electronics

MDPI

Article

Memristive Spiking Neural Networks Trained with Unsupervised STDP

Errui Zhou, Liang Fang * and Binbin Yang

Institute for Quantum Information & State Key Laboratory of High Performance Computing, College of Computer, National University of Defense Technology, Changsha 410073, China; zhou_microe@163.com (E.Z.); binbinyang1993@163.com (B.Y.)

* Correspondence: lfang@nudt.edu.cn; Tel.: +86-151-1645-1937

Received: 12 November 2018; Accepted: 3 December 2018; Published: 6 December 2018

Abstract: Neuromorphic computing systems are promising alternatives in the fields of pattern recognition, image processing, etc. especially when conventional von Neumann architectures face several bottlenecks. Memristors play vital roles in neuromorphic computing systems and are usually used as synaptic devices. Memristive spiking neural networks (MSNNs) are considered to be more efficient and biologically plausible than other systems due to their spike-based working mechanism. In contrast to previous SNNs with complex architectures, we propose a hardware-friendly architecture and an unsupervised spike-timing dependent plasticity (STDP) learning method for MSNNs in this paper. The architecture, which is friendly to hardware implementation, includes an input layer, a feature learning layer and a voting circuit. To reduce hardware complexity, some constraints are enforced: the proposed architecture has no lateral inhibition and is purely feedforward; it uses the voting circuit as a classifier and does not use additional classifiers; all neurons can generate at most one spike and do not need to consider firing rates and refractory periods; all neurons have the same fixed threshold voltage for classification. The presented unsupervised STDP learning method is time-dependent and uses no homeostatic mechanism. The MNIST dataset is used to demonstrate our proposed architecture and learning method. Simulation results show that our proposed architecture with the learning method achieves a classification accuracy of 94.6%, which outperforms other unsupervised SNNs that use time-based encoding schemes.

Keywords: memristive spiking neural networks (MSNNs); spike-timing dependent plasticity (STDP); unsupervised learning; memristor; hardware implementation

1. Introduction

With explosion of data today, conventional von Neumann architectures are undergoing severe challenges. The memory wall problem and the energy problem are main bottlenecks [1]. Researchers have put a lot of efforts to find new architectures to avoid or relieve these issues. Neuromorphic computing and logic-in-memory, which are promising alternatives for conventional von Neumann architectures, have attracted a lot of attention [1–6]. An emerging device, called memristor, is widely employed in neuromorphic computing systems. These systems use memristors as synapses to store synaptic weights and implement multiplication and addition by physical processes, which are energy-efficient. For instance, Payvand et al. proposed a neuromorphic system with non-ideal memristive devices and demonstrated the system using behavioral simulation [7]. There are mainly two kinds of memristive neuromorphic computing systems: voltage-based systems and spike-based systems [1,2,8–11]. Previous literature has shown that the main signals transmitted in mammalian brains are spikes [12]. Therefore, researchers think spike-based systems are more biologically plausible

than voltage-based systems. Memristive spiking neural networks (MSNNs), which are a kind of spike-based neuromorphic computing systems, have been studied broadly in recent years.

To execute specific tasks, MSNNs should be trained with proper learning methods. There are many learning algorithms for spiking neural networks (SNNs), and they can be employed in hardware frameworks (such as TrueNorth) with some constraints and modifications [13–16]. A lot of recent works focus on implementing error backpropagation in SNNs. Bohte et al. proposed a backpropagation method called SpikeProp to train SNNs [13]. Lee et al. also presented a backpropagation method that obtained excellent performance on the MNIST dataset [14]. Some works tried to convert pre-trained conventional artificial neural networks (ANNs) into SNNs and achieved competitive classification accuracies on the MNIST dataset [15,16]. However, backpropagation-based learning algorithms are not biologically plausible. Inspired by working mechanisms of human brains, lots of researchers pay attention to the spike-timing dependent plasticity (STDP) learning rules, which are more biologically plausible. Cohen et al. used the SKIM network to perform a large-scale and multi-class classification task [17]. Querlioz et al. and Diehl et al. gave architectures and unsupervised STDP learning methods for shallow SNNs [9,18]. Thiele et al. proposed a learning method for convolutional SNNs based on STDP [19]. Zhang et al. used a plasticity-centric approach to train multilayer SNNs and obtained the highest performance within STDP-based SNNs [20]. Note that the encoding schemes of above STDP-based SNNs are all rate-based, i.e., the information is transmitted by spike rate. Whereas, some researchers have showed that time-based SNNs may be more efficient in processing and hardware implementation [21,22]. Moreover, they think the first spike is enough to transmit information. Kheradpisheh et al. presented a time-based deep convolutional SNN and gave corresponding unsupervised learning rules [12]. Liu et al. proposed a time-based hierarchical structure based on supervised STDP learning rules [22].

However, most of above SNNs and learning rules are not friendly to be implemented in MSNNs due to their complex architectures and mechanisms. SNNs in ref. [9,18] need to consider firing rates and refractory periods of neurons, and they require lateral inhibition and adaptive threshold voltages. SNNs in ref. [19] also need to consider firing rates and require lateral inhibition. Lateral inhibition and adaptive threshold voltages are required in ref. [22] as well. Other SNNs have similar requirements. These requirements increase hardware complexity and are undoubtedly unfriendly to hardware implementation with memristors. For instance, consideration of firing rates and refractory periods increases the design complexity of CMOS neurons, lateral inhibition and adaptive threshold voltages add difficulties to hardware architectures.

In this paper, we present an MSNN architecture that is friendly to hardware implementation with memristor synapses and CMOS neurons. The architecture has an input layer and a feature learning layer. The input layer encodes input signals into time-based spikes, and the feature learning layer learns features. Additionally, there is a voting circuit after the feature learning layer for classification. To reduce the complexity of hardware implementation, the proposed MSNN has following constraints:

1. No leakage currents. Neurons in the proposed MSNN are integrate-and-fire. Their integrated voltages are reset only at the beginning of next processing duration.
2. No firing rates and no refractory periods. All neurons can generate at most one spike during a processing duration, and the spike is reduced to step signals for simplicity (this constraint reduces the complexity of CMOS neurons).
3. No lateral inhibition. The proposed MSNN is purely feedforward and has no lateral inhibitory neurons (the architecture complexity is undoubtedly reduced when no inhibitory neuron is required).
4. No adaptive threshold voltages. All neurons of the proposed MSNN have the same fixed threshold voltage for classification (the complexity of supplying a fixed threshold voltage is evidently lower than supplying various threshold voltages).

To train the proposed MSNN, we present an unsupervised STDP learning method that uses exact time difference between pre- and post-synaptic spikes. The learning process is unsupervised,

and labels are only required during label assignment of feature learning neurons. The winner-takes-all (WTA) mechanism is employed during the learning process, i.e., the first activated feature learning neuron learns the feature. The learning method uses no homeostatic mechanism due to the constraint of no adaptive threshold voltages. The proposed MSNN with the STDP learning method is tested with the MNIST dataset and achieves a classification accuracy of 94.6%, which outperforms other unsupervised SNNs that use time-based encoding schemes. The main contributions of this paper are:

1. We propose a hardware-friendly MSNN architecture, which decreases the hardware complexity by adding several constraints.
2. We present an unsupervised STDP learning method for the architecture.
3. The architecture with the learning method is validated by the MNIST dataset and achieves a classification accuracy that outperforms other time-based unsupervised SNNs.

The rest of this paper is organized as follows. Section 2 describes the detailed architecture of the proposed MSNN. The unsupervised STDP learning method for the proposed MSNN is presented in Section 3. Section 4 provides the results on the MNIST dataset, and Section 5 concludes this paper.

2. Architecture

The schematic of the proposed MSNN is shown in Figure 1. It has an input layer, a feature learning layer and a voting circuit. While previous SNNs have an inhibitory layer after the feature learning layer (each feature learning neuron corresponses to a inhibitory neuron), the hardware complexity is enhanced [9,18]. Unlike prior SNNs, our presented MSNN has no lateral inhibitory neurons and is purely feedforward. The neurons in the input layer generate spikes with time-based encoding scheme. Spikes are then propagated to the feature learning layer via synapses. For simplicity, spikes are reduced to step signals in the proposed MSNN. As shown in Figure 1, a memristor plays the role of a synapse, and there is a synapse between each pre- and post-synaptic neuron. The neurons in the feature learning layer learn different features. Finally, there is a voting circuit to determine which category the input pattern belongs to. Note that our time-based MSNN allows each neuron to generate at most one spike while other rate-based SNNs do not limit the number of spikes (when each neuron can generate multiple spikes during a process duration, firing rates and refractory periods should be considered, which increases complexity of hardware implementation).

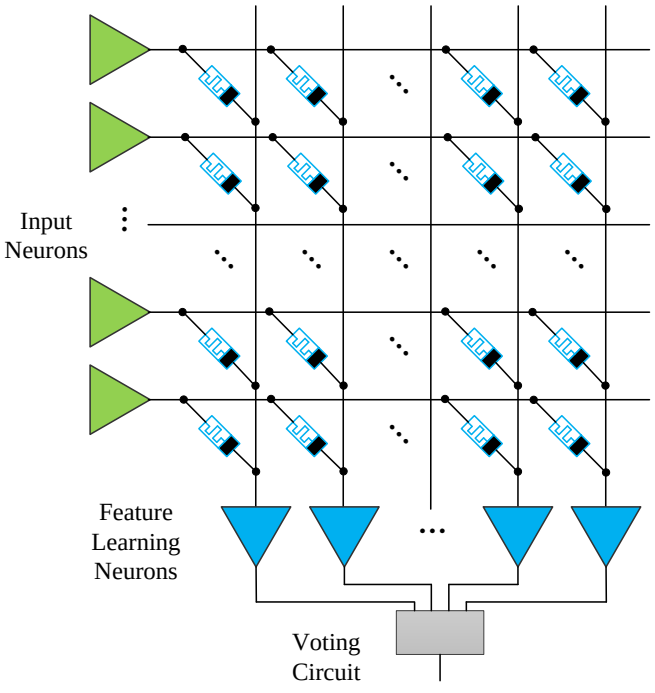

Figure 1. Schematic of the proposed MSNN.

2.1. Input Layer

The spike encoding scheme is essential for the MSNN because it transforms input signals into spikes. These spikes transmit all information that is necessary for learning and classification. There are two main kinds of spike encoding schemes: rate-based and time-based. The rate-based encoding scheme transforms input signals into spike rates of input neurons, and the time-based encoding scheme transforms input signals into spike times of input neurons [18,23]. A rate-based neuron needs to consider refractory periods, firing rates and so on, which will increase the hardware complexity. In this paper, we chose the time-based encoding scheme because of the simplicity of hardware implementation. The time-based neuron only needs to generate a step signal when it is activated (the CMOS neuron only needs to output a high voltage when it is activated, which is more simple than generating spikes with specific shapes; note that the unsupervised STDP learning method should be designed suitably for the step signal). For the MNIST dataset, each input neuron corresponds to a pixel. The stronger the intensity of a pixel is, the earlier the corresponding neuron fires. There are many schemes to convert input signals into spike times of neurons. In ref. [22], the spike time of an input neuron is inversely proportional to the input signal. In this paper, their relationship is described as follows:

$$t_i = p(1 - r_i/r_{max}), \tag{1}$$

where t_i is the spike time of an input neuron, p is the processing duration for an input pattern (p is set as 100 µs according to [11], and p can be modified to different scales according to specific requirements), r_i is the corrseponding pixel intensity and r_{max} is the maximum intensity of all pixels. In this paper, we set $r_{max} = 250$ directly. When an input neuron fires, it will generate a step signal with a fixed voltage. Assume the spike time of an input neuron is t_i, its output can be described as $V_f \cdot \varepsilon(t - t_i)$. V_f is the fixed voltage, and $\varepsilon(t - t_i)$ is the step function.

2.2. Synapse

There is a synapse between each pre- and post-synaptic neuron. The synaptic weights are all positive in our proposed MSNN. Therefore, a memristor is enough to play the role of a synapse, and the conductance of a memristor is employed as the synaptic weight of a synapse. A vital reason that MSNNs can implement multiplication operations efficiently is that the multiplication operation is implemented by Ohm's law [1]. The current flowing through a memristor is the product of the voltage applied on the memristor and the conductance of the memristor, which can be described by following equation:

$$I = VG, \tag{2}$$

where I is the current, V is the applied voltage and G is the conductance. Ohm's law has been demonstrated that it works well in memristive neuromorphic systems [1,2,11], hence, it is reasonable to use (2) in simulations. It is important to note that the conductance range of memristors is finite in practical. We set the range as $[10^{-6}$ S, 10^{-3} S] according to practical memristive devices [1]. Previous literatures have shown that conductance of memristors can be programmed to an arbitrary value within the conductance range by programming circuits [23]. Therefore, it is reasonable to assume that the conductance of memristors for classification can be modified to suitable values according to learning results (some researchers use the switching stochasticity of memristors to build stochstical spiking neurons, however, the switching stochasticity is not used when memristors are employed as synapses [24]). The programming scheme is not the key point of this paper, and the programming scheme in ref. [23] can be adopted.

2.3. Feature Learning Layer

Most CMOS neurons in previous works are rate-based, and they consider many complex characteristics such as refractory periods, firing rates, and leakage currents [25,26]. These CMOS

neurons are biologically plausible but inefficient for hardware implementation. However, a neuron can generate at most one spike in our proposed MSNN. Hence, neurons are hardware-efficient because they do not need consider refractory periods and firing rates. Moreover, neurons in the feature learning layer are integrate-and-fire (IF) neurons. There is no leakage currents during a processing duration, and all neurons are set to initial states at the beginning of next processing duration. Therefore, the CMOS neuron used in our MSNNs is a reduced version of [25], which only comprises the integration part and the firing part. Feature learning neurons integrate currents flowing from input neurons, which can be formulated as follows:

$$C\frac{dV_j}{dt} = \sum_i I_{ij} = \sum_i V_f \cdot \varepsilon(t - t_i) \cdot G_{ij}, \tag{3}$$

where i and j are indexes of the input neuron and the feature learning neuron, C is the integration capacitance, and V_j is the integrated voltage. Kirchhoff's current law is used here to implement addition operations [1]. Note that Kirchhoff's current law also has been demonstrated that it can work well in memristive neural networks [1,2]. For simplicity of hardware implementation, integration capacitance of all feature learning neurons is set as a fixed value. When the integrated voltage of a feature learning neuron exceeds a threshold voltage, the feature learning neuron is activated and fires a spike. Therefore, the output of the feature learning neuron can be written as $V_f \cdot \varepsilon(V_j - V_{th})$. V_{th} is the threshold voltage and it is set as a fixed value for simplicity of hardware implementation. The hardware complexity of supplying a fixed a threshold voltage is evidently lower than supplying various threshold voltages. When the spike time is noted as t_j, the output also can be written as $V_f \cdot \varepsilon(t - t_j)$.

Feature learning neurons learn features according to learning method described in Section 3. The learned features are described by weights of synapses between the input layer and the feature learning layer. After the learning process, all feature learning neurons have learned different features that correspond to different categories (i.e., the weights of synapses are modified to learned values). The training set is then applied to the MSNN again to assign each feature learning neuron to a category. The assigning scheme is described as follows: The number of times that a feature learning neuron is first activated is counted. The number of times is then assigned to different categories according to labels of input patterns. Finally, the feature learning neuron is assigned to the category with the largest number of first activations.

2.4. Voting Circuit

A classifier is crucial for an SNN with unsupervised learning rules to fulfill classification. In ref. [12], a support vector machine is used as the classifier, which requires extra supervised learning. A multilayer neural network is employed as the classifier in ref. [27]. However, these classifiers are unfriendly for MSNNs because they assume that the threshold voltages are infinite in the final layers, which is impossible in practical. To reduce hardware complexity, a compact and practical classifier is designed for the proposed MSNN in this paper. As shown in Figure 1, the classifier is a voting circuit. The voting circuit does not work during the learning process. It determines which category the input patter belongs to during the testing process. When an input pattern is applied to the MSNN, the voting circuit counts the first N activated feature learning neurons (N is the parameter that can be modified according to requirements). The category having the largest number of assigned feature learning neurons is given as the predicted category.

3. Learning Method

The synaptic weights between the input layer and the feature learning layer are learned by STDP. In refs. [9,18], several neurons can learn the same feature with different intensities according to their spike rates. However, our learning method uses the winner-takes-all mechanism, i.e., our learning method only allows the first activated feature learning neuron to learn the feature during a learning duration. This setup can help feature learning neurons learn features better according to our

experimental results. Our learning method is time-dependent because time plays a crucial role in the integration of voltages. The learning method is described as follows:

$$\Delta G_{ij} = \begin{cases} a^{+}(1 - e^{\frac{t_i - t_j}{\tau}}), & if \quad t_i \leq t_j, \\ a^{-}(1 - e^{\frac{t_i - t_j}{\tau}}), & if \quad t_i > t_j, \end{cases} \tag{4}$$

where i and j have been defined in (3), t_i and t_j are the corresponding spike times, a^+ and a^- are two parameters that specify the learning rates, τ is the time constant and ΔG_{ij} is the synaptic weight modification. Here τ is set as 20 µs according to the estimated values of spiking times of feature learning neurons. When memristors are employed as synapses, the range of synaptic weights is $[10^{-6}S, 10^{-3}S]$. Therefore, there is a limitation for all synaptic weights, which is described as follows:

$$G_{ij} = \begin{cases} G_{min}, & if \quad G_{ij} \leq G_{min}, \\ G_{max}, & if \quad G_{ij} \geq G_{max}, \\ G_{ij}, & if \quad G_{min} \leq G_{ij} \leq G_{max}. \end{cases} \tag{5}$$

G_{max} and G_{min} equal to 10^{-3} S and 10^{-6} S, respectively. For simplicity, all synaptic weights are normalized to a range of $[0, 1]$ in the following. Our learning method uses no homeostatic mechanisms because of the constraint of no adaptive threshold voltages, while many rate-based SNNs use homeostatic mechanisms to make feature learning neurons have approximately equal spike rates [9,18] .

As depicted in Figure 2, the synaptic weight is potentiated when $t_i - t_j \leq 0$, and the synaptic weight is depressed when $t_i - t_j > 0$. Note that the learning method used in this paper is different with classic STDP learning rules. For classic learning rules, the smaller the time difference, the larger the weight modification. However, the spikes in our proposed MSNN are step signals. The step signals keep influencing integrated voltages of post-synaptic neurons, which means the pre-synaptic spikes fired earlier have more contribution to post-synaptic spikes. Therefore, for the learning method in this paper, the larger the time difference, the larger the weight modification.

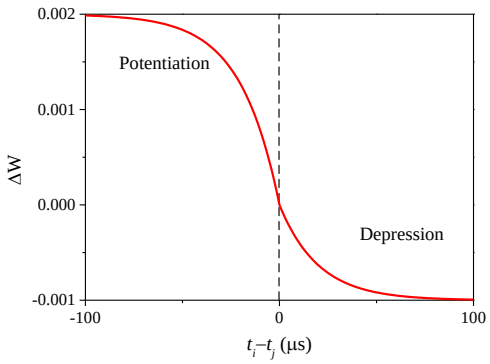

Figure 2. STDP rule used in the proposed MSNN.

When the range of synaptic weights is normalized to $[0, 1]$, the initial synaptic weights are set as random values drown from a uniform distribution with the mean of $u = 0.5$ and the variance of 0.012/12. By choosing a small u, feature learning neurons are hard to fire and learn features. By choosing a large u, the performance will decrease slightly. The variance also has significant influence on the learning of MSNNs: when the varience is too large, the learned features of feature learning neurons mainly depend on the initial synapyic weights. Additionally, the MSNN can get

better performance when the absolute value of a^- is slightly less than a^+. In this paper, a^+ and a^- are set as 0.002 and -0.001 respectively.

The learning process is described as follows: an input pattern is transformed into spikes by input neurons, and the spike time of the first activated feature learning neuron is recorded; the synaptic weights corresponding to the feature learning neuron are updated according to the learning method; all variables are set to initial states except synaptic weights, and training for another input pattern begins.

4. Results

In this paper, all simulations are implemented by MATLAB, and the memristor model in ref. [28] is adopted. Though our proposed architecture and learning method are demonstrated by simulation, the used models and parameters are set according to experimental results [1,2,7,11]. Parameters for simulation are shown in Table 1. The MNIST dataset is used in this paper to validate the proposed MSNN and learning method (i.e., the number of input neurons is 784). The 60,000 training images are used to train the MSNN, and the 10,000 testing images are employed to test the performance of trained MSNN. An image of "6" is shown in Figure 3a, and the corresponding spike times of input neurons after encoding are depicted in Figure 3b.

Table 1. Parameters for Simulation.

Parameters	Values
Voltage of Step Signals for learning (v_f)	1 V
Threshold Voltage of Neurons for Learning (V_{th})	0.5 V
Integration Capcitance (C)	1 nF
Processing Duration (p)	100 µs
τ	20 µs
a^+/a^-	$0.002/-0.001$

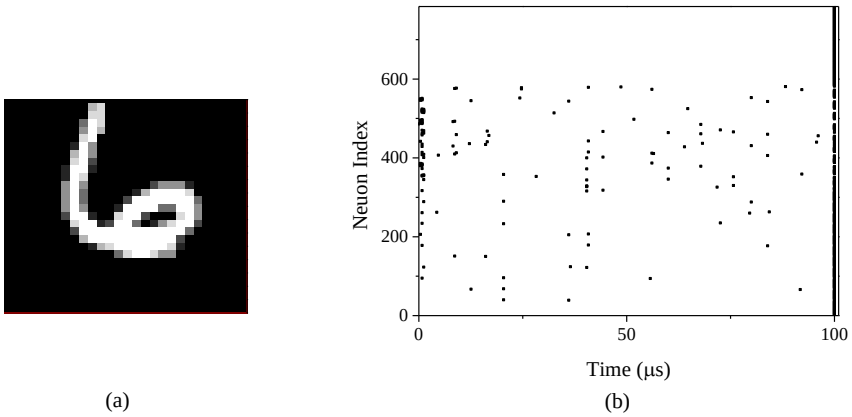

Figure 3. (**a**) An image of "6", (**b**) spike times of input neurons after encoding.

4.1. Feature Learning

Our first experiment was to see the feature learning process of the proposed MSNN. The number of feature learning neurons was set as 100. As shown in Figure 4a, the synaptic weights learned by a feature learning neuron are reshaped into a 2D structure. The feature learned by the feature learning neuron evolves with the number of training examples. It is easy to see that the feature learning neuron learns a feature of "1". The neuron learns nothing at the beginning, and the learned feature is more clearly when more examples are provided. The reason is that the MSNN learns the features with higher occurrence frequencies and tends to forget features with lower occurrence frequencies. When the whole training set is applied, the features learned by 100 neurons are depicted in Figure 4b. It can be

seen that most features of digits are learned while some features are not clear enough. The classification accuracies with different numbers of training examples are shown in Figure 4c. We can find that the classification accuracy is the lowest (about 27.4%) when the MSNN is not trained. This result depends on the initial stochastic synaptic weights. The classification accuracy then increases with the increase of the number of training examples, and it is almost unchanged when the number of training examples is larger than 10,000. These results mean that the proposed MSNN can achieve a convergent state after enough training examples are provided. The reason is that learned features are clearer when more examples are provided and learned features stay stable after enough examples are provided. For other MSNNs with more feature learning neurons, their classification accuracies also achieve almost unchanged values after more training examples are provided.

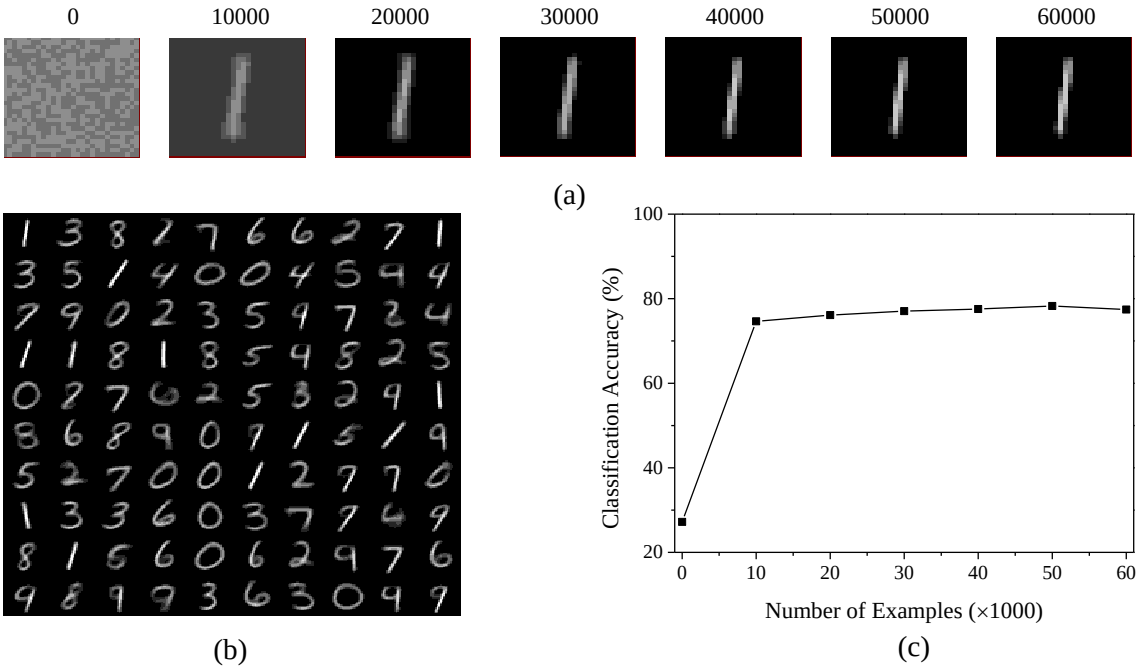

Figure 4. (**a**) Evolution of the learned feature, (**b**) Features learned after applying the whole training set, (**c**) impact of the number of training examples on classification accuracy.

4.2. *Impact of the Threshold Voltage for Testing*

While the threshold voltage of feature learning neurons for learning was fixed, we found that the threshold voltage for testing affected the performance of MSNNs in a follow-up experiment. We tested MSNNs with 100, 400, 1600 and 6400 feature learning neurons, respectively. The numbers of training examples were set as 30,000, 60,000, 300,000 and 360,000 according to experimental results. The number of voting neurons was set as one. Figure 5a shows the classification accuracies of MSNNs with different numbers of feature learning neurons. It shows that the MSNNs with more feature learning neurons obtain higher classification accuracies. The reason is that the MSNNs with more feature learning neurons can learn more features. However, the classification accuracy increases slower when the number of feature learning neurons is larger than 1600, because the learned features tend to be saturated. Figure 5b shows the classification accuracies of MSNNs with different threshold voltages. It depicts that the MSNNs with higher threshold voltages get higher classification accuracies. Nevertheless, the classification accuracy nearly keeps unchanged when the threshold voltage is higher than 2.5 V. The reason is concluded as follows: feature learning neurons need longer time to fire when the threshold voltage is higher; more features will be provided when the firing time is longer, but time length will have little influence after significant features have been provided. Note that the threshold voltage should be chosen carefully, because too high threshold voltage will result in no firing. We also

can find that the influence of threshold voltage is the greatest when the number of feature learning neurons is 100, and the influence is very weak when the number of feature learning neurons is larger than 1600. This means that the threshold voltage has little influence when there are enough feature learning neurons.

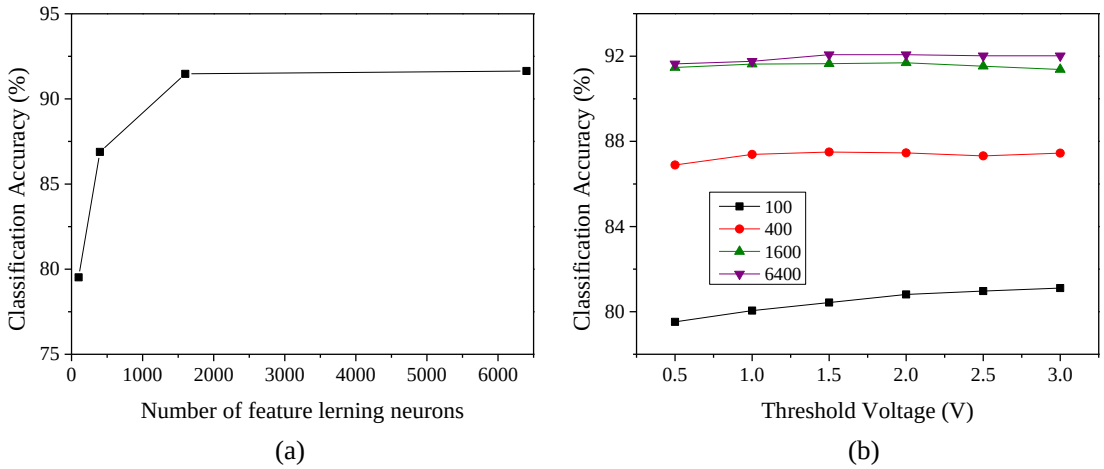

Figure 5. (**a**) Impact of the number of feature learning neurons on classification accuracy, (**b**) impact of the threshold voltage for testing on classification accuracy.

4.3. Impact of the Number of Voting Neurons

The relationship between the number of voting neurons and classification accuracy is shown in Figure 6. The threshold voltage is chosen as 2.5 V according to Section 4.2. The MSNN with 100 feature learning neurons obtains the highest classification accuracy with one voting neuron, and its classification accuracy decreases when the number of voting neurons increases. The reason may be that the learned features of 100 feature learning neurons are not balanced for different images. The MSNNs with 1600 and 6400 feature learning neurons achieve higher classification accuracies when the numbers of voting neurons increase. However, their classification accuracies nearly keep stable when the numbers of voting neurons are greater than five. This means that five voting neurons are nearly enough to determine the categories of input images. Interestingly, classification accuracies decrease evidently when the number of voting neurons is two. The reason is that the voting circuit cannot determine which category the input patter belongs to when the two voting neurons belongs to different categories. When the number of voting neurons is 10, the MSNN with 6400 feature learning neurons reaches the highest classification accuracy of 94.6%.

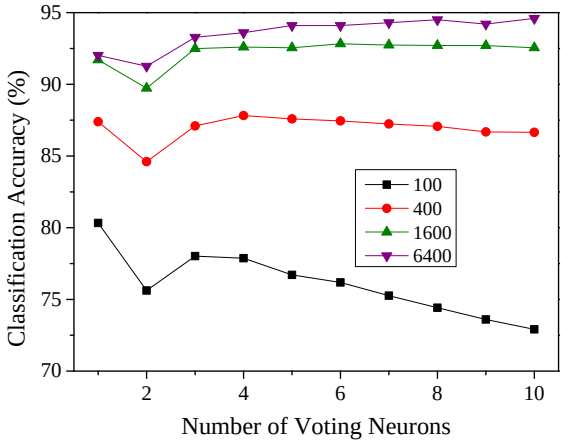

Figure 6. Impact of the number of voting neurons on classification accuracy.

4.4. Impact of Variations of Synaptic Weights

Memristors are regarded as nonvolatile devices when they are employed in neuromorphic computing systems. However, it is difficult for them to keep their conductance at the ideal trained values due to device variations. Therefore, it is essential to explore the impact of variations of synaptic weights on performance of the proposed MSNN. The MSNN that has 6400 feature learning neurons is used here to show the effect. Each synaptic weight is added with a random value drawn from a uniform distribution in range $\pm\alpha\%$ of 1 (because we have normalized the range of synaptic weights to $[0, 1]$). The parameter α is used to describe the variation level. Note that the synaptic weights are limited in $[0, 1]$ after adding random values. Figure 7 depicts the relationship between variation of synaptic weights and classification accuracy. The classification accuracy of the MSNN with $N = 10$ is higher than 90% when the variation level is up to 20% and is still higher than 80% when the variation level is up to 45%. This means that the variations of synaptic weights have weak influence on the MSNN with $N = 10$. However, the classification accuracy of the MSNN with $N = 1$ decreases sharply with the increase of variation level. These results demonstrate that the voting circuit using multiple voting neurons has good immunity to variations of synaptic weights.

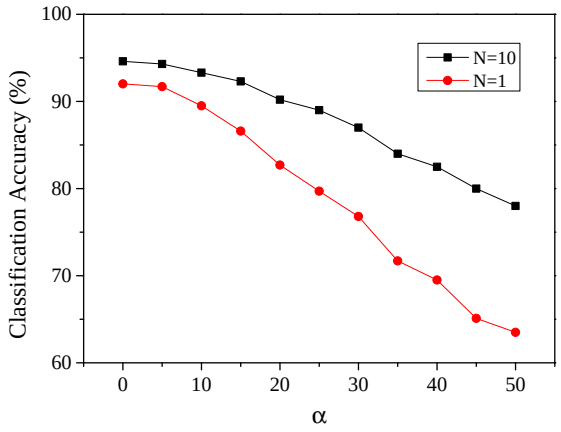

Figure 7. Impact of weight variations on classification accuracy.

4.5. Comparison

Our proposed MSNN achieves a classification accuracy of 94.6% for the MNIST dataset, which is competitive with previous works. As shown in Table 2, state-of-the-art SNNs with STDP learning methods are listed. The immunity to device variations of MSNNs with an unsupervised STDP is studied in ref. [9]. Rate-based encoding scheme, lateral inhibition and adaptive threshold voltages are required, which increases complexity of hardware implementation. A deep convolutional SNN with an extra classifier is successfully trained in ref. [12], which uses time-based encoding scheme and requires lateral inhibition. The SNN in ref. [18] is robust with several unsupervised STDP learning rules, and it requires rate-based encoding scheme, lateral inhibition and adaptive threshold voltages. The SNN in ref. [19] comprises several convolutional layers and only uses unsupervised STDP learning rules. It can learn online and is time-scale invariant. It achieves a relatively higher accuracy than other unsupervised SNNs while rate-based encoding scheme, lateral inhibition and adaptive threshold voltages are all required. The multilayer SNN in ref. [20] uses a plasticity-centric approach and achieves the highest accuracy. A hierarchical structure is proposed in ref. [22], which can implement continuous STDP learning. A SCNN is also presented in ref. [29], however, its classification accuracy is somewhat low. We should admit that the classification accuracy of the presented MSNN is lower than the highest accuracy of 98.52% and the proposed architecture is a somewhat shallow architecture. However, our proposed MSNN achieves the best classification performance within the unsupervised SNNs that use time-based encoding schemes. Moreover, the classification accuracy is even higher than rate-based

SNNs in refs. [9,29]. As disccussed in Section 1, rate-based encoding scheme, lateral inhibition and adaptive threshold voltages will increase complexity of hardware implementation. Our proposed MSNN adopts time-based encoding scheme and does not require lateral inhibition and adaptive threshold voltages, which means the complexity of hardware implementation is reduced. Therefore, our proposed MSNN trained with unsupervised STDP is friendlier to hardware implementation than listed SNNs.

Table 2. Comparison with previous works.

Architecture	(Un)-Supervised	Learning Method	Encoding Scheme	Lateral Inhibition	Adaptive Threshold Voltage	Accuracy
our work	Unsupervised	STDP	Time-based	No	No	94.6%
Two-layer MSNN [9]	Unsupervised	STDP	Rate-based	Yes	Yes	93.5%
SCNN [12]	Both	STDP + SVM	Time-based	Yes	No	98.4%
Two-layer SNN [18]	Unsupervised	STDP	Rate-based	Yes	Yes	95%
SCNN [19]	Unsupervised	STDP	Rate-based	Yes	Yes	96.58%
Multilayer SNN [20]	Both	STDP	Rate-based	No	No	98.52%
SCNN [22]	Supervised	STDP	Time-based	Yes	Yes	93%
SCNN [29]	Unsupervised	STDP	Rate-based	Yes	Yes	91.1%

5. Conclusions

In this paper, we propose a hardware-friendly architecture and an unsupervised STDP learning method for MSNNs. The presented architecture has an input layer, a feature learning layer, and a voting circuit. The input layer uses time-based encoding scheme and converts input signals into time-based spikes. The feature learning layer learns features by the unsupervised STDP learning method, and the voting circuit plays the role of a classifier. In order to reduce the complexity of hardware implementation, each neuron is allowed to generate at most one spike, and the spike is reduced to a step signal. No homeostatic mechanism is used during learning, and the threshold voltages of all neurons are the same fixed value. The presented architecture is fully feedforward and has no inhibitory neurons. The learning method is time-dependent and uses the WTA mechanism. We exploit the MNIST dataset to explore the effects of the threshold voltage for testing and the number of voting neurons on classification performance. Additionally, the impact of variation of synaptic weights is also studied, which shows that the proposed MSNN has good immunity to variation of synaptic weights.

Author Contributions: Data curation, E.Z.; Formal analysis, E.Z.; Investigation, E.Z. and B.Y.; Methodology, E.Z.; Resources, L.F.; Supervision, L.F.; Writing—original draft, E.Z.; Writing—review and editing, E.Z.

Funding: This work was supported by the National Natural Science Foundation of China (Grant No. 61332003) and National Natural Science Foundation of China(Grant No. 61832007).

Acknowledgments: The authors would like to thank Jingyan Xu and Lianhua Qu for their comments and beneficial suggestions.

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

References

1. Ambrogio, S.; Narayanan, P.; Tsai, H.; Shelby, R.M.; Boybat, I.; Di Nolfo, C.; Sidler, S.; Giordano, M.; Bodini, M.; Farinha, N.C.P.; et al. Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature* **2018**, *558*, 60–67. [CrossRef] [PubMed]
2. Li, C.; Belkin, D.; Li, Y.; Yan, P.; Hu, M.; Ge, N.; Jiang, H.; Montgomery, E.; Lin, P.; Wang, Z.; et al. Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. *Nat. Commun.* **2018**, *9*, 7–14. [CrossRef] [PubMed]
3. Zhou, J.; Yang, X.J.; Wu, J.J.; Zhu, X.; Fang, X.D.; Huang, D. A memristor-based architecture combining memory and image processing. *Sci. China Inf. Sci.* **2014**, *57*, 52111. [CrossRef]

4. Borghetti, J.; Snider, G.S.; Kuekes, P.J.; Yang, J.J.; Stewart, D.R.; Williams, R.S. Memristive switches enable stateful logic operations via material implication. *Nature* **2010**, *464*, 873–876. [CrossRef] [PubMed]
5. Zhu, X.; Yang, X.; Wu, C.; Xiao, N.; Wu, J.; Yi, X. Performing stateful logic on memristor memory. *IEEE Trans. Circuits Syst. II Express Briefs* **2013**, *60*, 682–686. [CrossRef]
6. Vato, A.; Bonzano, L.; Chiappalone, M.; Cicero, S.; Morabito, F.; Novellino, A.; Stillo, G. Spike manager: A new tool for spontaneous and evoked neuronal networks activity characterization. *Neurocomputing* **2004**, *58–60*, 1153–1161. [CrossRef]
7. Payvand, M.; Nair, M.V.; Muller, L.K.; Indiveri, G. A neuromorphic systems approach to in-memory computing with non-ideal memristive devices: From mitigation to exploitation. *Faraday Discuss.* **2018**. [CrossRef]
8. Hu, M.; Graves, C.E.; Li, C.; Li, Y.; Ge, N.; Montgomery, E.; Davila, N.; Jiang, H.; Williams, R.S.; Yang, J.J.; et al. Memristor-Based Analog Computation and Neural Network Classification with a Dot Product Engine. *Adv. Mater.* **2018**, *30*, 1705934. [CrossRef]
9. Querlioz, D.; Bichler, O.; Dollfus, P.; Gamrat, C.; Querlioz, D.; Bichler, O.; Dollfus, P.; Gamrat, C.; Querlioz, D.; Bichler, O.; et al. Immunity to Device Variations in a Spiking Neural Network with Memristive Nanodevices. *IEEE Trans. Nanotechnol.* **2013**, *12*, 288–295. [CrossRef]
10. Wang, J.J.; Hu, S.G.; Zhan, X.T.; Yu, Q.; Liu, Z.; Chen, T.P.; Yin, Y.; Hosaka, S.; Liu, Y. Handwritten-Digit Recognition by Hybrid Convolutional Neural Network based on HfO2 Memristive Spiking-Neuron. *Sci. Rep.* **2018**, *8*, 12546. [CrossRef]
11. Wang, Z.; Joshi, S.; Savel, S.; Song, W.; Midya, R.; Li, Y.; Rao, M.; Yan, P.; Asapu, S.; Zhuo, Y.; et al. Fully memristive neural networks for pattern classification with unsupervised learning. *Nat. Electron.* **2018**, *1*, 137–145. [CrossRef]
12. 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] [PubMed]
13. Bohte, S.M.; Kok, J.N.; La Poutré, H. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing* **2002**, *48*, 17–37. [CrossRef]
14. Lee, J.H.; Delbruck, T.; Pfeiffer, M. Training deep spiking neural networks using backpropagation. *Front. Neurosci.* **2016**, *10*, 508. [CrossRef] [PubMed]
15. Cao, Y.; Chen, Y.; Khosla, D. Spiking Deep Convolutional Neural Networks for Energy-Efficient Object Recognition. *Int. J. Comput. Vis.* **2015**, *113*, 54–66. [CrossRef]
16. 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 International Joint Conference on Neural Networks, Killarney, Ireland, 12–17 July 2015; pp. 1–8.
17. Cohen, G.K.; Orchard, G.; Leng, S.H.; Tapson, J.; Benosman, R.B.; van Schaik, A. Skimming digits: Neuromorphic classification of spike-encoded images. *Front. Neurosci.* **2016**, *10*, 1–11. [CrossRef] [PubMed]
18. Diehl, P.U.; Cook, M. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Front. Comput. Neurosci.* **2015**, *9*, 99. [CrossRef] [PubMed]
19. Thiele, J.C.; Bichler, O.; Dupret, A. Event-Based, Timescale Invariant Unsupervised Online Deep Learning with STDP. *Front. Comput. Neurosci.* **2018**, *12*, 46. [CrossRef] [PubMed]
20. Zhang, T.; Zeng, Y.; Zhao, D.; Shi, M. A Plasticity-centric Approach to Train the Non-differential Spiking Neural Networks. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018), New Orleans, LA, USA, 2–7 February 2018; pp. 620–627.
21. Delorme, A.; Thorpe, S.J. Face identification using one spike per neuron: Resistance to image degradation. *Neural Netw.* **2001**, *14*, 795–803. [CrossRef]
22. Liu, D.; Yue, S. Event-Driven Continuous STDP Learning With Deep Structure for Visual Pattern Recognition. *IEEE Trans. Cybern.* **2018**. [CrossRef] [PubMed]
23. Li, C.; Hu, M.; Li, Y.; Jiang, H.; Ge, N.; Montgomery, E.; Zhang, J.; Song, W.; Dávila, N.; Graves, C.E.; et al. Analogue signal and image processing with large memristor crossbars. *Nat. Electron.* **2017**, *1*, 52–59. [CrossRef]
24. Al-Shedivat, M.; Naous, R.; Cauwenberghs, G.; Salama, K.N. Memristors empower spiking neurons with stochasticity. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2015**, *5*, 242–253. [CrossRef]

25. Kwon, M.; Park, J.; Baek, M.; Hwang, S.; Park, B. Spiking Neural Networks with Unsupervised Learning Based on STDP Using Resistive Synaptic Devices and Analog CMOS Neuron Circuit. *J. Nanosci. Nanotechnol.* **2016**, *18*, 6588–6592. [CrossRef] [PubMed]
26. Wu, X.; Saxena, V.; Zhu, K.; Balagopal, S. A CMOS Spiking Neuron for Brain-Inspired Neural Networks with Resistive Synapses and in Situ Learning. *IEEE Trans. Circuits Syst. II Express Briefs* **2015**, *62*, 1088–1092. [CrossRef]
27. Ferré, P.; Mamalet, F.; Thorpe, S.J. Unsupervised Feature Learning With Winner-Takes-All Based STDP. *Front. Comput. Neurosci.* **2018**, *12*, 24. [CrossRef] [PubMed]
28. Zhou, E.; Fang, L.; Liu, R.; Tang, Z. An improved memristor model for brain-inspired computing. *Chin. Phys. B* **2017**, *26*, 118502. [CrossRef]
29. Lee, C.; Srinivasan, G.; Panda, P.; Roy, K. Deep Spiking Convolutional Neural Network Trained with Unsupervised Spike Timing Dependent Plasticity. *IEEE Trans. Cogn. Dev. Syst.* **2018**. [CrossRef]