GRADUAL SURROGATE GRADIENT LEARNING IN DEEP SPIKING NEURAL NETWORKS

Yi Chen, Silin Zhang, Shiyu Ren, Hong Qu

School of Computer Science and Engineering
University of Electronic Science and Technology of China
China

ABSTRACT

Spiking Neural Network (SNN) is a promising solution for ultra-low-power hardware. Recent SNNs have reached the performance of Deep Neural Networks (DNNs) in dealing with many tasks. However, these methods often suffer from a long simulation time to achieve the accurate spike train information. In addition, these methods are contingent on a well-designed initialization to effectively transmit the gradient information. To address these issues, we propose the Internal Spiking Neuron Model (ISNM), which uses the synaptic current instead of spike trains as the carrier of information. In addition, we design a gradual surrogate gradient learning algorithm to ensure that SNNs effectively back-propagate gradient information in the early stage of training and more accurate gradient information in the later stage of training. The experiments on various network structures on CIFAR-10 and CIFAR-100 datasets show that the proposed method can exceed the performance of previous SNN methods within 5 time steps.

Index Terms— Spiking Neural Networks, Spiking Neuron Model, Surrogate Gradient

1. INTRODUCTION

Benefiting from the hierarchical structure, Artificial Neural Networks (ANNs) are now the mainstream methods in machine learning[1]. Larger and more complex network structures effectively enhance the processing power of ANNs. However, these high-performance network models require a lot of computing resources and energy consumption during training and inference, which greatly limit their applications. There are many attempts to circumvent these concerns, such as model compression[2], neural network quantization[3] and distillation[4]. However, these attempts have not fundamentally solved these problems.

This work was supported by National Key R&D Program of China under Grant 2018AAA0100202, and in part by the National Science Foundation of China under Grant 61976043, and in part by the Zhejiang Lab's International Talent Fund for Young Professionals, and in part by the China Postdoctoral Science Foundation (Grant No. 2020M680148). (Corresponding author: Hong Qu, email:hongqu@uestc.edu.cn)

SNN shows promising potential for ultra-low-power hardware due to its highly parallel structure and event-driven processing manner[5][6]. The unique event-driven mechanism of SNN does not require constant computation, thus, consuming extremely low amount of computational resources[5]. On the other hand, information travels through discrete spike trains in SNNs. For different types of tasks, researchers designed different coding methods[7]. In the most classical latency-coding method, different information is represented by the firing time of a single spike. This further reduces the storage space and energy consumption[8].

However, due to the complex spatio-temporal dynamics and non-differentiable spiking activation function involved in SNN, the development of efficient learning algorithms for deep SNNs is an on-going research challenge. According to the learning processes of the existing algorithms, they can be divided into two categories.

The first category is ANN-to-SNN conversion methods, in which the parameters of a pre-trained ANN are converted into its SNN counterpart with similar structure. However, the conversion process inevitably degrades the SNN performance. Although various strategies have been proposed to circumvent the performance degradation, such as setting maximum activation as threshold[9], modifying ANN to make it more similar to SNN, the problem has not been completely solved. In addition, most of the conversion methods often adopt a rate-based coding scheme. This leads long simulation time and high energy consumption.

The second category of learning algorithms train the DSNNs directly. These methods mainly deal with the non-differentiable firing function to achieve effective Back Propagation (BP) learning[10][11][12][13]. These learning algorithms show competitive results compared with their ANN counterparts. However, these methods suffer from the gradient explosion problem and highly depend on the well-designed network initialization to ensure a successful BP processes in the early stage of training. In addition, the spike trains' discreteness limits the usage of ANNs' diverse structures, such as max-pooling and batch normalization.

To address these issues, we propose an Internal Spiking Neuron Model (ISNM), which takes the spike trains as a part of the internal dynamic process of the spiking neuron and uses

the synaptic current as the carrier of information transmission. In this way, the ISNM based SNNs can directly adopt the max-pooling and batch normalizaiton function. In addition, we designed a gradual surrogate gradient learning algorithm to ensure that: 1) In the early stage of learning, SNNs can effectively return gradient information even with a random initialization; 2) with the ongoing learning, we gradually tune the surrogate gradient function to approach the actual one with the aim to obtain a high accuracy.

2. METHODS

2.1. Internal Spiking Neuron Model

In the Leaky Integrate-and-Fire (LIF) spiking neuron model[14], the membrane potential U_i^t at time t for neuron i is given by

$$\tau_m \frac{\mathrm{d}U_i^l(t)}{\mathrm{d}t} = -[U_i^l(t) - U_{rest}] + RI_i^l(t). \quad (1)$$

τ_m is the constant used to control the change of neuron membrane potential over time, U is the resting potential, R is another constant used to control the effect of the input current on the membrane potential, I denotes the input current, which is defined as

$$I_i^l(t) = \sum w_{ji}^{l-1} s_j^{l-1}(t) \quad (2)$$

When membrane potential U_i^t exceed threshold θ from below, the spiking neuron fires a spike and this firing function can be written as:

$$s_i^l(t) = H(U_i^l(t) - \theta), \quad (3)$$

where H is the Heaviside function. For the convenience of calculation, we set $\frac{R}{\tau_m} = 1$ and resting potential $U_{rest} = 0$, and Eq.1, 2 and 3 can be merged into

$$s_i^l(t) = H(\frac{1}{\tau} U_i^l(t-1) + \sum_{j=1}^{N} w_{ji}^{l-1} s_j^{l-1}(t) - s_i^l(t-1) - \theta). \quad (4)$$

As we know, traditional ANNs consists of many functional layers, such as convolution layer, fully connected layer pooling layer and Batch Normalization (BN) layer. However,as shown in Eq.4, due to the binary representation of spikes ($s_i^l(t)$), there is no magnitude difference between spikes at the same time. This leads to serious errors if the maximum pooling layer or BN layer is directly adopted in SNNs.

To resolve this problem, we proposed the ISNM as shown in Eq. 5

$$I_i^l(t) = \sum_{j=1}^{N} w_{ji}^{l-1} H(\frac{1}{\tau} U_i^l(t-1) + I_i^{l-1}(t) - s_i^l(t-1) - \theta), \quad (5)$$

where the ISNM input $I_i^l(t)$ is no longer limited to a binary representation. Therefore, the ISNM-based SNN can perform maximum pooling and BN layer directly. In addition, this model does not require additional encoding or decoding methods.

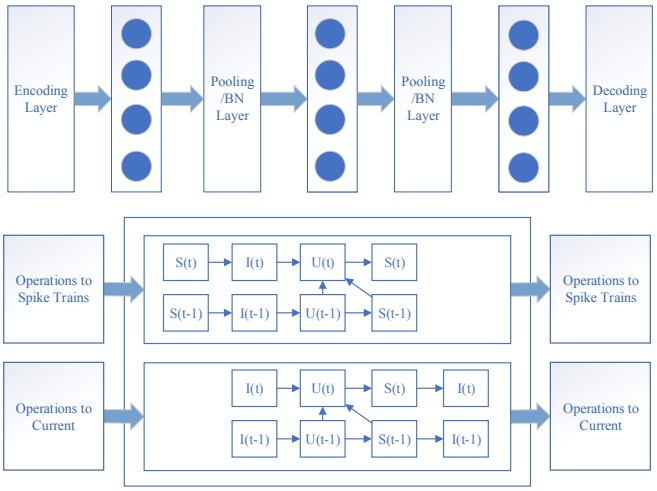

Fig. 1. Calculation of feedforward process in SNNs. The traditional spiking neural model, as shown in the figure above, uses spike trains as the carriers of information transmission between layers. The ISNM model is shown in the figure below. Input current is taken as the carrier of information transmission between layers, and spike trains are taken as a part of the dynamic process of neurons.

The method proposed here can be applied on top of existing model architecture. This gives us the flexibility to make orthogonal adjustment to the base model architecture. It should be noted that ISNM proposed in this paper is an attempt to model neurons from another perspective, which is no different from the dynamic process of traditional LIF neurons. In fact, when the synaptic weights and parameters are exactly the same, the neuron membrane potential of the two models behaves exactly the same as the output spike trains for the same input spike trains .

2.2. Gradual Surrogate Gradient Learning

Gradient surrogate-based methods often require well-designed weight initialization to ensure the effectively transmission of gradient information. At the same time, there is a gap between the surrogate and actual gradients at the early stage of training. This gap is beneficial at the early stage to activate more neuron to participate in the learning process. However, at the latter stage, this gap will be detrimental to the final accuracy of the trained SNNs. Based on the above understanding, we put forward Gradual Surrogate Gradient (GSG) learning method, defined as:

$$s(t) = \alpha \cdot (tanh[\beta(U_i^l - \theta)] + c), \quad (6)$$

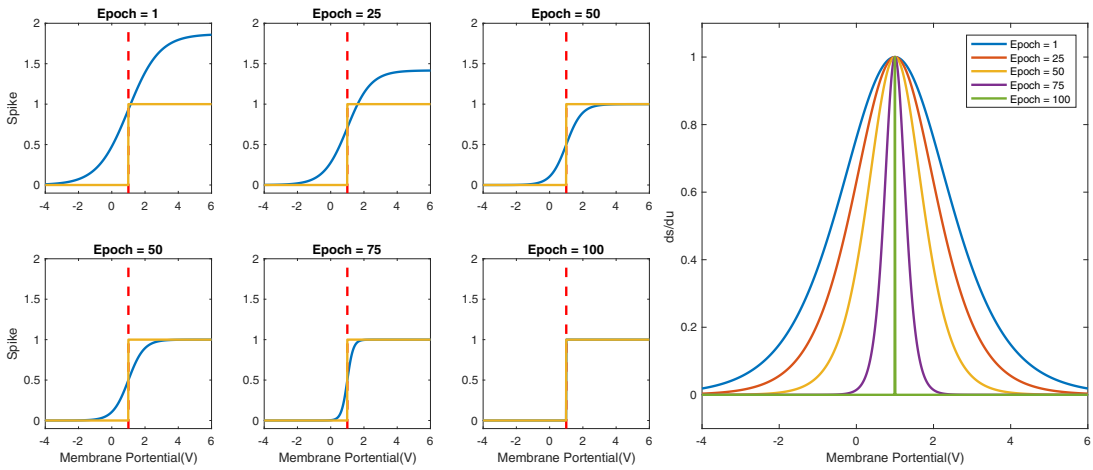

Fig. 2. The process of gradual surrogate gradient learning algorithm. At the beginning of training (Epoch$\leq$ 50), the approximation function will generate gradient information for a larger range of membrane voltages, ensuring that the network model can be effectively trained. At the end of the training (Epoch$\geq$ 50), the approximation function approximates the firing function more closely, resulting in more accurate gradient information.

where α and β are tunable parameters defined as

$$\beta = e^{\log(\beta_{min}) + \frac{(\log \beta_{max} - \log \beta_{min}) \cdot epoch}{max\_epoch}} \tag{7}$$

$$\alpha = max(\frac{1}{\beta}, 1) \tag{8}$$

According to Eq. 6, the derivative of $s(t)$ w.r.t. $u(t)$ is

$$\frac{\partial s(t)}{\partial u(t)} = \alpha\beta(1 - tanh^2((u(t) - \theta)\beta). \tag{9}$$

As shown in Fig.2, at the early stage of training, the approximation function will cover as much value range of membrane voltage as possible to ensure that more spike neurons can participate in learning. With the ongoing learning, the approximation function gradually approach the firing function, and the gradient information becomes increasingly accurate. By the end of the training, the approximation function is almost indistinguishable from the firing function.

3. EXPERIMENTS

3.1. Implementation Details

In order to test the effectiveness of the proposed model and training method, we use ISNM to build two networks with different structures, namely VGG-small and ResNet, on the CIFAR-10 and CIFAR-100 data sets. The structure of VGG-small network is a variant of VGG-9[15] with batch normalization. Specifically, we reduce the following multiple fully connected layers to one layer and add one convolutional layer as the coding layer. The structure of ResNet network refers to the classical ResNet-18[16] network structure. Since ISNM does not require additional coding methods, the first layer of the two networks adopts CNN instead of SNN as the coding layer to obtain more effective image feature information. Similarly, thanks to the nature of the ISNM, the output of the last layer does not require an additional decoding strategy and can be identified only by the average output current.

In the training process, we adopt cross entropy loss as loss function and use Adam optimizer[17] with initial learning rate of 10^{-3} with weight decay set to 10^{-4}. 200 and 300 epochs were trained on CIFAR-10 and CIFAR-100 datasets, respectively. During training, we used normal data augmentation methods, including normalization, random horizontal flipping, and random cropping.

3.2. Ablation Study

On the basis of ISNM and GSG, we introduce soft rest mechanism to further reduce information loss during network forward transmission. In order to demonstrate the effectiveness of our proposed model and algorithm, we conducted ablation experiments on the CIFAR10 data set through the VGG structure network, and the experimental results are shown in Table 1. The first row in the table represents the effect of training with a fixed approximation function using only ISNM. The second line builds on the previous one and adds the softrest mechanism. The third line is using both ISNM and GSG, without soft rest. The last one is the effect of using all three methods at once. It can be seen that the use of GSG ensures the accuracy of the regression gradient in the later training period, and improves the accuracy of the model by about 6%. The introduction of soft rest improved the accuracy of forward information transfer and improved model performance

by about 1%.

Table 1. Ablation Study

Method	Accuracy(%)
ISNM	84.63
ISNM+Soft Rest	85.4
ISNM+GSG	90.32
ISNM+GSG+Soft Rest	91.7

3.3. Comparison to Previous Work

Next, we compare proposed methods with previous works on CIFAR-10 and CIFAR-100 datasets. In Table 2 and Table 3, column 2 represents different architectures, column 3 represents the length of simulation time used for evaluation, and column 4 represents accuracy. It can be seen from Table 2 and Table 3 that the proposed method only needs 5 time steps to train SNN and inference, so as to obtain better test accuracy. As a result, the reasoning delay is improved 1-500 times compared to other SNNs.

Table 2. Comparison to Previous Work on CIFAR-10

Method	Architecture	TimeStep	Accuracy(%)
STDB[18]	VGG-16	680	91.4
Hybrid[19]	VGG-16	5	91.41
Conversion[20]	VGG-16	1500	91.2
Surrogate [21]	CifarNet	12	90.53
Surrogate[9]	VGG-16	2500	91.55
STDB[22]	ResNet-11	100	90.95
This work	VGG-small	**5**	**91.7**($\pm$ 0.31)
This work	ResNet	**5**	90.83($\pm$ 0.26)

Table 3. Comparison to Previous Work on CIFAR-100

Method	Architecture	TimeStep	Accuracy(%)
Conversion[23]	VGG-16	62	63.2
Surrogate[24]	VGG-9	50	66.6
Hybrid[19]	VGG-16	5	66.46
Surrogate[9]	ResNet-20	2048	64.09
This work	VGG-small	**5**	69.2($\pm$ 0.19)
This work	ResNet	**5**	**69.33**($\pm$ 0.46)

3.4. The effect of simulated time length

Previous high performance SNNs used frequency-based encoding schemes for statically typed data, such as images. The accuracy of such coding schemes is limited by the length of simulation time. Especially when the length of simulation time is very small (less than 5), this encoding scheme is difficult to transmit information effectively. The performance of SNNs based on this coding scheme is comparable to that of DNNs when the simulation duration is long, and its performance degrades sharply when the simulation duration is very small. Here, we design experiments to analyze the performance of the model and algorithm proposed in this paper under very short simulation time length, as shown in Table 4. It can be seen that although the reduction of the simulation time will slightly reduce the performance of the model, when the simulation time is very short, the proposed method can still maintain good performance. The reason for this result is that the coding layer is generated through training rather than manual setting, which increases the extraction of effective features and reduces information redundancy.

Table 4. The Effect of Simulated Time Length

Dataset	Architecture	TimeStep	Accuracy(%)
CIFAR-10	VGG	1	91.25
		5	91.7
	ResNet	1	90.56
		5	90.83
CIFAR-100	VGG	1	66.8
		5	69.2
	ResNet	1	67.65
		5	69.33

4. CONCLUSION

In this paper, we propose an Internal Spiking Neuron Model (ISNM) for SNN to directly perform max-pooling and batch normalizaiton function. We further proposed a gradual surrogate gradient (GSG) learning method for deep SNNs. In the GSG learning, the surrogate gradient function is tunable during the learning process. with the proposed GSG method, we can ensure more spiking neurons participate in learning at the early stage and higher accuracy after learning. we conducted experiments on different network structures on CIFAR-10 and CIFAR-100 data sets, respectively. Experimental results show that the proposed method outperforms the previous SNN model within 5 time steps. Moreover, our method works well even with one time step.

5. REFERENCES

[1] Jürgen Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[2] Song Han, Huizi Mao, and William J Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[3] Haotong Qin, Ruihao Gong, Xianglong Liu, Mingzhu Shen, Ziran Wei, Fengwei Yu, and Jingkuan Song, "Forward and backward information retention for accurate binary neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2250–2259.

[4] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 582–597.

[5] Malu Zhang, Hong Qu, Ammar Belatreche, Yi Chen, and Zhang Yi, "A highly effective and robust membrane potential-driven supervised learning method for spiking neurons," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 1, pp. 123–137, 2018.

[6] Stanisław Woźniak, Angeliki Pantazi, Thomas Bohnstingl, and Evangelos Eleftheriou, "Deep learning incorporating biologically inspired neural dynamics and in-memory computing," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 325–336, 2020.

[7] Yi Chen, Hong Qu, Malu Zhang, and Yuchen Wang, "Deep spiking neural network with neural oscillation and spike-phase information," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, vol. 35, pp. 7073–7080.

[8] Malu Zhang, Jiadong Wang, Jibin Wu, Ammar Belatreche, Burin Amornpaisannon, Zhixuan Zhang, Venkata Pavan Kumar Miriyala, Hong Qu, Yansong Chua, Trevor E Carlson, et al., "Rectified linear postsynaptic potential function for backpropagation in deep spiking neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[9] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: Vgg and residual architectures," *Frontiers in Neuroscience*, vol. 13, 2019.

[10] Xiaoling Luo, Hong Qu, Yun Zhang, and Yi Chen, "First error-based supervised learning algorithm for spiking neural networks," *Frontiers in neuroscience*, vol. 13, pp. 559, 2019.

[11] 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*, vol. 36, no. 6, pp. 51–63, 2019.

[12] Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian, "Incorporating learnable membrane time constant to enhance learning of spiking neural networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2661–2671.

[13] 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.

[14] Wulfram Gerstner and Werner M Kistler, *Spiking neuron models: Single neurons, populations, plasticity*, Cambridge university press, 2002.

[15] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[17] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[18] Seongsik Park, Seijoon Kim, Byunggook Na, and Sungroh Yoon, "T2fsnn: deep spiking neural networks with time-to-first-spike coding," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[19] Gourav Datta, Souvik Kundu, and Peter A Beerel, "Training energy-efficient deep spiking neural networks with single-spike hybrid input encoding," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[20] Jaehyun Kim, Heesu Kim, Subin Huh, Jinho Lee, and Kiyoung Choi, "Deep neural networks with weighted spikes," *Neurocomputing*, vol. 311, pp. 373–386, 2018.

[21] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi, "Direct training for spiking neural networks: Faster, larger, better," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 1311–1318.

[22] C. Lee, S. S. Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and K. Roy, "Enabling spike-based backpropagation for training deep neural network architectures," *arXiv preprint*, 2019.

[23] Sen Lu and Abhronil Sengupta, "Exploring the connection between binary and spiking neural networks," *Frontiers in Neuroscience*, vol. 14, pp. 535, 2020.

[24] Youngeun Kim and Priyadarshini Panda, "Revisiting batch normalization for training low-latency deep spiking neural networks from scratch," *arXiv preprint arXiv:2010.01729*, 2020.