Article

Quantized Weight Transfer Method Using Spike-Timing-Dependent Plasticity for Hardware Spiking Neural Network

Sungmin Hwang ^{1,2} , Hyungjin Kim ^{3,*} and Byung-Gook Park ^{1,2,*}

¹ Inter-University Semiconductor Research Center (ISRC), Seoul National University, Seoul 08826, Korea; smh88@snu.ac.kr

² Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Korea

³ Department of Electronic Engineering, Inha University, Incheon 22212, Korea

* Correspondence: hkim@inha.ac.kr (H.K.); bgpark@snu.ac.kr (B.-G.P.); Tel.: +82-32-860-7417 (H.K.); +82-2-880-7282 (B.-G.P.)

Abstract: A hardware-based spiking neural network (SNN) has attracted many researcher's attention due to its energy-efficiency. When implementing the hardware-based SNN, offline training is most commonly used by which trained weights by a software-based artificial neural network (ANN) are transferred to synaptic devices. However, it is time-consuming to map all the synaptic weights as the scale of the neural network increases. In this paper, we propose a method for quantized weight transfer using spike-timing-dependent plasticity (STDP) for hardware-based SNN. STDP is an online learning algorithm for SNN, but we utilize it as the weight transfer method. Firstly, we train SNN using the Modified National Institute of Standards and Technology (MNIST) dataset and perform weight quantization. Next, the quantized weights are mapped to the synaptic devices using STDP, by which all the synaptic weights connected to a neuron are transferred simultaneously, reducing the number of pulse steps. The performance of the proposed method is confirmed, and it is demonstrated that there is little reduction in the accuracy at more than a certain level of quantization, but the number of pulse steps for weight transfer substantially decreased. In addition, the effect of the device variation is verified.

Keywords: neuromorphic system; hardware-based SNN; offline training; spike-timing-dependent plasticity; weight quantization

Citation: Hwang, S.; Kim, H.; Park, B.-G. Quantized Weight Transfer Method Using Spike-Timing-Dependent Plasticity for Hardware Spiking Neural Network. *Appl. Sci.* **2021**, *11*, 2059. <https://doi.org/10.3390/app11052059>

Academic Editor: Eyad H. Abed

Received: 19 January 2021

Accepted: 23 February 2021

Published: 25 February 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Artificial neural network (ANN) has become a core technology that leads the modern artificial intelligence (AI) industry, thus it has been utilized in various fields such as image recognition, natural language processing, autonomous vehicles, and so on [1–3]. Given that the operation of ANN is based on vector-matrix multiplication (VMM), which is basically parallel calculation, the conventional computing system (called von Neumann architecture) where central processing units (CPUs) and memory are connected in series is not suitable for ANN [4,5]. In recent years, a neuromorphic computing system has been considered as a potential candidate for future computing architecture to overcome the limitation of von Neumann architecture [6–8]. The basic components consisting of the neuromorphic system are synapses and neurons, which are connected in parallel. An integrate-and-fire (I&F) neuron circuit receives pre-synaptic inputs and spatiotemporally integrates them into a membrane. When the membrane potential exceeds a threshold of the neuron, a spike is generated, and it is transmitted to the next neurons. Conventional I&F neuron circuits have been fabricated by the complementary metal-oxide-semiconductor process with a membrane capacitor [9,10]; however, in recent years, there have been some research about I&F neurons based on memristors such as resistive random-access memory (RRAM), phase-change random-access memory (PRAM), and magnetic random-access

memory (MRAM), which have advantages in energy and area [11,12]. A synapse adjusts the strength of the connection between neurons, which is called a synaptic weight. As synaptic devices, non-volatile memories, such as flash memory, memristors, etc., have been actively studied [13–16]. Conductance of the synaptic device changes according to its program (PGM) and erase (ERS) states, thus that the strength of the signal transmitted to the neuron (the weight) can be adjusted.

When implementing a hardware-based spiking neural network (SNN) using I&F neuron circuits and synaptic devices, there are two methods to train the network: Online and offline learning [17]. Online learning is capable of self-learning in hardware without the help of software through biologically plausible learning algorithms such as spike-timing-dependent plasticity (STDP). STDP is a Hebbian learning algorithm, which has been widely used for unsupervised learning, and supervised learning based on STDP has also been actively studied [18–20]; however, its performance has not yet reached state-of-the-art performance. Therefore, offline learning where weights are trained by a gradient descent algorithm in software-based ANN and transferred to the hardware-based SNN has attracted a lot of attention [21–23].

In order to use the hardware-based SNN as an inference system, it is important to precisely map trained weights to synaptic devices. When transferring the weights, two main issues should be taken into consideration. First of all, 32-bit full precision floating point numbers are generally used for trained weights by software-based ANN, but it is difficult to implement synaptic devices having multi-level conductance comparable to software-based weights. Moreover, it must be possible to access an individual synaptic device and be capable of mapping target weights by repeatedly applying PGM/ERS pulses. However, as the scale of the neural network increases, the number of synapses increases, thus it can be time-consuming to transfer all the weights.

In this paper, we propose a scheme for transferring quantized synaptic weights to hardware-based SNN through STDP. Firstly, we train a network for the Modified National Institute of Standards and Technology (MNIST) pattern recognition using a software-based learning algorithm and try to quantize the trained weights thus that they can fit in the characteristic of the synaptic device. STDP is generally used as a learning algorithm, but it is employed as a method for weight transfer. Based on the STDP characteristic, a timing difference to adjust the conductance to the target quantized weight level can be obtained, and then all synaptic devices with the same target weight connected to a neuron can be modulated simultaneously if a teaching signal having the extracted timing difference is applied to the synaptic devices. This can significantly reduce the number of PGM/ERS steps to transfer the weights to hardware-based SNN. The silicon-based four-terminal synaptic transistor previously reported by our group is used as a synaptic device for this work [24,25]. We demonstrate the validity of the proposed method by a system-level simulation of hardware-based SNN. All the simulation is performed by MATLAB 2019b.

2. Methods

2.1. Silicon-Based Four Terminal Synaptic Transistor

The synaptic device used in this work was a silicon-based 4-terminal synaptic transistor previously reported by our group [24,25]. The synaptic device was composed of an asymmetric dual gate. Only SiO₂ was formed below n⁺-doped polysilicon for gate 1 (G1), which received a pre-synaptic spike, and SiO₂/Si₃N₄/SiO₂ stack was formed below n⁺-doped polysilicon for gate 2 (G2), which was used for STDP. The detailed fabrication process is illustrated in the previous work [24,25]. Figure 1 illustrates the mechanism of the weight modulation by trapping electrons and holes in the charge trapping layer. If asymmetric spikes were applied to G1 and G2 with a timing difference Δt , the type of charge trapped in the charge storage layer and the amount of charge can be controlled. With the positive Δt between the spikes of G1 and G2, as shown in Figure 1a, hot holes generated by impact ionization were injected into the charge storage layer, which makes the threshold voltage of the synaptic device (V_{th}) decrease (potentiation). In the opposite case, as shown

in Figure 1b, hot electrons were injected into the charge storage layer with the negative Δt , which leads to the positive shift of V_{th} (depression). The changes of the synaptic weights with the timing difference Δt are plotted in Figure 1c, which was analogous to the biological STDP characteristic [26]. The STDP characteristic was measured with a Keithley 4200-SCS and pulse measurement unit (PMU). Curve fitting for STDP characteristic was performed, and the equations of potentiation and depression were extracted as:

$$\Delta I_{source} = \begin{cases} A_p \times e^{-\frac{\Delta t}{\tau_p}}, & \text{for } \Delta t > 0 \text{ (potentiation)} \\ A_d \times e^{\frac{\Delta t}{\tau_d}}, & \text{for } \Delta t < 0 \text{ (depression)} \end{cases} \quad (1)$$

where A_p and A_d correspond to the fitting constant for potentiation and depression, respectively, and τ_p and τ_d denote for the time constant of potentiation and depression, respectively. The extracted parameters for the synaptic device are summarized in the inset of Figure 1c. Based on Equation (1), we can calculate the timing difference to adjust the target quantized weights. The detailed scheme for the weight transfer will be explained in Section 3.

Figure 1. Weight adjustment of the silicon-based four-terminal synaptic transistor by: (a) Hot hole injection (potentiation) and (b) hot electron injection (depression); (c) measured spike-timing-dependent plasticity (STDP) characteristic and the fitting curve.

2.2. Network Configurations

As shown in Figure 2a, a network consisting of 784, 800, and 10 neurons in the input, hidden and output layers, respectively, were trained. As mentioned above, since the performance of learning algorithms for SNN had not reached the state-of-the-art, we trained the network using software-based learning techniques with MNIST dataset. The MNIST dataset is composed of 60,000 train and 10,000 test images, and each image has a label vector, which gives the answer [21–23]. The network was optimized by stochastic gradient descent (SGD) algorithm with a learning rate of 5×10^{-3} and momentum of 0.9 [27]. In order to prevent over-fitting, some nodes were randomly dropped out with the probability of 0.5 [28]. After training, we obtained an accuracy of 98.23% for 10,000 test images.

Figure 2. (a) Trained neural network for MNIST dataset consisting of 784, 800, and 10 neurons in the input, hidden, and the output layers; (b) Time-accuracy of spiking neural network (SNN) according to the simulation time; (c) Full-precision weight distribution; (d) The optimization process of the weight quantization according to the weight interval α .

The I&F neuron cannot fire more than one spike at one timestep. When too many input spikes were applied in one timestep or one input spike with a too large weight came in, the neuron could not properly represent the output activations [21]. In order to properly implement SNN with the trained weights by software-based learning rules, weight normalization must be performed. This can prevent the I&F neuron from underestimating the output activations [21]. In this work, we employed ‘Data-based Normalization’ where the trained weights were normalized by the maximum output activation or maximum weight [21]. In addition, an analog value of the input was encoded using rate-based coding. Rate-based spike trains were generated by applying the constant current inputs corresponding to the input values to the I&F neuron. Figure 2b shows the accuracy of SNN for 10,000 test images with the full-precision weight according to the simulation time. SNN determines the index of the maximum firing neuron in the output layer as the answer, thus we can extract the accuracy by comparing the answer with the label vector in the dataset. In SNN, there was an inherent latency since it takes an integration time for a certain neuron to fire. Thus, the accuracy increased with respect to the simulation time, and it finally converged to the ANN’s accuracy.

After the weight normalization, we needed to quantize the weights because it was impossible for the synaptic devices to have 32-bit full precision weights as the weights in software-based ANN. The weights were quantized by a linear quantization method, which equally divides the weight distribution into a linear scale [17]. As shown in Figure 2c, the normalized weight distribution can be divided by $(2n + 1)$ -level as $\{w^{n-}, \dots, w^{2-}, w^{1-}, w^0, w^{1+}, w^{2+}, \dots, w^{n+}\}$, and the interval between each level is α . Figure 2d indicates the converged accuracy of SNNs according to the quantization level when α changes. It can be seen that there is the optimal α , which can minimize the accuracy loss compared with the case using the full-precision weight. Figure 3 illustrates 3, 5, 7, and

9-level quantized weight distribution with the optimal α , which will be transferred to the hardware-based SNN using the proposed method.

Figure 3. Weight distributions quantized by: (a) 3; (b) 5; (c) 7 and (d) 9-level, each of which is optimized by the interval α .

3. Results

3.1. Configurations of Hardware SNNs

Figure 4 shows a schematic diagram of the hardware-based SNN. The silicon-based four-terminal synaptic transistors form a NOR-type synapse array. There are k^l wordlines (WLs) for the pre-synaptic inputs applied to G1 and the drain of the synaptic devices simultaneously. Here, k^l denotes for the number of neurons in l^{th} layer. Since the synaptic device does not flow current when there is no spike, the standby power can be suppressed in the hardware-based SNN. One synaptic weight was implemented by a pair of two synaptic transistors, which act as an excitatory and inhibitory synapse, respectively. There are $2k^l + 1$ bitlines (BLs) where $BL_{k^l+1}^+$ and $BL_{k^l+1}^-$ are connected to the same post-synaptic neuron k^l ; the excitatory synaptic current (I_{exc}) flows through $BL_{k^l+1}^+$, and it is accumulated as the membrane potential of the post-synaptic neuron; the inhibitory synaptic current (I_{inh}) on $BL_{k^l+1}^-$ withdraws the membrane potential; therefore, positive, zero, and negative weights can be implemented according to the relative weights of the two synaptic devices [29]. G2 of the synaptic device is connected in BL direction as $G2_{k^l+1}^+$ and $G2_{k^l+1}^-$, which are used only for the weight transfer. By connecting G2 in BL direction, all the synapses on the same column can be correlated with a spike by the same neuron, indicating the biological STDP characteristic [24].

Figure 4. Schematic view of the hardware-based SNN. Pre-synaptic inputs are applied to wordlines (WLs), and the current summation is performed in bitlines (BLs). Two BLs are paired for the excitatory and inhibitory synaptic currents. Gate 2 of the synaptic devices are connected in BL direction by $G2_{k^l+1}^+$ and $G2_{k^l+1}^-$, which are used for the weight adjustment.

For inference operation, the input image was encoded by the rate-based coding, which was applied to WL as the pre-synaptic inputs. It is assumed that the I&F neuron circuit follows the simple non-leaky I&F neuron model, and the detailed model equations are in the previous research [19]. Thus, the synaptic currents were delivered to the post-synaptic I&F neuron circuits and integrated as the membrane potential; the I&F neuron generated the spikes when the membrane potential was above its threshold; then, the spikes were transferred to the subsequent layers, and finally, the spikes occurred in the output layer. Decision making was performed by comparing the neuron index having the maximum firing rates in the output layer with the label vector in the dataset.

3.2. Weight Transfer by STDP

In order to transfer the weights to the synaptic devices, it was necessary to apply PGM/ERS pulses for the precise weight adjustment. Since the weight adjustment was performed through accessing the individual synaptic devices one by one, the number of pulses required for the weight transfer was at least as many as the number of synapses in the network. However, the proposed method was based on STDP, thus that it can adjust all the synaptic weights connected with the same BL simultaneously, which reduces the pulse steps for the weight adjustment.

Let us take an example with 5-level quantized weight for the weight transfer using STDP. It was assumed that all the synaptic devices were in the initial state since the STDP characteristic changes, given that the synaptic devices were not in the initial state [24]. First of all, we can find the exact timing difference to adjust the precise quantized weight based on the STDP characteristic of Equation (1), as shown in Figure 5a. The extracted timing differences were denoted as $\{\Delta t^{2-}, \Delta t^{1-}, \Delta t^{1+}, \Delta t^{2+}\}$. For the simplicity of the weight mapping process, we only adjusted the weights of the excitatory synapses while keeping those of the inhibitory synapses in the initial state. Because one synaptic weight was implemented by the conductance differences between the excitatory and inhibitory synapses, ΔI_{source} of the excitatory synapse directly corresponded to the synaptic weight. In order to implement the positive weight, the timing difference must be positive, which was applied to the excitatory synapses. On the other hand, the negative timing difference

can be obtained through the depression part of Equation (1), and it was also applied to the excitatory synapses. Here, it was unnecessary to apply STDP pulse for the zero-weight since it can be implemented with the initial states of the excitatory and inhibitory synapses.

Figure 5. (a) Example of extracting the timing differences from STDP curve with 5-level quantized weights; (b) extracted weight map having the value of w^{2-} , w^{1-} , w^{1+} , w^{2+} connected to the first neuron in the first hidden layer.

Next, select a neuron in the l^{th} layer. The weight maps connected to the neuron having only the value of each quantized level can be extracted. For example, four weight maps with the size of 28×28 were extracted for the first neuron in the first hidden layer of the trained network above, and the white pixels in each weight map corresponded to the synapses having the weight value of w^{2-} , w^{1-} , w^{1+} , w^{2+} , as shown in Figure 5b. Then, let us apply the image of the weight maps encoded by voltage input sequentially to neurons in $(l-1)^{\text{th}}$ layer. As illustrated in Figure 6a, the weight map of the value w^{1+} was applied to the input layer in which the weight pixels were encoded by the asymmetric voltage pulse while the black pixels were at zero voltage. At the same time, if a teaching signal of the asymmetric voltage with the timing difference Δt^{1+} was applied to $G2_{11}^+$, all the weights with the value of w^{1+} were transferred simultaneously. This process can be performed for the remaining weights w^{2+} , w^{1-} , and w^{2-} with the timing differences Δt^{2+} , Δt^{1-} , and Δt^{2-} , as shown in Figure 6b–d. It requires only 4 steps for 784 synaptic weights to be transferred. Now, we can map all synaptic weights by repeating the previous process for the remaining neurons in the network. Through this method, the minimum number of pulse steps (N) required to transfer the entire synaptic weight can be reduced, and it is defined as:

$$N = \sum_{l=1}^m (n-1)k^l \quad (2)$$

where l and k^l denote for the layer index ($l = 0, 1, 2, \dots, m$) and the number of neurons in layer l . The negative one in the first term of Equation (2) is caused by a zero-weight state.

3.3. Accuracy and Variation Effect

Figure 7a shows the accuracy according to the simulation time with the different quantization levels transferred by the proposed method. It can be seen that the accuracy of SNNs gradually increased over time. The accuracy loss increased as the quantization level decreased, but the accuracy with 3-level quantization at the steady-state was 97.58%, which was still quite close to that with the full-precision weight. The minimum number of pulse steps (N) and the accuracy with respect to the quantization level are summarized in Table 1. It indicates that the proposed method can substantially reduce the required pulse steps by more than $\times 10^2$, but there is still a trade-off among the quantization level, the ease of the weight transfer, and the accuracy.

Figure 6. Illustration of the weight mapping process by STDP. The extracted weight map is applied as the voltage inputs. At the same time, a teaching signal with the timing difference obtained from STDP characteristic is applied to the G2 line of the neuron. Then, the weights connected to the neuron are transferred sequentially with the value of w^{2-} , w^{1-} , w^{1+} , w^{2+} .

Figure 7. (a) Classification accuracy according to the simulation time when 3, 7, 5, and 9-level of the quantization weights transferred by the proposed method are used; (b) the variation effects on the classification accuracy with 3, 7, 5, and 9-level of the quantization weights.

Table 1. Summary of the N and the accuracy according to the quantization levels.

Weight Level	Minimum Steps (N)	Accuracy
Full-Precision	635,200	98.23%
3	1620	97.58%
5	3240	98.16%
7	4860	98.22%
9	6480	98.23%

Figure 7b shows the performance by variations of synaptic devices. Since the variation of the initial state of the excitatory and inhibitory synapses as well as the PGM/ERS variation of STDP affects the overall performance, it is necessary to simulate the effect of the variation on the performance of hardware-based SNN. We introduce the variation to the synaptic weights since all the variation sources change the synaptic weights. The variation of the synaptic weights was assumed to show a Gaussian distribution where μ and σ were the mean value and the standard deviation, respectively. Here, the mean value was defined as the target weight state, and the variation effect was simulated by varying σ/μ from 0% to 30% with 5% steps. All the simulation was conducted 10 times for generalization. Overall, as the variation increased, the accuracy decreased more severely. In addition, the accuracy was more vulnerable to the variation as the quantization level increased because the possibility of interfering with the weights of adjacent levels increased. Therefore, as the quantization level increased, the performance can be improved, but more precise variation control is required.

4. Conclusions

In this paper, we propose a method based on STDP for transferring quantized weights to a hardware-based spiking neural network. In order to precisely map the weights trained by software-based ANN to the hardware synapse array, each synaptic device must be accessed and adjusted repeatedly. However, as the scale of the neural network increases, the number of synapses needed to be tuned increases, which is quite time-consuming. STDP is originally a method for online learning of SNNs, but through this, we propose a method of mapping the quantized weights to a large number of synapses at once. The performance of the proposed method is confirmed with a system-level simulation of two-layer fully-connected SNNs for MNIST pattern recognition. It is demonstrated that there is little reduction in the accuracy at more than a certain level of quantization, but the number of pulse steps for weight transfer significantly decreased. In addition, the effect of the device variation is verified. Consequently, the hardware-based SNN with the quantized weights transferred by the proposed method has the trade-off among the accuracy, the number of pulse steps, and the tolerance to the variation.

Author Contributions: Conceptualization and writing—original draft preparation, S.H.; writing and editing, H.K. and B.-G.P.; supervision, H.K. and B.-G.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the NRF funded by the Korean government under Grant 2019M3F3A1A03079821, 2020M3F3A2A01081656, 2020M3H5A1081111, in part by the Brain Korea 21 FOUR program, and in part by INHA UNIVERSITY Research Grant.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [\[CrossRef\]](#)
2. Krizhevsky, A.; Sutskever, I.; Hinton, G. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [\[CrossRef\]](#)
3. Tian, Y.; Pei, K.; Jana, S.; Ray, B. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In Proceedings of the 40th International Conference on Software Engineering, Gothenburg, Sweden, 30 May–1 June 2018; pp. 303–314.
4. Upadhyay, N.K.; Jiang, H.; Wang, Z.; Asapu, S.; Xia, Q.; Joshua Yang, J. Emerging memory devices for neuromorphic computing. *Adv. Mater. Technol.* **2019**, *4*, 1800589. [\[CrossRef\]](#)
5. Sung, C.; Padovani, A.; Beltrando, B.; Lee, D.; Kwak, M.; Lim, S.; Larcher, L.; Della Marca, V.; Hwang, H. Investigation of I-V linearity in TaO x-Based RRAM devices for neuromorphic applications. *IEEE J. Electron. Devices Soc.* **2019**, *7*, 404–408. [\[CrossRef\]](#)
6. Mead, C. Neuromorphic electronic systems. *Proc. IEEE* **1990**, *78*, 1629–1636. [\[CrossRef\]](#)
7. Seo, J.-S.; Brezzo, B.; Liu, Y.; Parker, B.D.; Esser, S.K.; Montoye, R.K.; Rajendran, B.; Tierno, J.A.; Chang, L.; Modha, D.S. A 45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons. In Proceedings of the 2011 IEEE Custom Integrated Circuits Conference (CICC), San Jose, CA, USA, 17–20 September 2011; pp. 1–4.
8. Nawrocki, R.A.; Voyles, R.M.; Shaheen, S.E. A mini review of neuromorphic architectures and implementations. *IEEE Trans. Electron. Devices* **2016**, *63*, 3819–3829. [\[CrossRef\]](#)

9. Indiveri, G.; Corradi, F.; Qiao, N. Neuromorphic architectures for spiking deep neural networks. In Proceedings of the 2015 IEEE International Electron Devices Meeting (IEDM), Washington, DC, USA, 7–9 December 2015; pp. 4.2.1–4.2.4.
10. Park, J.; Kwon, M.-W.; Kim, H.; Hwang, S.; Lee, J.-J.; Park, B.-G. Compact neuromorphic system with four-terminal Si-based synaptic devices for spiking neural networks. *IEEE Trans. Electron. Devices* **2017**, *64*, 2438–2444. [\[CrossRef\]](#)
11. Wang, Z.; Joshi, S.; Savel'ev, S.; Song, W.; Midya, R.; Li, Y.; Rao, M.; Yan, P.; Asapu, S.; Zhuo, Y. Fully memristive neural networks for pattern classification with unsupervised learning. *Nat. Electron.* **2018**, *1*, 137–145. [\[CrossRef\]](#)
12. Tuma, T.; Pantazi, A.; Le Gallo, M.; Sebastian, A.; Eleftheriou, E. Stochastic phase-change neurons. *Nat. Nanotechnol.* **2016**, *11*, 693. [\[CrossRef\]](#)
13. Kornijcuk, V.; Lim, H.; Seok, J.Y.; Kim, G.; Kim, S.K.; Kim, I.; Choi, B.J.; Jeong, D.S. Leaky integrate-and-fire neuron circuit based on floating-gate integrator. *Front. Neurosci.* **2016**, *10*, 212. [\[CrossRef\]](#)
14. Burkitt, A.N. A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input. *Biol. Cybern.* **2006**, *95*, 1–19. [\[CrossRef\]](#)
15. Chu, M.; Kim, B.; Park, S.; Hwang, H.; Jeon, M.; Lee, B.H.; Lee, B.-G. Neuromorphic hardware system for visual pattern recognition with memristor array and CMOS neuron. *IEEE Trans. Ind. Electron.* **2015**, *62*, 2410–2419. [\[CrossRef\]](#)
16. Lee, S.-T.; Lee, J.-H. Neuromorphic Computing Using NAND Flash Memory Architecture With Pulse Width Modulation Scheme. *Front. Neurosci.* **2020**, *14*, 571292. [\[CrossRef\]](#)
17. Kwon, D.; Chung, I.-Y. Capacitive Neural Network Using Charge-Stored Memory Cells for Pattern Recognition Applications. *IEEE Electron. Device Lett.* **2020**, *41*, 493–496. [\[CrossRef\]](#)
18. Kheradpisheh, S.R.; Ganjtabesh, M.; Thorpe, S.J.; Masquelier, T. STDP-based spiking deep convolutional neural networks for object recognition. *Neural Netw.* **2018**, *99*, 56–67. [\[CrossRef\]](#)
19. Bengio, Y.; Mesnard, T.; Fischer, A.; Zhang, S.; Wu, Y.J.N.C. STDP-compatible approximation of backpropagation in an energy-based model. *Neural Comput.* **2017**, *29*, 555–577. [\[CrossRef\]](#)
20. Nessler, B.; Pfeiffer, M.; Maass, W. STDP enables spiking neurons to detect hidden causes of their inputs. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 7–10 December 2009; pp. 1357–1365.
21. Diehl, P.U.; Neil, D.; Binas, J.; Cook, M.; Liu, S.-C.; Pfeiffer, M. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 11–16 July 2015; pp. 1–8.
22. Hwang, S.; Chang, J.; Oh, M.-H.; Lee, J.-H.; Park, B.-G. impact of the Sub-Resting Membrane potential on Accurate inference in Spiking neural networks. *Sci. Rep.* **2020**, *10*, 1–10. [\[CrossRef\]](#)
23. Rueckauer, B.; Lungu, I.-A.; Hu, Y.; Pfeiffer, M.; Liu, S.-C. Conversion of Continuous-Valued Deep Networks to Efficient Event-Driven Networks for Image Classification. *Front. Neurosci.* **2017**, *11*, 682. [\[CrossRef\]](#) [\[PubMed\]](#)
24. Kim, H.; Hwang, S.; Park, J.; Park, B.-G. Silicon synaptic transistor for hardware-based spiking neural network and neuromorphic system. *Nanotechnology* **2017**, *28*, 405202. [\[CrossRef\]](#)
25. Kim, H.; Park, J.; Kwon, M.-W.; Lee, J.-H.; Park, B.-G. Silicon-based floating-body synaptic transistor with frequency-dependent short-and long-term memories. *IEEE Electron. Device Lett.* **2016**, *37*, 249–252. [\[CrossRef\]](#)
26. Roberts, P.D.; Bell, C.C. Spike timing dependent synaptic plasticity in biological systems. *Biol. Cybern.* **2002**, *87*, 392–403. [\[CrossRef\]](#) [\[PubMed\]](#)
27. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [\[CrossRef\]](#)
28. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.J.T.J.o.M.L.R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
29. Hwang, S.; Kim, H.; Park, J.; Kwon, M.-W.; Baek, M.-H.; Lee, J.-J.; Park, B.-G. System-Level Simulation of Hardware Spiking Neural Network Based on Synaptic Transistors and I&F Neuron Circuits. *IEEE Electron. Device Lett.* **2018**, *39*, 1441–1444.