ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10448202

OPTIMAL ANN-SNN CONVERSION WITH GROUP NEURONS

Liuzhenghao Lv[1] *Wei Fang*[2,3] *Li Yuan*[1,3] *Yonghong Tian*[1,2,3†]

[1]School of Electronic and Computer Engineering, Peking University, China
[2]School of Computer Science, Peking University, China
[3]Peng Cheng Laboratory, China

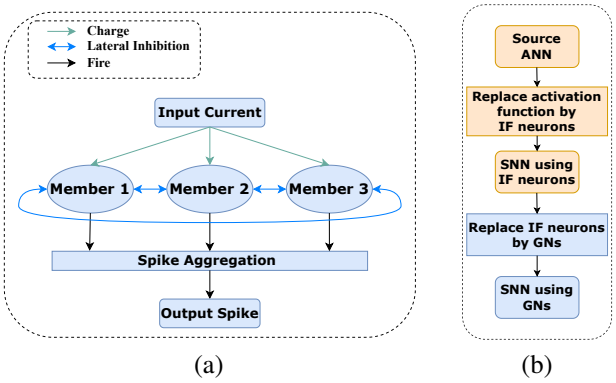
ABSTRACT

Spiking Neural Networks (SNNs) have emerged as a promising third generation of neural networks, offering unique characteristics such as binary outputs, high sparsity, and biological plausibility. However, the lack of effective learning algorithms remains a challenge for SNNs. For instance, while converting artificial neural networks (ANNs) to SNNs circumvents the need for direct training of SNNs, it encounters issues related to conversion errors and high inference time delays. In order to reduce or even eliminate conversion errors while decreasing inference time-steps, we have introduced a novel type of neuron called Group Neurons (GNs). One GN is composed of multiple Integrate-and-Fire (IF) neurons as members, and its neural dynamics are meticulously designed. Based on GNs, we have optimized the traditional ANN-SNN conversion framework. Specifically, we replace the IF neurons in the SNNs obtained by the traditional conversion framework with GNs. The resulting SNNs, which utilize GNs, are capable of achieving accuracy levels comparable to ANNs even within extremely short inference time-steps. The experiments on CIFAR10, CIFAR100, and ImageNet datasets demonstrate the superiority of the proposed methods in terms of both inference accuracy and latency. Code is available at `https://github.com/Lyu6PosHao/ANN2SNN_GN`.

***Index Terms*—** Spiking Neural Networks, Conversion, Spiking Neurons

1. INTRODUCTION

As a new generation of neural networks [1], spiking neural networks (SNNs) have attracted significant attention in the fields of artificial intelligence and computational neuroscience. The key feature of SNNs lies in the use of spiking neurons that closely resemble neurons in the human brain, endowing them with biological plausibility [2, 3]. Similar to real neurons, spiking neurons are activated and generate spikes only when their accumulated membrane potential surpasses a certain threshold. This means that spikes in SNNs are event-driven and highly sparse. Due to the differences between SNNs and artificial neural networks (ANNs), SNNs require specialized hardware in the form of neuromorphic hardware, which has already been researched and developed [4, 5, 6, 7]. When deployed on neuromorphic hardware, SNNs exhibit impressive capabilities in processing temporal information, achieving low power consumption, and high efficiency. However, developing efficient learning algorithms for SNNs has been a challenging task.



(a) (b)

Fig. 1. (a) A group neuron composed of three member neurons. (b) Comparison between the traditional ANN-SNN conversion framework and our optimized ANN-SNN conversion framework. In the traditional ANN-SNN conversion framework, the activation functions of the source ANN are typically replaced with IF neurons. Our optimized ANN-SNN conversion framework takes this a step further by replacing IF neurons in the SNN with group neurons.

While the SNN learning algorithms based on direct training often employ surrogate gradients to circumvent the non-differentiability issue of spiking neurons and utilize spatio-temporal back propagation (STBP) for training [8, 9, 10, 11], they typically require significant computational resources and may yield suboptimal model accuracy. In contrast, conversion-based SNN learning methods avoid the direct training of SNNs [12] and tend to achieve higher accuracy compared to directly trained SNNs. However, due to the clipping error, quantization error, and unevenness error intro-

This work is supported by the National Natural Science Foundation of China (No.62027804, 62332002, 62202014).
† Corresponding author (yhtian@pku.edu.cn).

duced during conversion [13], the accuracy of the converted SNNs is lower than that of the source ANN. This accuracy loss is pronounced, particularly when SNN inference time-steps are not enough. Thus, reducing or even eliminating accuracy loss in ANN-SNN conversion, especially in cases of limited inference time-steps, is crucial for the widespread adoption of SNNs.

Reducing accuracy loss, especially in scenarios with limited SNN inference time-steps, is challenging [14]. Existing ANN-SNN conversion frameworks commonly replace the activation functions in ANNs with Integrate-and-Fire (IF) neurons to obtain SNNs [14, 15, 16, 17, 18]. This practice relies on the mapping between the ANN activation function values and the firing rates of IF neurons. However, IF neurons have limited expressive capacity, making it difficult to achieve precise mapping within limited time-steps. [19] and [20] proposed improvements to the IF neurons, but the performances are not good enough.

Alternatively, we propose a novel neuron type called Group Neurons (GNs), as shown in Figure 1 (a), composed of multiple member neurons with unique neural dynamics that endow it with significantly greater expressive capacity than IF neurons. The firing rates of GNs can accurately map ANN activation values within limited inference time-steps.

Based on GNs, we optimize existing ANN-SNN conversion frameworks. While existing frameworks typically replace ANN activation functions with IF neurons, our method takes it a step further by replacing the IF neurons in the resulting SNN with GNs, as shown in Figure 1 (b). Our experiments demonstrate that SNN using GNs can achieve almost the same accuracy as that of the original ANN, even in extremely limited time-steps (e.g., 2 time-steps).

2. METHODS

In Section 2.1, we introduce the existing ANN-SNN conversion framework and its drawbacks. In Section 2.2, we propose a novel type of spiking neuron called Group Neurons (GNs). And in Section 2.3 we use GNs to optimize the existing ANN-SNN conversion framework.

2.1. Existing ANN-SNN Conversion Framework

In this section, we introduce IF neurons, the existing ANN-SNN conversion frameworks and their drawbacks.

The Integrate-and-Fire (IF) Neuron is the simplest type of spiking neuron, which is widely utilized in ANN-SNN conversion methods. This is due to the establishment of a mapping relationship between the firing rates of the IF neurons of SNN and the activation values of ANN. The dynamical behavior of the IF neuron can be described as follows:

$$p^l(t) = v^l(t-1) + W^l x^l(t) \tag{1}$$

$$s^l(t) = \text{Heaviside}(p^l(t) - \theta^l) \tag{2}$$

$$v^l(t) = p^l(t) - s^l(t)\theta^l \tag{3}$$

Here, Equation 1 describes the charging process of the IF neuron, where $v^l(t-1)$ denotes the membrane potential of the neuron in layer l at time-step $t-1$ and W^l denotes the synaptic connection weights between layer $l-1$ and layer l. $x^l(t)$ denotes the input current to the neuron in layer l at time-step t, and $p^l(t)$ denotes the membrane potential of the neuron in layer l just before spike firing at time-step t.

Equation 2 describes spike firing of the IF neuron, where the Heaviside$(\cdot)$ function outputs 0 if its input is negative and 1 otherwise. θ^l denotes the threshold of the neuron in layer l, and $s^l(t)$ denotes the output spike of the neuron in layer l.

Equation 3 describes the process of membrane potential reset in the neuron. We adopt soft reset mechanism, as it helps reduce the conversion error in ANN-SNN conversion [14, 15]. Soft reset implies that if the neuron fires, the membrane potential will be subtracted by the threshold value.

As mentioned earlier, in ANN-SNN conversion, we are interested in the firing rate of the IF neuron (or the average postsynaptic potential). We can combine equations 1 and 3 as follows:

$$v^l(t) = v^l(t-1) + W^l x^l(t) - s^l(t)\theta^l \tag{4}$$

By summing up the above equation from $t=1$ to $t=T$ and dividing both sides of the equation by T, we obtain:

$$\frac{v^l(T) - v^l(0)}{T} = \frac{W^l \sum_{t=1}^{T} x^l(t)}{T} - \frac{\sum_{t=1}^{T} s^l(t)\theta^l}{T} \tag{5}$$

We assume that if the neuron in layer $(l-1)$ fires, the input received by the neuron in the layer l is given by the postsynaptic potential θ^{l-1}, that is, $x^l(t) = s^{l-1}(t)\theta^{l-1}$. We substitute this into Equation 5, and replace $\frac{\sum_{t=1}^{T} s^l(t)\theta^l}{T}$ with $\phi^l(T)$:

$$\phi^l(T) = W^l \phi^{l-1}(T) + (-\frac{v^l(T) - v^l(0)}{T}) \tag{6}$$

Since $\phi^l(T) \geq 0$, $\phi^l(T)$ can be mapped to the activation value a^l of the ReLU activation function, with an associated mapping error of $|\frac{v^l(T)-v^l(0)}{T}|$. As evident, the fewer the inference time-steps T, the larger the mapping error (conversion error).

The quantization clip-floor-shift (QCFS) activation function was introduced by [13], which is used to reduce the conversion error and is defined in Equation 7. Training an ANN with the QCFS activation function instead of ReLU, and subsequently converting it into an SNN using IF neurons, has proven to be effective in reducing conversion errors during short inference time-steps. We use QCFS as the activation function in ANN due to its outstanding performance.

$$a^l = f(a^{l-1}) = \lambda^l clip(\frac{1}{L}\lfloor \frac{W^l a^{l-1} L}{\lambda^l} + 0.5 \rfloor, 0, 1) \tag{7}$$

Here, λ^l denotes the trainable threshold of ANN outputs of layer l, which is mapped to the threshold of IF neurons in SNN layer l. And L denotes the quantization level of ANN outputs.

However, even with the QCFS activation function, ANN-SNN conversion errors persist under very short inference time-steps due to the use of IF neurons, which have limited expressive capacity. For instance, accurately mapping an activation value of 0.125 to IF neuron firing rates requires at least 8 inference time-steps. In these 8 time-steps, the IF neuron must fire once while staying silent in the rest. Fewer than 8 time-steps lead to inevitable conversion errors.

In order to achieve nearly lossless ANN-SNN conversion in extremely short inference time-steps, it is imperative to design spiking neurons with greater expressive capacity. Therefore, we have proposed Group Neurons (GNs) to replace IF neurons.

2.2. Group Neuron

One GN is composed of τ IF neurons as members. Assuming the threshold of the IF neuron to be replaced is θ^l, the threshold of the i-th member neuron within the GN is $\theta_i^l = \frac{i\theta^l}{\tau}$. As a whole, the GN has its own threshold, denoted as $\theta_{GN}^l = \frac{\theta^l}{\tau}$. Each member neuron possesses an identical initial membrane potential.

The neural dynamics of the GN are as follows:

$$p^l(t) = v^l(t-1) + W^l x^l(t) \tag{8}$$

$$s_i^l(t) = Heaviside(p^l(t) - \theta_i^l) \quad (\forall i = 1, 2, \cdots, \tau) \tag{9}$$

$$v^l(t) = p^l(t) - \theta_{GN}^l \sum_{i=1}^{\tau} s_i^l(t) \tag{10}$$

$$s_{GN}^l(t) = \sum_{i=1}^{\tau} s_i^l(t) \tag{11}$$

Here, we denote the membrane potentials before spike firing of all member neurons in layer l as $p^l(t)$ since they are equivalent. For the same reason, we use $v^l(t)$ to denote the membrane potentials after spike firing of all member neurons. Equation 8 represents the charging process of each member neuron, with their charging currents being equal. Equation 9 describes the firing process of each member neuron i, with distinct thresholds as previously mentioned. Equation 10 describes the soft reset process, with a key distinction being that if a member neuron fires, it not only resets its own membrane potential by subtracting θ_{GN}^l but also resets the membrane potentials of all other members by subtracting θ_{GN}^l. Therefore, we can also conceptually refer to this process as lateral inhibition among member neurons. The process described in Equation 11 is termed spike aggregation. It aggregates the spikes fired by all member neurons at the current time-step to compute $s_{GN}^l(t)$, which serves as the output of the GN at the current time-step.

Fig. 2. Comparison between the firing rates of the IF neuron and the GN (τ=4), at the same time-step setting (T=4).

Based on equations 8, 9, 10, 11, following the derivation process in Section 2.1, we can also obtain an equation for the GN, similar to Equation 6 for the IF neuron:

$$\phi_{GN}^l(T) = W^l \phi_{GN}^{l-1}(T) + (-\frac{v^l(T) - v^l(0)}{T}) \tag{12}$$

Here, $\phi_{GN}^l(T) = \frac{\sum_{t=1}^{T} s_{GN}^l(t)\theta_{GN}^l}{T}$. Hence, the $\phi_{GN}^l(T)$ of the GN can also be mapped to a^l of the ANN.

GNs exhibit greater expressive capacity than IF neurons, resulting in smaller mapping errors. We plot the curves of the firing rate as a function of the input current x for the IF neuron and the GN, as shown in Figure 2. Here, the number of inference time steps is set to 4, τ is set to 4 in the GN , and the threshold of the IF neuron is 1.0. It can be observed that although both GN and IF neuron curves exhibit a step-like pattern, the GN curve is more detailed and closely resembles the activation function curve in ANN when compared to the IF neuron. By derivation, the width of one step of the IF neuron is $\frac{\theta^l}{T}$, while that of the GN is $\frac{\theta_{GN}^l}{T}$ where $\theta_{GN}^l = \frac{\theta^l}{\tau}$.

2.3. Optimal ANN-SNN Conversion Framework

Based on GNs, we have optimized the existing ANN-SNN conversion framework as shown in Figure 1. The conventional method typically involves replacing the activation functions in the source ANN with IF neurons to obtain an SNN as the conversion result. However, we have taken it a step further by replacing all the IF neurons in this SNN with GNs as shown in Algorithm 1, resulting in an SNN that utilizes GNs as the final conversion result.

3. EXPERIMENTS

3.1. Effectiveness of GN

We train a ResNet-20 on CIFAR-100 and converted it into two types of SNNs: one using IF neurons and the other using

Algorithm 1 Algorithm for replacing original IF neurons in SNN with GNs

Input: SNN model $M_{\text{SNN}}(\boldsymbol{x}; \boldsymbol{W})$ using IF neurons; Hyper parameter in GNs τ.

Output: $M_{\text{SNN}}(\boldsymbol{x}; \boldsymbol{W})$ using GNs

for $l = 1$ to M_{SNN}.layers **do**
  $M_{\text{SNN}}.\theta^l_{GN} \leftarrow M_{\text{SNN}}.\theta^l / \tau$
  for $i = 1$ to τ **do**
    $M_{\text{SNN}}.\theta^l_i \leftarrow M_{\text{SNN}}.\theta^l * i / \tau$
  end for
end for
return M_{SNN}

	T=1	T=2	T=4	T=8	T=16	T=32
SNN using GN	1.39	0.64	0.37	0.3	0.28	0.28
SNN using IF	11.65	6.36	3.72	2.02	0.92	0.45
ratio	11.9%	10.1%	9.9%	14.9%	30.4%	62.2%

Table 1. MSE of outputs between the last layer of ANN and the last layer of SNN at different time-steps.

GNs. We calculate the mean squared error (MSE) between the final layer outputs of both SNNs and the source ANN to gauge the magnitude of the conversion error. As shown in the Table 1, the SNN using IF neurons resulted in a significantly larger MSE than that using GNs. This indicates a substantial reduction in conversion error when replacing IF neurons with GNs. Furthermore, this reduction in conversion error is more pronounced within short inference time-steps, as the ratio is lower at T=1, 2, and 4.

3.2. Comparison with State-of-the-art Methods

We compare our method with other SOTA methods on the CIFAR-10/100 and ImageNet datasets in Table 2, including QCFS [13], OPI [21], COS [22], SRP [23], RMP [15], and SNNC-AP [24]. On CIFAR-10, for both ResNet-18 and VGG-16, our method outperforms the other methods at the same time-step setting. On CIFAR-100, for ResNet-20, our method outperforms the other methods within limited time-steps (T $\leq$ 8). For VGG-16, we achieve an accuracy almost the same with that of the source ANN with only 2 time-steps (76.36% vs 76.43%), which is 0.33% higher than COS (76.03%, T=2) and 12.57% higher than QCFS (63.79%, T=2). On ImageNet, we achieve an accuracy almost the same with that of the source ANN taking only 2 time-steps (73.61% vs 74.35%), while COS taking 4 time-steps (73.81% vs 74.22%) and QCFS taking more than 32 time-steps. Note that COS requires 8 additional time-steps for calibrating [22].

3.3. Effect of the Number of Member Neurons

Figure 3 illustrates the change in the accuracy for different values of τ. It is evident that increasing τ significantly improves the accuracy when the time-steps are limited (T $\leq$ 4).

Arch	Method	ANN	T=1	T=2	T=4	T=8	T=16	T=32
CIFAR-10 Dataset								
ResNet-18	QCFS	95.64%	88.84%	91.75%	93.83%	95.04%	95.56%	95.67%
	OPI	96.04%	-	-	-	75.44%	90.43%	94.82%
	COS	95.64%	95.25%	95.45%	95.46%	95.66%	95.68%	95.68%
	SRP	95.64%	94.59%	95.06%	95.25%	95.60%	95.55%	95.55%
	SNNC-AP	95.46%	-	-	-	-	-	94.78%
	***Ours*($\tau = 4$)**	96.48%	**96.01%**	**96.36%**	**96.57%**	**96.56%**	**96.50%**	**96.46%**
VGG-16	QCFS	95.52%	88.41%	91.18%	93.96%	94.95%	95.40%	95.54%
	OPI	94.57%	-	-	-	90.96%	93.38%	94.20%
	COS	95.51%	94.90%	95.36%	95.46%	95.51%	95.57%	95.61%
	SRP	95.52%	93.80%	94.47%	95.32%	95.52%	95.44%	95.42%
	SNNC-AP	95.72%	-	-	-	-	-	93.71%
	***Ours*($\tau = 4$)**	96.02%	**95.63%**	**95.84%**	**95.94%**	**96.00%**	**96.02%**	**95.97%**
CIFAR-100 Dataset								
ResNet-20	QCFS	69.94%	-	19.96%	34.14%	55.37%	67.33%	69.82%
	OPI	70.43%	-	-	-	23.09%	52.34%	67.18%
	COS	69.97%	59.22%	64.21%	65.18%	67.17%	**69.44%**	**70.29%**
	SRP	69.94%	46.48%	53.96%	59.34%	62.94%	64.71%	65.50%
	RMP	68.72%	-	-	-	-	-	27.64%
	***Ours*($\tau = 4$)**	68.41%	**63.89%**	**67.60%**	**68.87%**	**69.38%**	69.15%	69.18%
VGG-16	QCFS	76.28%	-	63.79%	69.62%	73.96%	76.24%	77.01%
	OPI	76.31%	-	-	-	60.49%	70.72%	74.82%
	COS	76.28%	74.24%	76.03%	76.26%	76.52%	76.77%	**76.96%**
	SRP	76.28%	71.52%	74.31%	75.42%	76.25%	76.42%	76.45%
	SNM	74.13%	-	-	-	-	-	71.80%
	SNNC-AP	77.89%	-	-	-	-	-	73.55%
	***Ours*($\tau = 4$)**	76.43%	**75.61%**	**76.36%**	**76.60%**	**76.85%**	**76.80%**	76.69%
ImageNet Dataset								
ResNet-34	SNNC-AP	75.36%	-	-	-	-	-	64.54%
	QCFS	74.32%	-	-	-	35.06%	59.35%	69.37%
	COS	74.22%	69.11%	72.66%	**73.81%**	**74.17%**	**74.14%**	**73.93%**
	SRP	74.32%	57.78%	64.32%	66.71%	67.62%	68.02%	68.40%
	***Ours*($\tau = 6$)**	74.35%	**71.46%**	**73.61%**	73.73%	73.57%	73.51%	73.46%

Table 2. Comparison with existing state-of-the-art ANN-SNN conversion methods

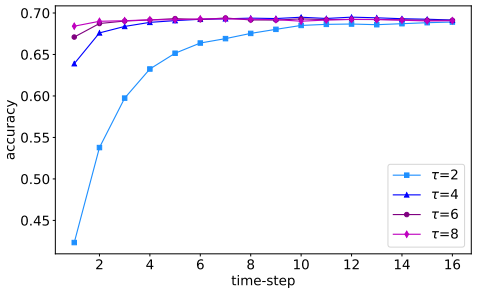


Fig. 3. Effect of different τ. ResNet-20 on CIFAR100.

When there are more time-steps available (T $>$ 8), the improvement is no longer as pronounced.

4. CONCLUSION

In this paper, we introduce GNs with enhanced expressive capabilities and optimize the existing ANN-SNN conversion frameworks. Instead of converting the source ANN into an SNN using IF neurons, we convert it into an SNN using GNs. Our method demonstrates excellent performance on the CIFAR-10, CIFAR-100, and ImageNet datasets, achieving low-latency, high-accuracy SNNs. Our method will contribute to the widespread adoption of SNNs.

5. REFERENCES

[1] Wolfgang Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[2] Eugene M Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[3] Warren S McCulloch and Walter Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

[4] Michael V DeBole, Brian Taba, Arnon Amir, Filipp Akopyan, Alexander Andreopoulos, William P Risk, Jeff Kusnitz, Carlos Ortega Otero, Tapan K Nayak, Rathinakumar Appuswamy, et al., "TrueNorth: Accelerating from zero to 64 million neurons in 10 years," *Computer*, vol. 52, no. 5, pp. 20–29, 2019.

[5] 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*, vol. 38, no. 1, pp. 82–99, 2018.

[6] Jing Pei, Lei Deng, Sen Song, Mingguo Zhao, Youhui Zhang, Shuang Wu, Guanrui Wang, Zhe Zou, Zhenzhi Wu, Wei He, et al., "Towards artificial general intelligence with hybrid tianjic chip architecture," *Nature*, vol. 572, no. 7767, pp. 106–111, 2019.

[7] Bo Xu, Yuhao Huang, Yuetong Fang, Zhongrui Wang, Shaoliang Yu, and Renjing Xu, "Recent progress of neuromorphic computing based on silicon photonics: Electronic–photonic co-design, device, and architecture," in *Photonics*. MDPI, 2022, vol. 9, p. 698.

[8] Qinyu Cheni, Bodo Rueckauer, Li Li, Tobi Delbruck, and Shih-Chii Liu, "Reducing latency in a converted spiking video segmentation network," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2021, pp. 1–5.

[9] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Frontiers in neuroscience*, vol. 12, pp. 331, 2018.

[10] 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*, vol. 34, pp. 21056–21069, 2021.

[11] Lang Feng, Qianhui Liu, Huajin Tang, De Ma, and Gang Pan, "Multi-level firing with spiking ds-resnet: Enabling better and deeper directly-trained spiking neural networks," *arXiv preprint arXiv:2210.06386*, 2022.

[12] Yongqiang Cao, Yang Chen, and Deepak Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *International Journal of Computer Vision*, vol. 113, pp. 54–66, 2015.

[13] Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, and Tiejun Huang, "Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks," *arXiv preprint arXiv:2303.04347*, 2023.

[14] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in neuroscience*, vol. 11, pp. 682, 2017.

[15] Bing Han, Gopalakrishnan Srinivasan, and Kaushik Roy, "Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13558–13567.

[16] Shikuang Deng and Shi Gu, "Optimal conversion of conventional artificial neural networks to spiking neural networks," *arXiv preprint arXiv:2103.00476*, 2021.

[17] 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.

[18] Nguyen-Dong Ho and Ik-Joon Chang, "Tcl: an ann-to-snn conversion with trainable clipping layers," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 793–798.

[19] Yang Li and Yi Zeng, "Efficient and accurate conversion of spiking neural network with burst spikes," *arXiv preprint arXiv:2204.13271*, 2022.

[20] Yuchen Wang, Malu Zhang, Yi Chen, and Hong Qu, "Signed neuron with memory: Towards simple, accurate and high-efficient ann-snn conversion," in *International Joint Conference on Artificial Intelligence*, 2022.

[21] Tong Bu, Jianhao Ding, Zhaofei Yu, and Tiejun Huang, "Optimized potential initialization for low-latency spiking neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, vol. 36, pp. 11–20.

[22] Zecheng Hao, Jianhao Ding, Tong Bu, Tiejun Huang, and Zhaofei Yu, "Bridging the gap between anns and snns by calibrating offset spikes," *arXiv preprint arXiv:2302.10685*, 2023.

[23] Zecheng Hao, Tong Bu, Jianhao Ding, Tiejun Huang, and Zhaofei Yu, "Reducing ann-snn conversion error through residual membrane potential," *arXiv preprint arXiv:2302.02091*, 2023.

[24] 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*. PMLR, 2021, pp. 6316–6325.