Gradient Scaling on Deep Spiking Neural Networks with Spike-Dependent Local Information

Seongsik Park [1] **Jeonghee Jo** [1] **Jongkil Park** [1] **Yeonjoo Jeong** [1] **Jaewook Kim** [1] **Suyoun Lee** [1]
Joon Young Kwak [1] **Inho Kim** [1] **Jong-Keuk Park** [1] **Kyeong Seok Lee** [1] **Gye Weon Hwang** [1] **Hyun Jae Jang** [1]

Abstract

Deep spiking neural networks (SNNs) are promising neural networks for their model capacity from deep neural network architecture and energy efficiency from SNNs' operations. To train deep SNNs, recently, spatio-temporal backpropagation (STBP) with surrogate gradient was proposed. Although deep SNNs have been successfully trained with STBP, they cannot fully utilize spike information. In this work, we proposed gradient scaling with local spike information, which is the relation between pre- and post-synaptic spikes. Considering the causality between spikes, we could enhance the training performance of deep SNNs. According to our experiments, we could achieve higher accuracy with lower spikes by adopting the gradient scaling on image classification tasks, such as CIFAR10 and CIFAR100.

1. Introduction

Deep learning with deep neural networks (DNNs) have been rapidly advancing artificial intelligence (AI) technology in various fields (LeCun et al., 2015; Tan & Le, 2019). However, as AI technology continues to progress, it demands more energy and computing resources, raising concerns about sustainable development and application. Spiking neural networks (SNNs) have received considerable attention as a solution to this problem. SNNs, which have been considered third-generation artificial neural networks, enable event-based computing, resulting in sparse operations compared to DNNs (Maass, 1997). Furthermore, SNNs hold great importance as the basis for neuromorphic computing, which imitates the operations of the human brain for its exceptional energy efficiency (Davies et al., 2018; Roy et al., 2019).

[1]Center for Neuromorphic Engineering, Korea Institute of Science and Technology, Seoul, Korea. Correspondence to: Seongsik Park <seong.sik.park@kist.re.kr>.

In *ICML Workshop on Localized Learning (LLW)*, Honolulu, Hawaii, USA. 2023. Copyright 2023 by the author(s).

Deep SNNs have been actively studied to combine the features of both DNNs and SNNs: the model capacity of the former and the energy efficiency of the latter. Deep SNNs have a similar synaptic topology to DNNs, with interconnected spiking neurons. While deep SNNs can leverage the advantages of both DNNs and SNNs, they face challenges in training. As an indirect training method for deep SNNs, DNN-to-SNN conversion has been proposed. Although this approach has enabled the implementation of various deep SNN models (Park et al., 2019; Kim et al., 2020; Park et al., 2020; Li et al., 2021; Bu et al., 2022), it has introduced issues, such as long inference latency (Han et al., 2020).

Recently, to improve the training performance, a gradient-based training algorithm, which is a successful training approach for DNNs, has been applied to training deep SNNs, such as spatio-temporal backpropagation (STBP) (Wu et al., 2018). This method with surrogate gradient, which can handle the non-differentiability of spiking neurons, has proven to be effective in training deep SNNs. Based on the successful training, gradient-based training approaches have inspired further research about improving the training performance of deep SNNs (Zheng et al., 2021; Deng et al., 2022; Yang et al., 2022). Furthermore, it has enabled the expansion of deep SNNs in various applications and algorithms, including Transformer models (Zhou et al., 2023) and neural architecture search algorithms (Na et al., 2022).

Gradient-based training algorithms have allowed deep SNNs to utilize their model capacity sufficiently. However, these algorithms cannot exploit the dynamic characteristics of SNNs as they are derived from DNNs. Unlike DNNs, SNNs have spatio-temporal features, and spiking neurons transmit information in the form of spikes. Thus, to maximize the training performance of deep SNNs, we proposed a training algorithm, called gradient scale, that can consider the spike dynamics in SNNs. We were inspired by spike-timing-dependent plasticity (STDP), which is a biologically plausible training algorithm of SNNs with local spike causality (Diehl & Cook, 2015). While utilizing the training performance of gradient-based algorithms, we adjusted gradients depending on the local spike relationships, which can be defined by the causality between spikes of pre- and post-

synaptic neurons. The proposed algorithm was evaluated on ResNet architectures (He et al., 2016) with image classification tasks, such as CIFAR10 and CIFAR100 (Krizhevsky, 2009).

2. Related Works

2.1. Spiking Neural Networks

SNNs consist of spiking neurons and synapses that connect them. Mimicking the behavior of the brain, spiking neurons exchange information with binary spikes through synapses. Because of the spike-based operation, SNNs have been expected to enable event-driven computing, which is a next-generation and energy-efficient computing paradigm. Thus, SNNs are promising neural networks for energy-efficient artificial intelligence as a fundamental component of neuromorphic computing that mimics the operations of the human brain.

Although there are various types of spiking neurons, such as izhikevich, leaky integrate-and-fire (LIF), and integrate-and-fire (IF) neuron models (Izhikevich, 2004), neurons commonly operate in an integrate-and-fire manner. Spiking neurons integrate incoming information into the internal state, called membrane potential, and fire spikes whenever the potential exceeds a threshold voltage. Due to the low complexity of computation, most deep SNNs adopt relatively simple spiking neuron models, such as IF and LIF. Thus, in this work, we used an LIF neuron model, which is described as

$$u_j^l(t) = \tau u_j^l(t\text{-}1) + z_j^l(t), \tag{1}$$

where τ is a leak constant, $u_j^l(t)$ and $z_j^l(t)$ are the membrane potential and incoming information of the jth spiking neuron in lth layer at time step t, respectively. The incoming information, called post-synaptic potential (PSP), is caused by pre-synaptic spikes (input spikes) as

$$z_j^l(t) = \sum_i w_{ij}^l s_i^{l\text{-}1}(t) + b_j^l, \tag{2}$$

where w and b are the synaptic weight and bias, respectively. When the accumulated information on the membrane potential exceeds a certain threshold, spikes are generated, and the information is transmitted to adjacent neurons through synapses. Spike generation can be expressed as

$$s_j^l(t) = H(u_j^l(t) - v_{\text{th},j}^l(t)), \tag{3}$$

where H is the Heaviside step function, and v_{th} is a threshold voltage. When a spike is generated, the membrane potential is reset. There are mainly two reset methods: soft and hard reset can be stated as

$$u_j^l(t) = \begin{cases} u_j^l(t) - s_j^l(t) v_{\text{th},j}^l(t) & \text{(soft)} \\ (s_j^l(t) - 1) u_j^l(t) + s_j^l(t) v_{\text{r},j}^l(t) & \text{(hard)}, \end{cases} \tag{4}$$

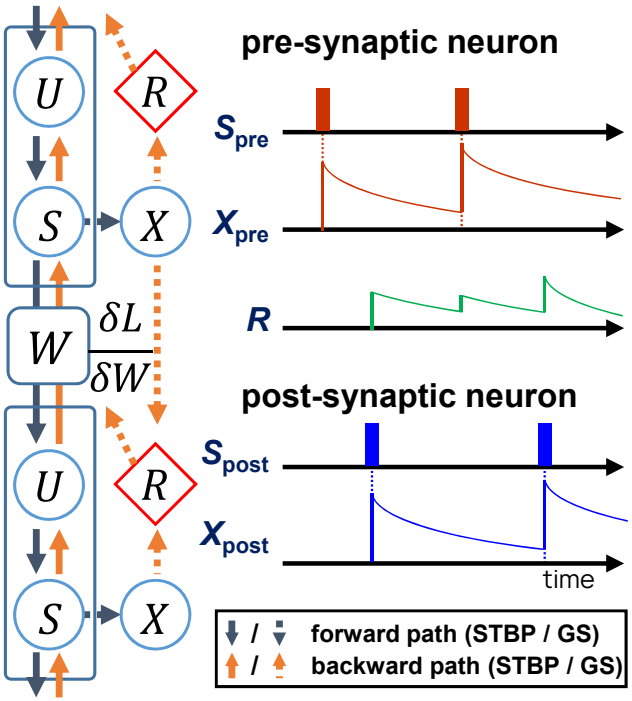

Figure 1. Spike trace and proposed gradient scaling (GS).

where v_{r} is a rest potential.

2.2. Training Methods of deep SNNs

Training algorithms of deep SNNs can be categorized into two approaches: indirect and direct training. Indirect training, which is represented by DNN-to-SNN conversion, transforms a pre-trained DNN model into deep SNN with the same topology, and the converted SNN only performs inference. This approach has been successfully applied to various neural network architectures (Sengupta et al., 2019; Han et al., 2020), applications (Kim et al., 2020), and neural codings (Park et al., 2019; Zhang et al., 2019; Park et al., 2020). However, it had drawbacks, such as long inference latency, due to disregarding features of SNNs during the training of DNNs. Certain studies have attempted to address these limitations with calibration (Li et al., 2021) and SNN-aware DNN training (Bu et al., 2022), but there still remain limitations that it is challenging to directly consider the dynamics of deep SNNs.

Direct training is a promising approach for high-performance and efficient deep SNNs. It can be mainly divided into unsupervised and supervised learning; which are represented by STDP and stochastic gradient descent (SGD), respectively. STDP is a biologically plausible training algorithm that considers the causal relationship between the spikes of pre- and post-synaptic neurons (Diehl & Cook,

Table 1. Accuracy and spikes on various configurations with CIFAR10 (training results of four times repetitions)

Methods	Reset	Accuracy (%)		# of Spikes (K)	
		Mean	Max	Mean	Max
ResNet20					
Baseline	soft	94.91	94.97	497	520
Gradient scale	soft	**95.05**	**95.11**	**492**	**506**
Baseline	hard	**93.68**	**93.86**	463	**470**
Gradient scale	hard	93.58	93.69	**451**	475
ResNet32					
Baseline	soft	95.00	95.16	814	879
Gradient scale	soft	**95.12**	**95.22**	**783**	**822**
Baseline	hard	90.52	90.71	**555**	572
Gradient scale	hard	**90.57**	**90.91**	563	**571**

Table 2. Accuracy and spikes on various configurations with CIFAR100 (training results of four times repetitions)

Methods	Reset	Test Accuracy		# of Spikes (K)	
		Mean	Max	Mean	Max
ResNet20					
Baseline	soft	74.83	75.18	641	649
Gradient scale	soft	**75.20**	**75.88**	**634**	**642**
Baseline	hard	72.26	72.37	562	573
Gradient scale	hard	**72.29**	**72.42**	**542**	**555**
ResNet32					
Baseline	soft	**75.57**	**75.79**	959	984
Gradient scale	soft	75.45	75.73	**954**	**979**
Baseline	hard	61.96	62.12	769	**782**
Gradient scale	hard	**62.03**	**62.19**	**759**	788

2015). While it takes into account the characteristics of SNNs, its low training performance compared to other algorithms has limited its application in deep SNN training.

The gradient-based training algorithm of deep SNNs leverages successful training algorithms from DNNs, such as SGD and error backpropagation. One of the significant obstacles to training deep SNNs with a gradient-based algorithm was the non-differentiability of spiking neurons as depicted in Eq. 3. To overcome this, STBP with a surrogate gradient, which approximates the gradient, was proposed and could train deep SNNs successfully (Wu et al., 2018). Since then, subsequent studies on improving the training performance of deep SNNs have been published, such as threshold-dependent batch normalization (tdBN) (Zheng et al., 2021), temporal effective batch normalization (Duan et al., 2022), and temporal efficient training with time-variant target distribution (Deng et al., 2022). However, these training algorithms did not utilize local information that can improve the training performance. Recently, a study using local information for training was published, but it did not utilize relationships between spikes (Yang et al., 2022).

3. Methods

With the introduction of scalable training algorithms, such as STBP, deep SNNs have become trainable with gradients. However, these existing gradient-based algorithms for deep SNNs have a limitation in that they do not effectively consider the causal relationship between spikes of pre- and post-synaptic neurons. Thus, in this work, we propose a method to exploit the local spike information in training deep SNNs with the gradient-based algorithm.

Before explaining the proposed method, we should define the relational expression of spikes. There are various representations for spike relation, but, in this work, we adopt trace-based representation for its low computational complexity, which is suitable for deep SNNs (Morrison et al., 2008). An example of the representation is shown in Fig. 1. S_{pre} and S_{post} indicate pre- and post-synaptic spike trains, respectively. The history of spike generation in each neuron is recorded in the spike trace x as follows:

$$x_i^l(t) = e^{-1} x_i^l(t\text{-}1) + s_i^l(t). \tag{5}$$

The spike trace increases by a spike when the neuron fires and exponentially decreases at each time step of the forward (blue dotted line in Fig. 1) Each layer has pre- and post-synaptic traces (X_{pre}^l, X_{post}^l) according to its connection. With these two spike traces, we defined the relationship of spikes R as

$$R^l(t) = f^l(X_{\text{pre}}^l(t), X_{\text{post}}^l(t)), \tag{6}$$

where f^l is a relationship function of lth layer. During training, it is calculated in the backward path (orange dotted line in Fig. 1). We used convolution and outer product operations for the relationship function f of convolution and fully connected layers, respectively.

We proposed a gradient scale that adjusts the gradient of synaptic weight according to the local relationship of the spike. Inspired by STDP, the proposed algorithm encourages training with the gradient when there is a causal relationship between pre- and post-synaptic spikes. Otherwise, if there is less relationship, the algorithm hinders the training. We implemented this encouragement and hindrance by scaling the gradients of synaptic weights obtained from STBP as follows:

$$\Delta W^l = -\eta g(\frac{\delta L}{\delta W^l}, R^l) = -\eta(\alpha \frac{\delta L}{\delta W^l} \circ R^l + (1-\alpha)\frac{\delta L}{\delta W^l}), \tag{7}$$

Table 3. Comparisons with previous methods on CIFAR10 (KD: Knowledge Distillation)

Methods	Training	Architecture	Neuron	Reset	T	ANN Acc. (%)	SNN Acc. (%)
Calibration (Li et al., 2021)	conversion	ResNet20	IF	soft	32	95.46	94.78
SNN-aware training (Bu et al., 2022)	conversion	ResNet18	IF	soft	4	96.04	90.43
tdBN (Zheng et al., 2021)	direct	ResNet19	LIF	hard	4	-	92.92
TET (Deng et al., 2022)	direct	ResNet19	LIF	hard	4	-	94.44
Local tandem (Yang et al., 2022)	KD	ResNet20	LIF	soft	16	95.36	94.76
Gradient scale (Ours)	direct	ResNet20	LIF	soft	4	-	95.11
Gradient scale (Ours)	direct	ResNet20	LIF	hard	4	-	93.69

where L is a loss, η is a learning rate, g is a gradient scaling function, α is a interpolation coefficient, and $\circ$ is element-wise multiplication (Hadamard product). The gradient scaling function g receives the gradients and spike relationship as inputs. As described in Eq. 7, we adopted a simple linear interpolation function for the scaling. In this work, we set the coefficient α to 0.1 empirically.

4. Experiments

4.1. Experimental Setup

To evaluate the effectiveness of the proposed gradient scaling, we set STBP (Wu et al., 2018) and tdBN (Zheng et al., 2021) as a baseline training algorithm. For each configuration, we trained deep SNN models for 300 epochs using SGD. We adopted a learning rate schedule in which the learning rate decreased to 0.1 times every 100 epochs. We used LIF neurons with the leak constant τ of 0.9, and the time step was fixed to four. We constructed deep SNN models based on ResNet20 and ResNet32 architectures and trained them on image classification datasets, such as CIFAR10 and CIFAR100. For data augmentation, Cutmix (Yun et al., 2019) was used, and for input encoding, real value encoding was applied as in other studies (Wu et al., 2018; Zheng et al., 2021).

4.2. Experimental Results

The experimental results on CIFAR10 and CIFAR100 are presented in Tables 1 and 2, respectively. We compared the training results of the baseline and proposed methods in various configurations of the model architecture and reset method of spiking neurons. For fair and precise evaluations, we recorded the mean and maximum results for test accuracy and spike count after training four times on each configuration. For the accuracy on CIFAR10 dataset, the mean and maximum accuracy were improved when the proposed gradient scale was applied in all cases except for the case of ResNet20 with the hard reset, as shown in Table 1. Furthermore, the proposed approach can reduce the number of spikes in most cases. There are similar trends in training results on CIFAR100, as depicted in Table 2. The accuracy and spike counts are improved in most cases except ResNet32 with the soft reset and the hard reset, respectively, with the proposed methods.

Table 3 presents the comparisons of the proposed method with other deep SNN training methods. For a fair comparison, we compared the results of a model structure similar to ResNet20 on the CIFAR10 dataset. Overall, the proposed approach shows higher training performance than the recent previous methods. In the case of soft reset, when the proposed method is applied, we achieve higher accuracy with shorter time steps than the conversion methods (Li et al., 2021; Bu et al., 2022) and local tandem learning (Yang et al., 2022). In the case of hard reset, it shows higher accuracy than tdBN (Zheng et al., 2021), but lower training performance than TET (Deng et al., 2022). It was difficult to compare other metrics, such as spike counts, in this study as they were not commonly reported in previous works.

5. Discussion

The proposed training algorithm can be further improved with optimization of relation function f, scaling function g, and hyperparameters, such as α in Eq. 7. In this paper, a simple spike relation function, which only considers the positive relation between the spike traces, and scaling function was used to show the feasibility of enhancement training performance with local spike information. In order to the improvement, we can consider more complicated relation functions of pre- and post-synaptic spike traces, which consider a negative relation of the spike traces as the STDP learning rule. Furthermore, we can use other scaling functions based on theoretical analysis of deep SNNs, instead of linear interpolation as in this work.

6. Conclusion

In this paper, we proposed a training method for deep SNNs with spike-dependent local information. The pro-

posed method, which is compatible with gradient-based training algorithms, such as STBP, scales the gradient of synaptic weight according to the relationship between spike traces of adjacent neurons. We verified the effectiveness of the proposed approach with ResNet architecture on CIFAR datasets. In the future, we will improve the proposed algorithm through exploration and optimization of the spike relation function and gradient scaling function. In addition, we will evaluate the algorithm with other model architectures and datasets. We believe that by taking into account the characteristics of SNNs and utilizing local information, the training performance of deep SNNs can be improved.

Acknowledgements

This work was supported in part by the Korea Institute of Science and Technology (KIST) through 2E32260 and the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) [NRF-2021R1C1C2010454].

References

Bu, T., Fang, W., Ding, J., Dai, P., Yu, Z., and Huang, T. Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks. In *ICLR*, 2022.

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

Deng, S., Li, Y., Zhang, S., and Gu, S. Temporal efficient training of spiking neural network via gradient reweighting. In *ICLR*, 2022.

Diehl, P. U. and Cook, M. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in computational neuroscience*, 9:99, 2015.

Duan, C., Ding, J., Chen, S., Yu, Z., and Huang, T. Temporal effective batch normalization in spiking neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 34377–34390, 2022.

Han, B., Srinivasan, G., and Roy, K. 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*, pp. 13558–13567, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Izhikevich, E. M. Which model to use for cortical spiking neurons? *IEEE transactions on neural networks*, 15(5): 1063–1070, 2004.

Kim, S., Park, S., Na, B., and Yoon, S. Spiking-yolo: Spiking neural network for energy-efficient object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 11270–11277, 2020.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, 2009.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553):436–444, 2015.

Li, Y., Deng, S., Dong, X., Gong, R., and Gu, S. A free lunch from ann: Towards efficient, accurate spiking neural networks calibration. In *International Conference on Machine Learning*, pp. 6316–6325. PMLR, 2021.

Maass, W. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9): 1659–1671, 1997.

Morrison, A., Diesmann, M., and Gerstner, W. Phenomenological models of synaptic plasticity based on spike timing. *Biological cybernetics*, 98:459–478, 2008.

Na, B., Mok, J., Park, S., Lee, D., Choe, H., and Yoon, S. Autosnn: towards energy-efficient spiking neural networks. In *International Conference on Machine Learning*, pp. 16253–16269. PMLR, 2022.

Park, S., Kim, S., Na, B., and Yoon, S. T2fsnn: Deep spiking neural networks with time-to-first-spike coding. In *DAC*, 2020.

Park, S. et al. Fast and efficient information transmission with burst spikes in deep spiking neural networks. In *DAC*, 2019.

Roy, K., Jaiswal, A., and Panda, P. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

Sengupta, A., Ye, Y., Wang, R., Liu, C., and Roy, K. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in neuroscience*, 13:95, 2019.

Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.

Wu, Y., Deng, L., Li, G., Zhu, J., and Shi, L. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12: 331, 2018.

Yang, Q., Wu, J., Zhang, M., Chua, Y., Wang, X., and Li, H. Training spiking neural networks with local tandem learning. In *NeurIPS*, 2022.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Zhang, L. et al. Tdsnn: From deep neural networks to deep spike neural networks with temporal-coding. In *AAAI*, 2019.

Zheng, H., Wu, Y., Deng, L., Hu, Y., and Li, G. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11062–11070, 2021.

Zhou, Z., Zhu, Y., He, C., Wang, Y., Yan, S., Tian, Y., and Yuan, L. Spikformer: When spiking neural network meets transformer. In *ICLR*, 2023.