*2025 International Conference on Intelligent Computing*
*July 26-29, Ningbo, China*
https://www.ic-icc.cn/2025/index.php

# PBSpikformer: A Pure Spike-Driven Spiking Neural Network with Fourier-Phase Attention and Dynamic Batch Context

Chengfan Yang[1[0009-0008-5478-0299]], Tao Deng[1(✉)[0000-0001-5094-5879]], Ran Li[1[0000-0002-0682-5578]] and Fei Yan[1[0000-0001-5177-4493]]

[1] School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China
tdeng@swjtu.edu.cn

**Abstract.** Spiking Neural Networks (SNNs), inspired by biological neurons, have gained increasing attention for their energy efficiency and event-driven computation. However, their binary nature and complex dynamics make it difficult to train high-performance, low-latency models, limiting their progress compared to Artificial Neural Networks (ANNs). To address these challenges, we propose PBSpikformer, a directly trainable spiking Transformer architecture that incorporates two novel components: Fourier-Phase Attention (FPA) and Dynamic Batch Context (DBC). FPA combines spike-based Q-K token attention with Spectral Cross-Modal Augmentation (SCMA) to effectively fuse spatial, temporal, and frequency-domain features while reducing computational complexity. DBC introduces batch-level global signals to modulate local and global activations, improving gradient flow and training robustness. Extensive experiments show that PBSpikformer outperforms existing SNN models across multiple benchmarks, achieving 96.7% accuracy on CIFAR10-DVS—a 12.7% improvement over previous methods—and becomes the first directly trained SNN to surpass 90% accuracy on this dataset.

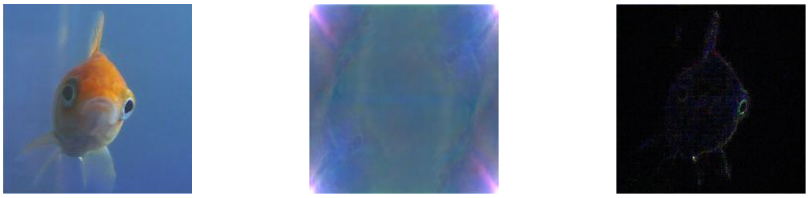
**Keywords:** Spiking Neural Network, Fourier Transform, Attention Mechanism.

## 1 Introduction

Spiking Neural Networks (SNNs), inspired by biological neurons, enable energy-efficient, event-driven computation through spike-based signaling. Recognized as the third generation of neural networks [1], SNNs exhibit rich temporal dynamics and map naturally onto neuromorphic hardware [2, 3]. However, the binary and non-differentiable nature of spikes presents challenges in training deep SNNs. Recent works have addressed this via surrogate gradients [4, 5] and residual learning [6].

Meanwhile, Transformer models [7] have achieved impressive results in vision tasks [8-10]. Combining Transformers with SNNs—known as Spiking Transformers [3, 11, 12]—has emerged as a promising direction for building high-performance, low-power

models. In vision applications, Fourier phase information captures critical semantic features like structure and contours, whereas amplitude reflects low-level textures [13-15].



(a) (b) (c)

**Fig. 1.** Illustration of phase and amplitude contributions to image reconstruction: (a) Original image; (b) Reconstruction using amplitude; (c) Reconstruction using phase.

As shown in Fig. 1, phase preserves object shape even when amplitude is discarded, highlighting its importance for robust representation.

In this paper, we propose PBSpikformer, a hierarchical spiking Transformer incorporating two key innovations: Fourier-Phase Attention (FPA) and Dynamic Batch Context (DBC). Our principal contributions are listed as below:

1. We propose FPA, a mechanism that extracts phase information from the Fourier spectrum to capture high-level semantic features like image structures and contours, while reducing computational complexity compared to traditional attention mechanisms.
2. We introduce DBC, a biologically inspired technique that dynamically adjusts local and global activation levels within a batch, improving gradient propagation, training robustness, and generalization in deep SNNs.
3. We develop PBSpikformer, an energy-efficient hierarchical spiking Transformer integrating FPA and DBC, supporting end-to-end training and enabling deep exploration of hierarchical representations in Transformer-based SNNs.
4. Extensive trials confirm that PBSpikformer pushes the performance frontier, with 96.7% accuracy on CIFAR10-DVS, surpassing the previous SOTA by 12.7%, and delivering competitive results on CIFAR10, CIFAR100, and DVS128 Gesture.

## 2 Related Work

### 2.1 Approaches to Training SNNs

SNN-learing techniques are divided into two main broad camps. The first thing is conversion of ANN-to-SNN [16, 17], where spiking neurons replace ReLU in pre-trained ANNs. While effective for reusing ANN structures, this method suffers from high latency and strong dependence on the original architecture [18]. The second is direct training using surrogate gradients [4-6], which enables end-to-end optimization despite spike non-differentiability. This approach provides better flexibility, lower latency, and broader architectural adaptability, making it the mainstream choice in recent research.

### 2.2 Applications of Phase Information in Fourier Transform

Extensive studies [13-15] show that phase information in the Fourier spectrum encodes key semantic features, such as image structure and contours, while amplitude mainly reflects low-level statistics like color and texture. Building on this, [19] enhanced image classification by manipulating phase in the frequency domain and reconstructing enriched features via inverse Fourier transform. Further, [20] emphasized that phase remains consistent across domains, while amplitude varies, proposing amplitude mix for robust data augmentation. Inspired by these insights, we designed a Fourier-based attention mechanism that prioritizes phase to improve semantic representation.

## 3 Method

### 3.1 Overall Architecture

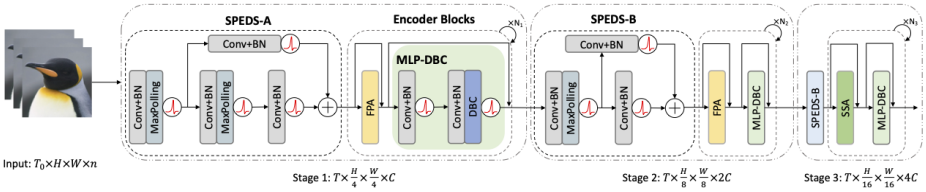



**Fig. 2.** Overview of PBSpikformer, a hierarchical spiking transformer architecture featuring Fourier-Phase Attention (FPA) and MLP with Dynamic Batch Context (DBC).

As shown in Fig. 2, PBSpikformer is a hierarchical spiking transformer architecture that integrates Fourier-Phase Attention (FPA) and MLPs with Dynamic Batch Context (DBC). The input tensor has shape $T_0 \times H \times W \times n$, where $T_0, H, W, n$ denote the time steps, height, width, and channels, respectively. For static images, $T_0 = 1, n = 3$; for neuromorphic data, $T_0 = T, n = 2$. The input is first split into $4 \times 4$ patches and embedded into a spiking feature space of dimension $C$ using the spiking patch embedding with deformable shortcut A (SPEDS-A), forming Stage 1 with spatial resolution reduced to $\frac{H}{4} \times \frac{W}{4}$. Stage 2 further downsamples by a factor of 2 using SPEDS-B, reducing resolution to $\frac{H}{8} \times \frac{W}{8}$ and doubling channels to $2C$, with FPA and MLP-DBC applied. Stage 3 again reduces the resolution to $\frac{H}{16} \times \frac{W}{16}$ and increases channels to $4C$, incorporating Spiking Self-Attention (SSA) and MLP-DBC. The numbers of PBSpikformer blocks in each stage are defined as $N_1, N_2, and N_3$, respectively. This hierarchical structure enables efficient multiscale spiking representation learning for both static and dynamic inputs.

### 3.2 Fourier-Phase Attention

The design of FPA is illustrated in Fig. 3a, consisting of two parts: Q-K Token Attention and Spectral Cross-Modal Augmentation. Unlike Vanilla Self-Attention (VSA) [8]

and Spiking Self-Attention (SSA) [11], which rely on full QKV triplets, FPA simplifies the computation by using only spike-based Q and K, followed by phase enhancement. As shown in Table 1, FPA achieves lower time and space complexity, making it highly suitable for event-driven architectures.

**Table 1.** Computational Complexity Comparison of Attention Methods

| Methods | Time Complexity | Space Complexity | Characteristics |
|---|---|---|---|
| VSA | $O(N^2D)$ | $O(N^2)$ | Dense, Softmax-based |
| SSA | $O(ND^2)$ or $O(ND^2)$ | $O(ND)$ | Sparse, Spike-based |
| FPA (Ours) | $O(TND)$ | $O(ND)$ | Sparse, Spike + FFT-based |

**Q-K Token Attention.** For clarity, we assume a single time step T = 1 and use single-head attention. The input spike tensor $S$ is converted into query ($Q$) and key ($K$) representations through convolution, batch normalization, and spiking activation:

$$Q = \mathrm{SN}_Q\left(\mathrm{BN}\big(\mathrm{Conv}(S)\big)\right), \quad K = \mathrm{SN}_K\left(\mathrm{BN}\big(\mathrm{Conv}(S)\big)\right) \tag{1}$$

Given the spike-form matrices $Q, K \in \mathbb{R}^{N\times D}$, we compute a token-wise attention vector $A_t \in \mathbb{R}^{N\times 1}$ as:

$$A_t = \mathrm{SN}\big(\textstyle\sum_{i=0}^{D} Q_{i,j}\big), \quad X' = \mathrm{BN}\big(\mathrm{Conv}(A_t \otimes K)\big) \tag{2}$$

Here, $A_t$ represents the binarized importance of each token, obtained by row-wise summation over $Q$ followed by a spiking neuron layer. The Hadamard product $\otimes$ applies token-wise masking on $K$, allowing attention to selectively emphasize informative spikes. The result is then refined via convolution and normalization to produce $X'$.

Since spike-based attention may miss certain global structures, we later fuse Fourier phase information to recover complementary frequency-domain semantics.

**Spectral Cross-Modal Augmentation.** We apply the one-dimensional Discrete Fourier Transform (1D DFT) along the temporal dimension to transform the spike-based spatial features into spectral features. Since the 1D DFT is efficiently computed using the implementation provided by Python's NumPy package, we omit the temporal dimension from Fig. [2] to highlight other essential operations. Specifically, given an input temporal sequence $x[i]$ of length $n$, the spectral representation is computed as follows:

$$X[k] = \mathcal{F}_{\text{time}}(x[i]) = \textstyle\sum_{i=0}^{n-1} x\,[i] e^{-j\frac{2\pi k i}{n}}, \quad k = 0,1,\dots,n-1 \tag{3}$$

where $X[k] \in \mathbb{C}^d$ denotes the complex-valued spectral tensor at frequency index $k$, $j$ is the imaginary unit, and $\mathcal{F}_{\text{time}}(\cdot)$ represents the 1D DFT applied along the temporal axis.

After obtaining the frequency-domain representation $X[k]$, we separate each frequency component into its real and imaginary parts:

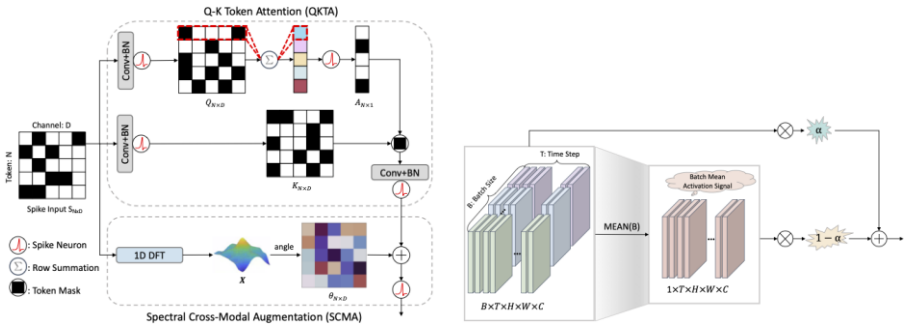
$$X[k] = \mathrm{Re}(X[k]) + j\mathrm{Im}(X[k]) \tag{4}$$

where $\mathrm{Re}(X[k])$ and $\mathrm{Im}(X[k])$ are the real and imaginary components of $X[k]$, respectively. The phase information $\theta[k]$ of each complex frequency component is subsequently extracted as the argument of the complex spectral number, defined by:

$$\theta[k] = \mathrm{angle}(X[k]) = \mathrm{atan2}\big(\mathrm{Im}(X[k]),\mathrm{Re}(X[k])\big) \tag{5}$$

where (X[k]) denotes the angle extraction operation, which in practice is implemented directly using Python's NumPy function np.angle() for numerical stability and efficiency, and it internally utilizes the atan2 function to accurately determine the phase angle in the range $[-\pi, \pi]$.

Finally, we integrate this frequency-domain phase information $\theta[k]$ into the spatial-domain features obtained from the QKTA module, denoted as $X'$. This fusion enhances multimodal representation, increases feature diversity, and improves the semantic expressiveness of the model. The resulting features pass through a spiking neuron layer, preserving the spike-based processing, sustaining biological interpretability, and lowering energy consumption.




(a) (b)

**Fig. 3.** Illustration of PBSpikformer components: (a) QKTA and SCMA modules in FPA; (b) Batch-level averaging in DBC.

### 3.3 Dynamic Batch Context

The activation of neurons is influenced by both local input and global brain states, such as attention [21, 22]. To simulate this biological mechanism, we propose DBC module, whose operational flow is illustrated in Fig. [fig:DBC]. Furthermore, inspired by the Spike-Element-Wise (SEW) addition strategy in residual spiking networks [6], DBC introduces a batch-level global signal to balance local and global activation levels before processing by spiking neurons, a process known as Activation-Pre-Batch-Context (APBC). Introducing APBC effectively improves gradient flow, stability, and feature representation in deep SNNs. In summary, DBC serves as an activation modulation mechanism designed to enhance the learning dynamics and robustness of deep SNNs

by regulating sample activation via batch-level global signals, thus improving SNN performance in a biologically plausible and computationally efficient manner.

The core idea of DBC is that the activation of each sample in a batch should refer to the global activation pattern of the entire batch. To this end, DBC introduces a batch mean activation signal $\mu$, calculated as:

$$\mu = \frac{1}{B}\sum_{i=1}^{B} x_i \tag{6}$$

where $x_i$ denotes the activation the $i$-th sample, and $B$ represents the batch size. Then, the activation $x_i$ of each sample is modulated based on the global signal $\mu$:

$$x' = \alpha x_i + (1-\alpha)\mu \tag{7}$$

where $\alpha \in [0,1]$ is the balance factor that controls the weighting of local activation and the global signal. Experimental results show that manually setting $\alpha$ (e.g., $\alpha = 0.5$) yields better performance, and the modulated signal $x'$ is then passed on to the spiking neural network for subsequent processing.

### 3.4 Theoretical Energy Consumption Calculation Analysis

In SNNs research, energy consumption is typically analyzed using theoretical estimates rather than specific hardware implementations, primarily for qualitative comparisons with ANNs [23, 24]. These estimates assume that BN layers can be combined with convolutional layers due to the homogeneity of convolution operations, thus neglecting the energy cost of BN layers during deployment [25]. The energy consumption of PBSpikformer is calculated based on prior works [23-27], assuming Multiply-Accumulate (MAC) and accumulate (AC) operations on 45 nm hardware. The energy per MAC and AC operation is $E_{\mathrm{MAC}} = 4.6\,\mathrm{pJ}$ and $E_{\mathrm{AC}} = 0.9\,\mathrm{pJ}$, respectively. The total energy consumption of PBSpikformer is given by:

$$E_{\mathrm{PBSpikformer}} = E_{\mathrm{AC}}\left(\sum_{i=2}^{N} SOPs_{\mathrm{Conv}}^{i} + \sum_{j=1}^{M} SOPs_{\mathrm{FPA}}^{j} + \sum_{k=1}^{Z} SOPs_{\mathrm{SSA}}^{k}\right) + E_{\mathrm{MAC}}\left(FLOPs_{\mathrm{Conv}}^{1} + \sum_{i=1}^{M+Z} FLOPs_{\mathrm{DBC}}^{i} + \sum_{j=1}^{M} FLOPs_{\mathrm{SCMA}}^{j}\right) \tag{8}$$

where Synaptic Operations (SOPs) and FLoating-point Operations (FLOPs) refer to the number of spike-based AC operations and the number of floating-point MAC operations, respectively.

## 4 Experiments

In this section, we evaluate the performance of the proposed method on neuromorphic datasets, including CIFAR10-DVS[31] and DVS128 Gesture[32], as well as on static datasets, such as CIFAR[33], which encompasses both CIFAR10 and CIFAR100. Additionally, several ablation trials are performed to gauge how each component of the proposed methods influences overall performance.

**Table 2.** Performance comparison of our method with existing methods on CIFAR10/100.

| Methods | Architecture | Param (M) | Time Step | CIFAR10 Acc(%) | CIFAR100 Acc(%) |
|---|---|---|---|---|---|
| Hybrid training[28] | ResNet-20/VGG-11[a] | 9.27 | 125 | 92.22 | 67.87 |
| STBP[4] | CIFARNet | 17.54 | 12 | 89.83 | - |
| STBP-tdBN[29] | ResNet-19 | 12.63 | 4 | 92.92 | 70.86 |
| TET[30] | ResNet-19 | 12.63 | 4 | 94.44 | 74.47 |
| MS-ResNet[26] | ResNet-110 | - | - | 91.72 | 66.83 |
| | ResNet-482 | - | - | 91.90 | - |
| ANN[11] | ResNet-19[b] | 12.63 | 1 | 94.97 | 75.35 |
| | Transformer | 9.32 | 1 | **96.73** | 81.02 |
| Transformer-based SNNs | Spikformer[11] | 9.32 | 4 | 95.51 | 78.21 |
| | QKFormer[12] | 6.74 | 4 | 96.18 | <u>81.15</u> |
| | **PBSpikformer** | 6.74 | 4 | <u>96.48</u> | **81.43** |

[a] X1/X2 denotes the architecture or time step for CIFAR10 and CIFAR100.
[b] denotes self-implementation results by [30].

### 4.1 Static Datasets Classification

For CIFAR datasets, PBSpikformer uses four blocks across three stages (N1, N2, N3 = 1, 1, 2). The AdamW optimizer is used with a batch size of 64, a learning rate of $1 \times 10^{-3}$ for CIFAR-10 and $2 \times 10^{-3}$ for CIFAR-100, and training lasts for 700 epochs with a cosine annealing scheduler and 20-epoch warm-up. The train:test ratio is 5:1. Following DeiT[34], we apply RandAugment[35], random erasing[36], and stochastic depth[37] for data augmentaion.

As Table 2 shows, PBSpikformer outperforms prior work on both CIFAR-10 and CIFAR-100 datasets. On CIFAR-10, it achieves 96.48%, outperforming most state-of-the-art methods, including QKFormer by 0.3% and falling just 0.25% behind the conventional Transformer model. On CIFAR-100, PBSpikformer reaches 81.43%, surpassing QKFormer and Transformer by 0.28% and 0.41%, respectively, confirming its position as a leading SOTA model in SNNs.

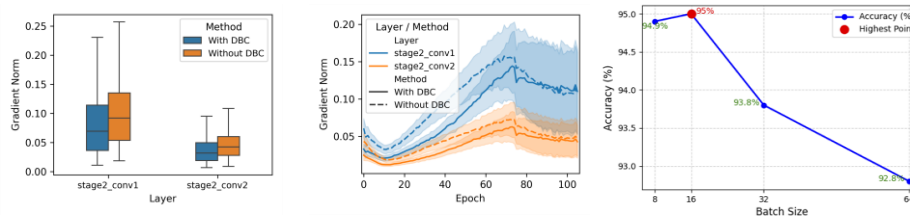
### 4.2 Neuromorphic Datasets Classification

In this experiment, we evaluate a compact version of PBSpikformer (mini-BatchSpikeformer), which consists of one encoder block in Stage 2 (N2=1) and one in Stage 3 (N3=1), while skipping Stage 1 (N1=0). For the CIFAR10-DVS dataset, the model is trained with a batch size of 16 for 500 epochs, using a learning rate of $5 \times 10^{-4}$. For the DVS128 Gesture dataset, the batch size is also 16, with a learning rate of $1 \times 10^{-3}$ and 200 epochs. The train:test ratio for both datasets is 9:1, and all other experimental settings align with those used for CIFAR-10 and CIFAR-100 to ensure consistency.

**Table 3.** Performance comparison of our method with existing methods on DVS128 Gesture and CIFAR10DVS.

| Methods | Architecture | Time Step | DVS128 Acc(%) | CIFAR10DVS Acc(%) |
|---|---|---|---|---|
| TA-SNN[38] | CNN-based SNN[39] | 60/10 | 98.6 | 72.0 |
| STBP-tdBN[29] | ResNet-17/19 | 40/10 | 96.9 | 67.8 |
| SEW-ResNet[40] | 7B-Net/Wide-7B-Net | 16 | 97.9 | 74.4 |
| SALT[41] | VGG-11 | -/20 | - | 67.1 |
| DSR[42] | VGG-11 | -/10 | - | 77.3 |
| MS-ResNet[26] | MS-ResNet20 | - | - | 75.6 |
| Transformer-based SNNs | Spikformer[11] | 16 | 98.3 | 80.9 |
| | QKFormer[12] | 16 | <u>98.6</u> | <u>84.0</u> |
| | **PBSpikformer** | 16 | **99.3** | **96.7** |

Table 3 presents the performance of PBSpikformer on the DVS128 Gesture and CIFAR10-DVS datasets, comparing it with state-of-the-art methods. PBSpikformer achieves 99.3% accuracy on the DVS128 Gesture dataset, matching the best-performing S-Transformer and outperforming Spikformer and QKFormer by 1% and 0.7%, respectively. On CIFAR10-DVS, it reaches 96.7%, significantly surpassing QKFormer and S-Transformer by 12.7% and 16.7%, respectively.




(a) (b) (c)

**Fig. 4.** Ablation study results of DBC: (a) Smoothed gradient trends with confidence intervals; (b) Accuracy under different batch sizes.

### 4.3 Ablation Study

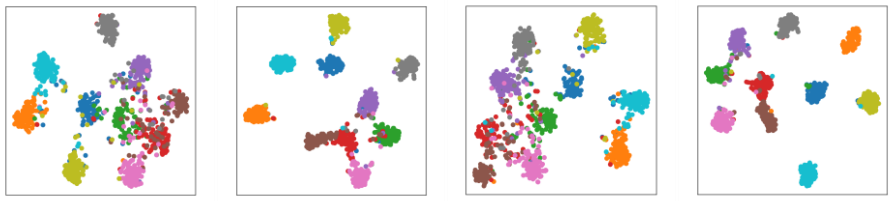
We run a suite of ablation studies to quantify each module's contribution. First, we analyze how DBC contributes to gradient stability and robustness under varying batch sizes. As shown in Fig. 4a and Fig. 4b show that DBC leads to lower and more stable gradient norms across layers. Moreover, Fig. 4c demonstrates that while smaller batch sizes (8 and 16) yield higher accuracy due to finer updates, DBC consistently enhances performance across all batch sizes.

**Table 4.** Effect of The Proposed Modules.

| Methods | Mem Usage (MB) | Training Speed (FPS) | Accuracy (%) |
|---|---|---|---|
| Baseline | 41.38 | 205 | 84 |
| + DBC[a] | 41.38 | 198 | 95 |
| + DBC'[b] | 41.39 | 203 | 85.4 |
| + FPA | 42.26 | 205 | 84.3 |
| + DBC + FPA | 42.26 | 206 | **95.2** |

[a] denotes $\alpha$ is set to 0.5, [b] denotes $\alpha$ is learnable.

We further evaluate the individual contributions of DBC and FPA. As shown in Table 4, introducing DBC with a fixed coefficient ($\alpha = 0.5$) boosts accuracy from 84% to 95%, with minimal overhead in memory and training speed. In contrast, making $\alpha$ learnable leads to unstable results. FPA alone provides only a modest improvement, but combining FPA with DBC achieves the best result—95.2% accuracy—without compromising efficiency. And t-SNE visualizations (see Fig. 5) show improved class separation when DBC and FPA are applied.



(a) (b) (c) (d)

**Fig. 5.** t-SNE visualization under different settings: (a) baseline, (b) with FPA, (c) with DBC, and (d) with FPA and DBC modules.

**Table 5.** Theoretical energy evaluation on CIFAR100.

| Methods | Architecture | Param (M) | OPs (G) | Power (mJ) | Accuracy (%) |
|---|---|---|---|---|---|
| ANN | Transformer[11] | 9.32 | 0.51 | 2.37 | 81.02 |
| SNN | Spikformer[11] | 9.33 | 0.30 | **0.30** | 77.77 |
| | QKformer[12] | 6.74 | 0.33 | 0.34 | 81.15 |
| | **PBSpikformer** | 6.74 | 0.37 | 0.36 | **81.43** |

Finally, we evaluate the theoretical energy efficiency of PBSpikformer. As shown in Table 4, our model achieves a low energy consumption of 0.36 mJ on CIFAR100, representing a 6.7× reduction compared to ANN-based Transformers. Although slightly higher than Spikformer (by 20%) and QKformer (by 6%), PBSpikformer attains the highest accuracy (81.43%), demonstrating an excellent balance between performance and energy efficiency.

## 5 Conclusion

In this paper, we present PBSpikformer, a novel SNN architecture that integrates Fourier-Phase Attention (FPA) and Dynamic Batch Context (DBC) to enhance image classification. By leveraging spectral phase features and batch-level global activation signals, our model captures domain-invariant semantics and improves training stability. Inspired by biological principles, this design addresses key limitations of traditional SNNs. PBSpikformer achieves state-of-the-art performance on both static and neuromorphic datasets, with strong energy efficiency. Our results highlight the potential of combining spiking dynamics with attention mechanisms for energy-constrained applications. Future work will extend this framework to real-time and large-scale scenarios.

**Acknowledgments.** This work was supported in part by the National Natural Science Foundation of China under Grants 62106208 and 62373311, in part by the Natural Science Foundation of Sichuan Province under Grant 2025ZNSFSC0471 and Grant 2023NSFSC1418, in part by the China Postdoctoral Science Foundation under Grants 2021TQ0272 and 2021M702715, in part by the Fundamental Research Funds for the Central Universities 2682025ZTPY007.

**Disclosure of Interests.** The authors report no conflicts of interest.

## References

1. W. Maass, “Networks of spiking neurons: the third generation of neural network models,” Neural networks, vol. 10, no. 9, pp. 1659–1671, 1997.
2. M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Di-mou, P. Joshi, N. Imam, S. Jain, et al., “Loihi: A neuromorphic manycore processor with on-chip learning,” Ieee Micro, vol. 38, no. 1, pp. 82–99, 2018.
3. M. Yao, O. Richter, G. Zhao, N. Qiao, Y. Xing, D. Wang, T. Hu, W. Fang, T. Demirci, M. De Marchi, et al., “Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip,” Nature Communications, vol. 15, no. 1, p. 4464, 2024.
4. Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, “Spatio-temporal backpropagation for training high-performance spiking neural networks,” Frontiers in neuroscience, vol. 12, p. 331, 2018.
5. E. O. Neftci, H. Mostafa, and F. Zenke, “Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks,” IEEE Signal Processing Magazine, vol. 36, no. 6, pp. 51–63, 2019.
6. W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, “Deep residual learning in spiking neural networks,” Advances in Neural Information Processing Systems, vol. 34, pp. 21056–21069, 2021.
7. A. Vaswani, “Attention is all you need,” Advances in Neural Information Processing Systems, 2017.
8. A. Dosovitskiy, “An image is worth 16x16 words: Transformers for image recognition at scale,” arXiv preprint arXiv:2010.11929, 2020.
9. Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in Proceedings of the IEEE/CVF international conference on computer vision, pp. 10012–10022, 2021.
10. W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L.Shao,“Pyramidvisiontransformer: A versatile backbone for dense prediction without

convolutions," in Proceedings of the IEEE/CVF international conference on computer vision, pp. 568–578, 2021.
11. Z. Zhou, Y. Zhu, C. He, Y. Wang, S. Yan, Y. Tian, and L. Yuan, "Spikformer: When spiking neural network meets transformer," arXiv preprint arXiv:2209.15425, 2022.
12. C. Zhou, H. Zhang, Z. Zhou, L. Yu, L. Huang, X. Fan, L. Yuan, Z. Ma, H. Zhou, and Y. Tian, "Qkformer: Hierarchical spiking transformer using qk attention," arXiv preprint arXiv:2403.16552, 2024.
13. A. Oppenheim, J. Lim, G. Kopec, and S. Pohlig, "Phase in speech and pictures," inICASSP'79. IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 4, pp. 632–637, IEEE, 1979.
14. A. V. Oppenheim and J. S. Lim, "The importance of phase in signals," Proceedings of the IEEE, vol. 69, no. 5, pp. 529–541, 1981.
15. L.N.PiotrowskiandF.W.Campbell,"A demonstration of the visual importance and flexibility of spatial-frequency amplitude and phase," Perception, vol. 11, no. 3, pp. 337–346, 1982.
16. Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy efficient object recognition," International Journal of Computer Vision, vol. 113, pp. 54–66, 2015.
17. Z. Wang, Y. Fang, J. Cao, Q. Zhang, Z. Wang, and R. Xu, "Masked spiking transformer," in Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1761–1771, 2023.
18. T. Bu, W. Fang, J. Ding, P. Dai, Z. Yu, and T. Huang, "Optimal ann snn conversion for high-accuracy and ultra-low-latency spiking neural networks," arXiv preprint arXiv:2303.04347, 2023.
19. Y. Rao, W. Zhao, Z. Zhu, J. Lu, and J. Zhou, "Global filter networks for image classification," Advances in neural information processing systems, vol. 34, pp. 980–993, 2021.
20. Q. Xu, R. Zhang, Y. Zhang, Y. Wang, and Q. Tian, "A fourier-based framework for domain generalization," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 14383–14392, 2021.
21. P. Fries, "Rhythms for cognition: communication through coherence," Neuron, vol. 88, no. 1, pp. 220–235, 2015.
22. F. Varela, J.-P. Lachaux, E. Rodriguez, and J. Martinerie, "The brainweb: phase synchronization and large-scale integration," Nature reviews neuroscience, vol. 2, no. 4, pp. 229–239, 2001.
23. P. Panda, S. A. Aketi, and K. Roy, "Toward scalable, efficient, and accurate deep spiking neural networks with backward residual connections, stochastic softmax, and hybridization," Frontiers in Neuroscience, vol.14, p.653, 2020.
24. B. Yin, F. Corradi, and S. M. Bohté, "Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks," Nature Machine Intelligence, vol. 3, no. 10, pp. 905–913, 2021.
25. X. Ding, Y. Guo, G. Ding, and J. Han, "Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks," in Proceedings of the IEEE/CVF international conference on computer vision, pp. 1911–1920, 2019.
26. Y. Hu, L. Deng, Y. Wu, M. Yao, and G. Li, "Advancing spiking neural networks towards deep residual learning," arXiv preprint arXiv:2112.08954, 2021.
27. M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC), pp. 10–14, IEEE, 2014.

28. N. Rathi, G. Srinivasan, P. Panda, and K. Roy, “Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation,” arXiv preprint arXiv:2005.01807, 2020.
29. H. Zheng, Y. Wu, L. Deng, Y. Hu, and G. Li, “Going deeper with directly-trained larger spiking neural networks,” in Proceedings of the AAAI conference on artificial intelligence, vol. 35, pp. 11062–11070, 2021.
30. S. Deng, Y. Li, S. Zhang, and S. Gu, “Temporal efficient training of spiking neural network via gradient re-weighting,” arXiv preprint arXiv:2202.11946, 2022.
31. H. Li, H. Liu, X. Ji, G. Li, and L. Shi, “Cifar10-dvs: an event-stream dataset for object classification,” Frontiers in neuroscience, vol. 11, p. 309, 2017.
32. A.Amir, B.Taba, D.Berg, T.Melano, J.McKinstry, C.DiNolfo, T.Nayak, A.Andreopoulos, G.Garreau, M.Mendoza, et al., “Alowpower,fully event-based gesture recognition system,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7243–7252, 2017.
33. A. Krizhevsky, G. Hinton, et al., “Learning multiple layers of features from tiny images,” 2009.
34. H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through attention,” in International conference on machine learning, pp. 10347–10357, PMLR, 2021.
35. E. D. Cubuk, B. Zoph, J. Shlens, and Q. 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, pp. 702–703, 2020.
36. Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, “Random erasing data augmentation,” in Proceedings of the AAAI conference on artificial intelligence, vol. 34, pp. 13001–13008, 2020.
37. G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, “Deep networks with stochastic depth,” in Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14, pp. 646–661, Springer, 2016.
38. M. Yao, H. Gao, G. Zhao, D. Wang, Y. Lin, Z. Yang, and G. Li, “Temporal-wise attention spiking neural networks for event streams classification,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10221–10230, 2021.
39. W. He, Y. Wu, L. Deng, G. Li, H. Wang, Y. Tian, W. Ding, W. Wang, and Y. Xie, “Comparing snns and rnns on neuromorphic vision datasets: Similarities and differences,” Neural Networks, vol. 132, pp. 108–120, 2020.
40. W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, “Deep residual learning in spiking neural networks,” Advances in Neural Information Processing Systems, vol. 34, pp. 21056–21069, 2021.
41. Y. Kim and P. Panda, “Optimizing deeper spiking neural networks for dynamic vision sensing,” Neural Networks, vol. 144, pp. 686–698, 2021.
42. Q. Meng, M. Xiao, S. Yan, Y. Wang, Z. Lin, and Z.-Q. Luo, “Training high-performance low-latency spiking neural networks by differentiation on spike representation,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12444–12453, 2022.