Proceedings of Machine Learning Research 222, 2023 ACML 2023

Training a General Spiking Neural Network with Improved Efficiency and Minimum Latency

Yunpeng Yao 202132748@mail.sdu.edu.cn
Information Science and Engineering, Shandong University, Shandong, China

Man Wu* wu.man.wi5@am.ics.keio.ac.jp
Department of Information and Computer Science, Keio University, Kanagawa, Japan

Zheng Chen chenz@sanken.osaka-u.ac.jp
SANKEN, Osaka University, Osaka, Japan

Renyuan Zhang rzhang@is.naist.jp
Division of Information Science, Nara Institute of Science and Technology, Nara, Japan
∗ *Corresponding Author*

Editors: Berrin Yanıkoğlu and Wray Buntine

1. Supplementary code

The online version contains supplementary material available at:
https://github.com/iverss1/ECML_SNN

2. Related Work

2.1. Learning Methods of SNNs

Current training methods in SNNs that achieve high performance can generally be divided into two branches: i) ANN to SNN conversion Cao et al. [2015] and ii) direct training with surrogate gradient. The conversion methods nearly maintain the accuracy of original ANNs by training the analog-spiking ANNs normally and converting them to spiking neurons by counting the fire rate Rueckauer et al. [2017]. Recent work has combined conversion and training processes to achieve near-lossless accuracy with VGG, ResNet, and their variants Deng and Gu [2021]; Li et al. [2021a]; Han et al. [2020]. However, the converted SNNs require longer time to rival the original ANN in precision due to pre-coding Rueckauer et al. [2017], which increases the SNN's latency and restricts its practical application Fang et al. [2021a]. Li et al. proposed a calibration method to improve the accuracy under fewer time steps Li et al. [2021a]. However, achieving competitive results still generally requires certain time steps (>100), which violates the hope of the low-energy costs to SNNs. Direct training of SNNs involves surrogate gradient-descent algorithms Wu et al. [2018]; Fang et al. [2021b], as the gradient with respect to threshold-triggered firing is non-differentiable Bengio et al. [2013]. To name a few, Spatio-Temporal Backpropagation Wu et al. [2018], Explicit Iterative LIF neuron Wu et al. [2019], and Threshold-Dependent Batch Normalization Zheng et al. [2021], allow gradient-based training methods to directly train SNNs using only a few time steps, such as $t = 10$ Rathi and Roy [2020], $t = 6$ Zheng et al. [2021], and unexpectedly, $t = 2$ Li et al. [2021b] in recent research.

2.2. Direct Training Framework

Recent studies have shown that incorporating advanced computational mechanisms and architectures from CNN and RNN with SNN neurons can improve performance and reduce the required time steps. The combination of convolution kernels and spiking neurons is a main trend that enables SNNs to inherit the powerful learning ability of CNNs on local areas or points Fang et al. [2021a]; Li et al. [2021b]. The earliest feedforward hierarchical spiking CNN for unsupervised learning of visual features was developed by Masquelier et al. Masquelier and Thorpe [2007]. As SNNs have evolved, Wu et al. Wu and He [2018] improved the leaky integrate-and-fire (LIF) model Hunsberger and Eliasmith [2015] to an iterative LIF model. Li et al. Li et al. [2022] further improved CNN-SNN by using a variant of convolution kernels, to obtain an optimal full-precision classification network. On the other hand, RNN-SNN methods are relatively rare. Given the adaptability of sequence models to time series, SNNs can still handle sequence features Bellec et al. [2018]. Recently, Rezaabad et al. Lotfi Rezaabad and Vishwanath [2020] developed an error backpropagation for LSTM-SNN for sequential datasets, while Datta et al. Datta et al. [2022] proposed a novel activation function in the source LSTM to jointly optimize the parameters on temporal MNIST.

This paper aims to address the trade-off issue between accuracy and time step by proposing a general-purpose framework. The theoretical feasibility of using CNNs and RNNs in our proposal is also demonstrated.

3. Related issues in window partition

3.1. Computational complexity between CNN-SNN and RNN-SNN in windows

The computational complexity of SNN (CNN/LSTM based) within a local window on an image of hw is 3.1 (Regardless of the bias):

$$\begin{aligned} \Omega(SCNN - W) &= d_x \ * \ k_h \ * \ k_w * d_h * 2 \\ \Omega(LSNN - W) &= \ d_x \ * \ d_h \ * \ 8 \ + \ d_h \ * \ (d_h \ * \ 8 \ + \ 20) \end{aligned} \tag{1}$$

$$\frac{\Omega(SCNN - W)}{\Omega(LSNN - W)} \sim \frac{d_x \ * \ k_h \ * \ k_w}{(d_x + \ d_h) * 4} \tag{2}$$

where d_x, d_h represent the dimension of input and output respectively,k_w, k_h are the size of convolution kernel. The complexity of an algorithm is dictated by its highest order term, thus the term with complexity $O(n)$ in the $\Omega(LSNN - W)$ can be omitted.Subsequently we compared the two models and obtained 2. Due to d_x generally equals d_h with the shortcuts in SNN to match the activations of the original input,2 is positive correlation according to the convolution kernel size and the kernel size usually set to $3 * 3$. When we consider a LIF cell with only one layer of convolution kernel (actually more than one layer in general) and one with a layer of LSTM, it is obvious that LSNN has smaller computational complexity.

Algorithm 1 Recomposed Computing

Input: $Dilated\ feature\ \{G_1, G_2, G_3, G_4\},\ Dilated\ window\ size\ M = \frac{3L}{4},$ $Window\ offset\ w_o = 2 * M - L\ \ (L\ is\ the\ size\ of\ fearture\ map)$

Output: $recomposed\ feature\ map\ X_l$

Weighted-Condense:

①$coverG_1G_2 = \frac{\left(G_1\left[:,\ :, \frac{w_o}{2} : \frac{3w_o}{2},\ :\right]\ +\ G_2[:,\ :, 0 : w_o,\ :]\right)}{2}$ ▷ condense overlaps region of G_1, G_2

②$G_1G_2 = cat\left(\left[G_1\left[:,\ :,\ 0 : \frac{w_o}{2},\ :\right], coverG_1G_2, G_2\left[:,\ :,\ w_o : \frac{3w_o}{2},\ :\right]\right]\right)$ ▷ recompose G_1, G_2 together

Repeat: ① and ② $for\ G_3G_4$:

$coverP = \frac{\left(G_1G_2\left[:, \frac{w_o}{2} : \frac{3w_o}{2},\ :,\ :\right]\ +\ G_3G_4[:, 0 : w_o, :,\ :]\right)}{2}$

$G = cat\left(\left[G_1G_2\left[:,\ 0 : \frac{w_o}{2},\ :,\ :\right],\ coverP, G_3G_4\left[:,\ w_o : \frac{3w_o}{2},\ :,\ :\right]\right]\right)$

Region Threshold: $X_l = G.where(Ms > Th_R) \rightarrow 1$

3.2. Weighted condense algorithm in dilated window

Weighted condense compresses the information streams X_l and M_l into the original feature size with certain weights, namely 4 and 2. More specifically, due to the division operation of weighted-condense, "median spikes (Ms)" (such as 0.25, 0.5, and 0.75) will be produced in overlaps region of recomposed X_{l+1}. For instance, if a spike appears in the overlapping area with weight 4, it becomes a Ms as 0.25. To maintain the low power advantage of event-driven, we set a region threshold (Th_R) to integrate these Ms into spikes. This threshold is set to 0.1.

4. Related computing process in proposed framework

4.1. Discrete computing process of $\partial I(t)$ and $\partial V(t)$

$$\frac{\partial I(l,t)}{\partial l} = \lim_{\Delta\delta_I \to 1} \frac{I[\iota, n] - I[\iota - \Delta\delta_I, n]}{\Delta\delta_I} - \frac{\Delta\delta_I}{2} \cdot \frac{d^2 I(l,t)}{dl^2} + O\left(\Delta\delta_I^2\right)$$

$$\begin{aligned} &\frac{\partial V(G_{m1}, G_{m2}, t)}{\partial G_{m1}} + \frac{\partial V(G_{m1}, G_{m2}, t)}{\partial G_{m2}} \\ = &\lim_{\Delta\delta_{m1} \to 1} \frac{V[\eta, \upsilon, n] - V[\eta - 1, \upsilon, n]}{\Delta\delta_{m1}} + \lim_{\Delta\delta_{m2} \to 1} \frac{V[\eta, \upsilon, n] - V[\eta, \upsilon - 1, n]}{\Delta\delta_{m2}} \\ &- \mathbb{H}(m_1, m_2) + O\left(\Delta\delta_V^2\right) \end{aligned}$$

Taylor Expansion of synaptic current $I(t)$ and membrane potential$V(t)$ are presented as above. Since both functions are not changing with respect to time steps(t) and the deepening of the network is a linear change, the second derivative of $I(l,t)$ does not exist. The accumulation of membrane potential$V(t)$ should be treated as a Taylor expansion of a multivariate composite function, and the variables change along with two directions. Where $\mathbb{H}$ is Hessian Matrix of

Table 1: ∇D comparison given different θ.

$\mathbb{S}(\theta) = 16 * [sin(\frac{3\theta\pi th}{2}) * sin(\frac{\theta\pi th}{4})^2] / \left(\theta^2\pi^2\right)$					
θ	0.1	0.3	0.5	0.7	0.9
$\mathbb{S}(th = 0.5)$	0.05	0.16	0.22	0.24	0.20
∇D	-0.31	-0.21	-0.14	-0.13	-0.17

membrane potential.

$$\mathbb{H}\left(m, m'\right) = \begin{bmatrix} \frac{\partial^2 f(M)}{\partial m_1^2} & \frac{\partial^2 f(M)}{\partial m_1 \partial m_2} & \cdots & \frac{\partial^2 f(M)}{\partial m_1 \partial m_n} \\ \frac{\partial^2 f(M)}{\partial m_2 \partial m_1} & \frac{\partial^2 f(M)}{\partial m_2^2} & \cdots & \frac{\partial^2 f(M)}{\partial m_2 \partial m_n} \\ \vdots & \vdots & \iddots & \vdots \\ \frac{\partial^2 f(M)}{\partial m_n \partial m_1} & \frac{\partial^2 f(M)}{\partial m_n \partial m_2} & \cdots & \frac{\partial^2 f(M)}{\partial m_n^2} \end{bmatrix}$$

As shown in the Matrix, higher order expansion composite function with m_1, m_2 is 0. There are two reasons for this. (i) Since deepening and sliding are both first order linear operation, Continuous second derivative does not exist in accumulation of membrane potential $V(t)$ with either direction. (ii)The two variables of a multivariate function are independent. Therefore, the joint derivative of the function is 0.

$$\Omega\left(\frac{\partial V_p(G,t)}{\partial G}\right) \tag{3}$$

$$\Rightarrow \Omega\left(\lim_{\Delta\delta_{m1}\to 1} \frac{V[\eta,\upsilon] - V[\eta-1,\upsilon]}{\Delta\delta_{m1}} + \lim_{\Delta\delta_{m2}\to 1} \frac{V[\eta,\upsilon] - V[\eta,\upsilon-1]}{\Delta\delta_{m2}}\right) \tag{4}$$

$$\Rightarrow V[\eta,\upsilon] - \alpha V[\eta-1,\upsilon-1] \tag{5}$$

Finally, we project the potentials in both directions onto a common space, constraining the membrane potential that ultimately affects neurons by both directional variables simultaneously as Eq. 5

4.2. The results of ∇D with different θ

$$\mathbb{S}(\theta) = 16 * [sin(\frac{3\theta\pi th}{2}) * sin(\frac{\theta\pi th}{4})^2] / \left(\theta^2\pi^2\right) \tag{6}$$

To detailed analysis over-activation issue, we use the area difference ∇D ranged in $R \in (th, 2 * th)$ between the fusion surface ($\mathbb{S}_{Surface} = \Omega(\mathbb{X}, \mathbb{Y})$) and the linear plane ($\mathbb{S}_{Plane} = \mathbb{X} + \mathbb{Y}$) as an indicator. This range is chosen because linear fusion only causes over-activation within this range. The area of the fusion surface $\mathbb{S}_{Surface}$ can be simplified as Eq. 6 which depends only on the radian factors θ.

The area of the projection plane $\mathbb{S}_{Plane}$ is determined by the threshold of the LIF model, the difference ∇D depends only on the radian factors θ. Theoretically, the greater the absolute difference, the greater the possibility of over-activation. $\mathbb{S}_{Surface}$ and ∇D are resulted in the Table. ?? with various θ, the over-activation of spiking neurons caused by membrane potential fusion can be suppressed with each θ. Different θ also lead to different levels of inhibition, when excessive activation occurs.

Table 2: Accuracy of RNN-SNN models on CIFAR10 and Cifar100 datasets.

Models/Dateset	MLP	RNN	Bi-RNN	GRU
CIFAR10	84.32	88.32	89.71	91.66
CIFAR100	×	64.33	64.17	65.41

5. Experimental details and additional exploration

We modify the ResMLP and VIT architectures slightly to facilitate ANN-SNN conversion. Patch-Merging is used for down sample, other architectures are same as Touvron et al. [2022] and Liu et al. [2021] The architectural details are:

ResMLP12: 48, F-48-shorcut , 48, PM, 96, F-96-shorcut, 96, PM, (192, F-192-shorcut)×3, 384, F-384, 384, C

VIT12: 48, MLP-48 , 48, PM, 96, MLP-96, 96, PM, (192, MLP-96)×3, 384, MLP-384, 384, C

5.1. Experiments settings

We evaluate the performance of the proposed framework in terms of classification accuracy and inference latency on the CIFAR10 Krizhevsky et al. [2009], CIFAR100 Krizhevsky et al. [2009], and Tiny-ImageNet Le and Yang [2015] datasets.

5.2. Training Hyperparameters

Standard data augmentation techniques are applied for image datasets such as padding by 4 pixels on each side, and 32×32 cropping by randomly sampling from the padded image or its horizontally flipped version (with 0.5 probability of flipping). The original 32×32 images are used during testing. Both training and testing data are normalized using channel-wise mean and standard deviation calculated from training set. Both SNN (CNN and RNN) are trained with cross-entropy loss with stochastic gradient descent optimization (weight decay=0.00002, momentum=0.9). We train the SNNs for 300 and 250 epochs for CIFAR and TinyImageNet respectively, with an initial learning rate of 0.05 and warmup learning rate is 0.001. The learning rate noise is limit in 0.67. The ANNs are trained with gradient clipping rather batch-norm (BN), the Gradient clipping mode is the normal version, clip-grad is set to 20.

Additionally, dropout Srivastava et al. [2014] is used as the regularizer with a constant dropout mask with dropout probability=0.1 training the SNNs. Since mix-up Yun et al. [2019] and augmentation splits Van Dyk and Meng [2001] causes significant information enhancing in training, we use mixup alpha as 0.1, augmentation splits as 2-6. During SNN training, the weights are mainly initialized using as initialization Chowdhury et al. [2022]. Upon conversion, at each training iteration with 1 time step, the SNNs are trained for 300 epochs with cross-entropy loss and adam optimizer (weight decay=0.0001). Initial learning rate is chosen as 0.001, which is decayed by 0.1.

5.3. Results of other RNN-SNN model

Table. 2 presents the results of different SNN model trained with proposed framework in RNN baseline. Performance of LSTM-SNN are detailed analysed in main paper. The proposed framework can still converge other RNN baseline networks, although the effect is inferior to LSTM based one.

5.4. Training deeper network

We also tested the convergence of our framework as the network deepened. As shown in Fig. 1, when the network layer number becomes 12, 20, 24, 36, the model can still be trained correctly, Whether it's based on CNN or LSTM. And in some cases, deeper networks achieve better accuracy. However, to ensure the energy consumption of the model, we abandoned some accuracy and adopted the 12 layer network in the main paper and experiments.

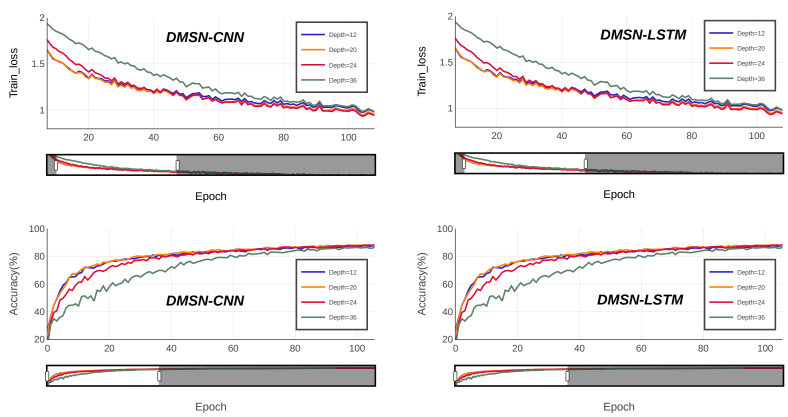

Figure 1: Deeper layers in our framework.

References

Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. *Advances in neural information processing systems*, 31, 2018.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

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.

Sayeed Shafayet Chowdhury, Nitin Rathi, and Kaushik Roy. Towards ultra low latency spiking neural networks for vision and sequential tasks using temporal pruning. In *ECCV*, pages 709–726, 2022.

Gourav Datta, Haoqin Deng, Robert Aviles, and Peter A Beerel. Towards energy-efficient, low-latency and accurate spiking lstms. *arXiv preprint arXiv:2210.12613*, 2022.

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

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*, 34:21056–21069, 2021a.

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*, pages 2661–2671, 2021b.

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.

Eric Hunsberger and Chris Eliasmith. Spiking deep networks with lif neurons. *arXiv preprint arXiv:1510.08829*, 2015.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Wenshuo Li, Hanting Chen, Jianyuan Guo, Ziyang Zhang, and Yunhe Wang. Brain-inspired multilayer perceptron with spiking neurons. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 783–793, 2022.

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*, pages 6316–6325. PMLR, 2021a.

Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Advances in Neural Information Processing Systems*, 34:23426–23439, 2021b.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

Ali Lotfi Rezaabad and Sriram Vishwanath. Long short-term memory spiking networks and their applications. In *International Conference on Neuromorphic Systems 2020*, pages 1–9, 2020.

Timothée Masquelier and Simon J Thorpe. Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS computational biology*, 3(2):e31, 2007.

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.

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.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

David A Van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1):1–50, 2001.

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.

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*, volume 33, pages 1311–1318, 2019.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11062–11070, 2021.