TRAINING DEEP SPIKING NEURAL NETWORKS FOR ENERGY-EFFICIENT NEUROMORPHIC COMPUTING

Gopalakrishnan Srinivasan[1], *Chankyu Lee*[1], *Abhronil Sengupta*[2], *Priyadarshini Panda*[3], *Syed Shakib Sarwar*[1], *and Kaushik Roy*[1]

[1]Purdue University, [2]Penn State University, [3]Yale University

ABSTRACT

Spiking Neural Networks (SNNs), widely known as the third generation of neural networks, encode input information temporally using sparse spiking events, which can be harnessed to achieve higher computational efficiency for cognitive tasks. However, considering the rapid strides in accuracy enabled by state-of-the-art Analog Neural Networks (ANNs), SNN training algorithms are much less mature, leading to accuracy gap between SNNs and ANNs. In this paper, we propose different SNN training methodologies, varying in degrees of bio-fidelity, and evaluate their efficacy on complex image recognition datasets. First, we present biologically plausible Spike Timing Dependent Plasticity (STDP) based deterministic and stochastic algorithms for unsupervised representation learning in SNNs. Our analysis on the CIFAR-10 dataset indicates that STDP-based learning rules enable the convolutional layers to self-learn low-level input features using fewer training examples. However, STDP-based learning is limited in applicability to shallow SNNs ($\leq$4 layers) while yielding considerably lower than state-of-the-art accuracy. In order to scale the SNNs deeper and improve the accuracy further, we propose conversion methodology to map off-the-shelf trained ANN to SNN for energy-efficient inference. We demonstrate 69.96% accuracy for VGG16-SNN on ImageNet. However, ANN-to-SNN conversion leads to high inference latency for achieving the best accuracy. In order to minimize the inference latency, we propose spike-based error backpropagation algorithm using differentiable approximation for the spiking neuron. Our preliminary experiments on CIFAR-10 show that spike-based error backpropagation effectively captures temporal statistics to reduce the inference latency by up to 8$\times$ compared to converted SNNs while yielding comparable accuracy.

Index Terms— SNN, Stochastic STDP, ANN-SNN conversion, Spike-based error backpropagation, Surrogate gradient backpropagation

This work was supported in part by the Center for Brain Inspired Computing (C-BRIC), one of the six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, by the Semiconductor Research Corporation, the National Science Foundation, Intel Corporation, the DoD Vannevar Bush Fellowship, and by the U.S. Army Research Laboratory and the U.K. Ministry of Defense under Agreement Number W911NF-16-3-0001.

1. INTRODUCTION

Deep learning Analog Neural Networks (ANNs) have revolutionized the field of machine learning by redefining the state-of-the-art for a variety of artificial intelligence tasks including object recognition, gesture recognition, and natural language processing, among other tasks. The superhuman performance offered by the deep learning models has been achieved by expending significantly high computational energy, which is believed to be more than three orders of magnitude compared to the human brain. In the quest for more computationally efficient neural architectures, recent research has been directed towards models that are inspired by the computation and communication capability of the human brain to mimic its efficiency for cognitive tasks. The next generation of neural networks with added bio-fidelity are known as Spiking Neural Networks (SNNs). SNNs process information temporally using spikes in an event-driven manner, which can be exploited for obtaining higher energy efficiency in special-purpose neuromorphic chips [1, 2, 3].

Training SNNs is challenging because of the need to effectively use spike timing information while overcoming the discontinuous nature of spike trains. We present SNN training approaches with varying levels of bio-fidelity as shown in Fig. 1. Spike Timing Dependent Plasticity (STDP) is an experimentally observed biological phenomenon used for the unsupervised training of SNNs. STDP-based learning rules [4, 5] modify a synaptic weight based on the temporal correlation between the input and output neuronal spike trains. STDP-based unsupervised learning has been demonstrated for fully-connected [5] and convolutional SNNs [6, 7, 8, 9, 10]. In this work, we propose STDP-based stochastic algorithms [11, 12] for energy- and memory-efficient learning in binary SNNs. Stochastic STDP encodes spike timing information in the switching probability of the binary synapses. We demonstrate efficient low-level feature learning using stochastic STDP for binary convolutional SNNs, resulting in $>$20$\times$ kernel memory compression compared to full-precision (32-bit) SNN with comparable, albeit lower, accuracy. However, the efficacy of STDP deteriorates for deep SNNs with more than four layers, leading to much lower than acceptable accuracy for complex recognition tasks.

In order to realize large-scale SNNs capable of inferring

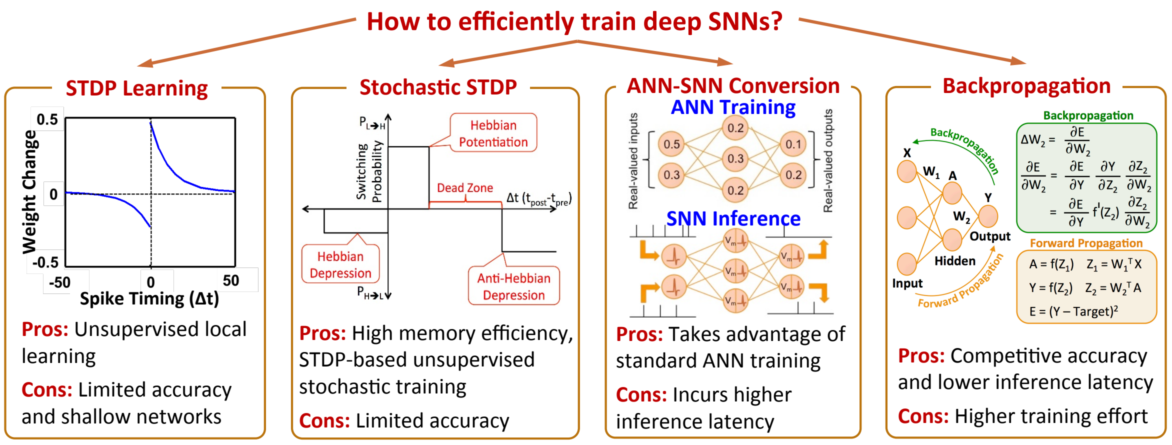
How to efficiently train deep SNNs?
STDP Learning
Weight Change
Spike Timing (Δt)
Pros: Unsupervised local learning
Cons: Limited accuracy and shallow networks
Stochastic STDP
Hebbian Potentiation
Dead Zone
Hebbian Depression
Anti-Hebbian Depression
Switching Probability
Pros: High memory efficiency, STDP-based unsupervised stochastic training
Cons: Limited accuracy
ANN-SNN Conversion
ANN Training
SNN Inference
Real-valued inputs
Real-valued outputs
Pros: Takes advantage of standard ANN training
Cons: Incurs higher inference latency
Backpropagation
Forward Propagation
Input
Hidden
Output
Pros: Competitive accuracy and lower inference latency
Cons: Higher training effort

Fig. 1. Training methodologies varying in degrees of bio-fidelity for deep spiking neural networks.

real-world datasets including ImageNet, we propose methodology to convert off-the-shelf trained ANN consisting of artificial ReLU neurons to SNN composed of Integrate-and-Fire (IF) neurons [13]. The ANN-to-SNN conversion approaches [14, 15, 16, 17] leverage backpropagation algorithms well developed for training ANNs. We present threshold initialization strategy based on ANN-trained weights and SNN spike statistics to significantly reduce the accuracy loss during SNN inference, and report the best accuracy to date for deep SNNs on ImageNet. However, the high accuracy is achieved by trading off inference latency, which is in the order of hundreds of time-steps for ImageNet [17, 13].

The high inference latency incurred by the conversion approaches stems from their inability to incorporate temporal statistics during training. Recent works have attempted to integrate spike timing information by directly training SNNs using spike-based error backpropagation algorithms [18, 19, 20, 21, 22]. We propose spike-based error backpropagation using low-pass filtered spike train as the differentiable approximation for the Leaky-Integrate-and-Fire (LIF) neuronal activation [23]. Our preliminary results indicate that the proposed algorithm efficiently incorporates spike timing to train deep SNNs ($\leq$ 11 layers) for achieving competitive accuracy using fewer time-steps. As a result, SNN trained using spike-based error backpropagation can provide larger computational energy savings relative to those trained with conversion approaches while yielding comparable accuracy.

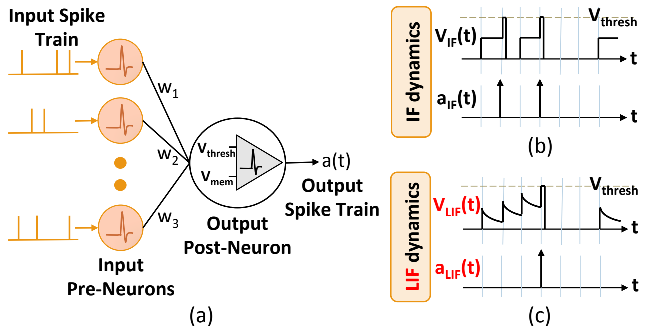

Fig. 2. **(a)** Core computational unit of an SNN composed of input (pre) neurons connected via synaptic weights to output (post) neuron. **(b)** Integrate-and-Fire (IF) neuron dynamics. **(c)** Leaky-Integrate-and-Fire (LIF) neuron dynamics.

2. SNN PRELIMINARIES

The core building block of an SNN is a set of input (pre) neurons connected via synaptic weights to output (post) neuron as shown in Fig. 2(a). For object recognition tasks, the input neurons are formed by mapping the image pixels to Poisson spike trains firing at a rate proportional to the corresponding intensities. The input pre-spikes get modulated by the synaptic weights to produce resultant current into the post-neuron. The spiking dynamics of the post-neuron are typically emulated using either Integrate-and-Fire (IF) or Leaky-Integrate-and-Fire (LIF) model [24]. Both IF and LIF neurons integrate the input current into their respective state or membrane potential. The key difference between the models is that the IF neuronal membrane potential remains constant in the time period between successive inputs while the LIF neuronal membrane potential exhibits exponential decay as shown in Figs. 2(b) and 2(c), respectively. We use LIF neurons for SNN trained using STDP or spike-based error backpropagation to effectively capture more temporal dependencies. On the other hand, we use IF neurons for SNN obtained from ANN, trained with ReLU activations, due to equivalency between the ReLU and IF transfer functions as will be explained in section 3.2.

3. SNN TRAINING APPROACHES AND RESULTS

3.1. Spike Timing Dependent Plasticity (STDP)

STDP is a bio-plausible localized learning mechanism, which postulates that the change in the synaptic weight connecting a pair of pre- and post-neurons depends on the time difference between the respective spiking instants as shown in Fig. 1. STDP-based learning rules capture spike timing information in the bit-precision of multilevel weights. In order to achieve memory-efficient learning in ultra-low precision SNNs composed of binary weights, we present *stochastic-STDP* learning rule where spike timing is embedded in the switching probability of binary synaptic weights as illustrated in Fig. 1. The ultra-low precision SNNs trained using stochastic-STDP can be efficiently realized using CMOS or emerging device technologies [11] to enable on-chip training and inference in edge devices. Based on stochastic-STDP, a binary weight is probabilistically potentiated for small positive time difference between the pre- and post-spikes, which conforms to the Hebbian learning theory [25]. The binary weight is probabilistically depressed for small negative (Hebbian in nature) or large positive spike timing difference (anti-Hebbian in nature). In addition, we incorporate *dead zone* with zero switching probability to maintain optimal balance between the potentiation and depression weight updates as shown in Fig. 1.

We demonstrate stochastic-STDP based representation learning using binary convolutional SNN composed of stacked convolutional layers for hierarchical input feature extraction, spatial pooling layers for dimensionality reduction, and fully-connected layers trained using backpropagation for inference. We train the binary kernels (or filters) forming the convolutional layers in a greedy layer-wise unsupervised manner using stochastic-STDP. We introduce dropout (proxy for lateral inhibition) and homeostasis (firing threshold adaptation) among the output maps in every layer to enable the binary filters to learn diverse input features. We first trained 36C3-2P-128FC-10FC SNN, composed of single convolutional (C) layer with 36 3×3 binary filters trained with stochastic-STDP on 10,000 MNIST digits, whose activations are pooled (P) and fed to fully-connected (FC) layers trained on the entire dataset for inference. We obtain 98.54% accuracy with $39.5\times$ kernel memory compression relative to full-precision (32-bit) convolutional SNN benchmarked in prior works as shown in [12]. We then trained wider 256C3-2P-1024FC-10FC SNN, consisting of single convolutional layer with 256 3×3 binary filters trained with stochastic-STDP on 5,000 CIFAR-10 images. We achieve 66.23% accuracy with $21\times$ kernel memory compression relative to full-precision (32-bit) convolutional SNN as shown in [12]. Finally, we scaled up to three-layer deep convolutional SNN in the presence of residual connections. However, the accuracy saturates or even deteriorates for deep SNNs because of the inability of STDP-based unsupervised learning to effectively combine the low-level features into meaningful high-level features.

3.2. ANN-SNN Conversion

ANN-SNN conversion entails the following steps:

1. Train the ANN with constraints that include removing batch normalization layers and bias neurons, and using average pooling instead of max pooling operation.

2. Transfer the trained weights from ANN to SNN.

3. Initialize the firing threshold of the IF neurons in every SNN layer so that the spike rates approximately match the corresponding ANN ReLU activations as depicted in Fig. 3.

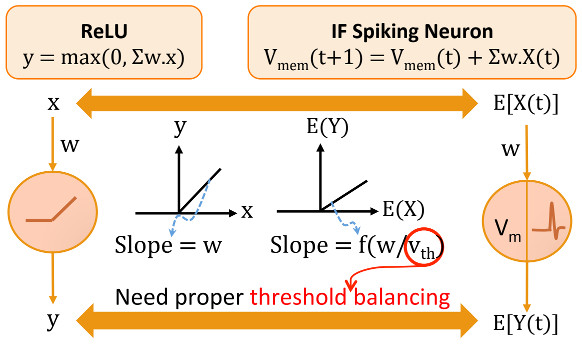

Fig. 3. Methodology for converting ANN, trained with ReLU activations, to SNN composed of IF neurons for inference.

The second step, which is known as "threshold balancing", is the key to minimizing accuracy loss during inference. Too low a threshold deteriorates the recognition capability of the SNN, resulting in accuracy degradation. On the other hand, too high a threshold significantly increases the inference latency. We present the *spike-norm* algorithm [13] for determining the thresholds in a layer-wise manner based on the SNN spike statistics and transferred weights as described below. We first generate Poisson spike trains for the entire training set and propagate them to the first layer of the SNN, initialized with the ANN-trained weights. We then record the weighted spike-input received by the first layer and heuristically set the threshold to be the maximum weighted spike-input across time-steps over the entire training dataset. Once the threshold for a particular layer is set, the same process is carried out sequentially for the remaining layers of the SNN. We experimentally find that the proposed *spike-norm* algorithm, on account of using spike statistics, yields thresholds that provide optimal balance between inference latency and accuracy. We demonstrate competitive accuracy for VGG and residual SNNs on the CIFAR-10 and ImageNet datasets. VGG-16 SNN, converted using *spike-norm*, achieves 91.55% accuracy on CIFAR-10 and 69.96% (best accuracy to date for SNNs) on ImageNet. We find that the average number of accumulate (AC) operations for the VGG16-SNN per inference is $1.975\times$ more than the number of multiply-and-accumulate (MAC) operations for the equivalent ANN due to relatively higher inference latency. However, AC operation can be up to an order of magnitude energy-efficient over MAC operation [26], leading to much improved computational efficiency for the SNN.

3.3. Spike-Based Error Backpropagation

The standard ANN backpropagation technique cannot be directly applied for training SNNs due to the discontinuous nature of the spike trains produced by the LIF neurons. The derivative of the spiking neuronal output with respect to the

Table 1. Accuracy-efficiency results for SNN, trained with different approaches on CIFAR-10, over ANN.

Model	Training Method	Accuracy (%)	Latency	# Synaptic Ops	Efficiency (int*)	Efficiency (float*)
ResNet-4 SNN [12]	Stochastic STDP	66.50	100	1.32E8 8-bit AC‡	12.5$\times$	8.7$\times$
VGG9-ANN [23]	ANN backprop	91.98	1	1.95E8 32-bit MAC	1$\times$	1$\times$
VGG9-SNN [23]	Conversion	90.45	800	1.26E9 32-bit AC	4.8$\times$	0.6$\times$
VGG9-SNN [23]	SNN backprop	90.45	100	7.03E8 32-bit AC	8.6$\times$	1.1$\times$
VGG16-SNN [13]	Conversion	91.55	500	1.975$\times$#Ops$_{ANN}$	15.7$\times$**	2.1$\times$**

‡Model uses ReLU-based fully-connected layers, incurring additional 1.45E7 MACs, which are factored into the efficiency results. Also, 8-bit AC is sufficient since the model uses binary weights and activations. Energy$_{8bit-AC}$ is assumed to be 0.25$\times$Energy$_{32bit-AC}$.
*Efficiency (Energy$_{ANN}$/Energy$_{SNN}$) is computed using energy numbers in [26] for int/float AC and MAC ops. Memory energy is ignored.
**Efficiency of VGG-16 SNN is with respect to VGG-16 ANN. Efficiency of remaining SNN models is with respect to VGG-9 ANN.

weighted spike-input is not defined at the spiking timing instants and is zero elsewhere. The Dirac delta gradient would preclude training convergence since it does not provide useful information during error backpropagation. We present spike-based backpropagation using approximate LIF neuronal activation and surrogate- or pseudo-derivative for effectively backpropagating error gradients. We propose using low-pass filtered spike train as the differentiable approximation for the LIF neuronal activation. The LIF pseudo-derivative is then computed as

$$LIF\ pseudo{-}derivative = \frac{1}{v_{th} + \alpha} \qquad (1)$$

where v_{th} is the neuronal firing threshold and α is the correction term to account for the leaky neuronal dynamics. Higher the threshold, lower is the rate of change of spiking neuronal output with respect to the input, and vice versa. We refer the readers to [23] for a detailed analysis of the proposed approximations. In addition, we also adopt commonly used regularization algorithms such as dropout and L2 regularization to achieve good generalization during training.

We demonstrate our spike-based backpropagation algorithm on VGG and residual SNNs consisting of 9-11 layers. Our results on CIFAR-10 indicate that the proposed algorithm efficiently integrates spike timing to train deep SNNs that yield competitive accuracy. VGG9-SNN provided 90.45% accuracy, which is comparable, albeit slightly lower, to that obtained using the conversion approach. However, the comparable accuracy is obtained with 8$\times$ fewer time-steps as depicted in Fig. 4. Interestingly, Fig. 4 also shows that the accuracy drops as the number of inference time-steps is increased beyond that (100 time-steps) used during backpropagation-based training. Finally, we trained ResNet-11 SNN and found that it offers higher accuracy (90.95%) compared to that provided by the conversion approach (90.15%), indicating that capturing temporal dependencies is imperative for architectures with residual (or skip) connections.

4. DISCUSSION

Table 1 summarizes the accuracy-efficiency trade-offs provided by SNN, trained using different approaches on CIFAR-10, over ANN. ResNet-4 SNN, composed of binary filters trained using stochastic-STDP, offers high degree of computational efficiency while yielding lower accuracy. It can enable energy- and memory-efficient training as well as inference in edge devices for low-complexity tasks. VGG-9 SNN, obtained through ANN-SNN conversion, provides comparable (slightly lower) accuracy with up to 4.8$\times$ reduction in computational energy over ANN. However, the energy savings are limited, especially while using floating-point hardware, due to the high inference latency (800 time-steps). On the other hand, VGG-9 SNN, trained using spike-based backpropagation, offers comparable accuracy to that provided by converted SNN with only 100 time-steps, leading to 1.1-8.6$\times$ improvement in computational energy efficiency over ANN across floating- and fixed-point hardware. However, scaling the spike-based backpropagation algorithm for training much deeper SNNs on ImageNet is challenging due to high training time and memory requirements incurred for integrating error gradients through time. Deep SNNs can offer higher computational energy benefits than shallow SNNs as shown in Table 1 for VGG-16 SNN (obtained via conversion), since spiking sparsity increases exponentially with depth. Work has started in earnest to investigate hybrid conversion-backpropagation methodology for training deep SNNs that can provide the best trade-off between accuracy and computational efficiency.

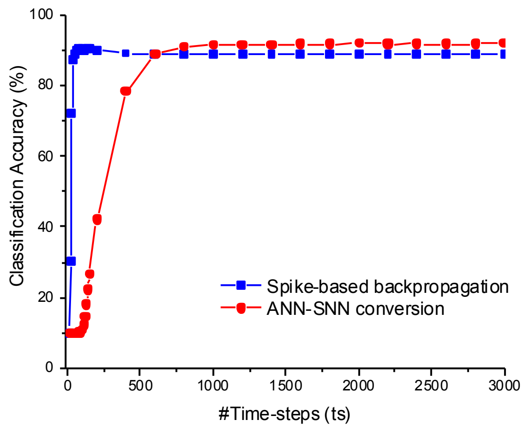

Fig. 4. Accuracy of VGG-9 SNN, trained using spike-based error backpropagation and ANN-SNN conversion on CIFAR-10, versus the number of inference time-steps.

5. REFERENCES

[1] P. A. Merolla *et al.*, “A million spiking-neuron integrated circuit with a scalable communication network and interface,” *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[2] M. Davies *et al.*, “Loihi: A neuromorphic manycore processor with on-chip learning,” *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[3] P. Blouw *et al.*, “Benchmarking keyword spotting efficiency on neuromorphic hardware,” in *Proceedings of the 7th Annual Neuro-inspired Computational Elements Workshop*. ACM, 2019, p. 1.

[4] S. Song *et al.*, “Competitive hebbian learning through spike-timing-dependent synaptic plasticity,” *Nature neuroscience*, vol. 3, no. 9, p. 919, 2000.

[5] P. U. Diehl and M. Cook, “Unsupervised learning of digit recognition using spike-timing-dependent plasticity,” *Frontiers in computational neuroscience*, vol. 9, p. 99, 2015.

[6] T. Masquelier and S. J. Thorpe, “Unsupervised learning of visual features through spike timing dependent plasticity,” *PLoS computational biology*, vol. 3, no. 2, p. e31, 2007.

[7] G. Srinivasan *et al.*, “Stdp-based unsupervised feature learning using convolution-over-time in spiking neural networks for energy-efficient neuromorphic computing,” *J. Emerg. Technol. Comput. Syst.*, vol. 14, no. 4, pp. 44:1–44:12, Nov. 2018.

[8] S. R. Kheradpisheh *et al.*, “Stdp-based spiking deep convolutional neural networks for object recognition,” *Neural Networks*, vol. 99, pp. 56–67, 2018.

[9] C. Lee *et al.*, “Deep spiking convolutional neural network trained with unsupervised spike timing dependent plasticity,” *IEEE Transactions on Cognitive and Developmental Systems*, pp. 1–1, 2018.

[10] P. Ferré *et al.*, “Unsupervised feature learning with winner-takes-all based stdp,” *Frontiers in computational neuroscience*, vol. 12, p. 24, 2018.

[11] G. Srinivasan *et al.*, “Magnetic tunnel junction based long-term short-term stochastic synapse for a spiking neural network with on-chip stdp learning,” *Scientific reports*, vol. 6, p. 29545, 2016.

[12] G. Srinivasan and K. Roy, “Restocnet: Residual stochastic binary convolutional spiking neural network for memory-efficient neuromorphic computing,” *Frontiers in Neuroscience*, vol. 13, p. 189, 2019.

[13] A. Sengupta *et al.*, “Going deeper in spiking neural networks: Vgg and residual architectures,” *Frontiers in neuroscience*, vol. 13, 2019.

[14] P. U. Diehl *et al.*, “Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing,” in *International Joint Conference on Neural Networks (IJCNN)*. Killarney, Ireland: 2015, pp. 1–8.

[15] Y. Cao *et al.*, “Spiking deep convolutional neural networks for energy-efficient object recognition,” *International Journal of Computer Vision*, vol. 113, no. 1, pp. 54–66, 2015.

[16] E. Hunsberger and C. Eliasmith, “Training spiking deep networks for neuromorphic hardware,” *arXiv preprint arXiv:1611.05141*, 2016.

[17] B. Rueckauer *et al.*, “Conversion of continuous-valued deep networks to efficient event-driven networks for image classification,” *Frontiers in neuroscience*, vol. 11, p. 682, 2017.

[18] P. Panda and K. Roy, “Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition,” in *2016 International Joint Conference on Neural Networks (IJCNN)*. Vancouver, British Columbia, Canada: IEEE, 2016, pp. 299–306.

[19] J. H. Lee *et al.*, “Training deep spiking neural networks using backpropagation,” *Frontiers in neuroscience*, vol. 10, p. 508, 2016.

[20] G. Bellec *et al.*, “Long short-term memory and learning-to-learn in networks of spiking neurons,” in *Advances in Neural Information Processing Systems*, Montral, Quebec, Canada, 2018, pp. 787–797.

[21] S. B. Shrestha and G. Orchard, “Slayer: Spike layer error reassignment in time,” in *Advances in Neural Information Processing Systems*, Montral, Quebec, Canada, 2018, pp. 1412–1421.

[22] E. O. Neftci *et al.*, “Surrogate gradient learning in spiking neural networks,” *arXiv preprint arXiv:1901.09948*, 2019.

[23] C. Lee *et al.*, “Enabling spike-based backpropagation in state-of-the-art deep neural network architectures,” *arXiv preprint arXiv:1903.06379v3*, 2019.

[24] P. Dayan and L. F. Abbott, *Theoretical neuroscience*. Cambridge, MA, USA: The MIT Press, 2001, vol. 806.

[25] D. Hebb, “The organization of behavior,” 1949.

[26] S. Han *et al.*, “Learning both weights and connections for efficient neural network,” in *Advances in neural information processing systems*, 2015, pp. 1135–1143.