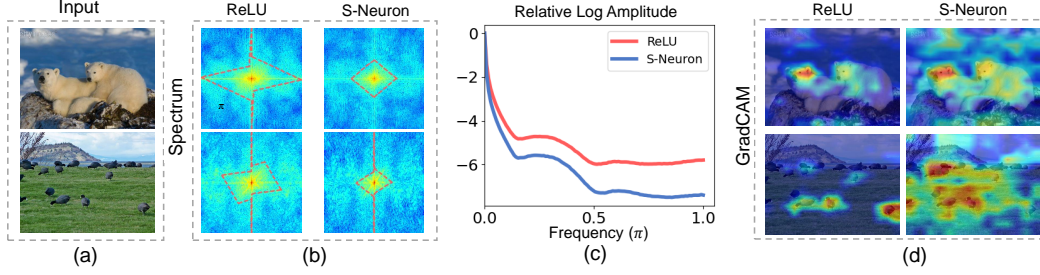
---

*Corresponding author

Figure 2: Comparison between ReLU and spiking neuron (S-Neuron): (a) Input images; (b) Fourier spectrum analysis of output features processed as input→ activation→ weighting, with high-frequency regions marked (red dashed boxes: regions >0.55× max amplitude) and (c) the corresponding relative log amplitude; (d) GradCAM comparison with identical architectural setting following [9], with the converted Spiking Transformer using 256 timesteps. Spiking neurons cause the rapid dissipation of high-frequency components, which consequently leads to the degradation of feature representations.

1 Introduction

Spiking neural networks (SNNs) are emerging as an energy-efficient alternative to conventional artificial neural networks (ANNs) [3, 4]. Their efficiency arises from spiking neurons that utilize spatiotemporal dynamics to mimic biological computation in the human brain [5]. In ANNs, all neurons within the same layer must await the complete processing of real-valued, dense tensors before any information can flow to the subsequent layer. SNNs, however, transmit information asynchronously, with spiking neurons consuming energy only when receiving or emitting spikes ("1"), otherwise remaining inactive [6, 7]. This binary activation pattern enables SNNs to replace the energy-intensive multiply-and-accumulate (MAC) operations that are essential in ANNs with much simpler spike-based accumulation. Leveraging the energy-efficiency benefits, modern SNN variants, such as Spiking Transformers that integrate the powerful Transformer architecture with spike-based computing, have gained growing attention [8, 9].

Despite their energy efficiency, the discrete nature of spike-based computation presents both opportunities and challenges. A major obstacle for SNNs remains their performance gap relative to ANNs. This gap is often attributed to the so-called "representation error" [10–12], which argues binary spike trains inherently limit the precision of feature representations compared to continuous activations. However, this seems inconsistent with the established consensus in the standard deep learning literature, where low-bit and even binary networks can still achieve comparable accuracy [13, 14]. Further, it should be noted that SNNs operate on temporal sequences: while spiking neurons strictly transmit binary signals at each individual time step, a train of spikes spanning n simulation timesteps can encode activation values with at least $\log(n)$-bit precision [15, 16]. The conflicting observation reveals an unexplored dimension in understanding SNN performance limitations.

It is natural to think from the frequency domain. Spiking neurons produce discrete, pulse-like activations, fundamentally distinguishing their frequency response from continuous activation functions commonly used in standard networks (e.g., ReLU [17]). Prior works have suggested that spiking neurons may enrich signals with local details (high frequencies) [18, 19]. However, in Figure 2 (b-c), we observed a surprising phenomenon: when examining the end-to-end information flow of input → activation → weighting, rather than focusing solely on the property of activation functions, spiking neurons tend to propagate low-frequency information more prominently than ReLU.

Feature degradation observed in SNNs may instead originate from the rapid dissipation of high-frequency components, which prevents the network from effectively capturing local, fine-grained information (Figure 2 (d)). To support this finding, we perform a simple experiment in which non-parametric pooling operators, i.e., Max-Pool and Avg-Pool, serve as token mixers in Spiking Transformers, shown in Figure 1. From the frequency domain perspective, Max-Pool excels at capturing local high-frequency details (e.g., local edges/textures), whereas Avg-Pool favors global low-frequency patterns. Intriguing, Spiking Transformers exhibit opposite preferences to ANNs in token mixing: while standard Transformers typically employ Avg-Pool for token mixing [1, 2], replacing it with Max-Pool in Spiking Transformers yields a +2.39% improvement on CIFAR-100, making it surpass the well-tuned Spikformer [8] baseline by 0.97%.

Overall, this work provides further theoretical and empirical evidence supporting the view that high-frequency information is essential for SNNs:

- We provide the first theoretical proof that spiking neurons inherently act as low-pass filters at the network level, revealing their tendency to suppress high-frequency features.
- We propose **Max-Former**, which restores high-frequency information in Spiking Transformers via two lightweight modules: extra Max-Pool in patch embedding and Depth-Wise Convolution (DWC) in place of early-stage self-attention.
- Restoring high-frequency information significantly improves performance while saving energy cost. On ImageNet, **Max-Former** achieves 82.39% top-1 accuracy (+7.58% over Spikformer) with 30% energy consumption and lower parameter count (63.99M vs. 66.34M).
- Extending the insight beyond transformers, **Max-ResNet-18** achieves state-of-the-art performance on convolution-based benchmarks: 97.17% on CIFAR-10 and 83.06% on CIFAR-100.

We believe this straightforward yet powerful solution will motivate future research to explore the unique properties of SNNs, beyond the established practice in standard deep learning.

2 Preliminary and Related Works

2.1 Spiking Neuron Models

SNNs implement spike-driven processing through biologically-inspired neuron models for non-linear activations. The Leaky Integrate-and-Fire (LIF) model is a widely adopted abstraction of this behavior, offering an effective balance between biological plausibility and computational efficiency [20]. The discretized LIF model under each simulation timestep n can be formulated as:

$$U[n] = f(V[n-1], I[n]), \qquad (1)$$
$$S[n] = H\big(U[n] - V_{\text{th}}\big), \qquad (2)$$

$$V[n] = \begin{cases} U[n] - V_{\text{th}}, & S[n] = 1, \\ U[n], & S[n] = 0, \end{cases} \qquad (3)$$

where β is the decay factor, V_{th} is the firing threshold, and $H(\cdot)$ refers to the Heaviside step function that determining spike generation: $S[n] = H\big(U[n] - V_{\text{th}}\big) = 1$ when $U[n] \geq V_{\text{th}}$, otherwise remains inactive ($S[n] = 0$). The charging process of the LIF neuron is determined by $f(\cdot)$:

$$f(V[n-1], I[n]) = \beta V[n-1] + (1-\beta)I[n] \qquad (4)$$

At each timestep n, the current membrane potential $U[n]$ is updated by integrating the time-domain signal $I[n]$ corresponding to input data or intermediate operations like Conv and MLP. If $U[n]$ exceeds the threshold V_{th}, the neuron fires a spike ($S[n] = 1$). $V[n]$ records membrane potential over time given the decay factor β and output spike activity. If the neuron does not fire, then $V[n] = U[n]$. Notably, the LIF model simplifies to integrate-and-fire (IF) neurons when eliminating the membrane potential decay process between timesteps. Its charging process can be formulated as:

$$f(V[n-1], I[n]) = V[n-1] + I[n] \qquad (5)$$

2.2 Spiking Neural Networks

Drawing inspiration from biological neurons, SNNs extend conventional ANNs by incorporating temporal dynamics and discrete spike-based communication [5]. Leveraging this spike-driven mechanism, neuromorphic chips implement computation through event-driven spike routing and accumulation, which substitutes energy-intensive matrix–vector multiplications [7, 6]. This facilitates high parallelism, scalability, and exceptional power efficiency, with power consumption typically in the range of tens to hundreds of milliwatts [21].

Recently, the development of modern SNNs, e.g., Spiking Transformer, has demonstrated both attractive performance and reduced energy consumption [8, 9, 22]. Spikformer [8] pioneered the spike-based self-attention mechanism called Spiking Self Attention (SSA) that utilizes sparse spike-form Query, Key, and Value vectors to eliminate the need for energy-intensive softmax operations. Following its success, many works endeavor to enhance Spiking Transformers by adapting advanced ANN Transformer architectures [23, 24] or devising complicated spike coding mechanisms to reduce representation error (e.g., multi-threshold [25]/ multi-spike neurons [26, 27]).

Here, we instead address a fundamental question: What truly limits SNNs' performance compared to ANNs? Our investigation reveals that the answer lies in frequency properties -— specifically, that spiking neurons function as low-pass filters, impeding the propagation of high-frequency detail

within the network. In addition to theoretical proof, we probe the importance of high-frequency information through our Max-Former, which features two frequency-enhancing operators: Max-Pool in patch embedding and DWC in place of early-stage self-attention. We further validate this principle in convolutional architectures with our proposed Max-ResNet.

3 Methods

In this section, we first present a theoretical analysis of the frequency properties of spiking neurons. We show that, although the raw output spike trains of spiking neurons appear spectrally all-pass due to their impulse-shaped spike waveform, the resulting high-frequency components are merely superficial and cannot be propagated through the network. In fact, spiking neurons act as low-pass filters at the network level. This is a fundamental problem that has been overlooked in previous works. Building on this insight, we probe the importance of high-frequency information in SNNs through Max-Former, which strategically employs high-pass operators (Max-Pool and DWC) to restore high-frequency details and avoid feature degradation.
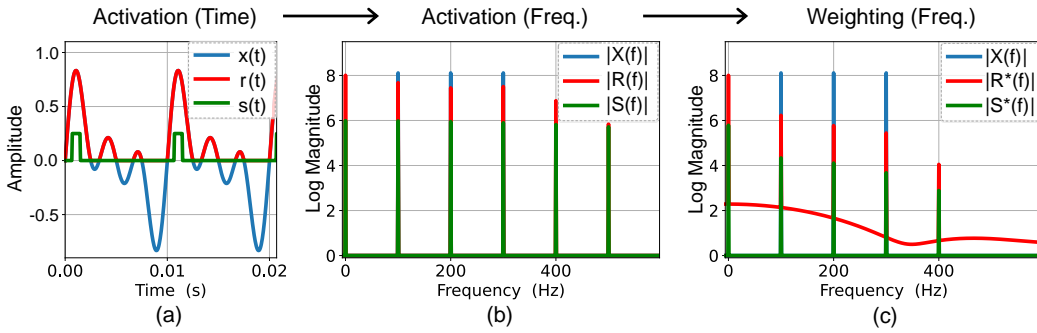

Figure 3: Time-frequency analysis of ReLU and spiking neurons. (a) Time-domain signals of input $x(t) = \frac{1}{3}(\sin(2\pi \cdot 100t) + \sin(2\pi \cdot 200t) + \sin(2\pi \cdot 300t))$ (blue), ReLU-processed: $r(t)$ (red), spiking output of a LIF neuron with the $\beta = 0.25$: $s(t)$ (green). (b) Fourier analysis of $x(t)$, $r(t)$, and $s(t)$. (c) Fourier analysis of linear transformed (CONV/MLP) activations, where ReLU expands the frequency bandwidth of the input signal, while the spiking neuron shows high-frequency attenuation.

3.1 Spiking Neurons are Low-pass Filters

We begin with an intuitive time-frequency analysis using an input $x(t) = \frac{1}{3}(\sin(2\pi \cdot 100t) + \sin(2\pi \cdot 200t) + \sin(2\pi \cdot 300t))$ as shown in Figure 3. The results reveal three key observations: (1) In the time domain, the ReLU output ($r(t)$) perfectly follows $x(t) > 0$ while spiking neurons selectively respond to 100Hz (Figure 3 (a)); (2) However, the spiking outputs' spectral response |S(f)| still appears nearly all-pass, which contradicts the low-frequency behavior observed in the time domain. These spurious high-frequency components actually arise from the impulse-shaped spike waveform itself rather than from genuine (Figure 3 (a-b)); (3) The waveform-induced high-frequency components cannot be propagated across layers, resulting in a network-level low-pass behavior. When considering the whole process from input → activation → linear transform, ReLU expands the frequency bandwidth of $x(t)$ [28], whereas the spiking neuron exhibits strong high-frequency attenuation (Figure 3 (c)).

We first examine the charging process of spiking neurons to theoretically analyze their frequency-selective properties. Given Equ. (3) and Equ. (4), this can be formulated as:

$$V[n] = \beta V[n-1] + (1-\beta)I[n] \tag{6}$$

Applying the Z–transform with $\mathcal{Z}\{V[n-1]\} = z^{-1}V(z)$ yields:

$$V[z] = \beta z^{-1}V[z] + (1-\beta)I[z] \tag{7}$$

which can be rearranged to formulate the transfer function from input current to membrane potential:

$$H(z) = \frac{V(z)}{I(z)} = \frac{1-\beta}{1-\beta z^{-1}}, \quad 0 \leq \beta < 1 \tag{8}$$

Equ. (8) is exactly the form of a first-order infinite-impulse-response (IIR) low-pass filter with a single pole at $z = \beta$. Accordingly, as β, i.e., the pole, approaches 1, LIF neurons exhibit stronger

4

low-frequency selectivity. Notably, the decay factor β correlates to the membrane time constant τ as $\beta = 1 - \frac{1}{\tau}$, with τ ranges from 1 to $+\infty$, a smaller τ yields a smaller β. From an intuitive standpoint, a shorter time constant allows the membrane potential to respond within narrower temporal windows, rendering the neuron more sensitive to higher-frequency inputs.

From the individual neuron to the network-level information transmission, the average membrane potential is positively and consistently correlated with the possibility of spike firing during operation. We approximate the inherently nonlinear spike-generation process as linear around the firing threshold V_{th}. Denoting the firing rate by $fr(V)$, we define the local gain k and approximate the Z–domain spike train $S(z)$ by:

$$k = \left.\frac{\partial fr}{\partial V}\right|_{V=V_{th}} \quad\quad (9) \quad\quad\quad\quad S(z) \approx k\,V(z). \quad\quad (10)$$

When output spike trains are weighted by a causal synaptic kernel $w[n]$, the Z-transformed output current ($y[n] = w[n] * s[n]$) can be denoted as $Y(z) = W(z)\,S(z)$. The overall input-to-output transfer function is obtained by combining this with (8) and (10):

$$H'(z) = \frac{Y(z)}{I(z)} = S(z)\,W(z)\,H(z) = k\,W(z)\,\frac{1-\beta}{1-\beta\,z^{-1}}. \quad\quad (11)$$

The first-order low-pass IIR characteristics of $H(z)$ make the system $Y(z)$ inherently favor low-frequency signal components, regardless of whether the synaptic kernel $W(z)$ or the spike coding process $S(z)$ changes the gain or phase response. The low-pass term $(\frac{1-\beta}{1-\beta z^{-1}})^L$ further amplifies the system's frequency selectivity when the process $H'(z)$ is cascaded L times (layers). The complete formula is as follows:

$$H'_L(z) = \frac{Y_L(z)}{I(z)} = \prod_{i=1}^{L}\big[S_i(z)\,W_i(z)\,H(z)\big] = \Big(\prod_{i=1}^{L} k_i\,W_i(z)\Big)\Big(\frac{1-\beta}{1-\beta z^{-1}}\Big)^L \quad\quad (12)$$

In the special case of the non-leaky IF neurons, which obey the charging process in (5), $H(z)$ is formulated as:

$$H_{(z)} = \frac{1}{1-z^{-1}}, \quad\quad (13)$$

This corresponds to an ideal discrete-time low-pass filter with a pole at $z = 1$, which can yield a consistent conclusion with our previous analysis.

3.2 Max-Former

It remains unclear whether high-frequency information is truly important for SNNs and whether restoring it can improve performance. Therefore, we systematically investigate the low-pass filtering characteristics of spiking neurons through Max-Former. To decouple frequency effects from model complexity, we: (1) replace self-attention with high-frequency-preserving DWC in the early stages, and (2) add Max-Pool in patch embedding to compensate spiking neurons' low-pass preference. Notably, compared to the quadratic computational complexity of self-attention, DWC and Max-Pool only require linear complexity relative to the sequence length and are more parameter-efficient. We consistently adopt the LIF neuron model throughout this work.

3.2.1 Overall Architecture

Figure 4(a) illustrates the overall framework of Max-Former. The architecture consists of 3 stages with $\frac{H}{4} \times \frac{W}{4}$, $\frac{H}{8} \times \frac{W}{8}$, and $\frac{H}{16} \times \frac{W}{16}$ tokens respectively, where H and W denote the height and width of the input image. Critically, MaxFormer processes information through discrete spikes over time. This spike-driven computing paradigm supports two types of input:

(1) Event Streams: Asynchronous events $e = [x, y, t, p]$ containing spatial coordinates (x, y), timestamp t, and polarity p are converted to event frames through temporal binning. Given original resolution dt_o and target $dt = \alpha dt_o$, events are aggregated over α consecutive bins:

$$I_t = \sum_{k=\alpha t}^{\alpha(t+1)-1} S_k \in \mathbb{R}^{2 \times h \times w} \quad\quad (14)$$

where S_k denotes raw event data. The whole process denoises raw events and converts them to frame sequences at target temporal resolutions.

(2) Static Images: Conventional images are converted to spike sequences by: 1) Repeating static frames T times, 2) Encoding pixel intensities to spikes using spiking neurons. The resulting input is formulated as: $I = \text{Spiking\_Embed}(\{I_t\}_{t=1}^{T})$, which contains identical information across timesteps.
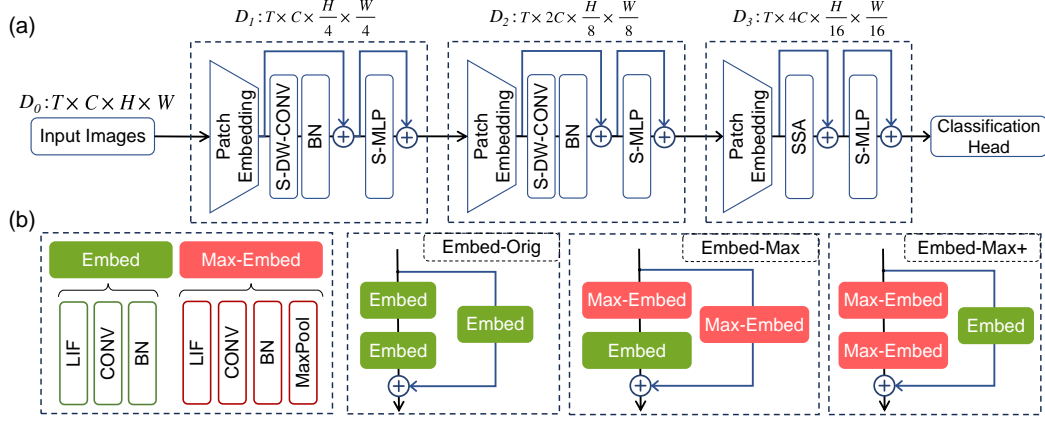
Figure 4: (a) Overview of Max-Former: we restore high-frequency signals by using lightweight DWCs instead of self-attention in the early stages. Following the hierarchical design of [29], Max-Former adopts a 3-stage architecture. D_i: feature dimensions of stage-i. (b) In Max-Former's patch embedding stage, we propose three configurations (Embed-orig, Embed-Max, and Embed-Max+) to enhance high-frequency components.

3.2.2 Patch Embedding

To transform the input into a tokenized representation, given input $\{S\} \in \mathbb{R}^{T \times C \times H \times W}$, the process of patch embedding is formulated as:

$$\mathbf{Y} = (\mathcal{G}_1(\{S\}) + G_2(\{S\})), \quad \mathbf{Y} \in \mathbb{R}^{T \times C' \times H' \times W'} \tag{15}$$

where $C' = 2C$, and $H' = \lfloor H/P \rfloor$, $W' = \lfloor W/P \rfloor$ with the patch size $P = 4$. To address spiking neurons' inherent frequency preference, we present three patch embedding configurations as shown in Figure 4(b) :

$$\text{Embed-Orig} \quad : (\mathcal{G}_1, \mathcal{G}_2) = (\text{Embed}, \text{Embed}) \tag{16}$$

$$\text{Embed-Max} \quad : (\mathcal{G}_1, \mathcal{G}_2) = (\text{Max-Embed}, \text{Embed}) \tag{17}$$

$$\text{Embed-Max+} : (\mathcal{G}_1, \mathcal{G}_2) = (\text{Max-Embed}, \text{Max-Embed}) \tag{18}$$

where: Embed $\equiv$ {LIF - CONV - BN}, and Max-Embed $\equiv$ {LIF - CONV - BN - MaxPool}.

3.2.3 Token Mixing

In Transformers, lower layers typically require more high-frequency details, while higher layers benefit from more global information [30, 31]. Like biological vision, high-frequency details enable early stages to learn low-level features while progressively building local-to-global representations. Accordingly, we replace early-stage self-attention with DWC to preserve high frequencies essential for local feature learning. Given input embedding $\mathbf{Y} \in \mathbb{R}^{T \times C \times H \times W}$, the spiking DWC is defined as:

$$\mathbf{Z}_c(\mathbf{Y})[i] = \text{LIF}(\sum_{j \in \Omega(i)} w_{c,j} \cdot \mathbf{Y}_c[j]) \tag{19}$$

where $\Omega(i)$ denotes the local neighborhood of position i, $w_{c,j}$ represents learnable convolution weights for channel c, and $X_c, Z_c \in \mathbb{R}^{T \times H \times W}$ is the input and output slice for channel c. For the final stage, we implement token mixing via Spiking Self-Attention (SSA) [8]. The SSA computation follows:

$$\mathbf{Z} = \text{LIF}(\text{BN}(\mathbf{YW})), \quad \mathbf{Z} \in \{Q, K, V\} \tag{20}$$

$$\text{SSA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{LIF}(\mathbf{QK^TV} \cdot s) \tag{21}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{T \times N \times H \times W}$ are spike-form tensors generated by learnable linear layers, s is a scaling factor. SSA eliminates floating-point multiplications, ensuring spike-driven compatibility.

3.3 Membrane Shortcut

Residual learning and shortcuts enable very deep networks to be trained effectively by providing identity shortcuts that preserve information flow and mitigate vanishing gradients [32]. For SNNs, a crucial consideration is maintaining compatibility with the spike-driven computing paradigm throughout all operations. As Fig. 5 shows, existing SNN shortcut implementations fall into three categories: (1) Vanilla Shortcut [33], (2) Pre-Spike Shortcut [34], and (3) Membrane Shortcut [35].
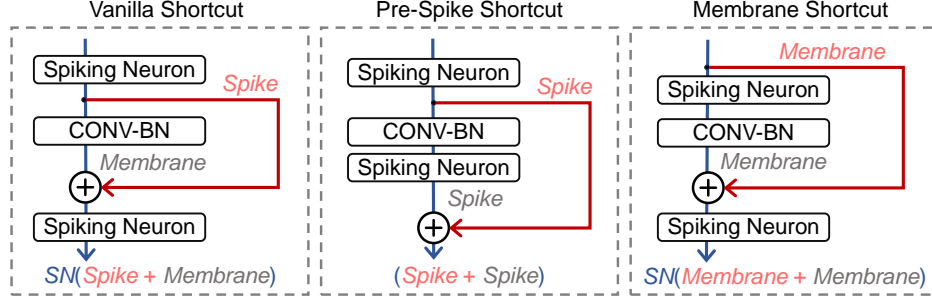
6

Figure 5: Shortcut connection in SNNs. (Left) Vanilla Shortcut that combines spike and membrane potential. (Middle) Pre-Spike Shortcut that adds spike signals before neuron charging. (Right) Membrane Shortcut that directly connects membrane potentials, ensuring identical potential mapping while strictly preserving the spike-driven computing paradigm throughout the network.

The Vanilla Shortcut scheme [33] directly connects spikes (binary) to membrane potentials (continuous), leading to a distribution mismatch that inherently violates the identity mapping principle. The Pre-Spike Shortcut [34] adds spike signals before neuron charging, resulting in summation values that range from 0 to 2. This disrupts the binary spike representation and spike-driven flow in SNNs.

In this work, we adopt the Membrane Shortcut [35] for its dual advantages: it preserves identity mapping by directly connecting membrane potentials while maintaining binary spike outputs that remain inherently compatible with spike-driven computation. Unlike Vanilla or Pre-Spike Shortcuts, this approach ensures both mathematical consistency with residual learning principles and seamless compatibility with spike-driven operations. We provide a detailed analysis of its impact on model performance and energy costs in **Appendix D**.

4 Experiment

As shown in Figure 6, Max-Former restores high-frequency information by grafting the merits of frequency-enhancing operators: Max-Pool and DWC. To empirically probe the importance of high-frequency information in Spiking Transformers, we evaluate Max-Former through comprehensive experiments on static datasets (CIFAR-10 [36], CIFAR-100 [37] and ImageNet [38]) and neuromorphic datasets (CIFAR10-DVS [39], DVS128 Gesture [40]), with architecture configurations detailed in Table 1. In addition, we design Max-ResNet to further investigate the effect of high-frequency restoration in convolutional architectures (model implementation is detailed in **Appendix A.3**). Experimental settings and energy analysis methods are detailed in **Appendix A** and **B**.
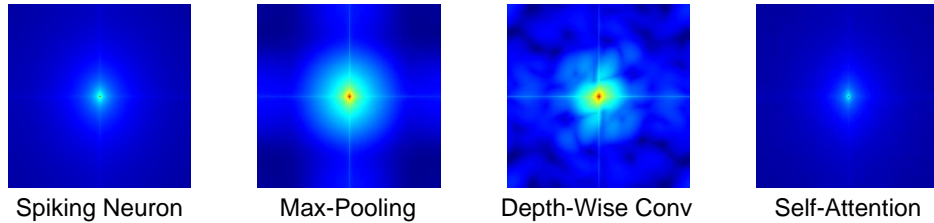
Figure 6: Fourier spectrum of Spiking Neurons, Spiking Max-Pool, Spiking Depth-Wise Convolution and Spiking Self-attention.

Table 1: Max-Former architecture configurations for different classification tasks. Notation DWC-N denotes depth-wise convolution with kernel size $N \times N$. For block settings: CIFAR-10/100: 3 stages (1/1/2 blocks); ImageNet: 3 stages (1/3/7 blocks); Neuromorphic: 2 stages (1/1 blocks).

Dataset	Stage 1		Stage 2		Stage 3	
	Patch Embed	Token Mix	Patch Embed	Token Mix	Patch Embed	Token Mix
Cifar10 [36]/100 [37]	Embed-Orig	Identity	Embed-Max	DWC-3	Embed-Max	SSA
ImageNet [38]	Embed-Orig	DWD-7	Embed-Max	DWC-5	Embed-Max	SSA
Neuromorphic [39, 40]	Embed-Max+	DWC-3	Embed-Max	SSA	—	—

7

4.1 Results on CIFAR and Neuromorphic Datasets

As shown in Table 2, Max-Former delivers performance improvements across both static datasets (CIFAR10/CIFAR100) and neuromorphic datasets (DVS128/CIFAR10-DVS). Notably, for CIFAR10/100 classification, its first stage only uses identity mapping for token mixing (Table 1, yet still attains attractive results. Max-Former achieves 97.04% accuracy on CIFAR10 with only 6.57M parameters at T=4, surpassing Spikformer (95.51%, 9.32M), S-Transformer (95.60%, 10.28M), and QKFormer (96.18%, 6.74M). Similarly, on CIFAR100, Max-Former attains 82.65% accuracy, significantly outperforming Spikformer (78.21%), S-Transformer (78.40%), and QKFormer (81.57%).

Max-Former and QKFormer share a similar hierarchical architecture, though QKFormer originally employs pre-spike shortcuts [24]. For a fair comparison, we additionally implemented QKFormer with the Membrane Shortcut (denoted as MS-QKFormer in the table) using identical training configurations. Max-Former still outperforms MS-QKFormer by 0.2% on CIFAR10 (97.04% vs. 96.84%) and by 1.08% on CIFAR100 (82.65% vs. 81.57%), while requiring slightly fewer parameters (6.57M vs. 6.74M). For neuromorphic datasets, Max-Former maintains this performance advantage. On DVS128, it achieves 98.6% accuracy, matching MS-QKFormer with the membrane shortcut. On CIFAR10-DVS, Max-Former reaches 84.2% accuracy, exceeding MS-QKFormer (82.3%) by 1.9% and surpassing other spike-driven models like S-Transformer (80.0%) and SWformer (83.9%).

Table 2: Performance Comparison on CIFAR10 [36], CIFAR100 [37], DVS128 [40] and CIFAR10-DVS [39]. Param.: Parameter (M); Acc.:Top-1 Accuracy (%); T: Simulation Timestep. [*]Models trained from scratch using identical configurations.

Method	CIFAR10			CIFAR100			DVS128			CIFAR10-DVS			Membrane Shortcut
	Param.	T	Acc.	Param.	T	Acc.	Param.	T	Acc.	Param.	T	Acc.	
ResNet-19 (ANN) [24]	12.63	1	94.97	12.63	1	75.35	—	—	—	—	—	—	—
Max-Former (ANN)	6.57	1	96.82	6.60	1	82.41	—	—	—	—	—	—	—
Spikformer [8]	9.32	4	95.51	9.32	4	78.21	2.57	16	98.3	2.57	16	80.9	✗
S-Transformer [41]	10.28	4	95.60	10.28	4	78.40	2.57	16	99.3	2.57	16	80.0	✓
SWformer [19]	7.51	4	96.10	7.51	4	79.30	-	-	-	2.05	16	83.9	✓
QKFormer [24]	6.74	4	96.18	6.74	4	81.15	1.50	16	98.6	1.50	16	84.0	✗
MS-QKFormer[*]	6.74	4	96.84	6.74	4	81.57	1.50	16	98.6	1.50	16	82.3	✓
Max-Former[*]	**6.57**	**4**	**97.04**	**6.60**	**4**	**82.65**	**1.45**	**16**	**98.6**	**1.45**	**16**	**84.2**	✓

Table 3: ImageNet performance comparison. Notation: A2S: ANN-to-SNN conversion; Model-L-D: models with L blocks and D channels. Input resolution: 224×224. [*]Identical training configurations.

Methods	Type	Architecture	Param (M)	Power (mJ)	Time Step	Top-1 Acc (%)	Membrane Shortcut
ViT [42]	ANN	ViT-L/16	304.3	80.96	1	79.70	—
DeiT [43]	ANN	DeiT-B	86.6	80.50	1	81.80	—
PVT [44]	ANN	PVT-Large	61.4	45.08	1	81.70	—
MST [9]	A2S	Swin Transformer-T	28.50	—	512	78.51	✗
Spikformer [8]	SNN	Spikformer-8-384	16.81	7.73	4	70.24	✗
		Spikformer-8-768	66.34	21.48	4	74.81	✗
S-Transformer [41]	SNN	S-Transformer-8-384	16.81	3.90	4	72.28	✓
		S-Transformer-8-768	66.34	6.10	4	76.30	✓
Meta-Spikformer [45]	SNN	—	31.3	7.80	1	75.4	✓
		—	31.3	32.80	4	77.2	✓
SWformer [19]	SNN	SWformer-8-512	27.6	5.08	4	75.43	✓
QKFormer [24]	SNN	HST-10-384	16.47	15.13	4	78.80	✗
		HST-10-768	64.96	8.52	1	81.69	✗
MS-QKFormer[*]	SNN	HST-10-384	16.47	5.52	4	76.48	✓
		HST-10-768	64.96	6.79	1	77.78	✓
Max-Former[*]	SNN	Max-10-384	16.23	**4.89**	4	**77.82**	✓
		Max-10-512	28.65	**2.50**	1	**75.47**	✓
		Max-10-512	28.65	**7.49**	4	**79.86**	✓
		Max-10-768	63.99	**5.27**	1	**78.60**	✓
		Max-10-768	63.99	**14.87**	4	**82.39**	✓

4.2 Results on ImageNet Classification

Table 3 shows Max-Former's performance on ImageNet classification, demonstrating its effectiveness for complex visual tasks. Max-Former-10-768 (T=4) achieves 82.39% accuracy (+7.58% over Spikformer) with 30% lower energy (14.87mJ vs 21.48mJ), despite using only lightweight for early-stage token mixing. It also outperforms the ANN-to-SNN MST model (78.51%) that requires 512 timesteps. Training/inference speed and memory usage are analyzed in **Appendix C**.

Our analysis focuses on models using the Membrane Shortcut, which eliminates the energy-inefficient ternary spike transmissions ($\{0, 1, 2\}$ in Pre-Spike Shortcut) while maintaining full compatibility with standard neuromorphic hardware without additional hardware overhead (see **Appendix D**). For fair comparison, we implemented a membrane potential variant of QKFormer (denoted MS-QKFormer), where MS-QKFormer shows 64% lower energy (5.52mJ vs 15.13mJ) in HST-10-384.

Max-Former-10-384 (16.23M, T=4) achieves 77.82% accuracy, outperforming MS-QKFormer-10-384 (16.47M, 76.48%), S-Transformer-8-768 (66.34M, 76.3%), and Meta-Spikformer (31.3M, 77.2%). For energy efficiency under identical settings, Max-Former-10-384 consumes 4.89mJ, significantly lower than MS-QKFormer-10-384(5.52mJ), S-Transformer(6.10mJ), and Meta-Spikformer (32.8mJ). Compared to conventional ANN models, Max-Former demonstrates concrete advantages in energy efficiency while maintaining competitive accuracy. Specifically, when compared to PVT-Large (a representative hierarchical ANN), Max-Former-10-768 (T=4) achieves comparable accuracy (82.34% vs. 81.70%) with 67% lower energy consumption (14.87mJ vs. 45.08mJ). These results confirm the importance of high-frequency information in Spiking Transformers: replacing energy-intensive self-attention with lightweight DWC in early stages actually produces better performance.

Table 4: Ablation of Patch-Embedding/ Token-Mixing Strategies on CIFAR100 and CIFAR10-DVS.

CIFAR100			CIFAR10-DVS		
Patch Embed	Token Mix	Acc (%)	Patch Embed	Token Mix	Acc (%)
Orig/Max/Max	**Identity/DWC-3/<u>SSA</u>**	**82.65**	**Max+/Max**	**DWC-3/SSA**	**84.2**
Orig/**Orig/Orig**	Identity/DWC-3/SSA	81.63	**Orig/Orig**	DWC-3/SSA	79.2
Orig/<u>Max/**Orig**</u>	Identity/DWC-3/SSA	81.88	<u>Orig</u>/Max	DWC-3/SSA	81.5
Orig/Max/<u>Max</u>	Identity/**Identity**/SSA	81.28	<u>Max</u>+/Max	**DWC-1**/SSA	81.2
Orig/Max/Max	Identity/<u>**DWC-5**</u>/SSA	82.02	Max+/Max	**DWC-5**/SSA	82.7
Orig/Max/Max	**DWC-7**/<u>**DWC-5**</u>/SSA	82.42	Max+/Max	<u>**DWC-7**</u>/SSA	82.1
Orig/Max/Max	**SSA/SSA**/SSA	82.23	Max+/Max	**SSA**/SSA	83.9
Orig/Orig/Orig	**SSA/SSA**/SSA	81.43	Orig/Orig	**SSA**/SSA	79.8

4.3 Ablation Study

We provide direct evidence for the critical role of high-frequency information in Spiking Transformers through in-depth ablation studies.

(1) **Ablation on Patch Embedding Strategies**: Proper patch embedding strategies help to unlock the performance limit of Spiking Transformers. On CIFAR100, changing Max-Former's patch embedding strategy from the proposed Embed-Orig/Embed-Max/Embed-Max to the default Embed-Orig/Embed-Orig/Embed-Orig configuration reduces performance from 82.65% to 81.63%. Neuromorphic datasets exhibit stronger dependence on high-frequency components. On CIFAR10-DVS, replacing the first stage's Embed-Max+ with Embed-Orig causes a significant performance drop from 84.2% to 81.5%. Incorporating high-frequency information through patch embedding proves effective. When purely using SSA token mixing, optimizing the patch embedding strategy can improve accuracy by +0.8% on Cifar 100 and +4.1% on CIFAR10-DVS, highlighting its critical role in spiking architectures.

(2) **Ablation on Token Mixing Strategies**: Max-Former further restores high-frequency information through the token mixing strategy. Despite SSA taking higher energy/parameter costs, Max-Former achieves better performance by simply replacing early-stage self-attention with lightweight DWC. On Cifar100, this substitution leads to +0.42% performance gain (All SSA: 82.23% vs. Max-Former: 82.65%). On CIFAR10-DVS, Max-Former with hybrid token mixing (DWC-3 + SSA) achieves 84.2% accuracy, outperforming the full-SSA variant (83.9%) by 0.3%. The kernel size selection for high-frequency preservation is also of great significance. Larger kernels (DWC-5/7) degrade performance (-0.63% CIFAR100, -2.1% CIFAR10-DVS) due to excessive feature smoothing. Insufficient filtering

with DWC-1 also underperforms (81.2% vs 84.2% on CIFAR10-DVS). This highlights the necessity of balancing high-frequency and low-frequency components in Spiking Transformers.

Overall, Max-Former's performance gains stem from effective high-frequency information propagation, independent of parameter count, which is evidenced by: (1) Max-pooling patch embedding (Embed-Max/Embed-Max+) consistently outperforms original versions despite similar parameter budgets. (2) Larger DWC kernels (DWC-5/DWC-7) increasing parameters but degrading accuracy (-0.63% on CIFAR100, -2.1% on CIFAR10-DVS vs Max-Former with DWC-3). See **Appendix D** for visualizations and **E** for limitations discussion.

4.4 Generality across Convolutional Architectures

We further extend the effectiveness of Max-Former to convolutional architectures by proposing Max-ResNet. The key modification in our Max-ResNet lies in the inclusion of only two additional max-pooling operations compared to MS-ResNet [46]. The detailed implementation of Max-ResNet is provided in Appendix A.3. Training settings are listed in Appendix A.

High-frequency information is essential for SNNs. As shown in Table 5, Max-ResNet achieves a remarkable performance improvement over the baseline MS-ResNet, despite having identical model sizes. Specifically, with the block configuration [2, 2, 2, 2], Max-ResNet improves the CIFAR-10 accuracy by +2.41% (from 94.4% to 96.81%) and the CIFAR-100 accuracy by +6.48%. Similarly, under the [3, 3, 2] configuration, the accuracy increases by +2.25% and +6.65% on CIFAR-10 and CIFAR-100, respectively.

In short, Max-ResNet-18 sets state-of-the-art benchmarks across convolutional baselines, with only a moderate model size and an extremely straightforward high-frequency restoration strategy. Therefore, preserving high-frequency information is fundamental to effective feature representation in SNNs, regardless of architecture.

Table 5: Comparison of different ResNet architectures on CIFAR-10 and CIFAR-100.

Architecture	Training Method	Block Config.	Params (M)	Time Step	CIFAR-10 Acc. (%)	CIFAR-100 Acc. (%)
KDSNN-ResNet-18 [47]	Knowledge Distillation	[2, 2, 2, 2]	11.22	4	95.72	78.46
MS-ResNet-18 [46]	Direct Training	[2, 2, 2, 2]	11.22	4	94.40	75.06
		[3, 3, 2]	12.50	4	94.92	76.41
MS-ResNet-34 [46]		[2, 2, 2, 2]	21.33	4	94.69	75.34
Max-ResNet-18	Direct Training	[2, 2, 2, 2]	11.22	4	96.81 (+2.41)	81.54 (+6.48)
		[3, 3, 2]	12.50	4	97.17 (+2.25)	83.06 (+6.65)

5 Conclusion

This work challenges the prevailing assumption that binary activation constraints are the primary cause of SNNs' performance gap. Through theoretical analysis and empirical validation, we demonstrate for the first time that spiking neurons inherently function as low-pass filters at the network level, resulting in the rapid attenuation of high-frequency components that critically degrade feature representation. **We demonstrate that high-frequency information is crucial for effective spiking computation**: Max-Former (63.99M parameters) achieves 82.39% top-1 accuracy on ImageNet, outperforming Spikformer (74.81%, 66.34M parameters) by +7.58%, while reducing energy consumption by 30% at a comparable model size; Max-ResNet-18 further achieves state-of-the-art performance among convolutional baselines – 97.17% on CIFAR-10 and 83.06% on CIFAR-100. **Notably, all these improvements are achieved through extremely simple modifications, even slightly reducing the overall model size.** We believe this simple yet effective solution will motivate future research to explore the unique properties of SNNs, beyond the established practices in ANN studies.

6 Acknowledgements

References

[1] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10819–10829, 2022.

[2] Weihao Yu, Chenyang Si, Pan Zhou, Mi Luo, Yichen Zhou, Jiashi Feng, Shuicheng Yan, and Xinchao Wang. Metaformer baselines for vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):896–912, 2023.

[3] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[4] Chunming He, Yuqi Shen, Chengyu Fang, Fengyang Xiao, Longxiang Tang, Yulun Zhang, Wangmeng Zuo, Zhenhua Guo, and Xiu Li. Diffusion models in low-level vision: A survey. *arXiv preprint arXiv:2406.11138*, 2024.

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

[6] Filipp Akopyan, Jun Sawada, Andrew Cassidy, Rodrigo Alvarez-Icaza, John Arthur, Paul Merolla, Nabil Imam, Yutaka Nakamura, Pallab Datta, and Gi-Joon Nam. 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, 2015.

[7] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018.

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

[9] Ziqing Wang, Yuetong Fang, Jiahang Cao, Qiang Zhang, Zhongrui Wang, and Renjing Xu. Masked spiking transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1761–1771, 2023.

[10] Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Advances in Neural Information Processing Systems*, 34:23426–23439, 2021.

[11] Yufei Guo, Yuanpei Chen, Liwen Zhang, Xiaode Liu, Yinglei Wang, Xuhui Huang, and Zhe Ma. Im-loss: information maximization loss for spiking neural networks. *Advances in Neural Information Processing Systems*, 35:156–166, 2022.

[12] Jianhao Ding, Zhaofei Yu, Yonghong Tian, and Tiejun Huang. Optimal ann-snn conversion for fast and accurate inference in deep spiking neural networks. *arXiv preprint arXiv:2105.11654*, 2021.

[13] Yanjing Li, Sheng Xu, Baochang Zhang, Xianbin Cao, Peng Gao, and Guodong Guo. Q-vit: Accurate and fully quantized low-bit vision transformer. *Advances in neural information processing systems*, 35:34451–34463, 2022.

[14] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016.

[15] Edgar D Adrian and Yngve Zotterman. The impulses produced by sensory nerve endings: Part 3. impulses set up by touch and pressure. *The Journal of physiology*, 61(4):465, 1926.

[16] Yuhang Li, Shikuang Deng, Xin Dong, Ruihao Gong, and Shi Gu. A free lunch from ANN: Towards efficient, accurate spiking neural networks calibration. In *International Conference on Machine Learning*, pages 6316–6325. PMLR, 2021.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[18] Jaldert Rombouts and Sander Bohte. Fractionally predictive spiking neurons. *Advances in Neural Information Processing Systems*, 23, 2010.

[19] Yuetong Fang, Ziqing Wang, Lingfeng Zhang, Jiahang Cao, Honglei Chen, and Renjing Xu. Spiking wavelet transformer. In *European Conference on Computer Vision*, pages 19–37. Springer, 2024.

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

[21] Arindam Basu, Lei Deng, Charlotte Frenkel, and Xueyong Zhang. Spiking neural network integrated circuits: A review of trends and future directions. In *2022 IEEE Custom Integrated Circuits Conference (CICC)*, pages 1–8. IEEE, 2022.

[22] Yuetong Fang, Ziqing Wang, Deming Zhou, Hongwei Ren, Shibo Zhou, and Renjing Xu. Dynamic token masking in spiking neural network. 2025.

[23] 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, 2024.

[24] 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. *arXiv preprint arXiv:2403.16552*, 2024.

[25] Zecheng Hao, Xinyu Shi, Yujia Liu, Zhaofei Yu, and Tiejun Huang. Lm-ht snn: Enhancing the performance of snn to ann counterpart through learnable multi-hierarchical threshold model. *arXiv preprint arXiv:2402.00411*, 2024.

[26] Ziqing Wang, Yuetong Fang, Jiahang Cao, Hongwei Ren, and Renjing Xu. Adaptive calibration: A unified conversion framework of spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 1583–1591, 2025.

[27] 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.

[28] Christodoulos Kechris, Jonathan Dan, Jose Miranda, and David Atienza. Dc is all you need: describing relu from a signal processing standpoint. *arXiv preprint arXiv:2407.16556*, 2024.

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[30] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34:12116–12128, 2021.

[31] Chenyang Si, Weihao Yu, Pan Zhou, Yichen Zhou, Xinchao Wang, and Shuicheng Yan. Inception transformer. *Advances in Neural Information Processing Systems*, 35:23495–23509, 2022.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.

[33] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11062–11070, 2021.

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

[35] Yifan Hu, Lei Deng, Yujie Wu, Man Yao, and Guoqi Li. Advancing spiking neural networks toward deep residual learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[36] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[37] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[38] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009.

[39] Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. CIFAR10-DVS: An Event-Stream Dataset for Object Classification. *Frontiers in Neuroscience*, 11, 2017.

[40] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, et al. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7243–7252, 2017.

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

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

[43] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[44] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 568–578, 2021.

[45] 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. In *The Twelfth International Conference on Learning Representations*.

[46] Yifan Hu, Lei Deng, Yujie Wu, Man Yao, and Guoqi Li. Advancing spiking neural networks towards deep residual learning. *arXiv preprint arXiv:2112.08954*, 2021.

[47] Qi Xu, Yaxin Li, Jiangrong Shen, Jian K Liu, Huajin Tang, and Gang Pan. Constructing deep spiking neural networks from artificial neural networks with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7886–7895, 2023.

[48] Wei Fang, Yanqi Chen, Jianhao Ding, Ding Chen, Zhaofei Yu, Huihui Zhou, Timothée Masquelier, Yonghong Tian, and other contributors. Spikingjelly. `https://github.com/fangwei123456/spikingjelly`, 2020. Accessed: 2023-02-21.

[49] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[50] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[51] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[52] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[53] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

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

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

[56] Philipp Weidel and Sadique Sheik. Wavesense: Efficient temporal convolutions with spiking neural networks for keyword spotting. *arXiv preprint arXiv:2111.01456*, 2021.

[57] Ismael Balafrej, Soufiyan Bahadi, Jean Rouat, and Fabien Alibart. Enhancing temporal learning in recurrent spiking networks for neuromorphic applications. *Neuromorphic Computing and Engineering*, 2025.

[58] Ashish Gautam and Takashi Kohno. Adaptive stdp-based on-chip spike pattern detection. *Frontiers in Neuroscience*, 17:1203956, 2023.

[59] Shikuang Deng, Yuhang Wu, Kangrui Du, and Shi Gu. Spiking token mixer: An event-driven friendly former structure for spiking neural networks. *Advances in Neural Information Processing Systems*, 37:128825–128846, 2024.

[60] Svea Marie Meyer, Philipp Weidel, Plank Philipp, Leobardo Campos-Macias, Sumit Bam Shrestha, Philipp Stratmann, Jonathan Timcheck, and Mathis Richter. A diagonal state space model on loihi 2 for efficient streaming sequence processing. In *NeurIPS 2024 Workshop Machine Learning with new Compute Paradigms*.

[61] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.

[62] Zhaokun Zhou, Kaiwei Che, Wei Fang, Keyu Tian, Yuesheng Zhu, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer v2: Join the high accuracy club on imagenet with an snn ticket. *arXiv preprint arXiv:2401.02020*, 2024.

[63] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar, et al. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2020.

A Analysis of High-Frequency Information in Spiking Transformers

To validate our approach, we conduct experiments across three static and two neuromorphic datasets. In this section, we first provide the detailed experimental setup used to obtain the results presented in our paper. We then perform additional analysis on the importance of high-frequency information in Spiking Transformers. For complete parameter configurations, please refer to our public code repository at https://github.com/bic-L/MaxFormer.

Table 6: Hyperparameters for image classification on different datasets.

Hyper parameters	ImageNet	CIFAR-10	CIFAR-100	Neuromorphic
Model Size	10–384 / 10–512 / 10–768	4–384	4–384	2–256
Epochs	200	400	400	106
Resolution	224×224	32×32	32×32	128×128
Batch Size	$512 \ (8 \times 64)$	128	64	16
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning rate	$1.2 \times 10^{-3} \ (T = 1)$ $1.35 \times 10^{-3} \ (T = 4)$	1.50×10^{-3}	1.50×10^{-3}	6.00×10^{-3}
Learning rate decay	Cosine	Cosine	Cosine	Cosine
Warmup epochs	5	20	20	10
Weight decay	0.05	0.06	0.06	0.06
Rand Augment	9 / 0.5	9-n1 / 0.4	9-n1 / 0.4	—
Mixup	0.25 / 0.4/ 0.8	0.5	0.75	0.5
CutMix	1	0.5	0.5	—
Mixup prob	0.5	1	1	0.5
Erasing prob	0.0	0.25	0.25	—
Label smoothing	0.1	0.1	0.1	0.1

A.1 Experimental Details

Datasets: We evaluate Max-Former through comprehensive experiments on static datasets (CIFAR-10 [36], CIFAR-100 [37] and ImageNet [38]) and neuromorphic datasets (CIFAR10-DVS [39], DVS128 Gesture [40]). The training and inference pipeline are implemented in SpikingJelly [48].

Static Datasets: For static image classification, we evaluate on three standard benchmarks. ImageNet-1k [38] is one of the most widely used datasets in computer vision. It contains 1.28 million training, 50,000 validation, and 100,000 test images covering the common 1K classes. Both CIFAR-10 [36] and CIFAR-100 [37] include 50,000 training images and 10,000 testing images with 32×32 resolution. The main difference between them is that CIFAR-10 has 10 categories for classification, while CIFAR-100 has 100 categories.

Neuromorphic Datasets: For event-based vision tasks, we evaluate on two standard benchmarks. CIFAR10-DVS [39] is an event-based version of the CIFAR-10 dataset, created by capturing moving image samples using the Dynamic Vision Sensor (DVS). It includes 10,000 event-based images (128×128 pixels) spread across 10 classes, with 9,000 samples for training and 1,000 for testing. The DVS128 Gesture dataset [40] contains 1,342 event-based recordings of 11 different hand gesture types performed by 29 people under 3 different lighting conditions. Each gesture recording lasts about 6 seconds on average.

Hyper Parameters: Our training scheme mainly follows [24] and [41]. Specifically, MixUp [49], CutMix [50] and RandAugment [51] are used for data augmentation. The models are trained using AdamW optimizer [52] with the weight decay of 0.05 for ImageNet-1K classification tasks and the weight decay of 0.06 for all other datasets. Label Smoothing [53] is set as 0.1. Detailed training

hyperparameters are shown in Table 6. For our ImageNet experiments, we used 8 NVIDIA A30 GPUs to train most models. However, for the MaxFormer-10-512 (T=4) and MaxFormer-10-768 (T=4) models, we used 8 NVIDIA H20 GPUs instead. For the smaller datasets (CIFAR10, CIFAR100, DVS128 Gesture, and CIFAR10-DVS), we used a single A30 GPU for training.

A.2 Impact of High-Frequency Information on Model Performance

Table 7: Patch embedding and token mixing schemes on CIFAR-100. Acc.:Top-1 Accuracy (%). DWC-N: spiking depth-wise convolution with kernel size $N \times N$. SSA: Spiking Self-Attention.

	Patch Embed	**Token Mix**	**Acc. (%)**
(1)	Embed-Orig/Embed-Orig/Embed-Orig	Avg-Pool/Avg-Pool/Avg-Pool Max-Pool/Max-Pool/Max-Pool	76.73 79.12
(2)	Embed-Orig/Embed-Orig/**Embed-Max** Embed-Orig/**Embed-Max/Embed-Max**	Identity/Identity/Identity Identity/Identity/Identity	80.11 80.46
(3)	Embed-Orig/**Embed-Max/Embed-Max**	Avg-Pool/Avg-Pool/Avg-Pool Max-Pool/Max-Pool/Max-Pool Identity/Max-Pool/Identity Identity/Identity/Max-Pool	77.61 79.78 79.99 80.12
(4)	Embed-Orig/**Embed-Max/Embed-Max**	SSA/SSA/SSA DWC-3/DWC-5/SSA DWC-5/DWC-3/SSA DWC-5/DWC-7/SSA DWC-7/DWC-5/SSA DWC-3/DWC-3/SSA Identity/DWC-3/SSA	82.13 82.36 82.46 82.45 82.42 82.59 82.65
(5)	Embed-Orig/**Embed-Max/Embed-Max** Embed-Orig/Embed-Orig/Embed-Orig Embed-Orig/Embed-Orig/Embed-Orig	SSA+DWC-5/SSA+DWC-5/SSA+DWC-5 SSA+DWC-3/SSA+DWC-3/SSA+DWC-3 SSA+DWC-5/SSA+DWC-5/SSA+DWC-5	82.09 82.56 82.73
(6)	Embed-Orig/Embed-Max/Embed-Max Embed-1Max/Embed-Max/Embed-Max Embed-Max/Embed-Max/Embed-Max	SSA/SSA/SSA SSA/SSA/SSA SSA/SSA/SSA	82.13 79.98 78.78
(7)	Embed-Orig/Embed-1Max/Embed-1Max Embed-1Max/Embed-1Max/Embed-1Max	SSA/SSA/SSA SSA/SSA/SSA	82.02 81.86
(8)	Embed-Orig/Embed-Max/Embed-Max Embed-Orig/Embed-1Max/Embed-1Max	SDSA/SDSA/SDSA SDSA/SDSA/SDSA	81.77 81.52

In Table 7, we provide additional analysis on the impact of high-frequency information. We conducted these experiments on CIFAR-100 [37], using the experimental settings detailed in Table 6. We discuss the ablation below according to the following aspects:

(a) Extra Max-Pool in Patch Embedding/Token Mixing:

High-frequency information plays a critical role in the performance of Spiking Transformer due to the inherent low-pass filter characteristics of spiking neurons. Experimental results shown in Table 7 (1) reveal that strategically preserving these frequencies through max-pooling operations significantly enhances model accuracy, with a 2.39% improvement when replacing average pooling with max-pooling across all stages (76.73% to 79.12%). In Table 7 (2), the performance of Spiking Transformer increases progressively when extending Embed-Max from the last patch embedding blocks (80.11%) to include the middle block (80.46%).

However, excess high-frequency information will instead impair model performance. For instance, in Table 7 (3), switching from using avg-pool for all token mixing to max-pool improves top-1 accuracy from 77.61% to 79.78%. Yet we found even better results with a more targeted setting: when employing max-pooling exclusively in the middle stage increases accuracy to 79.99%, while restricting it to only the last stage further pushes performance to 80.12%.

This happens because spiking neurons act like low-pass filters that naturally reduce high-frequency components as information moves deeper through the network. Therefore, strategically adding back

high-frequency components at specific points in the network is crucial for pushing the performance limit of Spiking Transformers.

In general, **for the hierarchical spiking transformer, Embed-Max can improve performance when used for patch embedding**, as shown in Table 7 (6-8). However, for non-hierarchical ones, although enhancing high-frequency components can in principle be beneficial, its actual impact is limited since patch embedding is only performed once throughout the entire process, causing the improvement to diminish in deeper layers. A more promising solution lies in optimizing token mixing, as discussed in [19].

(b) Spiking Transformers Benefit from High-Frequency Information:

In biological vision, high-frequency details help early processing stages learn elementary features,, which are then gradually built from local to global representations. Similarly, in standard non-spiking Transformers, the lower layers typically need more high-frequency details, while higher layers work better with global information. Spiking Transformers follows the same design philosophy, but with an important difference: they need additional frequency-enhancing operations (e.g., max-pooling and depth-wise convolution) for restoring high-frequency information that would otherwise be lost.

In Table 7 (4), we show that, a proper token mixing strategy can effectively restore high-frequency information in Spiking Transformer, resulting in significant performance gains. By replacing Spiking Self-Attention (SSA) with depth-wise convolution (DWC) for token mixing, we improved performance from 82.13% to 82.65%. Importantly, this improvement does not come from adding more parameters/ computational burden. For example, the Identity/DWC-3/SSA combination works 0.29% better than DWC-3/DWC-5/SSA, even though the former one has a lower computational cost. Our further experiments in Table 7 (4-5) confirm that these findings hold true in the full-SSA network: restoring high-frequency components can significantly optimize the performance from 82.13% to 82.73% (+0.6%) on Cifar100. The proper high-frequency enhancement strategy is essential to unlocking the full potential of Spiking Transformers.

A.3 Max-ResNet Implementation

As shown in Figure 7, Max-ResNet introduces only a minor architectural change to MS-ResNet [46]: all layers are replaced with Max-ResNet layers, while the first layer remains unchanged. Code implementation is available at https://github.com/bic-L/MaxFormer.
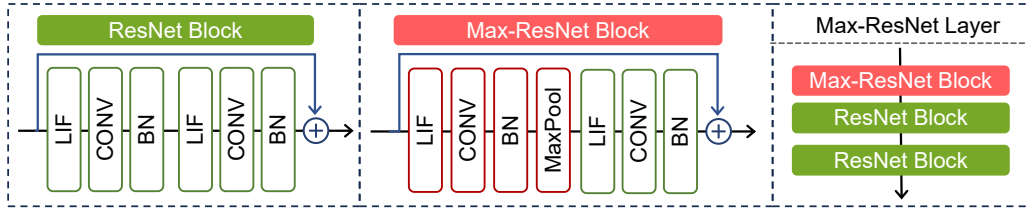

Figure 7: Overview of Max-ResNet. A single Max-Pool operation is added per block and layer.

B Energy Analysis

To estimate the theoretical energy consumption of Spiking Transformers, we follow the methodology used in previous studies [8, 54, 41, 19]. It is worth noting that the batch normalization (BN) layers and linear scaling transformations that follow convolution layers can be combined directly into the convolution layers themselves with added bias terms during deployment. Thus, in common practice [8, 54, 41, 19], the energy consumption of BN is typically excluded when calculating theoretical energy usage. For fair comparison, our work adopts the same strategy. In Spiking Transformers, energy consumption is directly proportional to synaptic operations (SOPs), which can be calculated as:

$$\text{SOPs}(l) = fr \times \text{T} \times \text{FLOPs}(l) \tag{22}$$

where l represents a specific block or layer in the Spiking Transformer architecture, fr refers to the firing rate of the input spike train for that particular block or layer, and T is the simulation time step of spiking neurons. Assuming the multiply-accumulate (MAC) and accumulate (AC) operations

are implemented on the 45nm neuromorphic chip described in [7], where each MAC operation uses $E_{MAC} = 4.6$pJ of energy and each AC operation uses $E_{AC} = 0.9$pJ, we can estimate the total energy consumption of a Spiking Transformer by adding up the energy used by all MAC and AC operations across all layers:

$$E_{SNN} = E_{MAC} \times FLOP^1_{CONV} + E_{AC} \times (\sum_{n=2}^{N} SOP^n_{SNN\ Conv} + \sum_{j=1}^{M} SOP^j_{SNN\ FC}) \quad (23)$$

$FLOP^1_{CONV}$ represents the floating-point operations in the first layer, which converts non-spike inputs into spike form for static image classification tasks. Since this layer performs floating-point computations, we estimate its energy consumption using E_{MAC}. For all subsequent layers, which process spike data, we estimate energy consumption using E_{AC}. For mainstream non-spiking Transformers, the energy consumption is estimated through:

$$E_{ANN} = E_{MAC} \times FLOPs \quad (24)$$

C   Comparison on Train/Inference Time and Memory Consumption

Table 8: Comparison of training/inference time and memory usage between QKFormer-10-768 and Max-Former-10-768 models. All measurements were conducted with the simulation timestep of 1 and the batch size of 32. MS-QKFormer indicates the QKFormer variant with the membrane shortcut.

	Train Time (s)	Train Memory (MB)	Infer. Time (s)	Infer. Memory (MB)
QKFormer (64.96M, T=1, B=32)	0.214	18227	0.053	5000
MS-QKFormer* (64.96M, T=1, B=32)	0.208	17496	0.048	4822
Max-Former* (63.99M, T=1, B=32)	0.179	15431	0.044	4354

Max-Former delivers faster training and inference speeds while consuming less memory. We compared the performance metrics of QKFormer [24], its membrane shortcut variant (MS-QKFormer), and Max-Former on ImageNet using 224×224 input resolution. All tests were conducted on a CentOS 7.9 server equipped with the Intel Xeon Gold 6348 CPU (2.60GHz) and the Nvidia A30 GPU. As Table 8 shows, when compared to MS-QKFormer with the same hierarchical architecture and shortcut configuration, Max-Former reduces training time by 14% and both inference time and memory usage by 10%. Additionally, our results indicate that the pre-spike shortcut strategy used in the original QKFormer increases both processing time and memory demands.

D   Residual Connections in Spiking Neural Networks

(a) Performance and Energy Tradeoffs:

The unique asynchronous nature of spike-based computation makes implementing residual connections challenging. As a result, the research community of spiking neural networks (SNNs) has not yet reached a consensus on the standardized residual learning approach, either in terms of algorithm or hardware implementation. While the main focus of our work is not related to residual learning, we want to offer a detailed comparison between the pre-spike shortcut [55] and the membrane shortcut [54], the two most representative residual learning methods that emerged in recent years.

As explained in Section **3.3**, the pre-spike shortcut [55] implements residual connections between spiking outputs, while the membrane shortcut [54] connects membrane potentials directly. In algorithm designs, membrane shortcuts have been reported to facilitate better performance in prior works [24, 41], especially on small datasets. However, our findings indicate this advantage is not universal across all scenarios. As shown in Table 9, the patch embedding stages of Spiking Transformers account for the majority of energy consumption. Consequently, the multiple patch embedding stages in hierarchical architectures like QKFormer [24], while enabling efficient feature learning with fewer parameters, also come at higher energy usage. This makes the choice of shortcut scheme particularly impactful on overall energy efficiency. When processing ImageNet images at 224×224 resolution, the pre-spike QKFormer consumes three times more energy than

its membrane shortcut variant, reflecting the substantially higher SOPs required. Nevertheless, this increased computational overhead does translate to notable performance gains (+2.32% accuracy when comparing QKFormer to MS-QKFormer).

From the hardware perspective, implementing either shortcut type on neuromorphic chips is technically feasible but presents significant challenges [56, 57]. Implementing the pre-spike shortcut specifically requires the chips to support multi-spike operations, as the ternary spike transmissions (values of 0, 1, or 2) can occur. This results in either higher energy consumption or increased hardware complexity [58]. Yao et al. [41] proposed implementing membrane shortcuts through the addressing function that passes the membrane potential to corresponding neurons in subsequent layers for merging. While membrane shortcuts do strictly adhere to the spike-driven computing paradigm and could possibly be supported by standard neuromorphic hardware, transmitting membrane potentials can create substantial communication overhead, making practical implementation non-trivial. Many discussions on hardware deployment still advocate avoiding shortcuts as the current preferred approach [59, 60]. However, given the critical role of residual learning in modern deep learning, avoiding shortcuts altogether is not a sustainable long-term strategy for advancing neuromorphic computing.

In our work, we primarily compare with MS-QKFormer using the membrane shortcut to ensure fair comparisons. We hope the SNN community can establish a consensus on standardized shortcut implementations in the near future, considering the significant impact of shortcut schemes on both energy efficiency and model performance.

Table 9: Energy consumption comparison (in mJ) across processing stages for QKFormer, QKFormer with membrane shortcut (MS-QKFormer), and our proposed Max-Former model. * Identical training configurations.

Model	Stage 1			Stage 2			Stage 3			Classifier	Total Energy
	Patch Embed	Token Mix	MLP	Patch Embed	Token Mix	MLP	Patch Embed	Token Mix	MLP		
QKFormer (16.47M, T=4, 78.80%)	1.26	0.16	0.41	2.68	0.35	0.77	2.97	2.71	3.81	0.006	15.13
MS-QKFormer* (16.47M, T=4, 76.48%)	1.19	0.052	0.15	0.96	0.11	0.25	0.88	0.88	1.06	0.007	5.52
Max-Former* (16.23M, T=4, 77.82%)	0.41	0.02	0.17	0.89	0.01	0.36	0.91	0.96	1.16	0.001	4.89

E Visualization

We demonstrate the GradCAM visualizations [61] of four Spiking Transformers with the membrane shortcut of similar size. Compared to Spike-Driven Transformer [41] and SWFormer [19], the hierarchical architecture used in both QKFormer [24] and our Max-Former allows for more precise focusing on target objects. Compared to MS-QKFormer, Max-Former shows more concentrated activation patterns. For instance, in the polar bear image, Max-Former completely skips the background and precisely focuses on the bear's key features (the head, rather than the outline or fur).

F Impact and Limitation

Our work serves as the theoretical foundation for many existing architectural choices in Spiking Transformers. Specifically, in [41], the authors found that directly applying known practices of MetaFormer does not achieve good results in Spiking Transformers: employing average pooling operators to replace SDSA as the token mixer surprisingly results in substantial performance degradation from 61.0% to 41.2%. Similar phenomena have been discussed in earlier works. For instance, Spikformer v2 [62] discovered that removing the max-pooling operator in the original Spikformer [8] leads to a substantial performance drop, while adding convolution layers (which act as high-pass filters) in the patch embedding stage significantly improves the performance. Our work reveals the underlying principles of these architectural designs: spiking transformers need to enhance high-frequency components to alleviate the feature degradation caused by their inherent low-pass activation.

We are well aware that there is still much space to be explored, and we hope that our Max-Former can serve as a good starting point for future research. For instance, similar to [31], Max-Former requires manually balancing frequency components, which demands considerable expertise when adapting to different tasks. Incorporating direct frequency learning approaches like Fourier-based [63]
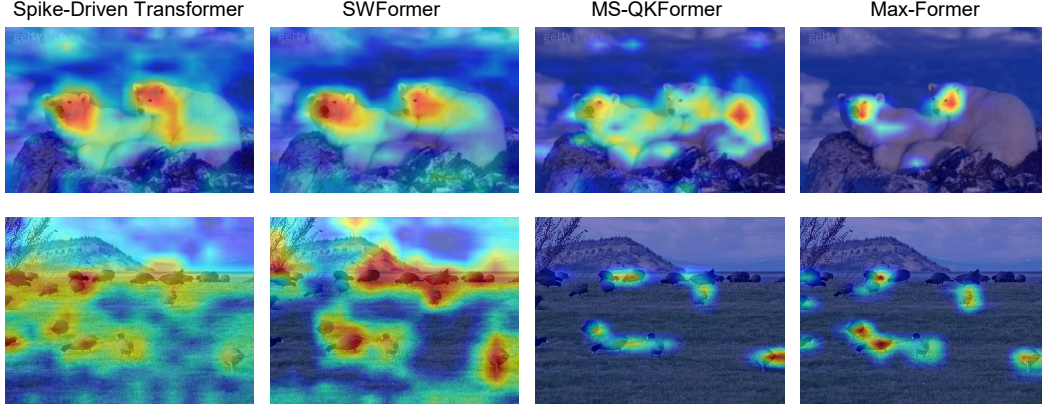
Figure 8: GradCAM visualizations [61] comparing four Spiking Transformers of similar size: Spike-Driven Transformer-8-512 [41] (29.68M), SWFormer-8-512 [19] (27.6M), QKFormer-10-512 with membrane shortcut (MS-QKFormer) (29.08M), and our Max-Former-10-512 (28.65M).

or Wavelet-based[19] methods would offer more straightforward solutions. The main challenge, however, lies in developing efficient spike-based frequency representations. Overall, we believe our work will inspire more future research to advance neuromorphic computing through exploring the unique properties of spiking neurons, rather than expending excessive effort to adapt established practices from standard non-spiking neural networks.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction reflect the paper's contributions and scope clearly.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in **Appendix E**.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All the theorems, formulas, and proofs are clearly stated in the main paper.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide experiment details in the **Appendix A, B**.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is available: https://github.com/bic-L/MaxFormer.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All training and test details have been revealed in the **Appendix A**.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not reported because it would be too computationally expensive.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See **Appendix A**.

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: This research is in every respect with the NeurIPS Code of Ethics.

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: This research belongs to foundational research and is not tied to particular applications.

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper properly credits creators and mentions the license and terms of use for existing assets.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not release new assets.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing or research with human subjects.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing or research with human subjects.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The paper does not involve LLMs as any important, original, or non-standard components.