Signed Neuron with Memory: Towards Simple, Accurate and High-Efficient ANN-SNN Conversion

Yuchen Wang, **Malu Zhang**, **Yi Chen** and **Hong Qu***
School of Computer Science and Engineering,
University of Electronic Science and Technology of China
yuchenwang@std.uestc.edu.cn, maluzhang@uestc.edu.cn, chenyi@std.uestc.edu.cn,
hongqu@uestc.edu.cn

Abstract

Spiking Neural Networks (SNNs) are receiving increasing attention due to their biological plausibility and the potential for ultra-low-power event-driven neuromorphic hardware implementation. Due to the complex temporal dynamics and discontinuity of spikes, training SNNs directly usually suffers from high computing resources and a long training time. As an alternative, SNN can be converted from a pre-trained artificial neural network (ANN) to bypass the difficulty in SNNs learning. However, the existing ANN-to-SNN methods neglect the inconsistency of information transmission between synchronous ANNs and asynchronous SNNs. In this work, we first analyze how the asynchronous spikes in SNNs may cause conversion errors between ANN and SNN. To address this problem, we propose a signed neuron with memory function, which enables almost no accuracy loss during the conversion process, and maintains the properties of asynchronous transmission in the converted SNNs. We further propose a new normalization method, named neuron-wise normalization, to significantly shorten the inference latency in the converted SNNs. We conduct experiments on challenging datasets including CIFAR10 (95.44% top-1), CIFAR100 (78.3% top-1) and ImageNet (73.16% top-1). Experimental results demonstrate that the proposed method outperforms the state-of-the-art works in terms of accuracy and inference time. The code is available at https://github.com/ppppps/ANN2SNNConversion_SNM_NeuronNorm.

1 Introduction

In recent years, Artificial Neural Networks (ANNs) surpass human performance in a variety of tasks, such as image processing, natural language processing, speech processing and mind sports [He *et al.*, 2015; Devlin *et al.*, 2018; Lam *et al.*, 2019; Silver *et al.*, 2018]. However, ANNs suffer from substantial computational costs, which limits their applications in edge computing and power-critical applications. Recently, The brain-inspired Spiking Neural Networks (SNNs) are proposed to offer a low-power alternative. Due to their sparse-asynchronous spikes and event-driven computation, SNNs are more suitable to implement on ultra-low-power neuromorphic hardware, such as TrueNorth, Loihi, and Tianjic [Akopyan *et al.*, 2015; Davies *et al.*, 2018; Pei *et al.*, 2019]. However, because of the complex temporal dynamics and discontinuity of spike activities, training SNNs efficiently remains an open question.

At present, there are two promising SNN training approaches, which are direct training SNNs from scratch and ANN-SNN conversion. Direct training SNNs needs to tackle the problem of non-differentiable binary activation function, popular methods are training SNNs with the surrogate gradient [Wu *et al.*, 2018; Neftci *et al.*, 2019] and synaptic plasticity [Kheradpisheh *et al.*, 2018]. However, their training procedure requires tremendous resources and suffers from unsatisfactory accuracy. ANN-SNN conversion methods directly map the parameters of a pre-trained ANN to SNN with same structure [Diehl *et al.*, 2015; Rueckauer *et al.*, 2017]. These methods not only bypass the non-differentiable issue of the spike activity, but also save training time and resources compared with the direct training rules. Therefore ANN-SNN conversion methods are promising to train large-scale SNNs. Although ANN-SNN conversion has made some progress, all the up-to-date methods are still much less mature, and the limited time steps ($<$256) will result in a remarkable accuracy drop. As a consequence, the existing converted SNNs need thousand of inference time steps to reach the level of ANNs in terms of accuracy, leading to the power consumption and latency of SNNs being higher than ANNs in applications. We believe that there are still covert errors in the conversion process that have not been discovered.

In this work, we solve two difficult problems to achieve near loss-less ultra low-latency ANN-SNN conversion. In particular, we meticulously analyze the difference between SNNs and ANNs, then found that asynchronous transmission characteristic of SNNs causes conversion error, and the information loss after parameter normalization results in long SNN inference latency. We propose optimal solutions and yield a conversion model that enables high accuracy and low latency. Experimental results on benchmark datasets manifest our model is better than existing state-of-the-art models

*Contact Author

in terms of accuracy and latency. The contributions of this work are summarized as follows:

- We first analyze that the different information transmission mechanisms between synchronous ANNs and asynchronous SNN may cause conversion errors. Then, we propose a signed neuron with memory function (SNM), which makes the firing rates of spiking neurons exactly equal to corresponding ReLU activation values and maintains the properties of asynchronous transmission in the converted SNNs.
- We find the existing layer-based parameter normalization methods lead to a long inference latency in the converted SNNs. To resolve this problem, we put forward a neuron-wise parameter normalization (NeuronNorm) strategy that can significantly reduce SNN inference latency.
- We conduct experiments on large-scale datasets such as CIFAR and ImageNet through popular deep neural network structures. Results demonstrate that the proposed model can surpass the accuracy of current state-of-the-art models with a significantly short inference time (less than 128 time steps).

2 Related Work

ANN-SNN conversion is in burgeoning research, which is first applied to object recognition in the work of Cao *et al.* [2015]. For the conversion of ANN to SNN, the most common techniques are divided into (1) soft-reset mechanism and (2) parameter normalization. These two technologies are usually used in combination. In order to retain the residual membrane potential above hard-reset neurons' firing thresholds, Rueckauer *et al.* [2016] applied the membrane potential reset-by-subtraction mechanism [Diehl *et al.*, 2016] to converted SNNs, which is also called soft-reset mechanism, leading to more accurate ANN-SNN conversion. Han *et al.* [2020] further verified its effectiveness in challenging classification tasks by adding constraints when training ANNs. Although soft-reset mechanism narrows the difference of ANN activation value and SNN firing rate, SNN is still plagued by the problem caused by spike asynchronous transmission.

The family of parameter normalization [Diehl *et al.*, 2015; Rueckauer *et al.*, 2016; Rueckauer *et al.*, 2017; Sengupta *et al.*, 2019] bridges the gap between spiking neuron firing rates and ReLU activation values. The parameter here refers to the weight and bias of networks, and normalization scales first adopted are maximum activation values of each layer in ANN. Rueckauer *et al.* [2017] analyzed 16,666 CIFAR10 samples and found that 99.9% ReLU activations are much smaller than the maximum activation value, which indicates maximum activation value is not a suitable scale. As a consequence, a lot of following works are studying how to find a better normalization scale. Kim *et al.* [2020] proposed Spiking-YOLO with channel-wise data-based normalization. Subsequently, Ding *et al.* [2021] presented Rate Norm Layer to replace the ReLU function, and obtain the scale through a gradient-based algorithm. Eventually, a layer-wise parameters calibration algorithm proposed by Li *et al.* [2021] becomes state-of-the-art model based on the work of Deng and Gu [2021]. However, all of them are complicated procedures vulnerable to high inference latency. Additionally, converting low-latency SNNs with Batch Normalization (BN) layers still remains an ongoing challenge.

3 Preliminaries

Soft-Reset Neuron. ReLU activation function is typically used in a large number of deep neural networks at present. Fortunately, the functional relationship between Integrate-Fire (IF) neurons input and output is similar to ReLU. The membrane potential $v_j^l(t)$ of IF neuron j in layer l at time step t described by:

$$\tilde{v}_j^l(t) = v_j^l(t-1) + \sum_i w_{ij} s_i^{l-1}(t) + b_j, \tag{1}$$

$$s_j^l(t) = \begin{cases} 1, & \tilde{v}_j^l(t) \geq \theta_j^l \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

$$v_j^l(t) = (1 - s_j^l(t))\tilde{v}_j^l(t) + s_j^l(t) v_{\text{rest}}, \tag{3}$$

where s_i^{l-1} is the spike from presynaptic neuron i and v_{rest} donates the resting potential. When $\tilde{v}_j^l(t)$ exceeds firing threshold θ_j^l, neuron j will send a spike $s_j^l(t)$ to its postsynaptic neuron. In terms of the hard-reset neuron, the membrane potential will immediately return to resting potential after firing a spike. Therefore, hard-reset neurons ignore residual potential at firing instants, resulting in an accuracy drop for the converted SNN. Soft-reset neurons avoid the above problem and are widely used in various models. For soft-reset neurons, Eq. (3) is described as:

$$v_j^l(t) = \tilde{v}_j^l(t) - s_j^l(t). \tag{4}$$

In this case, the soft-reset neuron will not return to resting potential after firing a spike, but maintains the residual potential above firing threshold.

Parameter normalization. According to the calculation of spiking neuron output rate, SNN can meet the accuracy of ANN only if there are plenty of time steps [Li *et al.*, 2021]. But this will cause the converted SNN to have more power consumption than ANN. Since the output rate of IF neurons cannot exceed 1, which requires the output range of ReLU function to be $[0, 1]$, it is needed to transform the parameters of ANN. Let us introduce the well-known layer-wise parameter normalization (LayerNorm) [Rueckauer *et al.*, 2017], which rescales all parameters by the maximum output value λ^l of layer l as follows:

$$W^l = W^l \frac{\lambda^{l-1}}{\lambda^l}, \quad b^l = \frac{b^l}{\lambda^l}. \tag{5}$$

Parameter normalization is equivalent to transmitting threshold value in SNN [Li *et al.*, 2021]. Since the firing rate of neurons in SNN cannot exceed 1, the firing threshold can be set to the maximum activation value of ANN. So Eq. (5) is also equivalent to directly modifying the firing threshold and spike value, take the soft-reset neuron as an example:

$$\theta_j^l = \lambda^l, \quad s_j^l(t) = \begin{cases} \theta_j^l, & v_j^l(t) \geq \theta_j^l \\ 0, & \text{otherwise} \end{cases}. \tag{6}$$

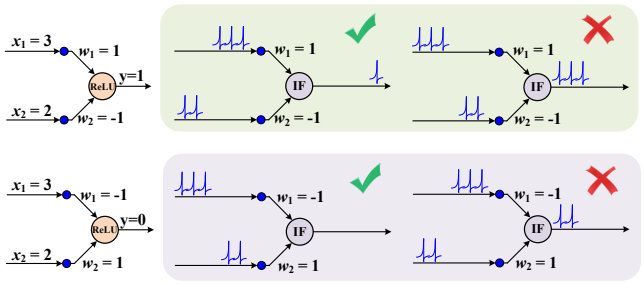

Figure 1: Explanation about SNN asynchronous transmission information error. Occurs when spikes input from the negative synapses arrive too late, which leads to a higher neuron output rate.

After replacing the neurons of ANN with soft-reset neurons, and setting the firing threshold of soft-reset neurons to maximum activation value of their corresponding layer, a pre-trained ANN is transformed into SNN that can be used directly. As for the input of SNN, which can be regarded as the input current of spiking neurons, we don't apply additional encoding.

4 Method

4.1 Signed Neuron with Memory

In this section, we first analyze the covert conversion error caused by asynchronous spikes. Then, we propose the new model, Signed Neuron with Memory (SNM), which can solve the above problem.

Error Analysis

In ANN, one neuron will adds the input simultaneously to get an activation value. But in SNN, since spikes may be scattered at any time steps along the time axis, the output rate of neurons may have unexpected variation. Through analysis, we summarize this error as: improper handling of the late arrival of spikes transmitted by negative weights leads to a higher output rate of spiking neurons, which makes ANN and converted SNN unable to correspond. Let us explain through the examples in Fig. 1.

Assuming that input data are $x_1 = 3$ and $x_2 = 2$, the example in the first row is that when corresponding weights $w_1 = 1, w_2 = -1$, the output of ANN neuron is 1. The example in the second row is that when corresponding weights $w_1 = -1, w_2 = 1$, the output of ANN neuron is 0. According to rate-based encoding, which encodes the data into a proportional number of spikes, we can assume that x_1 corresponds to three spikes and x_2 corresponds to two spikes. In addition, the resting potential of IF neuron is 0, and the firing threshold is 1.

Output spikes of neurons in a deep SNN may show various distributions on the time axis. We analyze two special cases, i.e., the early arrival or late arrival of spikes transmitted by the negative weight, which correspond to column two and column three in Fig. 1. For column two, when spikes transmitted by the negative weight arrive firstly, the neuron output rate can correspond to the output of ANN neuron. However, when spikes transmitted by the positive weight arrive firstly, the neuron will fire immediately, causing a high output rate.

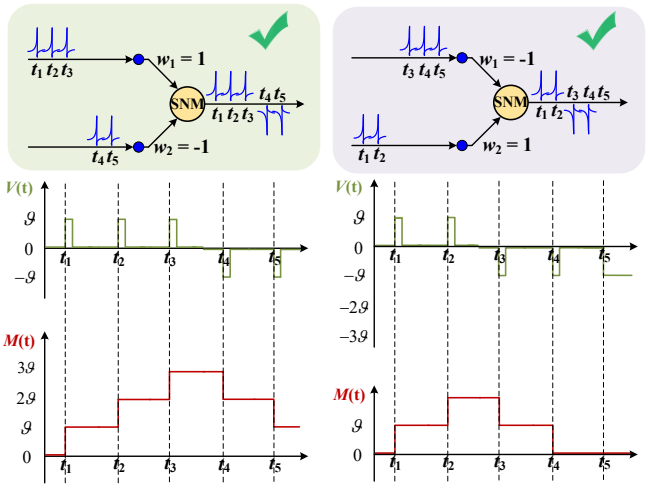

Figure 2: Exemplification of signed neuron with memory. $V(t)$ represents the membrane potential of the SNM neuron at t, and $M(t)$ represents the memory value of the SNM neuron at t, which equals to the sum of currently transmitted spikes.

For the SNN using rate-based encoding, although we can control the distribution of input spikes, the distribution of output spikes in hidden layers cannot be controlled, making the above error unavoidable.

The Proposed Solution: Signed Neuron with Memory

If all spikes are evenly distributed on the time axis, or the spikes transmitted by the negative synapse always reach the postsynaptic neurons earlier, then the above problem can be avoided. From this, it is easy to find that in hidden layers, we can shift all spikes to the first few time steps, or place spikes on equidistant time steps to make them evenly distributed. But the consequence is that we have to wait to ensure that all possible spikes of layer l are triggered before sending the spike trains to layer $l + 1$. In other words, the network only has the synchronous transmission ability.

A key contribution of this work is that we design a new neuron model named Signed Neuron with Memory (SNM) to address the above problem. The negative spike of SNM can enable SNN to maintain both asynchronous transmission and synchronous transmission capabilities while effectively solve the problem of excessively high output rate. And usage of memory mechanism ensures that SNM remembers the sum of its transmitted spikes (positive and negative spikes can be offset). Only when the memory value is greater than zero, negative spikes can be transmitted to the next layer. By this means, the dynamic of SNM neuron is described by:

$$\tilde{v}_j^l(t) = v_j^l(t-1) + \sum_i w_{ij} s_i^{l-1}(t) + b_j, \tag{7}$$

$$\tilde{m}_j^l(t) = m_j^l(t-1), \tag{8}$$

$$s_j^l(t) = \begin{cases} \theta_j^l, & \tilde{v}_j^l(t) \geq \theta_j^l \\ -\theta_j^l, & \tilde{v}_j^l(t) \leq -\theta_j^l \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad \tilde{m}_j^l(t) > 0\ , \tag{9}$$

$$v_j^l(t) = \tilde{v}_j^l(t) - s_j^l(t), \tag{10}$$

$$m_j^l(t) = \tilde{m}_j^l(t) + s_j^l(t), \tag{11}$$

where m_j^l is the memory value of neuron j in layer l.

Let us illustrate the SNM through Fig. 2. The examples in green and purple boxes respectively correspond to the error conditions of the same color box in Fig. 1. For the example in the green box, three spikes are input to an SNM neuron from synapse w_1 at t_1, t_2, and t_3 respectively. At t_3, the neuron membrane potential is 0, and the memory value is 3θ, which is the amount of all spikes that have been transmitted. At t_4, the SNM will emit a negative spike since its membrane potential is less than $-\theta$ and the memory value is greater than 0. The same is true at t_5. For the example in purple box, two spikes input to SNM neuron from w_2 cause the memory value larger than 0 at t_3, so SNM can fire a negative spike at t_3. But the negative spikes fire at t_3 and t_4 cause the memory value to drop to 0, then even if the membrane potential at t_5 is lower than the negative threshold again, it cannot fire a negative spike.

Through the above exemplification, we can say that the output rate of an SNM is equal to its memory value. Ordinary signed neuron brings new problems, i.e., if more negative spikes are transmitted than positive spikes, then the output rate of a signed neuron is negative, which contradicts ReLU function. But SNM can ensure that firing frequency will not be lower than 0 due to the existence of memory.

4.2 Neuron-Wise Parameter Normalization

In this section, we first show that layer-wise normalization suffers from huge information loss during ANN-SNN conversion, which causes the long inference latency of SNN. Then we present an optimal solution: neuron-wise normalization.

Problem of Long Inference Latency

A proper parameter normalization can reduce information loss when the ReLU activation function is converted to the spiking activation function, which is critical to shortening SNN inference latency. Information loss here refers to the amount of information that can be transmitted in ANN but is missed in SNN. By analyzing the activation value of the first convolutional layer, [Rueckauer *et al.*, 2017] made a point that setting normal scale to the maximum activation value λ^l of ANN layer l results in low firing rates. We will further analyze below the reason why LayerNorm needs a high inference latency.

A large number of activation values in ANN are far below maximum activation value [Rueckauer *et al.*, 2017]. Regrettably, the transmission of small values by SNN requires longer simulation length. For example, assume the maximum activation value λ of ANN's input layer is 10, so the firing threshold of neurons in SNN's input layer will be set to 10. When the input to neuron i of input layer is 0.01, then T must be greater than or equal to 1000 for neuron i to fire a spike (0.01 accumulated T times). Obviously, a smaller value requires a longer T to transmit, so the limited simulation length will cause a lot of data loss. A simple solution is to add a

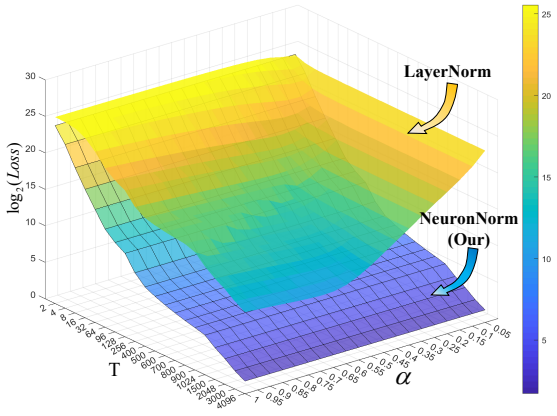

Figure 3: Comparison of the amount of information loss between LayerNorm and NeuronNorm.

coefficient α ($\alpha \in [0, 1]$) to the maximum activation value λ before setting threshold. Lowering threshold can retain more small data, but fails to distinguish larger data, because the neuron output rate will always equal to 1 when input is greater than $\alpha\lambda$.

In summary, there are two factors that affect the information loss, which are coefficient α of the maximum activation value λ and simulation length T. For a given α and T, we formulate the missing information of layer l as:

$$Loss^l = \sum_c^C \sum_w^W \sum_h^H x_{c,w,h}^l, \tag{12}$$

$$x_{c,w,h}^l = \begin{cases} a_{c,w,h}^l, & 0 < a_{c,w,h}^l < \frac{\alpha\lambda^l}{T} \\ a_{c,w,h}^l - \alpha\lambda^l, & a_{c,w,h}^l > \alpha\lambda^l \\ 0, & \text{otherwise} \end{cases}. \tag{13}$$

Where, $a_{c,w,h}^l$ is the activation value of neuron located at (w, h) in channel c of layer l. Obviously, a larger T will result in fewer losses, this is why converted SNNs using layer normalization usually requires T greater than 1024.

The Proposed Solution: Neuron-Wise Normalization

A lot of existing works focus on how to select optimal normalization scale [Kim *et al.*, 2020; Li *et al.*, 2021; Ding *et al.*, 2021], but we propose an alternative simple and efficient method named neuron-wise normalization (NeuronNorm). We separately record the maximum activation value of each neuron in ANN, and use them as the firing threshold of corresponding spiking neurons. Since the maximum activation values of different neurons in the same layer are relatively independent, the amount of information loss is greatly reduced. Therefore, Eq. (5) should be rewritten as:

$$W_{ij}^l = W_{ij}^l \frac{\lambda_i^{l-1}}{\lambda_j^l}, \quad b_j^l = \frac{b_j^l}{\lambda_j^l}, \tag{14}$$

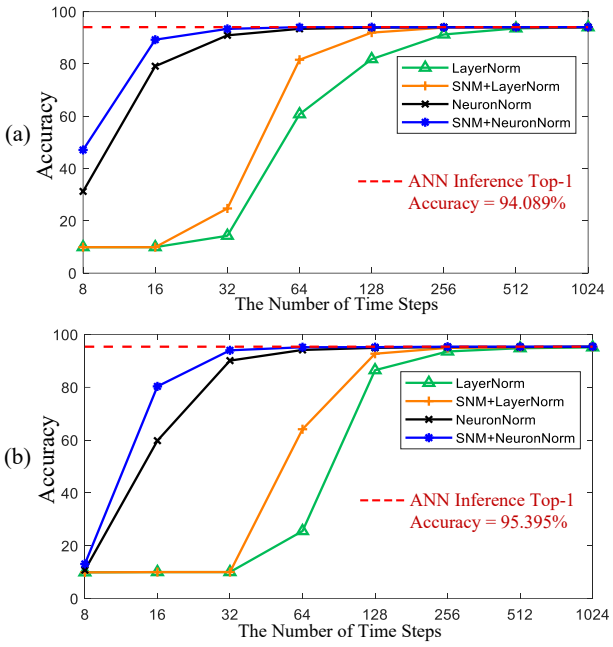

Figure 4: The inference top-1 accuracy on CIFAR10 dataset w.r.t different simulation lengths. (a) The accuracy of network using the VGG16 structure. (b) The accuracy of network using the ResNet18 structure.

where λ_i^{l-1} represents the maximum activation value of presynaptic neuron i and λ_j^l represents the maximum activation value of postsynaptic neuron j.

Fig. 3 shows the change of loss when applying 20 different T and 20 different α. The vertical axis is the logarithm of the loss calculated by Eq. (12) through 9,500 samples of CIFAR10 in the first convolutional layer of VGG16. The upper surface is the amount of converted SNN's information loss using layer-wise normalization, which changes with α and T. The surface below is the amount of SNN's information loss using our neuron-wise normalization. We fix the value of α to constant 1, meaning that our method does not perform a parametric search. It can be seen from Fig. 3 that the surface obtained by our method is lower than LayerNorm loss surface of any combination of α and T, which proves that our method achieves both simplicity and accuracy.

5 Experiments

5.1 Details

In order to verify the effectiveness of our method, we use VGG and ResNet network structures to conduct experiments on CIFAR10, CIFAR100[1], and ImageNet2012[2] datasets. The initialization of ANN network parameters adopts Kaiming normal initialization [He *et al.*, 2015], and the network training adopts SGD algorithm with 0.9 momentum followed by milestones learning rate decay. The L2 penalty with a value of $5e^{-4}$ is also added. In addition, to prevent over-fitting, we add Batch Normalization (BN) layers to ANNs. In the SNN inference stage, we use the method introduced by [Rueckauer *et al.*, 2017] to fold the parameters of BN layers into convolutional layers. The usage of parameter normalization assumes that train data and test data have the same distribution, so the maximum activation value we choose for neuron-wise normalization is obtained from training set.

[1]https://www.cs.toronto.edu/~kriz/cifar.html

[2]https://image-net.org/challenges/LSVRC/2012/

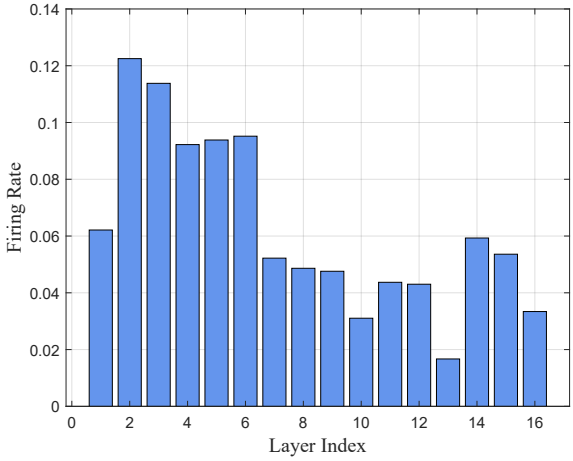

Figure 5: Firing rate of VGG16 on ImageNet dataset.

5.2 Ablation Study

In this section, we will prove that the proposed method enables SNN with higher prediction accuracy and shorter simulation time. The result here is inference top-1 accuracy on CIFAR10 dataset.

Fig. 4(a) and Fig. 4(b) respectively show the SNN inference accuracy of the VGG16 structure and ResNet18 structure when using four different conversion methods. In the benchmark model for comparison, we use the layer-wise normalization with maximum activation value ($\alpha = 1$) of each layer to normalize weight and bias, which is marked as green line in Fig. 4. After changing IF neuron of the benchmark model to SNM neuron, as shown by the orange line in Fig. 4, we can see a significant improvement in accuracy at any simulation duration. The black line shows that the role of neuron-wise normalization is significant, it can use 32 time steps to achieve the accuracy of layer-wise normalization with 256 inference time steps.

The blue line shows the effect of combining two techniques proposed in this work. It can be seen that the inference accuracy of converted SNN now exceeds the accuracy of any method used alone, and there are only 32 time steps to achieve a near-lossless ANN-SNN conversion.

5.3 Comparison with Related Work

To further illustrate the effectiveness of our work, here we use VGG and ResNet structures to conduct experiments on three datasets, and compare top-1 accuracy with some of the best models as shown in Table 1. For CIFAR datasets, regardless of using VGG or ResNet, the accuracy of our model can surpass all previous state-of-the-art models. The VGG16 trained on CIFAR10 can reach top-1 accuracy 94.1%, and

Method	Net. Arch.	ANN Acc.	T=32	T=64	T=128	T≥512
			CIFAR10			
RMP [Han *et al.*, 2020]	VGG16	93.63	60.3	90.35	92.41	93.63(T=2048)
RateNorm [Ding *et al.*, 2021]	VGG16	92.9	85.4	91.15	92.51	92.86(Unknown)
Opt. [Deng and Gu, 2021]	VGG16	92.09	92.29	92.22	92.24	92.03(T=512)
Ours (SNM+NeuronNorm)	VGG16	94.09	**93.43**	**94.07**	**94.07**	**94.1**(T=512)
RateNorm [Ding *et al.*, 2021]	ResNet18	93.06	83.95	91.96	93.27	93.45(Unknown)
Opt. [Deng and Gu, 2021]	ResNet20	92.32	93.3	93.55	93.56	93.58(T≤600)
Ours (SNM+NeuronNorm)	ResNet18	95.39	**94.03**	**94.03**	**95.19**	**95.44**(T=1024)
			CIFAR100			
RMP [Han *et al.*, 2020]	VGG16	71.22	-	-	63.76	70.93(T=2048)
Opt. [Deng and Gu, 2021]	VGG16	70.21	56.16	62.93	67.45	70.35(T≥2048)
Ours (SNM+NeuronNorm)	VGG16	74.13	**71.8**	**73.69**	**73.95**	**74.1**(T=512)
RMP [Han *et al.*, 2020]	ResNet20	68.72	27.64	46.91	57.69	67.82(T=2048)
Opt. [Deng and Gu, 2021]	ResNet20	77.16	51.27	70.12	75.81	77.19(T≥2048)
Ours (SNM+NeuronNorm)	ResNet18	78.26	**74.48**	**77.59**	**77.97**	**78.3**(T=1024)
			ImageNet			
RMP [Han *et al.*, 2020]	VGG16	73.49	-	-	-	73.09(T=4096) (-0.4)
Opt. [Deng and Gu, 2021]	VGG16	72.4	54.92 (-17.48)	66.51 (-5.89)	69.94 (-2.46)	72.09(T≥2048) (-0.31)
Calibration [Li *et al.*, 2021]	VGG16	75.36	63.64 (-11.72)	70.69 (-4.67)	**73.32** (-2.04)	**75.32**(T≥2048) (-0.04)
Ours (SNM+NeuronNorm)	VGG16	73.18	**64.78** (**-8.4**)	**71.5** (**-1.68**)	72.86 (**-0.32**)	73.16(T=1024) (**-0.02**)

Table 1: Comparison with other works. The best accuracy of the same network structure on the same dataset is displayed in bold. The ImageNet dataset additionally lists the conversion loss in brackets under accuracy, which is $Acc_{snn} - Acc_{ann}$.

trained on CIFAR100 can reach top-1 accuracy 74.1%. The ResNet18 trained on CIFAR10 reach top-1 accuracy 95.44%, and trained on CIFAR100 can reach top-1 accuracy 78.3%. In order to prove that our model does not require excessive inference latency (T>128), Table 1 also lists the inference accuracy of different time steps, and compares them with other works. The results show that, our accuracy is the highest at any number of time steps in CIFAR datasets. Since the ANN we trained on ImageNet is not satisfactory, we further compare the conversion losses which are listed in brackets. Results indicate the accuracy drop of our model is minimal with any simulation length on ImageNet dataset.

5.4 Energy Estimation

In this section we will prove the efficiency of proposed method. This experiment is conducted on ImageNet dataset using VGG16 network structure. Through statistics of firing rate, we are pleased to find that the calculation of our model is very sparse. It should be noted that the firing rate here is calculated by the number of spikes. Unlike section 4.1, the positive and negative spikes will not offset. We show the firing rate of each layer when simulation length is 32 in Fig. 5, and the average firing rate is only 0.063. To further estimate the energy consumption on chips, we use the energy estimation equation in the work of [Rathi and Roy, 2020]. In our work, except for neurons in the first layer to perform multiplication operations, the rest of neurons only perform addition operations. So the ratio of SNN and ANN energy consumption is:

$$\frac{Energy_{snn}}{Energy_{ann}} = \frac{c * \alpha + (1 - \frac{c}{b}) * b * \beta}{a * \alpha}, \tag{15}$$

where α represent the energy cost for multiplication which is 4.6 pJ and β for addition which is 0.9 pJ [Horowitz, 2014]. a, b and c represent the number of operations in ANN, SNN and first layer of SNN. Based on Eq. (15), we calculated that our model only needs 50.12% of ANN's energy consumption when T is 32.

6 Conclusion

In this paper, we propose a new ANN-SNN conversion method from the perspective of neuron model and normalization method, which can achieve accurate and low-latency inference. We find and analyze the conversion error caused by SNN asynchronous transmission, and propose a new neuron model named Signed Neuron with Memory (SNM), which can ensure the correspondence of spiking neuron output rate to ReLU activation value without losing the asynchronous transmission capability of SNN. Then, we find that the layer-wise normalization will lose much information during ANN-SNN conversion, which severely increases the inference latency of SNNs. In response to this problem, we propose neuron-wise normalization, which can significantly reduce the information loss, thereby shortening the SNN inference latency. We convert deep neural networks such as VGG16 and ResNet to SNNs with this model, converted SNNs can

reach new state-of-the-art performances for different simulation lengths on challenging datasets.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0100202, in part by the National Science Foundation of China under Grant 61976043 and Grant 62106038, and in part by the CCF-Hikvision Open Fund.

References

[Akopyan *et al.*, 2015] Filipp Akopyan, Jun Sawada, Andrew Cassidy, Rodrigo Alvarez-Icaza, John Arthur, Paul Merolla, Nabil Imam, Yutaka Nakamura, Pallab Datta, Gi-Joon Nam, et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE transactions on computer-aided design of integrated circuits and systems*, 34(10):1537–1557, 2015.

[Cao *et al.*, 2015] Yongqiang Cao, Yang Chen, and Deepak Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66, 2015.

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

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

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Diehl *et al.*, 2015] Peter U Diehl, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International joint conference on neural networks (IJCNN)*, pages 1–8. ieee, 2015.

[Diehl *et al.*, 2016] Peter U Diehl, Bruno U Pedroni, Andrew Cassidy, Paul Merolla, Emre Neftci, and Guido Zarrella. Truehappiness: Neuromorphic emotion recognition on truenorth. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 4278–4285. IEEE, 2016.

[Ding *et al.*, 2021] 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.

[Han *et al.*, 2020] 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*, pages 13558–13567, 2020.

[He *et al.*, 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

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

[Kheradpisheh *et al.*, 2018] Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, Simon J Thorpe, and Timothée Masquelier. Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67, 2018.

[Kim *et al.*, 2020] Seijoon Kim, Seongsik Park, Byunggook Na, and Sungroh Yoon. Spiking-yolo: Spiking neural network for energy-efficient object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11270–11277, 2020.

[Lam *et al.*, 2019] Max WY Lam, Xie Chen, Shoukang Hu, Jianwei Yu, Xunying Liu, and Helen Meng. Gaussian process lstm recurrent neural network language models for speech recognition. In *ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 7235–7239. IEEE, 2019.

[Li *et al.*, 2021] Yuhang Li, Shikuang Deng, Xin Dong, Ruihao Gong, and Shi Gu. A free lunch from ann: Towards efficient, accurate spiking neural networks calibration. *arXiv preprint arXiv:2106.06984*, 2021.

[Neftci *et al.*, 2019] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[Pei *et al.*, 2019] 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*, 572(7767):106–111, 2019.

[Rathi and Roy, 2020] Nitin Rathi and Kaushik Roy. Diet-snn: Direct input encoding with leakage and threshold optimization in deep spiking neural networks. *arXiv preprint arXiv:2008.03658*, 2020.

[Rueckauer *et al.*, 2016] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, and Michael Pfeiffer. Theory and tools for the conversion of analog to spiking convolutional neural networks. *arXiv preprint arXiv:1612.04052*, 2016.

[Rueckauer *et al.*, 2017] 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*, 11:682, 2017.

[Sengupta *et al.*, 2019] Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in neuroscience*, 13:95, 2019.

[Silver *et al.*, 2018] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

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