Towards Accurate Binary Spiking Neural Networks: Learning with Adaptive Gradient Modulation Mechanism

Yu Liang[1], Wenjie Wei[1], Ammar Belatreche[2], Honglin Cao[1], Zijian Zhou[1], Shuai Wang[1], Malu Zhang[1*], Yang Yang[1]

[1]University of Electronic Science and Technology of China
[2]Northumbria University

Abstract

Binary Spiking Neural Networks (BSNNs) inherit the event-driven paradigm of SNNs, while also adopting the reduced storage burden of binarization techniques. These distinct advantages grant BSNNs lightweight and energy-efficient characteristics, rendering them ideal for deployment on resource-constrained edge devices. However, due to the binary synaptic weights and non-differentiable spike function, effectively training BSNNs remains an open question. In this paper, we conduct an in-depth analysis of the challenge for BSNN learning, namely the frequent weight sign flipping problem. To mitigate this issue, we propose an Adaptive Gradient Modulation Mechanism (AGMM), which is designed to reduce the frequency of weight sign flipping by adaptively adjusting the gradients during the learning process. The proposed AGMM can enable BSNNs to achieve faster convergence speed and higher accuracy, effectively narrowing the gap between BSNNs and their full-precision equivalents. We validate AGMM on both static and neuromorphic datasets, and results indicate that it achieves state-of-the-art results among BSNNs. This work substantially reduces storage demands and enhances SNNs' inherent energy efficiency, making them highly feasible for resource-constrained environments.

Introduction

Deep Neural Networks (DNNs) have achieved significant breakthroughs in various fields, such as computer vision (He et al. 2016; Liu et al. 2024b), natural language processing (Vaswani et al. 2017; Achiam et al. 2023), and speech processing (Wu et al. 2018a; Pan et al. 2021). However, these successes rely on complex model architectures and large model parameters, making their deployment on resource-limited devices highly challenging (Qin et al. 2024). In contrast, brain-inspired spiking neural networks (SNNs) emulate the spike generation mechanism of biological neurons, utilizing sparse binary spikes as the basic unit for information transmission (Ghosh-Dastidar and Adeli 2009). Due to their sparse and spike-driven computational paradigm, SNNs have been regarded as the energy-efficient alternative to traditional DNNs (Davies et al. 2018). Nonetheless, despite their energy efficiency, SNNs often struggle to achieve satisfactory performance when dealing with complex tasks (Guo, Huang, and Ma 2023).

To ensure the competitive performance of SNNs on complex tasks, the SNN community is committed to developing effective training strategies and large-scale architectures (Wu et al. 2021; Yao et al. 2024; Guo et al. 2022b), often neglecting the inherent energy efficiency trait of SNNs. This negligence increases the difficulty of deploying SNNs on resource-constrained edge computing environments (Liu et al. 2023; Guo et al. 2022a). Therefore, a deeper understanding of the energy efficiency advantage of SNNs is crucial to enable their widespread deployment, especially in resource-limited scenarios (Wei et al. 2024b). In recent years, some studies have conducted preliminary exploration on compressing SNNs, including techniques such as pruning (Li et al. 2024; Shi et al. 2023), neural architecture search (Liu et al. 2024a; Yan et al. 2024), and quantization (Hu, Zheng, and Pan 2024).

Among these compression techniques, quantization is a promising solution, which reduces computational and storage demands by representing weights with low bit width (Gong et al. 2014; Li, Dong, and Wang 2019). The most extreme method within this domain is binarization, which compresses networks by reducing the parameter bit width to a mere 1-bit (Rastegari et al. 2016; Qin et al. 2022; Gholami et al. 2022). Incorporating this binarization into SNNs allows computationally intensive convolutional operations to be replaced with efficient bitwise operations. This transformation significantly reduces both computation and storage requirements, making networks more hardware-friendly and energy-efficient. However, despite the significant advantages in binarizing SNNs, these approaches struggle to scale to complex tasks when compared to full-precision SNNs (FP-SNNs) (Yin et al. 2024). There is no extensive research on the underlying reason behind this critical challenge.

In this paper, we reveal that due to the two possible values of their weights (-1 and +1), binary SNNs (BSNNs) face a more severe challenge of weight sign flipping during the learning process. This issue causes the performance of BSNNs to lag behind that of FP-SNNs equivalents, limiting their application in complex scenarios. To address this issue, we propose an adaptive gradient modulation mechanism (AGMM) to adaptively regulate the gradient magnitude dur-

*Corresponding author: maluzhang@uestc.edu.cn

ing the learning process. By integrating AGMM, BSNNs can mitigate the frequent weight sign flipping problem, leading to faster convergence and higher accuracy. We summarize the main contributions of this paper as follows:

- We provide a comprehensive analysis of the difficulty in BSNNs learning, i.e., the frequent weight sign flipping problem. Furthermore, we reveal that the frequency of flipping is proportional to the mean and variance of gradients via rigorous reasoning.
- We propose an Adaptive Gradient Modulation Mechanism (AGMM), aiming to adjust the gradient magnitude in the learning process to solve the frequent weight sign flipping problem. This mechanism effectively narrows the gap between BSNNs and their FP-SNN equivalents.
- Extensive experiments conducted on both static and neuromorphic datasets demonstrate that our method achieves state-of-the-art performance when compared with other existing BSNNs approaches. Moreover, we thoroughly validate the effectiveness and efficiency of the AGMM.

Related Work

Spiking Neural Network

Brain-inspired SNNs have emerged as a new computing paradigm (Zhang et al. 2019; Tang et al. 2024). Unlike traditional DNNs, SNNs perform computations using binary spikes over time, making them especially suitable for energy-efficient computing applications (Wang et al. 2024b; Wei et al. 2023). Given the non-differentiability of the spike generation function, the SNN research community has devoted considerable efforts to developing effective training algorithms. These approaches can be broadly classified into two categories: conversion-based methods and direct learning algorithms. Conversion methods may not fully exploit the temporal processing capabilities of SNNs and often require multiple steps for accurate inference (Roy, Chakraborty, and Roy 2019; Wang et al. 2020). In contrast, direct learning methods can achieve high accuracy with low-latency inference (Wu et al. 2018b, 2019; Wang et al. 2024a). Considering activation values in SNNs are binary, further quantizing the weights to binary would allow computation-intensive convolution operations to be executed using simple bitwise operations. This would further enhance the inherent efficiency advantages of SNNs.

Binary Neural Network

As an extreme form of network quantization, binary neural networks (BNNs) constrain both weights and activations to -1 and +1, substantially reducing memory storage and computational overhead. However, BNNs experience substantial performance degradation compared to their full-precision counterparts. Subsequent research has focused on narrowing this performance gap between BNNs and full-precision neural networks. Several approaches have been proposed to address this issue. XNOR-Net (Rastegari et al. 2016) employs a deterministic binarization scheme and minimizes quantization error by incorporating scaling factors in each layer. Bi-Real (Liu et al. 2018) explores piecewise polynomial functions and network architectures for quantization. Previous work has also focused on constraining the distribution of weights or activations in BNNs (Xu et al. 2021). Building on these advancements in the binary domain, binarization methods in SNNs remain a promising area for exploration.

Binarization techniques in SNNs

Several researchers have combined SNNs with binarization techniques. For example, (Qiao et al. 2021) employ the surrogate gradient (SG) method to train BSNNs directly. (Jang, Skatchkovsky, and Simeone 2021) propose a novel Bayesian-based algorithm for BSNNs, which shows significant advantages in accuracy and calibration compared to SG methods. (Kheradpisheh, Mirsadeghi, and Masquelier 2022) integrate binarization into temporal-coded SNNs, where each neuron emits at most one spike, offering substantial energy benefits. Recently, (Wei et al. 2024a) take inspiration from information theory and propose a weight-spike dual regulation (WS-DR) method to enhance the information capacity of BSNNs, achieving significant performance improvements. However, current BSNNs continue to encounter challenges when scaling to complex datasets such as ImageNet. Therefore, it is essential to investigate the challenges inherent in the BSNN training process and to bridge the performance gap between BSNNs and FP-SNNs.

Preliminary

Leaky Integrate-and-Fire model. Various spiking neuron models have been proposed (Hodgkin and Huxley 1952; Zhang et al. 2021). In this paper, we select the Leaky Integrate-and-Fire (LIF) model due to its computational efficiency. Typically, the Euler method is employed to discretize the dynamic equations of LIF models (Wu et al. 2018b), resulting in the following iterative form:

$$U^n[t] = \tau U^n[t-1] + X^n[t], \tag{1}$$

where τ is the constant leaky factor, $U^n[t]$ represents the membrane potential of neurons in layer n at time t, and $X^n[t] \in \mathbb{R}^{(C,H,W)}$ is their input current. $X^n[t]$ integrates inputs from all presynaptic neurons, described as:

$$X^n[t] = \mathcal{BN}\left(W^n S^{n-1}[t]\right), \tag{2}$$

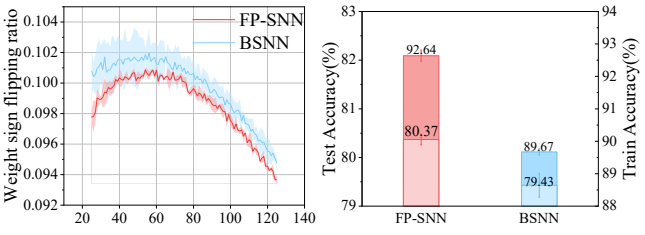
where $\mathcal{BN}(\cdot)$ denotes the batch normalization operation, W^n is the weights of the n-th convolutional layer, and $S^{n-1}[t]$ is the output spikes of neurons in layer $n-1$ at time t. When the membrane potential $U^n[t]$ reaches the firing threshold V_{th}, the neuron will generate a spike, defined as:

$$S^n[t] = \begin{cases} 1, & \text{if } U^n[t] \geq V_{th}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

After spike emission, the hard reset function is invoked, mathematically defined as:

$$U^n[t] = U^n[t] \cdot \left(1 - S^n[t]\right). \tag{4}$$

Clearly, $U^n[t]$ is reset to zero upon spike emission while remaining unchanged if no spike occurs.



(a) (b)

Figure 1: Weight sign flipping ratio and performance comparison between FP-SNN and vanilla BSNN. The darker color represents the training accuracy, while the lighter color indicates the test accuracy.

Weight binarization To achieve significant model compression, BSNNs adopt the sign function to convert W^n in Eq. 2 into binary representations. Typically, the scaling factor γ is introduced to match the distribution of float-point weights, thus minimizing the quantization error. Therefore, the binarization on $\omega \in W^n$ can be formulated as:

$$\omega_b = \gamma \cdot \text{sign}(\omega) = \begin{cases} -\gamma, & \text{if } \omega < 0, \\ +\gamma, & \text{otherwise}, \end{cases} \tag{5}$$

where ω_b is a binary weight ($\omega_b \in W_b^n$), and γ is computed as the average of the absolute value of weights in each channel (Rastegari et al. 2016). To solve the non-differentiability of $\text{sign}(\cdot)$, the straight-through estimator (STE) is employed to approximate the gradient of binary weights (Bengio, Léonard, and Courville 2013), i.e., $\frac{\partial \text{sign}(\cdot)}{\partial W^n} = 1_{|W^n| \leq 1}$.

Learning algorithm In order to train SNNs successfully, we use the spatial-temporal backpropagation (STBP) algorithm (Wu et al. 2018b), where the gradient of the loss function $\mathcal{L}$ to weight W_b^n can be formulated as:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial W_b^n} = \sum_t \Bigg(& \frac{\partial \mathcal{L}}{\partial S^n[t]} \frac{\partial S^n[t]}{\partial U^n[t]} + \\ & \frac{\partial \mathcal{L}}{\partial U^n[t+1]} \frac{\partial U^n[t+1]}{\partial U^n[t]} \Bigg) \frac{\partial U^n[t]}{\partial W_b^n}, \end{aligned} \tag{6}$$

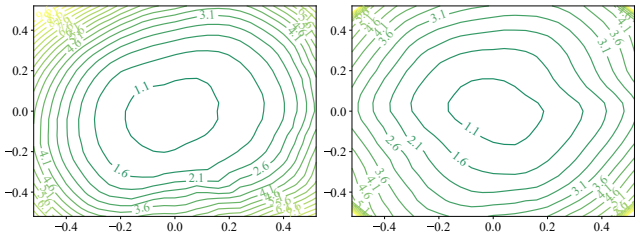
where $\frac{\partial S^n[t]}{\partial U^n[t]}$ is the derivative of the non-differentiable spike step function. Therefore, the surrogate gradient is employed to replace this gradient. In this paper, we use the triangular-shaped function, which is mathematically defined as:

$$\frac{\partial S^n[t]}{\partial U^n[t]} = \max\left(0, \beta - |U^n[t] - V_{th}|\right), \tag{7}$$

where β is a hyperparameter that controls the gradient range.

Method

In this section, we first conduct an in-depth analysis of the frequent weight sign flipping problem in BSNNs. Subsequently, we propose an AGMM method to adaptively adjust the gradient magnitude in the BSNN learning process, thereby addressing the aforementioned issue.



(a) FP-SNN (b) Vanilla BSNN

Figure 2: The loss landscape of FP-SNN and vanilla BSNN.

Problem analysis

Compared to FP-SNNs, BSNNs offer substantial advantages in energy efficiency, making them highly suitable for deployment on edge devices. However, scaling BSNNs to more complex tasks or datasets is still a major challenge. This is primarily attributed to the frequent weight sign flipping problem in the learning process of BSNNs.

Specifically, FP-SNNs update the weight ω in a continuous space, allowing fine-grained adjustments. In contrast, BSNNs update ω in a discrete space, namely -1 and 1, essentially flipping the sign. While an appropriate flipping frequency can help achieve stable convergence and high-performance networks, BSNNs suffer from more frequent weight sign flipping compared to FP-SNNs. To measure this difference, we adopt the metric proposed by (Helwegen et al. 2019) to calculate the weight sign flipping ratio during the learning process. This metric is defined as:

$$\pi_e = \frac{\sum_n \text{sum}(\text{sign}(W_e^n) \oplus \text{sign}(W_{e+1}^n))}{\sum_n |W_e^n|}, \tag{8}$$

where W_e^n is the weight in layer n at epoch e, $\text{sum}(\cdot)$ is the sum function, and $|W_e^n|$ is the number of entries in W_e^n.

We record the flipping ratio and performance of FP-SNN and BSNN throughout the overall training process on CIFAR-100 with the ResNet-19 and plot them in Fig. 1. From Fig. 1(a), where the x-axis denotes the number of epochs, we observe that BSNN consistently exhibits a higher flipping ratio compared to FP-SNN throughout the training process. Besides, we can also find that the flipping ratio is gradually decreased as the BSNN converges. The performance results for FP-SNN and BSNN are shown in Fig. 1(b), revealing a significant accuracy gap between FP-SNN and BSNN. To demonstrate the training difference between FP-SNN and BSNN, we also analyze their loss lanscapes(Li et al. 2018). According to Fig. 2, in contrast to FP-SNN, the loss for BSNN increases sharply around the borders of the loss landscape. This sharpness of the network's loss landscape correlates well with its generalization error. This result shows that FP-SNN outperforms BSNN in terms of generalization.

Given the frequent weight sign flipping issue occurring during BSNNs training, it is obvious that balancing the frequency of weight flipping is crucial for better convergence and performance. Fortunately, the frequency of weight sign flipping is positively correlated with the mean and variance of gradients. Next, we conduct a rigorous analysis of this.

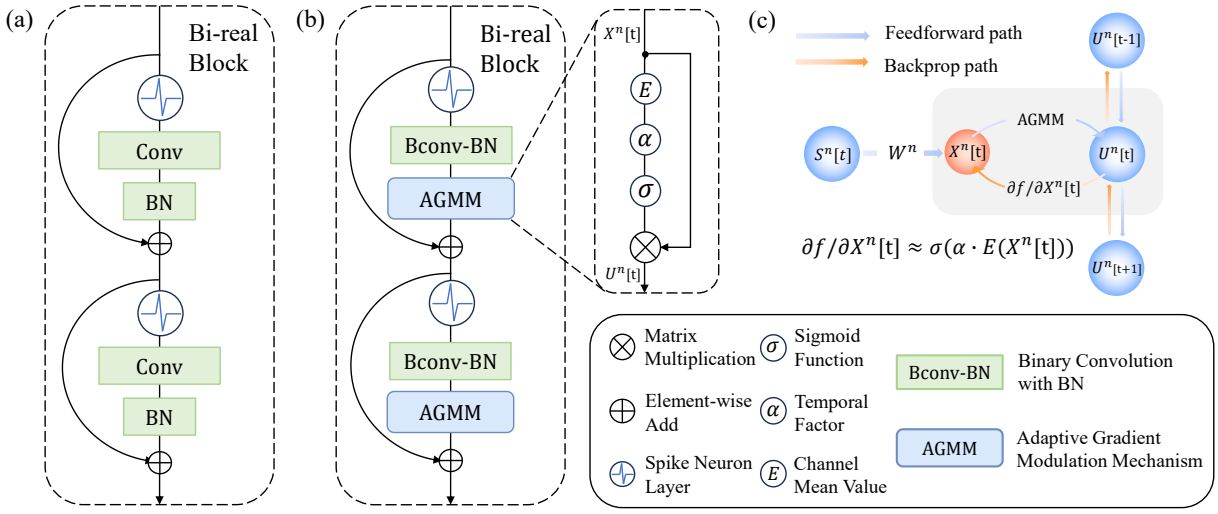


Figure 3: (a) The network architecture of FP-SNNs, which is inherited from the work of Bi-Real Net (Liu et al. 2018). (b) The proposed AGMM module in BSNNs, where each Bconv-BN module is followed by an AGMM module. (c) The backpropagation computation graph of the proposed AGMM.

Proposition 1. *The frequency of weight sign flipping is positively correlated with the mean and variance of gradients. Specifically, the larger the mean (μ) or variance (σ^2) of gradients, the higher the frequency of weight sign flipping.*

Proof 1. Consider a weight ω for analysis. According to gradient descent optimization, its update can be described as:

$$\omega_{e+1} = \omega_e - \eta \nabla \mathcal{L}(\omega_e), \tag{9}$$

where e is the epoch, η is the learning rate, and $\nabla \mathcal{L}(\omega_e)$ is the gradient of the loss function to the weight ω. Based on Eq. 8, we define the weight sign flipping event A_e as $\text{sign}(\omega_e) \neq \text{sign}(\omega_{e+1})$.

Assume $\omega_e > 0$ (the case for $\omega_e < 0$ is similar), then the condition for A_e to occur is $\omega_e - \eta \nabla \mathcal{L}(\omega_e) < 0$. Generally, gradients in neural networks follow a normal distribution (Glorot and Bengio 2010; He et al. 2015; Zhang et al. 2018), i.e., $\nabla \mathcal{L}(\omega_e) \sim N(\mu, \sigma^2)$. Under this distribution, the probability of event A_e can be calculated as:

$$\begin{aligned} P(A_e) &= P(\nabla \mathcal{L}(\omega_e) > \frac{\omega_e}{\eta}) \\ &= 1 - F(\frac{\omega_e}{\eta}; \mu, \sigma) \\ &= 1 - \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\frac{\omega_e}{\eta}} \exp\left(-\frac{(t-\mu)^2}{2\sigma^2}\right) dt, \end{aligned} \tag{10}$$

where $F(\cdot)$ is the cumulative distribution function of $\nabla \mathcal{L}(\omega_e)$. From Eq. 10, it can be observed that $P(A_e)$ is positively correlated with μ and σ^2. Therefore, in BSNNs, the frequent weight sign flipping problem can be alleviated by reducing μ and σ^2 of the gradients in the backward process.

Adaptive Gradient Modulation Mechanism

Based on the above analysis, we propose an Adaptive Gradient Modulation Mechanism (AGMM), which adaptively reduces gradient magnitudes during the learning process, as shown in Fig. 3. In the following, we analyze how AGMM works from its forward and backward propagation.

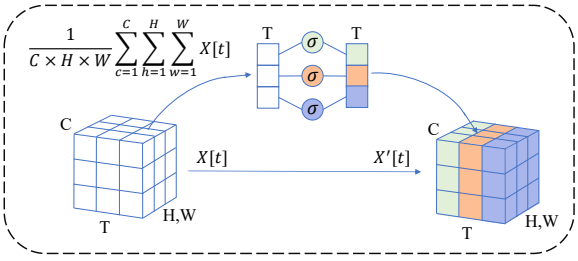


Figure 4: Diagram of AGMM. $X[t]$ is computed by Bconv-BN module, where T denotes the time step, C denotes the channel, and H, W represents the spatial resolution.

Forward propagation In the forward propagation, the proposed AGMM is performed on the spatial feature map $X^n[t]$, which is calculated by the Bconv-BN module, as shown in Fig. 4. For simplicity, we ignore the layer index n in the following analysis. The AGMM is mathematically defined as follows:

$$\begin{aligned} X'[t] &= \sigma(E(X[t])) \cdot X[t], \\ E(X[t]) &= \frac{\alpha[t]}{C \times H \times W} \sum_{c=1}^{C} \sum_{h=1}^{H} \sum_{w=1}^{W} X[t], \end{aligned} \tag{11}$$

where $X'[t]$ denotes the output feature map calculated by AGMM, $\sigma(\cdot)$ is the sigmoid function, and $\alpha[t]$ is the trainable temporal-wise factor. During the forward pass, AGMM computes the time-scaled average $E(X[t])$ with the temporal factor $\alpha[t]$, and modulates the feature map via a sigmoid function to generate the enhanced output $X'[t]$.

AGMM preserves the network's spike-driven character-

istics despite introducing negligible parameters. Furthermore, the sigmoid function of AGMM lowers the firing rate of the network, which in turn lowers synaptic operations (SOPs). The Experiments section offers comprehensive evidence supporting these claims.

Backward propagation To better understand the effectiveness of AGMM in training, we analyze the gradient magnitude throughout the network and provide a thorough explanation of AGMM's backpropagation mechanism. In AGMM-BSNN, the derivative of the loss function with respect to the binary weight ω_b is described as:

$$\frac{\partial \mathcal{L}_{\text{AGMM}}}{\partial \omega_b} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}_{\text{AGMM}}}{\partial X'[t]} \frac{\partial X'[t]}{\partial X[t]} \frac{\partial X[t]}{\partial \omega_b}. \tag{12}$$

Compared to that of vanilla BSNN, this derivative includes an additional term $\partial X'[t]/\partial X[t]$. In the following analysis, we primarily focus on the additional term. By applying the chain rule, this additional term can be expressed as follows:

$$\begin{aligned} \frac{\partial X'[t]}{\partial X[t]} &= \frac{\partial(\sigma(E(X[t]) \cdot X[t]))}{\partial X[t]} \\ &= \sigma(E(X[t])) \frac{\partial X[t]}{\partial X[t]} + X[t] \frac{\partial \sigma(E(X[t]))}{\partial X[t]}. \end{aligned} \tag{13}$$

The first term in Eq. 13 is simplified to $\sigma(E(X[t]))$, while the second term can be expanded using the derivative of the sigmoid function, i.e., $\sigma'(x) = \sigma(x)(1 - \sigma(x))$. Therefore, the derivative in the second term of Eq. 13 can be rewritten as:

$$\frac{\partial \sigma(E(X[t]))}{\partial X[t]} = \frac{\partial E(X[t])}{\partial X[t]} \sigma(E(X[t]))(1 - \sigma(E(X[t]))), \tag{14}$$

where $\partial E(X[t])/\partial X[t] = \frac{\alpha[t]}{C \times H \times W}$ and $\sigma(E(X[t]))(1 - \sigma(E(X[t])) \in [0, 0.25]$. Therefore, the numerator of the term in Eq. 14 is substantially smaller than the denominator. Building upon this, the second term of Eq. 13 can be neglected. Consequently, it can be rewritten as a simplified form:

$$\frac{\partial X'[t]}{\partial X[t]} \approx \sigma(E(X[t])). \tag{15}$$

Building on the above analysis, the gradients of the binary weight ω_b in a BSNN with AGMM can be expressed as follows:

$$\nabla \mathcal{L}_{\text{AGMM}}(\omega_b) \approx \sum_{t=1}^{T} \nabla \mathcal{L}_t(\omega_b) \cdot \sigma(E(X[t])), \tag{16}$$

where the gradient distribution $\nabla \mathcal{L}_t(\omega_b)$ at time t follows a normal distribution $N(\mu_t, \sigma_t^2)$, and $\nabla \mathcal{L}_{\text{AGMM}}(\omega_b)$ follows a normal distribution $N(\mu, \sigma^2)$. As a result, both the mean and variance of the gradients at time t are scaled by $\sigma(E(X[t]))$. Given that $0 < \sigma(\cdot) < 1$, this scaling reduces both the mean and variance of the gradient. By combining with Proof 1, we can derive:

$$P_{\text{AGMM}}(A_e) < P(A_e). \tag{17}$$

Therefore, using the AGMM can effectively reduce the probability of frequent weight sign flipping. The sigmoid function $\sigma(\cdot)$ reduces the original gradient values while the term $E(X[t])$ determines the extent of gradient reduction based on the time step. By combining these two characteristics in backpropagation, AGMM can adaptively reduce the gradient magnitude to mitigate the problem of frequent weight sign flipping in BSNNs.

Experiments

Implementation details

The experimental evaluation focuses on image classification tasks across two categories of datasets: static and neuromorphic. The static datasets include CIFAR-10, CIFAR-100 and ImageNet while the neuromorphic datasets comprise DVS-Gesture and DVS-CIFAR10. These datasets are highly significant in the fields of machine learning and neuromorphic computing, serving as standard benchmarks for evaluating a variety of methods. Following previous studies (Rastegari et al. 2016), we maintain full-precision weights in the final fully-connected layer and the first convolutional layer. We evaluate our AGMM using ResNet-19 on CIFAR-10 and CIFAR-100 with 400 epochs, and ResNet-18 on ImageNet with 300 epochs. For ResNet architecture, we use the double skip connections (Liu et al. 2018). For neuromorphic datasets, we implement the structure of VGGSNN commonly used in SNNs (Deng et al. 2022). We employ SGD as the optimizer, coupled with a cosine annealing schedule for learning rate adjustment.

Comparison with related work

In this section, we evaluate the efficacy and efficiency of the proposed AGMM by comparing its performance and model size with existing quantized SNN approaches.

We demonstrate the efficacy of AGMM-BSNN by comparing it with other quantized methods. As shown in Tab. 1, AGMM-BSNN consistently outperforms all quantized methods across various datasets. On the CIFAR-10 dataset, AGMM-BSNN achieves 96.13% and 96.33% accuracy with 2 and 4 timesteps respectively, outperforming other existing approaches. This performance exceeds the previous best result, Q-SNN (Wei et al. 2024a), by 0.59% under the same experimental settings. On CIFAR-100, AGMM-BSNN also demonstrates superior performance, achieving the accuracy of 80.25% and 80.70% with 2 and 4 timesteps. This surpasses the state-of-the-art (SOTA) performance Q-SNN by 1.43% with the same timestep and structure. For ImageNet, AGMM-BSNN outperforms the reported best result Bit-SNN(Hu, Zheng, and Pan 2024), with an accuracy of 64.67%. On neuromorphic datasets, AGMM-BSNN matches the SOTA performance of Q-SNN on DVS-Gesture and exceeds it on DVS-CIFAR10, obtaining 82.40% accuracy on DVS-CIFAR10.

Our model achieves superior performance while maintaining computational efficiency comparable to existing approaches. We analyze the AGMM-BSNN model size in comparison to existing quantized SNN methods on CIFAR-100, with the results illustrated in Fig. 5. AGMM-BSNN

Dataset	Method	Architecture	Learning	Timestep	Accuracy
CIFAR-10	(Wang et al. 2020)	6Conv3FC	ANN2SNN	100	90.19%
	(Yoo and Jeong 2023)	VGG16	ANN2SNN	32	91.51%
	(Deng et al. 2021)	7Conv3FC	Direct train	8	89.01%
	(Pei et al. 2023)	5Conv1FC	Direct train	1	92.12%
	(Wei et al. 2024a)	ResNet-19	Direct train	2	95.54%
	AGMM	ResNet-19	Direct train	2	**96.13%**
				4	**96.33%**
CIFAR-100	(Lu and Sengupta 2020)	VGG15	ANN2SNN	400	62.07%
	(Wang et al. 2020)	6Conv2FC	ANN2SNN	300	62.02%
	(Yoo and Jeong 2023)	VGG16	ANN2SNN	32	66.53%
	(Deng et al. 2021)	7Conv3FC	Direct train	8	55.95%
	(Pei et al. 2023)	6Conv1FC	Direct train	1	69.55%
	(Wei et al. 2024a)	ResNet-19	Direct train	2	78.82%
	AGMM	ResNet-19	Direct train	2	**80.25%**
				4	**80.70%**
ImageNet	(Yoo and Jeong 2023)	ResNet-18	Direct train	4	54.34%
		ResNet-34			60.10%
	(Hu, Zheng, and Pan 2024)	ResNet-18	Direct train	1	62.06%
	AGMM	ResNet-18	Direct train	4	**64.67%**
DVS-Gesture	(Pei et al. 2023)	5Conv1FC	Direct train	20	94.63%
	(Qiao et al. 2021)	2Conv2FC	Direct train	150	97.57%
	(Yoo and Jeong 2023)	15Conv1FC	Direct train	16	97.57%
	(Wei et al. 2024a)	VGGSNN	Direct train	16	97.92%
	AGMM	VGGSNN	Direct train	16	**97.92%**
DVS-CIFAR10	(Qiao et al. 2021)	2Conv2FC	Direct train	25	62.10%
	(Pei et al. 2023)	5Conv1FC	Direct train	10	68.98%
	(Yoo and Jeong 2023)	16Conv1FC	Direct train	16	74.70%
	(Wei et al. 2024a)	VGGSNN	Direct train	10	81.60%
	AGMM	VGGSNN	Direct train	10	**82.40%**

Table 1: Performance comparison on both static and neuromorphic datasets.

achieves 80.25% accuracy with only 2.46MB, outperforming Q-SNN (Wei et al. 2024a) by 1.13% in accuracy with a minimal increase in parameters. These experiments demonstrate that our method significantly enhances the accuracy of BSNN while maintaining BSNN's efficiency.

Ablation study

In this section, we conduct ablation experiments to prove that the proposed AGMM mitigate the frequent weight sign flipping problem by regulating the gradient of BSNNs. And we evaluate AGMM-BSNN's energy-efficient advantages compared to vanilla BSNN and FP-SNN. All ablation experiments are performed on the CIFAR-100 dataset with the architecture of ResNet-19.

Efficacy of AGMM We first visualized the weight sign flipping ratio and performance between AGMM-BSNN, vanilla-BSNN, and FP-SNN to validate the efficacy of our proposed AGMM. As shown in Fig. 6(a), where the x-

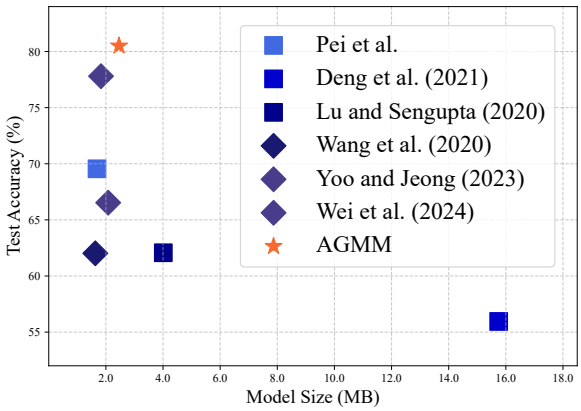


Figure 5: Comparison of test accuracy and model size of AGMM and other methods on CIFAR-100.

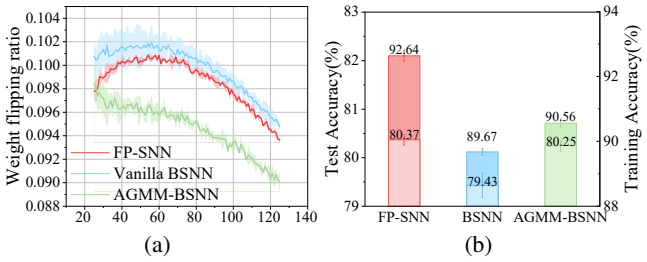


Figure 6: Weight sign flipping ratio and performance of FP-SNN, vanilla BSNN and AGMM-BSNN. The darker color represents the training accuracy, while the lighter color indicates the test accuracy.

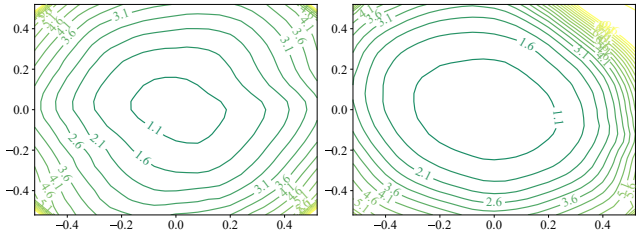


Figure 7: The loss landscape of vanilla BSNN and AGMM-BSNN. AGMM-BSNN performs better in terms of generalization around the local minima.

axis denotes the number of epochs, the flipping ratio of AGMM-BSNN is generally lower than that of the vanilla BSNN during the training process. Furthermore, Fig. 6(b) demonstrates that AGMM-BSNN significantly outperforms the vanilla BSNN in terms of accuracy. Although AGMM-BSNN's training accuracy is lower than that of FP-SNN, its test accuracy is comparable to FP-SNN, demonstrating the generalization capability of AGMM-BSNN. We further inspect the 2D landscapes of AGMM-BSNN and vanilla BSNN around their local minima to demonstrate why AGMM-BSNN generalizes better than vanilla BSNN. As shown in Fig. 7, AGMM-BSNN converges to a lower loss compared to vanilla BSNN, with a flatter convergence region. Moreover, AGMM-BSNN exhibits lower loss in areas distant from the convergence region. These observations indicate that AGMM enhances BSNN's convergence and improves its generalization performance. Based on the aforementioned results, it can be deduced that AGMM enhances BSNN's performance by decreasing weight sign flipping frequency, hence improving convergence.

Efficiency of AGMM Although the proposed AGMM introduces a small number of learnable parameters, it does not compromise the efficiency of the BSNN. To demonstrate this, we first compare the firing rates of AGMM-BSNN and vanilla-BSNN. As shown in Fig. 8, since AGMM compresses the input to the spiking neurons during the forward pass via the sigmoid function, resulting in a lower firing rate for AGMM-BSNN. This phenomenon is evident in

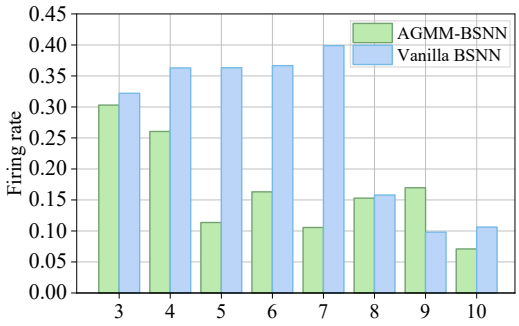


Figure 8: Firing rate in convolution layer of ResNet-19.

Model	Size (MB)	SOPs (G)	MACs (G)	Energy (mJ)
AGMM-BSNN	2.46	0.045	0.028	**0.064**
Vanilla BSNN	2.31	0.056	0.027	0.066
FP-SNN	50.46	0.750	0.027	0.330

Table 2: Energy estimation.

the middle layers of the network. Additionally, we calculate the SOPs and power consumption of AGMM, following the methodology outlined in (Yao et al. 2024), as shown in Tab. 2. While our method introduces additional parameters compared to the vanilla BSNN, the reduced firing rate results in lower SOP and energy consumption for AGMM, as compared to the vanilla BSNN.

Conclusion

In this paper, we provide a comprehensive analysis of the weight sign flipping problem during the learning process of BSNNs and introduce an adaptive gradient modulation mechanism to address this challenge. Our theoretical analysis shows that the frequency of weight sign flipping is positively correlated with the mean and variance of gradients during training. To mitigate this issue, AGMM adaptively modulates the gradient magnitude, thereby reducing the mean and variance metrics in BSNNs. Extensive experiments on both static and neuromorphic datasets demonstrate that AGMM-BSNN consistently outperforms existing quantized SNN methods. These results underscore the effectiveness of the AGMM approach in enhancing the performance of BSNNs while maintaining their inherent energy efficiency advantages, making them more suitable for deployment in resource-constrained environments. In our future work, we will explore extending AGMM to complex large-scale SNN architectures, such as large spiking language models, and investigating its applicability to more complex tasks.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under grant U20B2063, 62220106008, and 62106038, the Sichuan Science and Technology Program under Grant 2024NSFTD0034 and 2023YFG0259, the Open Research Fund of the State Key

Laboratory of Brain-Machine Intelligence, Zhejiang University (Grant No.BMI2400020).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Davies, M.; Srinivasa, N.; Lin, T.-H.; Chinya, G.; Cao, Y.; Choday, S. H.; Dimou, G.; Joshi, P.; Imam, N.; Jain, S.; et al. 2018. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1): 82–99.

Deng, L.; Wu, Y.; Hu, Y.; Liang, L.; Li, G.; Hu, X.; Ding, Y.; Li, P.; and Xie, Y. 2021. Comprehensive snn compression using admm optimization and activity regularization. *IEEE transactions on neural networks and learning systems*, 34(6): 2791–2805.

Deng, S.; Li, Y.; Zhang, S.; and Gu, S. 2022. Temporal efficient training of spiking neural network via gradient reweighting. *arXiv preprint arXiv:2202.11946*.

Gholami, A.; Kim, S.; Dong, Z.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2022. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, 291–326. Chapman and Hall/CRC.

Ghosh-Dastidar, S.; and Adeli, H. 2009. Spiking neural networks. *International journal of neural systems*, 19(04): 295–308.

Glorot, X.; and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249–256. JMLR Workshop and Conference Proceedings.

Gong, Y.; Liu, L.; Yang, M.; and Bourdev, L. 2014. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

Guo, Y.; Huang, X.; and Ma, Z. 2023. Direct learning-based deep spiking neural networks: a review. *Frontiers in Neuroscience*, 17: 1209795.

Guo, Y.; Tong, X.; Chen, Y.; Zhang, L.; Liu, X.; Ma, Z.; and Huang, X. 2022a. RecDis-SNN: Rectifying Membrane Potential Distribution for Directly Training Spiking Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 326–335.

Guo, Y.; Zhang, L.; Chen, Y.; Tong, X.; Liu, X.; Wang, Y.; Huang, X.; and Ma, Z. 2022b. Real spike: Learning real-valued spikes for spiking neural networks. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XII*, 52–68. Springer.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. 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*, 630–645. Springer.

Helwegen, K.; Widdicombe, J.; Geiger, L.; Liu, Z.; Cheng, K.-T.; and Nusselder, R. 2019. Latent weights do not exist: Rethinking binarized neural network optimization. *Advances in neural information processing systems*, 32.

Hodgkin, A. L.; and Huxley, A. F. 1952. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4): 500.

Hu, Y.; Zheng, Q.; and Pan, G. 2024. BitSNNs: Revisiting Energy-efficient Spiking Neural Networks. *IEEE Transactions on Cognitive and Developmental Systems*.

Jang, H.; Skatchkovsky, N.; and Simeone, O. 2021. BiSNN: training spiking neural networks with binary weights via bayesian learning. In *2021 IEEE Data Science and Learning Workshop (DSLW)*, 1–6. IEEE.

Kheradpisheh, S. R.; Mirsadeghi, M.; and Masquelier, T. 2022. Bs4nn: Binarized spiking neural networks with temporal coding and learning. *Neural Processing Letters*, 54(2): 1255–1273.

Li, H.; Xu, Z.; Taylor, G.; Studer, C.; and Goldstein, T. 2018. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31.

Li, Y.; Dong, X.; and Wang, W. 2019. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. *arXiv preprint arXiv:1909.13144*.

Li, Y.; Xu, Q.; Shen, J.; Xu, H.; Chen, L.; and Pan, G. 2024. Towards Efficient Deep Spiking Neural Networks Construction with Spiking Activity based Pruning. *arXiv preprint arXiv:2406.01072*.

Liu, H.; Chen, Y.; Zeng, Z.; Zhang, M.; and Qu, H. 2023. A low power and low latency FPGA-based spiking neural network accelerator. In *2023 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Liu, Q.; Yan, J.; Zhang, M.; Pan, G.; and Li, H. 2024a. LitE-SNN: Designing Lightweight and Efficient Spiking Neural Network through Spatial-Temporal Compressive Network Search and Joint Optimization. *arXiv preprint arXiv:2401.14652*.

Liu, Y.; Zhang, K.; Li, Y.; Yan, Z.; Gao, C.; Chen, R.; Yuan, Z.; Huang, Y.; Sun, H.; Gao, J.; et al. 2024b. Sora: A review on background, technology, limitations, and opportunities of large vision models. *arXiv preprint arXiv:2402.17177*.

Liu, Z.; Wu, B.; Luo, W.; Yang, X.; Liu, W.; and Cheng, K.-T. 2018. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *Proceedings of the European conference on computer vision (ECCV)*, 722–737.

Lu, S.; and Sengupta, A. 2020. Exploring the connection between binary and spiking neural networks. *Frontiers in neuroscience*, 14: 540201.

Pan, Z.; Zhang, M.; Wu, J.; Wang, J.; and Li, H. 2021. Multi-tone phase coding of interaural time difference for sound source localization with spiking neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29: 2656–2670.

Pei, Y.; Xu, C.; Wu, Z.; and Yang, Y. 2023. ALBSNN: ultra-low latency adaptive local binary spiking neural network with accuracy loss estimator. *Frontiers in Neuroscience*, 17: 1225871.

Qiao, G.; Ning, N.; Zuo, Y.; Hu, S.; Yu, Q.; and Liu, Y. 2021. Direct training of hardware-friendly weight binarized spiking neural network with surrogate gradient learning towards spatio-temporal event-based dynamic data recognition. *Neurocomputing*, 457: 203–213.

Qin, H.; Ding, Y.; Zhang, M.; Yan, Q.; Liu, A.; Dang, Q.; Liu, Z.; and Liu, X. 2022. Bibert: Accurate fully binarized bert. *arXiv preprint arXiv:2203.06390*.

Qin, H.; Ma, X.; Zheng, X.; Li, X.; Zhang, Y.; Liu, S.; Luo, J.; Liu, X.; and Magno, M. 2024. Accurate LoRA-Finetuning Quantization of LLMs via Information Retention. *arXiv preprint arXiv:2402.05445*.

Rastegari, M.; Ordonez, V.; Redmon, J.; and Farhadi, A. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, 525–542. Springer.

Roy, D.; Chakraborty, I.; and Roy, K. 2019. Scaling deep spiking neural networks with binary stochastic activations. In *2019 IEEE International Conference on Cognitive Computing (ICCC)*, 50–58. IEEE.

Shi, X.; Ding, J.; Hao, Z.; and Yu, Z. 2023. Towards Energy Efficient Spiking Neural Networks: An Unstructured Pruning Framework. In *The Twelfth International Conference on Learning Representations*.

Tang, H.; Gu, P.; Wijekoon, J.; Alsakkal, M. A.; Wang, Z.; Shen, J.; Yan, R.; and Pan, G. 2024. Neuromorphic auditory perception by neural spiketrum. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need.(Nips), 2017. *arXiv preprint arXiv:1706.03762*, 10: S0140525X16001837.

Wang, S.; Zhang, D.; Belatreche, A.; Xiao, Y.; Qing, H.; We, W.; Zhang, M.; and Yang, Y. 2024a. Ternary Spike-based Neuromorphic Signal Processing System. *arXiv preprint arXiv:2407.05310*.

Wang, S.; Zhang, D.; Shi, K.; Wang, Y.; Wei, W.; Wu, J.; and Zhang, M. 2024b. Global-Local Convolution with Spiking Neural Networks for Energy-efficient Keyword Spotting. *arXiv preprint arXiv:2406.13179*.

Wang, Y.; Xu, Y.; Yan, R.; and Tang, H. 2020. Deep spiking neural networks with binary weights for object recognition. *IEEE Transactions on Cognitive and Developmental Systems*, 13(3): 514–523.

Wei, W.; Liang, Y.; Belatreche, A.; Xiao, Y.; Cao, H.; Ren, Z.; Wang, G.; Zhang, M.; and Yang, Y. 2024a. Q-SNNs: Quantized Spiking Neural Networks. *arXiv preprint arXiv:2406.13672*.

Wei, W.; Zhang, M.; Qu, H.; Belatreche, A.; Zhang, J.; and Chen, H. 2023. Temporal-coded spiking neural networks with dynamic firing threshold: Learning with event-driven backpropagation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10552–10562.

Wei, W.; Zhang, M.; Zhang, J.; Belatreche, A.; Wu, J.; Xu, Z.; Qiu, X.; Chen, H.; Yang, Y.; and Li, H. 2024b. Event-Driven Learning for Spiking Neural Networks. *arXiv preprint arXiv:2403.00270*.

Wu, J.; Chua, Y.; Zhang, M.; Li, H.; and Tan, K. C. 2018a. A spiking neural network framework for robust sound classification. *Frontiers in neuroscience*, 12: 836.

Wu, J.; Xu, C.; Han, X.; Zhou, D.; Zhang, M.; Li, H.; and Tan, K. C. 2021. Progressive tandem learning for pattern recognition with deep spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11): 7824–7840.

Wu, Y.; Deng, L.; Li, G.; and Shi, L. 2018b. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12: 323875.

Wu, Y.; Deng, L.; Li, G.; Zhu, J.; Xie, Y.; and Shi, L. 2019. Direct training for spiking neural networks: Faster, larger, better. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 1311–1318.

Xu, Z.; Lin, M.; Liu, J.; Chen, J.; Shao, L.; Gao, Y.; Tian, Y.; and Ji, R. 2021. Recu: Reviving the dead weights in binary neural networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, 5198–5208.

Yan, J.; Liu, Q.; Zhang, M.; Feng, L.; Ma, D.; Li, H.; and Pan, G. 2024. Efficient spiking neural network design via neural architecture search. *Neural Networks*, 173: 106172.

Yao, M.; Hu, J.; Zhou, Z.; Yuan, L.; Tian, Y.; Xu, B.; and Li, G. 2024. Spike-driven transformer. *Advances in neural information processing systems*, 36.

Yin, R.; Li, Y.; Moitra, A.; and Panda, P. 2024. MINT: Multiplier-less INTeger Quantization for Energy Efficient Spiking Neural Networks. In *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 830–835. IEEE.

Yoo, D.; and Jeong, D. S. 2023. CBP-QSNN: Spiking Neural Networks Quantized Using Constrained Backpropagation. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 13(4): 1137–1146.

Zhang, G.; Wang, C.; Xu, B.; and Grosse, R. 2018. Three mechanisms of weight decay regularization. *arXiv preprint arXiv:1810.12281*.

Zhang, M.; Wang, J.; Wu, J.; Belatreche, A.; Amornpaisannon, B.; Zhang, Z.; Miriyala, V. P. K.; Qu, H.; Chua, Y.; Carlson, T. E.; et al. 2021. Rectified linear postsynaptic potential function for backpropagation in deep spiking neural networks. *IEEE transactions on neural networks and learning systems*, 33(5): 1947–1958.

Zhang, M.; Wu, J.; Chua, Y.; Luo, X.; Pan, Z.; Liu, D.; and Li, H. 2019. Mpd-al: an efficient membrane potential driven aggregate-label learning algorithm for spiking neurons. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 1327–1334.