Deep Spiking Neural Network with Neural Oscillation and Spike-Phase Information

Yi Chen,[1] **Hong Qu,**[1] **Malu Zhang,**[1,2] **Yuchen Wang,**[1]
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, China
[2]Department of Electrical and Computer Engineering, National University of Singapore, Singapore
{chenyi, wangyuchen}@std.uestc.edu.cn, hongqu@uestc.edu.cn, maluzhang@u.nus.edu

Abstract

Deep spiking neural network (DSNN) is a promising computational model towards artificial intelligence. It benefits from both the DNNs and SNNs through a hierarchy structure to extract multiple levels of abstraction and the event-driven computational manner to provide ultra-low-power neuromorphic implementation, respectively. However, how to efficiently train the DSNNs remains an open question because of the non-differentiable spike function that prevents the traditional back-propagation (BP) learning algorithm directly applied to DSNNs. Here, inspired by the findings from the biological neural networks, we address the above-mentioned problem by introducing neural oscillation and spike-phase information to DSNNs. Specifically, we propose an Oscillation Postsynaptic Potential (Os-PSP) and phase-locking active function, and further put forward a new spiking neuron model, namely Resonate Spiking Neuron (RSN). Based on the RSN, we propose a Spike-Level-Dependent Back-Propagation (SLDBP) learning algorithm for DSNNs. Experimental results show that the proposed learning algorithm resolves the problems caused by the incompatibility between the BP learning algorithm and SNNs, and achieves state-of-the-art performance in single spike-based learning algorithms. This work investigates the contribution of introducing biologically inspired mechanisms, such as neural oscillation and spike-phase information to DSNNs and providing a new perspective to design future DSNNs.

Introduction

Deep neural networks (DNNs) have achieved great success in computer vision(He et al. 2016), natural language processing(Young et al. 2018; Ghaeini et al. 2018; Lee and Li 2020), speech processing(Abdel-Hamid et al. 2014; Lam et al. 2019) and other fields(Liu et al. 2018; Shao, Liew, and Wang 2020). The hierarchical organization inspired by the biological brain enables DNNs to extract multiple levels of features and the back-propagation (BP) learning algorithm offers an efficient global learning method. However, DNNs' training performs highly rely on large-scale computing resources (e.g., GPUs and server cluster). The high requirements for hardware devices limit the deployment of DNNs on power-critical computation platforms, such as edge devices. Spiking neural networks (SNNs) simulate the biological brain at neuronal level and hold the potential of ultra-low-power neuromorphic implementation through event-driven computation. Besides, the spiking neurons model the signal processing of biological neurons in more detail, then the SNNs model can capture more types of information such as neural oscillation and spike phase information. The study of deep spiking neural networks (DSNNs) benefits from both DNNs and SNNs, and is a promising way towards human-level artificial intelligence. However, how to efficiently train DSNNs remains an open question as the binary nature of spike train and non-differentiable spike function limit usage of BP learning algorithm. To resolve this problem, many learning algorithms and training strategies have been proposed. Depending on the training mechanisms, they can be divided into three categories.

The first category adopts the ANN-SNN transformation strategy, which means that the parameters of a pre-trained ANN are transplanted to the SNN with a similar structure (Esser et al. 2015; Hunsberger and Eliasmith 2015; Esser et al. 2016; Peter et al. 2013; Liu, Chen, and Furber 2017; Diehl et al. 2015, 2016; Bodo et al. 2017; Rueckauer and Liu 2018; Han, Srinivasan, and Roy 2020). The original intention of this transformation strategy is to make full use of the state-of-the-art ANNs, and make the converted SNN have a comparable performance. However, these attempts do not meet the expectation that the loss of accuracy caused by approximation in the transformation process seems inevitable. Although some mitigation strategies have been proposed, such as weight/activation normalization (Diehl et al. 2015, 2016; Bodo et al. 2017) and additional noise (Peter et al. 2013; Liu, Chen, and Furber 2017), the problem is not completely resolved. Furthermore, in order to better match the activation of ANNs, the corresponding SNNs normally use the spike rate coding, which means that the SNN needs to generate a relatively large number of spikes, resulting in obscure time information of spike activity and huge energy consumption (Mostafa 2017).

The second category is the local learning method based on spike-timing-dependent plasticity (STDP), an unsupervised form of synaptic plasticity observed in different brain areas. This category update DSNNs layer-by-layer so that derivative transport between layers in BP is not needed. Besides, STDP updates synaptic weights by considering the

time difference between presynaptic and postsynaptic spikes so that the derivative of spike function is not needed either. Although STDP is capable of training spiking neuron to detect coincidence spike pattern, like other unsupervised learning algorithms, it is difficult to recognize rare but diagnostic unique features. To enhance the neuron's decision-making ability, various STDP-variants have been proposed, such as neuron competition(Kheradpisheh et al. 2018), lateral inhibition, and reward system(Mozafari et al. 2018). However, the improvements from these methods are limited, and most of these methods need an external non-spiking encoding or readout layer, e.g. support vector machines. Thus, these methods Cannot take full advantage of SNNs.

Different from the above mentioned two categories, the third category of learning algorithms directly trains the deep spiking neural networks. The typical examples are SpikeProp(Bohte, Kok, and La Poutre 2002) and its various improvements(Shrestha and Song 2017a,b; Xu et al. 2013). By a linear assumption, the problem of non-differentiable spike function is addressed. However, their methods still suffer from the problems of gradient exploding and dead neurons. These two problems have been partially addressed by adding extra strategies during the training process, such as constraints on weights and gradient normalization (Mostafa 2017). In addition, some works resolve these problems by applying non-leaky PSP function,such as S4NN (Kheradpisheh and Masquelier 2020)and STBDP(Zhang et al. 2020b), or multi-bits spikes(Voelker, Rasmussen, and Eliasmith 2020; Xu et al. 2020). Another path of the third category is the surrogate derivatives with the typical examples of STBP(Wu et al. 2018, 2019; Zhang and Li 2020),and SLAYER (Shrestha and Orchard 2018; Neftci, Mostafa, and Zenke 2019). These learning algorithms show competitive results as compared with their ANN counterparts. However, the computational and memory demands of these algorithms are high and the advantage of event-driven learning is not fully utilized since they need to store the state of neurons at all times in order to learn.

Recent neuroscience researchers have discovered that neuron assemblies fire together and synchronize across brain regions orchestrated by theta oscillations, to encode(Battaglia et al. 2011) and process information. Instead of relying on the increasing firing rates as ANNs, episodic encoding in hippocampal may rely on the temporal precision of single-unit firing with respect to the concurrent theta phase(Hanslmayr, Staresina, and Bowman 2016). A large number of experimental findings suggest that neurons do not perform a simple weighted sum of their inputs and fire based on that sum as in most neural network models. Moreover, theta phase precession is ubiquitous throughout hippocampal subregions, and has also been observed in the entorhinal cortex(Bagur and Benchenane 2018). The phase precession can synchronize the encoding of sequential spatial information. These findings raise the possibility that the biological neuron can be modeled with neural oscillation, and spike-phase contains important information.

Among the existing learning algorithms, directly training algorithms adjust the DSNNs' synaptic weights more efficiently, and are compatible with the sparse encoding method like temporal coding. Hence, the directly training algorithms hold the potential of providing an ultra-low-power neuromorphic implementation. With these considerations, we focus on developing a new spiking neuron model to solve problem proposed before with neural oscillation and spike-phase information. Our main contributions are summarized as follows:

1) We analyze the issues that limit the usage of BP learning algorithm for training DSNNs, including: non-differentiable spike function, gradient explosion and dead neurons during training. Based on these understandings, we propose Oscillation Postsynaptic Potential (Os-PSP) and phase-locking active function for spiking neurons to solve these problems with a single spike.

2) Based on the proposed Os-PSP and phase-locking active function, we propose a Spike-Level-Dependent Back-Propagation (SLDBP) learning algorithm for DSNNs. After that, we explain how to solve the above mentioned three problems with the proposed algorithm.

3) We design the XOR problem from both absolute-oriented and relative-oriented to test the ability of the single-layer resonate SNN to deal with nonlinear problems. Then we design resonate DSNN and apply it to MNIST and CIFAR10 dataset to test the model's ability to process real data.

Experimental results demonstrate that the proposed learning algorithm achieves the state-of-art performance in spike time based learning algorithms of SNNs. This work explores the relationship between neural oscillations and phase-locking from another perspective, providing a new direction for brain-like computing systems.

Problem Description

The great success of DNNs in multiple fields because of the BP algorithm, and this arouses the interest of applying the BP algorithm to DSNN. However, the mechanisms of information encoding and processing between typical artificial neurons in DNNs and SNNs are different. Due to the non-differential spike function, dead neurons, and gradient explosion problems, BP cannot be applied to DSNNs directly. Next, we will discuss these three problems in depth.

In a fully connected DSNN, the dynamic process of a typical spiking neuron can be expressed as follow. Each spike fired at t_i^{l-1} by presynaptic neuron i will generate presynaptic potential (PSP), and further influence the membrane potential $V_j^l(t)$ of the postsynaptic neuron j according to the synaptic efficacy w_{ij}^l, as in Eq.1. As soon as the accumulated membrane potential reaches the threshold θ from below, the postsynaptic neuron fires a spike, like Eq.2. For brevity, we assume each neuron fires only one spike, and the membrane potential of a spiking neuron can be expressed as:

$$V_j^l(t) = \sum_i^N w_{ij}^l K(t - t_i^{l-1}), \quad t_i^{l-1} \leq t, \tag{1}$$

$$t_j^l = t, \quad if \quad V_j^l(t) = \theta, \tag{2}$$

where $V_j^l(t)$ is the membrane potential on neuron j in layer l with N presynaptic neurons, w_{ij}^l is the synaptic efficacy between presynaptic neuron i and postsynaptic neuron j. The

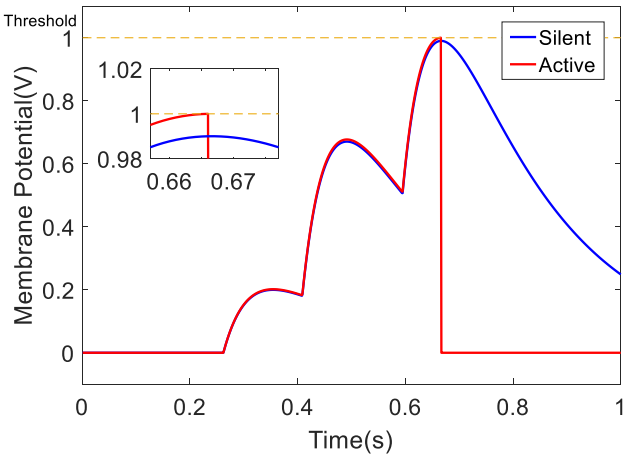

Figure 1: Hair trigger problem. The red line shows the membrane potential of an active spiking neuron which just reaches the threshold and fires a spike at 0.667s. While the blue line indicates if the membrane potential is slightly reduced, this spiking neuron will remain silent without generating any output spike. It can be seen from the inserted figure, the membrane potential reaches the threshold almost in the horizontal direction.

kernel K controls the shape of PSPs, and a typical one used in (Gütig 2016) is:

$$K(s) = V_{norm}(\exp(\frac{s}{\tau_m}) - \exp(\frac{s}{\tau_s})). \tag{3}$$

In order to train DSNN with BP, the derivative of spike time t_j^l with respect to synaptic efficacy w_{ij}^l needs to be calculated, and it can be expanded by the chain rule:

$$\frac{\partial t_j^l}{\partial w_{ij}^l} = \frac{\partial t_j^l}{\partial V_j^l}\frac{\partial V_j^l}{\partial w_{ij}^l}. \tag{4}$$

As show in Eq. 2, the discrete spike function is mathematically nondifferentiable, making it impossible to calculate $\frac{\partial t_j^l}{\partial V_j^l}$ in Eq.4. To alleviate this problem, (Bohte, Kok, and La Poutre 2002) assume the linear relationship between membrane potential V_j^l and spike time t_j^l within an infinite tiny time period before t_j^l, as in Eq.5:

$$\frac{\partial t_j^l}{\partial V_j^l} = -(\frac{\partial V_j^l}{\partial t_j^l})^{-1}. \tag{5}$$

However, the error caused by this approximation will increase with the complexity of the network, and this assumption leads to another problem: gradient exploding.

The gradient explosion problem raises when a spiking neuron almost fires a spike, which also be known as "hair trigger" neuron. As shown in Fig.1, the membrane potential reaches the threshold almost in the horizontal direction, i.e. $\frac{\partial V_j^l}{\partial t_j^l} \approx 0$, and leads to gradient exploding according to Eq.4 and 5. Various attempts, e.g. adaptive learning rate, limited gradient strategy(Bohte, Kok, and La Poutre 2002; Mostafa 2017; Luo et al. 2019; Zhang and Li 2020), have been proposed to alleviate the problem, but the problem has not been completely resolved.

According to Eq.3, the back propagation of the error is entirely dependent on the firing of the neuron. If the neuron doesn't fire, then there's no error return. In a more extreme case, if a neuron doesn't fire a spike for all its synaptic inputs, its synaptic weight will never be "pop-up" again and it becomes a "dead neuron".

Methods

In this section, we integrate Oscillation Postsynaptic Potential (Os-PSP) and phase-locking active functions to spiking neurons, and put forward a new spiking neuron model, namely Resonate Spiking Neuron (RSN) to overcome the three problems that restrict the usage of BP in DSNNs. Then we analyze how these three problems are solved with the proposed RSN. After that, we propose a Spike-Level-Dependent Back-Propagation (SLDBP) learning algorithm for DSNNs formed by RSNs.

Resonate Spiking Neuron Model

Inspired by the recent findings that neural oscillation and phase information play a critical role in information processing in brain, we propose the RSN with Oscillation Postsynaptic Potential (Os-PSP) and phase-locking active function. Instead of the binary spike representation, we use the amount of ions a_i^{l-1} released by spiking neuron i in one spike to increase the encoding capacity of spike trains. The oscillation membrane potential $V_j^l(t)$ of RSN j in layer l at time t can be written as:

$$V_j^l(t) = \sum_i w_{ij}^l a_i^{l-1} K(t - (t_i^{l-1} + d_{ij}^l)), \tag{6}$$

$$K(s) = a_i^{l-1} cos(s), \quad t_i^{l-1} + d_{ij}^l \leq t. \tag{7}$$

As shown in Fig.2, an RSN contains two phases: the collecting phase and the resonating phase. During the collecting phase, the RSN will receive all the ions from presynaptic neurons and its membrane potential starts oscillating. In the resonating phase, the membrane potential of the neuron will gradually stabilize and show periodicity. The neuron will spike when the membrane potential reaches its maximum V_{max}^j in one cycle, the magnitude of the spike $a(t)$ is related to V_{max}^j at the moment of firing, and then enters a resting state. In Fig.2 upper, the amplitude of neural oscillation is reduced when the 3rd and 4th spike coming because the time of the last two spikes is about half a period T different from the time of the first two spikes. On the contrary, in Fig.2 lower, with proper delays, all presynaptic potentials are resonated and generate a much larger membrane potential oscillation.

The information accuracy that a standard binary spike train can encode is limited by its breadth (number of neurons, e.g. population coding) and length (time step, e.g. temporal-based coding or rate-based coding). Inspired by the phase-lock mechanism found in auditory system, we proposed a phase-locking active function. The relation between

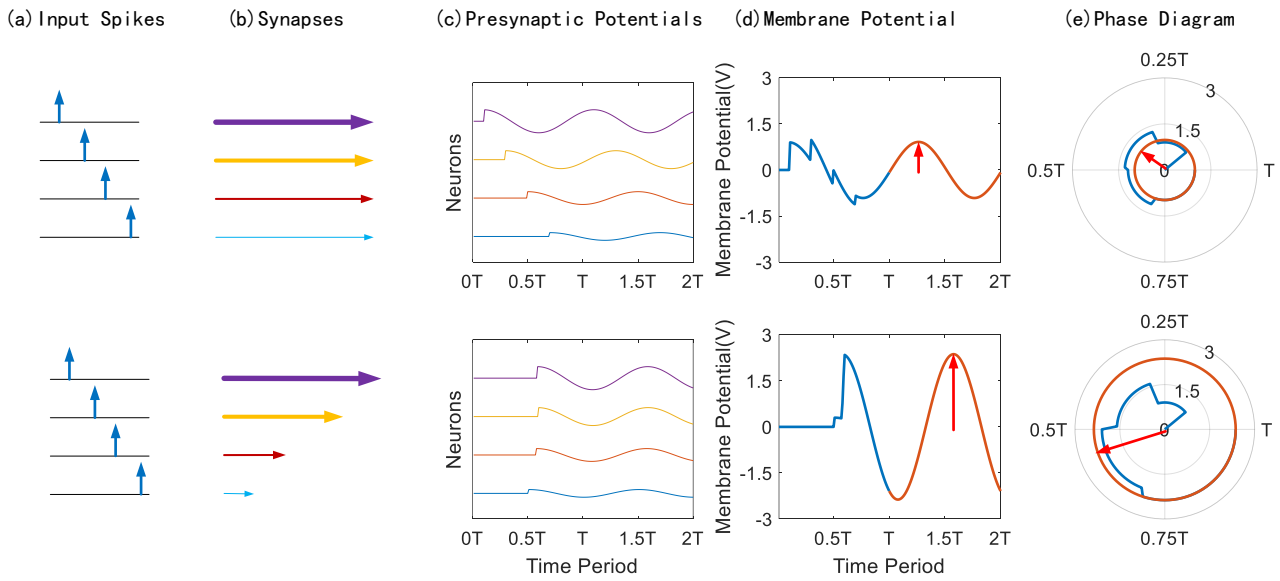

Figure 2: Resonate Spiking Neuron. (a) Input spike train generated by four input neurons. (b) Synaptic weights and delays are represented by the thickness and length of the arrows, respectively. (c) The presynaptic potential is not only related to the amount of ions in spike, but also to the spike time and synaptic delay. (d) The resonate spiking neuron accumulates membrane potential in the first cycle T and then releases a pulse at the crest of the next cycle. (e) The phase diagram corresponding to (d). The amplitude of neural oscillation is represented by the radius in the figure. The length and radius of the red arrows indicate the spike size and time, respectively.

ion amount $a_j(t)$ of the RSN j and firing time t can be defined by:

$$a_j(t) = \begin{cases} F(V_j(t)), & V_j(t) = V_j^{max} \\ 0, & others \end{cases}, \tag{8}$$

where V_j^{max} is the maximum membrane potential in one cycle. Function F is defined to represent relation between ion amount $a_j(t)$ and $V_j(t)$. The specific formula for the function F can be set as needed. In this paper, for simplicity, we define function F as:

$$F(V) = V. \tag{9}$$

According to Eq.6, the RSN membrane potential is not only related to the amount of ions in spikes, but also to the time between spikes. When all input neurons fire spikes at the same time, the resonance scale of the neuron is the largest, then the RSN will be completely equivalent to ANN. In contrast, when the time difference between the two input spikes is exactly half a cycle, the membrane potential oscillations generated by them will inhibit each other, resulting in the decreased amplitude of the membrane potential oscillations of the neurons.

Through the above mechanism, spike trains between RSN neurons convey two kinds of information: one is the size of the spike, which represents the activity level of neurons; the other one is the relative time of the spike, which represents the degree of association between different neurons. By adjusting the relative delay between neurons, a single neuron can show both excitation and inhibition.

Next, we analyse how non-differential spike function, gradient explosion and dead neuron problems can be solved with the proposed RSN.

Solution for the Three Problems

For a DSNN formed with RSN, the binary spike trains are replaced by the amount of ions, the outputs of DSNN are a_o. When training this DSNN with BP, the derivative of ions amount with respect to synaptic efficacy can be calculated as:

$$\frac{\partial a_j^l(t_j^l)}{\partial w_{ij}^l} = \frac{\partial a_j^l(t_j^l)}{\partial V_j^l(t_j^l)} \frac{\partial V_j^l(t_j)}{\partial w_{ij}^l}. \tag{10}$$

The calculation of non-differential part in Eq.2 is replaced by Eq.8. The non-differential spike function is solved.

By introducing Eq.8 and Eq.9, the first term $\frac{\partial a_j^l(t_j^l)}{\partial V_j^l(t_j^l)}$ on the right-hand side of Eq.10 can be calculated as:

$$\frac{\partial a_j^l}{\partial V_j^l(t_j^l)} = 1. \tag{11}$$

And from Eq.6 the second term $\frac{\partial V_j^l(t_j^l)}{\partial w_{ij}^l}$ is:

$$\frac{\partial V_j^l(t_j^l)}{\partial w_{ij}^l} = a_i^{l-1} cos(t_j^l - (t_i^{l-1} + d_{ij}^l)). \tag{12}$$

The range of Eq.11 is $[-\sum_i a_i^{l-1}, \sum_i a_i^{l-1}]$, rather than $(-\infty, +\infty)$. The gradient explosion problem caused by trigger neuron is solved.

As the phase-locking active function defined in Eq.8, RSN always fires a spike when the membrane potential reaches its maximum in one cycle in resonating phase. Information can be fed forward without losing and so does the required error in BP, which solves the dead neuron problem.

Spike-Level-Dependent Back-Propagation (SLDBP) for Resonate SNN

The goal of the algorithm is to learn a set of target activity amounts, denoted a_d^l, at the output neurons for a given set of input patterns $P[a_1^1, .., a_i^1]$. In order to be consistent with the following vision task experiments, we adopt the cross entropy error function as the cost function. Given target class d and actual activity amounts a_d^l, we define the following loss function L:

$$L = -log\big(\frac{\exp(a_d^l)}{\sum_j \exp(a_j^l)}\big). \tag{13}$$

Combining Eq.6 and Eq.8, it can be seen that L is a function of w_{ij}^l and d_{ij}^l. To apply the BP algorithm, we need to calculate:

$$\Delta w_{ij}^l = -\eta \frac{\partial L}{\partial w_{ij}^l}, \tag{14}$$

where η is the learning rate and w_{ij}^l the synaptic weight from neuron i to neuron j. According to the chain rule, we can get:

$$\frac{\partial L}{\partial w_{ij}^l} = \frac{\partial L}{\partial a_j^l}\frac{\partial a_j^l}{\partial w_{ij}^l}. \tag{15}$$

and from Eq.11 and Eq.12

$$\begin{aligned}\frac{\partial a_j^l}{\partial w_{ij}^l} &= 1 \cdot a_i^{l-1} cos(t_j^l - (t_i^{l-1} + d_{ij}^l)) \\ &= a_i^{l-1} cos(t_j^l - (t_i^{l-1} + d_{ij}^l)).\end{aligned} \tag{16}$$

Similar to the traditional BP algorithm, the above weight adjustment rules can be extended to networks with multiple hidden layers. Combine Eq.6 and Eq.11, we get:

$$\begin{aligned}\frac{\partial a_j^l}{\partial a_i^{l-1}} &= \frac{\partial a_j^l}{\partial V_j^l(t_j^l)}\frac{\partial V_j^l(t_j^l)}{\partial a_i^{l-1}} \\ &= 1 \cdot w_{ij}^l cos(t_j^l - (t_i^{l-1} + d_{ij}^l)) \\ &= w_{ij}^l cos(t_j^l - (t_i^{l-1} + d_{ij}^l)).\end{aligned} \tag{17}$$

The synaptic delays can also be adjusted by learning rules similar to weights, where the difference lies in the calculation of partial derivatives $\frac{\partial a_j^l}{\partial d_{ij}^l}$, which is:

$$\begin{aligned}\frac{\partial a_j^l}{\partial d_{ij}^l} &= \frac{\partial a_j^l}{\partial V_j^l(t_j^l)}\frac{\partial V_j^l(t_j^l)}{\partial d_{ij}^l} \\ &= w_{ij}^l a_i^{l-1} sin(t_j^l - (t_i^{l-1} + d_{ij}^l)).\end{aligned} \tag{18}$$

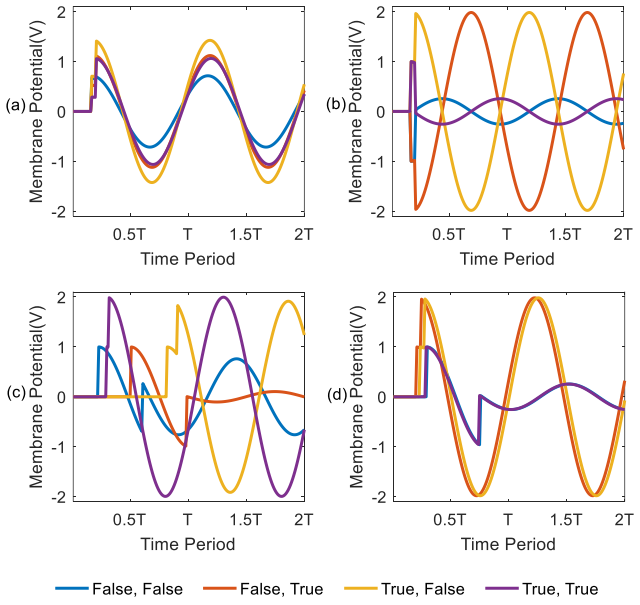

Figure 3: The XOR problems. (a)(b) Membrane potential of RSN before and after learning in absolute-oriented XOR problem. (c)(d) Membrane potential of RSN before and after learning in relative-oriented XOR problem.

.

Experiments and Results

In this section, we first apply the RSN model to the XOR problem. Then the proposed learning algorithm is tested on multiple image data sets. We compare the proposed method with several recent results with the same or similar network sizes previously reported, including traditional ANNs, converted SNNs and different SNN's BP methods.

XOR Problem

In this part, we will apply the RSN model to multiple XOR problems. As mentioned in previous section, the information is encoded as the size and time of the spike. The size of the spike can be used to represent positive and negative classes, and the time of the spike can be used to represent the relationship between two inputs. We trained the model for 20 epochs and then test the ability of the model to handle XOR problems from both absolute-oriented and relative-oriented.

First, we encode $True$ and $False$ as the size of the spike $a^{True} = 0.9$ and $a^{False} = 0.1$, respectively, and the time of two corresponding spike events is the same. After that, we only adjust the weight of a single RSN with two input neurons, and the delay of each synapse is always 0. The results are shown in the Fig.3 upper. It can be seen when the input is $False, False$ or $True, True$, the activity of RSN is significantly higher than other cases in Fig.3(b). From the absolute-oriented, RSN can handle the XOR problem.

Then, we encode the XOR input pattern from relative-oriented: the size of the spike $a^{True} = a^{False} = 1$, which means that both neurons have input, and the difference is the time of the spike. The time corresponding to the same input,

Model	Structure	Method	Accuracy
(Hunsberger and Eliasmith 2016)	MLP	Converted SNN	98.51%
(Hunsberger and Eliasmith 2016)	CNN	Converted SNN	99.07%
(Diehl and Cook 2015)	CNN	STDP SNN	95.00%
(Zhang et al. 2020a)	MLP	Surrogated SNN	96.80%
(Deng et al. 2020)	MLP	Surrogated SNN	98.41%
(Deng et al. 2020)	CNN	Surrogated SNN	99.22%
This work	MLP	Resonate SNN	98.73%
This work	CNN	Resonate SNN	99.26%

Table 1: Comparison of the proposed algorithm against other baseline algorithms on MNIST

such as $False, False$ or $True, True$, is the same, and the time corresponding to different inputs differs by half a cycle T. This time, we still use a single RSN with two inputs for testing. The difference is that this time we only learn the synaptic delay, that is, the relationship between the two inputs, and do not care about their specific input size. Fig.3 shows that When the input is $False, False$ or $True, True$, the activity of RSN is significantly higher than other cases.

Vision Task

To demonstrate the capability of the proposed model and learning algorithm, we choose MNIST and CIFAR10, which are two commonly used datasets for benchmarking vision classification algorithms. All reported experiments below are conducted on an NVIDIA 1080 GPU with Pytorch framework. The experimented SNNs are based on the RSN model described in the last section and all experiment results are obtained by repeating the experiments five times. For the MNIST dataset, we adopted Adaptive moment estimation(Adam) as the optimizer and trained for 150 epochs. For the CIFAR10 dataset, we adopted Stochastic Gradient Descent(SGD) as the optimizer and trained for 200 epochs .

Network Structure The network models we trained or compared with are either multi-layered perceptron(MLP) or convolutional neural networks(CNNs). These two network structures are shown in Table.2 with C stands for convolutional layer, P stands for pooling layer and FC stands for fully a connected layer.

Dataset	Network Structure
MNIST	784-400-10
MNIST	6C5-P2-16C5-P2-128FC-10
CIFAR10	64C3-P2-128C3-128C3-P2-256FC-10

Table 2: Network structure used for vision task.

In the spike-pooling layer, we select the spike with the largest amount and retain its phase information to achieve a winner-takes-all effect while retaining its temporal information with other spike.

Encoding When SNNs are used to process real-world data, the data should be encoded into spike patterns. Here, we encode each pixel value of image into spike's amount a^i directly and time t^i base on latency coding method. It should be noticed that the firing time is limited in $[0 \cdots 0.5T]$, T is the period of neural oscillation. For example, the input spike amount corresponding to pixel value of $0.4I_{max}$ is $a^i = 0.4$, and the firing time is $t^i = 0.4 \times 0.5T$. In this way, the pixel with higher value in the original image fires a larger spike, and the pixels with similar value fire at the same time.

Accuracy Table.1 compares the performance of different network architectures and algorithms on the MNIST dataset. It can be seen that the proposed algorithm achieves the best results without additional optimization strategies. The accuracy reported below is obtained by repeating the experiments five times.

Model	Method	Accuracy
(Nair and Hinton 2010)	CNN with ReLU	84.15%
(Sengupta et al. 2019)	Converted SNN	76.81%
(Deng et al. 2020)	Surrogated SNN	74.23%
This work	Resonate SNN	84.87%

Table 3: Comparison of the proposed model against other baseline models on CIFAR10

Table.3 lists the results of the existing state-of-the-art SNNs-based learning methods on CIFAR10 dataset. To ensure the fairness of the comparison, all the learning algorithms are compared with same network structure as shown in Table. 2. We use the CNN with ReLU as the activation function as the baseline, which contains only the convolutional layer and the max-pooling layer. As shown in Table. 3, both conversion and surrogate SNNs-based learning algorithms suffer from accuracy loss because of the approximation. However, our work achieves an accuracy of 84.87%, which is 8.06% higher than its counterparts.

Conclusion

In this paper, we put forward a new DSNN model composed of the proposed RSN that considers the neural oscillation mechanism and spike-phase information. In addition, we also propose a Spike-Level-Dependent Back-Propagation (SLDBP) learning algorithm for DSNNs. Experimental results demonstrate that the proposed method helps to resolve the problems caused by the incompatibility between the BP

learning algorithm and SNNs. We also design both fully-connected and convolution architectures to compare the proposed algorithm with other state-of-the-art SNN models on the classical visual processing data sets MNIST and CIFAR10. The experimental results show that the proposed model achieves the best results, and exceeds the best model by 0.04% and 8.06% respectively on MNIST and CIFAR10. This work investigates the contribution of introducing biologically inspired mechanisms, such as neural oscillation and spike-phase information to DSNNs and providing a new perspective to design future DSNNs.

Acknowledgments

This work was supported by National Key R&D Program of China under Grant 2018AAA0100202, and in part by the National Science Foundation of China under Grant 61976043, and in part by the Zhejiang Lab's International Talent Fund for Young Professionals, and in part by the China Postdoctoral Science Foundation (Grant No. 2020M680148). (Corresponding author: Hong Qu, email:hongqu@uestc.edu.cn)

References

Abdel-Hamid, O.; Mohamed, A.-r.; Jiang, H.; Deng, L.; Penn, G.; and Yu, D. 2014. Convolutional Neural Networks for Speech Recognition. *IEEE ACM Transactions on Audio, Speech, and Language Processing* 22(10): 1533–1545.

Bagur, S.; and Benchenane, K. 2018. The Theta Rhythm Mixes and Matches Gamma Oscillations Cycle by Cycle. *Neuron* 100(4): 768–771.

Battaglia, F. P.; Benchenane, K.; Sirota, A.; Pennartz, C. M.; and Wiener, S. I. 2011. The Hippocampus: Hub of Brain Network Communication for Memory. *Trends in Cognitive Sciences* 15(7): 310–318.

Bodo, R.; Iulia-Alexandra, L.; Hu, Y.; Michael, P.; and Liu, S. C. 2017. Conversion of Continuous-Valued Deep Networks to Efficient Event-Driven Networks for Image Classification. *Frontiers in Neuroscience* 11: 682.

Bohte, S. M.; Kok, J. N.; and La Poutre, H. 2002. Error-Backpropagation in Temporally Encoded Networks of Spiking Neurons. *Neurocomputing* 48(1-4): 17–37.

Deng, L.; Wu, Y.; Hu, X.; Liang, L.; Ding, Y.; Li, G.; Zhao, G.; Li, P.; and Xie, Y. 2020. Rethinking the Performance Comparison Between SNNS and ANNS. *Neural Networks* 121: 294–307.

Diehl, P. U.; and Cook, M. 2015. Unsupervised Learning of Digit Recognition Using Spike-Timing-Dependent Plasticity. *Frontiers in computational neuroscience* 9: 99.

Diehl, P. U.; Neil, D.; Binas, J.; Cook, M.; and adn Michael Pfeiffer, S.-C. L. 2015. Fast-Classifying, High-Accuracy Spiking Deep Networks Through Weight and Threshold Balancing. 1–8.

Diehl, P. U.; Pedroni, B. U.; Cassidy, A.; Merolla, P.; Neftci, E.; and Zarrella, G. 2016. Truehappiness: Neuromorphic emotion recognition on truenorth. In *2016 International Joint Conference on Neural Networks (IJCNN)*, 4278–4285. IEEE.

Esser, S. K.; Appuswamy, R.; Merolla, P.; Arthur, J. V.; and Modha, D. S. 2015. Backpropagation for Energy-Efficient Neuromorphic Computing. *In Advances in Neural Information Processing Systems* 1117–1125.

Esser, S. K.; Merolla, P.; Arthur, J.; Cassidy, A. S.; Appuswamy, R.; Andreopoulos, A.; Berg, D.; McKinstry, J. L.; Melano, T.; Barch, D.; Nolfo, C. D.; Datta, P.; Amir, A.; Taba, B.; Flickner, M.; and Modha, D. 2016. Convolutional networks for fast, energy-efficient neuromorphic computing. *Proceedings of the National Academy of Sciences* 113: 11441 – 11446.

Ghaeini, R.; Hasan, S. A.; Datla, V.; Liu, J.; Lee, K.; Qadir, A.; Ling, Y.; Prakash, A.; Fern, X.; and Farri, O. 2018. DR-BiLSTM: Dependent Reading Bidirectional LSTM for Natural Language Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 1460–1469.

Gütig, R. 2016. Spiking Neurons can Discover Predictive Features by Aggregate-Label Learning. *Science* 351(6277).

Han, B.; Srinivasan, G.; and Roy, K. 2020. RMP-SNN: Residual Membrane Potential Neuron for Enabling Deeper High-Accuracy and Low-Latency Spiking Neural Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 13555–13564.

Hanslmayr, S.; Staresina, B. P.; and Bowman, H. 2016. Oscillations and Episodic Memory: Addressing the Synchronization/Desynchronization Conundrum. *Trends in Neurosciences* 39(1): 16–25.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hunsberger, E.; and Eliasmith, C. 2015. Spiking Deep Networks with LIF Neurons. *ArXiv* abs/1510.08829.

Hunsberger, E.; and Eliasmith, C. 2016. Training Spiking Deep Networks for Neuromorphic Hardware. *ArXiv* abs/1611.05141.

Kheradpisheh, S. R.; Ganjtabesh, M.; Thorpe, S.; and Masquelier, T. 2018. STDP-Based Spiking Deep Convolutional Neural Networks for Object Recognition. *Neural Networks* 99: 56 – 67.

Kheradpisheh, S. R.; and Masquelier, T. 2020. S4NN: Temporal Backpropagation for Spiking Neural Networks with One Spike per Neuron. *International Journal of Neural Systems* 30(6): 2050027.

Lam, M. W.; Chen, X.; Hu, S.; Yu, J.; Liu, X.; and Meng, H. 2019. Gaussian Process Lstm Recurrent Neural Network Language Models for Speech Recognition. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7235–7239. IEEE.

Lee, G.; and Li, H. 2020. Modeling Code-Switch Languages Using Bilingual Parallel Corpus. In *Proceedings of the 58th*

Annual Meeting of the Association for Computational Linguistics, 860–870.

Liu, C.; Zhu, E.; Zhang, Q.; and Wei, X. 2018. Modeling of Agent Cognition in Extensive Games via Artificial Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* 29(10): 4857–4868.

Liu, Q.; Chen, Y.; and Furber, S. 2017. Noisy Softplus: an activation function that enables SNNs to be trained as ANNs. *ArXiv* abs/1706.03609.

Luo, X.; Qu, H.; Zhang, Y.; and Chen, Y. 2019. First error-based supervised learning algorithm for spiking neural networks. *Frontiers in Neuroscience* 13: 559.

Mostafa, H. 2017. Supervised Learning Based on Temporal Coding in Spiking Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* 29(7): 3227–3235.

Mozafari, M.; Kheradpisheh, S. R.; Masquelier, T.; Nowzari-Dalini, A.; and Ganjtabesh, M. 2018. First-Spike-Based Visual Categorization Using Reward-Modulated STDP. *IEEE Transactions on Neural Networks and Learning Systems* 29(12): 6178–6190.

Nair, V.; and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*.

Neftci, E. O.; Mostafa, H.; and Zenke, F. 2019. Surrogate Gradient Learning in Spiking Neural Networks. *IEEE Signal Processing Magazine* 36: 61–63.

Peter, O.; Daniel, N.; Liu, S. C.; Tobi, D.; and Michael, P. 2013. Real-Time Classification and Sensor Fusion with a Spiking Deep Belief Network. *Frontiers in Neuroscience* 7(7): 178.

Rueckauer, B.; and Liu, S. C. 2018. Conversion of Analog to Spiking Neural Networks Using Sparse Temporal Coding. In *IEEE International Symposium on Circuits & Systems (ISCAS)*, 1–5.

Sengupta, A.; Ye, Y.; Wang, R.; Liu, C.; and Roy, K. 2019. Going Deeper in Spiking Neural Networks: Vgg and Residual Architectures. *Frontiers in Neuroscience* 13: 95.

Shao, Y.; Liew, S.; and Wang, T. 2020. AlphaSeq: Sequence Discovery With Deep Reinforcement Learning. *IEEE Transactions on Neural Networks and Learning Systems* 31: 3319–3333.

Shrestha, S. B.; and Orchard, G. 2018. SLAYER: Spike Layer Error Reassignment in Time. In *Advances in Neural Information Processing Systems*, 1412–1421.

Shrestha, S. B.; and Song, Q. 2017a. Robust Spike-Train Learning in Spike-Event Based Weight Update. *Neural Networks* 96: 33–46.

Shrestha, S. B.; and Song, Q. 2017b. Robustness to Training Disturbances in Spikeprop Learning. *IEEE Transactions on Neural Networks and Learning Systems* 29(7): 3126–3139.

Voelker, A. R.; Rasmussen, D.; and Eliasmith, C. 2020. A Spike in Performance: Training Hybrid-Spiking Neural Networks with Quantized Activation Functions. *ArXiv* abs/2002.03553.

Wu, Y.; Deng, L.; Li, G.; Zhu, J.; and Shi, L. 2018. Spatio-Temporal Backpropagation for Training High-Performance Spiking Neural Networks. *Frontiers in Neuroscience* 12: 331.

Wu, Y.; Deng, L.; Li, G.; Zhu, J.; Xie, Y.; and Shi, L. 2019. Direct Training for Spiking Neural Networks: Faster, Larger, Better. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1311–1318.

Xu, C.; Zhang, W.; Liu, Y.; and Li, P. 2020. Boosting Throughput and Efficiency of Hardware Spiking Neural Accelerators Using Time Compression Supporting Multiple Spike Codes. *Frontiers in Neuroscience* 14.

Xu, Y.; Zeng, X.; Han, L.; and Yang, J. 2013. A Supervised Multi-Spike Learning Algorithm Based on Gradient Descent for Spiking Neural Networks. *Neural Networks* 43: 99–113.

Young, T.; Hazarika, D.; Poria, S.; and Cambria, E. 2018. Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Computational Intelligence Magazine* 13(3): 55–75.

Zhang, M.; Luo, X.; Chen, Y.; Wu, J.; Belatreche, A.; Pan, Z.; Qu, H.; and Li, H. 2020a. An Efficient Threshold-Driven Aggregate-Label Learning Algorithm for Multimodal Information Processing. *IEEE Journal of Selected Topics in Signal Processing* 14(3): 592–602.

Zhang, M.; Wang, J.; Zhang, Z.; Belatreche, A.; Wu, J.; Chua, Y.; Qu, H.; and Li, H. 2020b. Spike-Timing-Dependent Back Propagation in Deep Spiking Neural Networks. *ArXiv* abs/2003.11837.

Zhang, W.; and Li, P. 2020. Temporal Spike Sequence Learning via Backpropagation for Deep Spiking Neural Networks. *ArXiv* abs/2002.10085.