frontiers

Neuronal-plasticity and Reward-propagation improved Recurrent Spiking Neural Networks

Shuncheng Jia[1,2,#], **Tielin Zhang**[1,2,#,*], **Xiang Cheng**[1,2], **Hongxing Liu**[1,4] **and Bo Xu**[1,2,3,*]

[1] *Institute of Automation, Chinese Academy of Sciences, Beijing, China*
[2] *School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China*
[3] *Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Shanghai, China*
[4] *Beijing University of Technology, Beijing, China*
[#] *Contributed equally*

Correspondence*:
Tielin Zhang and Bo Xu
tielin.zhang@ia.ac.cn, xubo@ia.ac.cn

Tielin Zhang and Bo Xu in Institute of Automation, Chinese Academy of Sciences, Beijing, China, tielin.zhang@ia.ac.cn, xubo@ia.ac.cn.

ABSTRACT

Different types of dynamics and plasticity principles founded out from natural neural networks have been well applied on Spiking neural networks (SNNs) for their biologically-plausible efficient and robust computations compared to their counterpart deep neural networks (DNNs). Here we further propose a special Neuronal-plasticity and Reward-propagation improved Recurrent SNN (NRR-SNN). The historically-related adaptive threshold with two channels is highlighted as important neuronal plasticity for increasing the neuronal dynamics, and then global labels instead of errors are used as a reward for the paralleling gradient propagation. Besides, a recurrent loop with proper sparseness is designed for robust computation. Higher accuracy and stronger robust computation are achieved on two sequential datasets (i.e., TIDigits and TIMIT datasets), which to some extent, shows the power of the proposed NRR-SNN with biologically-plausible improvements.

Keywords: Spiking Neural Network, Neuronal Plasticity, Synaptic Plasticity, Reward Propagation, Sparse Connections.

1 INTRODUCTION

Many different types of deep neural networks (DNNs) have been proposed for the efficient machine learning on image classification [1], recognition [2], memory association [3] and prediction [4]. However, with the rapid development of DNNs, some shortcomings come in line with their prosperity.

- The first problem is the increasing number of synaptic parameters. Different types of structures instead of neurons play important roles in different functions of DNNs, where nearly all artificial neurons use a Sigmoid-like activation function for simple non-linear input-output mapping. The unbalanced complexity between artificial neurons and networks makes DNNs contain a large number of network parameters to be tuned.

- The second problem is the slow backpropagation (BP) with high computational cost, which is also considered not biologically-plausible. In DNNs, the BP interleaves with feedforward propagation sequentially, and the error signals have to be backpropagated from the output neurons to hidden neurons layer-by-layer, with a risk of gradient disappearance or gradient explosion, especially for extremely-deep networks. The nature of supervised and synchronous computation of DNNs also makes them hard to be accelerated by parallel computation.
- The third problem is that all of the artificial neurons in DNNs during BP procedure have to satisfy the limitation of mathematical differentiability, which obviously lacks support from biological verification, where the non-differential spike-type signals are everywhere, caused by the time slot of membrane potential at firing threshold, the probabilistic firing of a specific spike, or the hard refractory time for stop firing.
- The fourth problem is the separation of spatial and temporal information with different network architectures. For example, the convolutional kernels are carefully designed for efficient spatial information integration, and the recurrent loops (sparse or dense types) are successfully introduced for effective sequential information prediction, instead of simultaneously spatially-temporal information processing in biological networks.

Unlike DNNs, some other networks are designed to contain both biologically-realistic network structures and biologically-plausible tuning methods. Spiking neural network (SNN) is one of them, which contains spiking neurons with dynamic membrane potential and also dynamic synapses for spatially-temporal information processing. There are many advantages of SNNs compared to their counterpart DNNs. For example, the two-bit efficient encoding of information at the neuron scale; the balanced complexity between the neuron and network scales, i.e., with proper-sparseness connections (neurons only connect in a certain area) and far-more complex neurons (neurons with dynamic somas and dendrites).

Besides, SNNs prefer using the biologically-plausible tuning methods, such as spike-timing-dependent plasticity (STDP) [5], short-term plasticity (STP) [6], pre-post membrane balanced plasticity [7, 8], and excitatory-inhibitory balanced plasticity [9]. The long-term depression (LTD) [10] shows that the repeated low-frequency activation into postsynaptic neurons will reduce the transmission efficiency of synapses, while that with repeated high-frequency (long-term potentiation, LTP [11]) will lead to synaptic enhancement. STDP [12] shows presynaptic and postsynaptic activations of different neurons in chronological order would result in different (with an increment or decrement) synaptic changes, i.e., if the postsynaptic neuron fired within 20 milliseconds after the activation of the presynaptic neuron, it will cause LTP, or else LTD. Besides, more effective plasticity propagation rules are founded out and have been well applied to the training of SNNs. The reward propagation [13] describes an efficient label-based instead of error-based gradient propagation. Synaptic plasticity propagation describes LTP/LTD propagation in neighborhood synapses [14]. Most of these plasticity propagation rules are biologically-plausible for the efficient learning of SNNs.

There are also some shortcomings of SNNs. First, due to the non-differential character of biological neurons in SNNs, the gradient backpropagation that is power on tuning DNNs is not directly applicable on the training of SNNs; Second, ordinary SNNs have limited neuronal dynamics, by omitting dynamic threshold and other related features of biological networks. These phenomenons make the current SNNs more closed to DNNs towards an unbalanced complexity between local neurons and global networks, instead of that with a balanced complexity in biological networks.

This paper focuses more on the research on neuronal dynamics, learning plasticity, and sparseness architectures of SNNs, toward a more efficiently biologically-plausible computation. Hence, under these considerations, the Neuronal-plasticity and Reward-propagation improved Recurrent SNN (NRR-SNN) is proposed for efficient and robust computations. We think the contribution of this paper can be concluded as the following parts:

- Firstly, the historically-related two-channel adaptive threshold is highlighted as an important neuronal plasticity for increasing the neuronal dynamics. This additional neuronal dynamics will integrate well with other dynamic membrane potentials (e.g., the leaky integrated-and-fire, LIF) for a stronger temporal information computation.
- Secondly, the global labels, instead of errors, are used as a reward for the gradient propagation. This new learning method can also be parallelly computed in order to save more computational cost.
- Thirdly, dynamic neurons are then connected in a recurrent loop with defined sparseness for the robust computation. Moreover, an additional parameter is set for representing the degree of sparseness for analyzing the proposed algorithm's anti-noise performance.

The paper is organized as follows: The Section 2 will give a brief introduction of related works. Section 3 will show some basic background knowledge about dynamic neurons, the procedure of feedforward propagation, and plasticity propagation in standard SNNs. A detailed description of the proposed NRR-SNN is given in Section 4, including the dynamic nodes with neuronal plasticity, the architecture with different sparseness, and the tuning method reward propagation. Section 5 will show the proposed algorithm's performance on two standard sequential datasets (i.e., TIDigits and TIMIT) on their efficient and robust computations. A further discussion and conclusion will be given to analyze these inspirations from the brain in Section 6.

2 RELATED WORKS

The multi-scale plasticity in SNN covers the neuronal plasticity, synaptic plasticity, and plasticity propagations. Neuronal plasticity plays a critical role in the dynamic information processing of the biological neural network [15, 16]. The standard neurons in SNNs include the H-H model [17, 18, 19], LIF model [20], SRM model [21, 22], and Izhikevich model [23]. The VPSNN (short for voltage-dependent and plasticity-centric SNN) has been proposed, which contained the neuronal plasticity focusing more on membrane potential dynamics with a static firing threshold [7]. Yu et al. [24] have also proposed several plasticity algorithms to deal with spike coding's neuronal plasticity during training.

Synaptic plasticity refers to the dynamic changes of synapses according to different tasks. Zenke et al. [25] have proposed the SuperSpike, where a non-linear voltage-based three-factor learning rule was used to update neuronal plasticity at synapse scale dynamically. Kheradpisheh et al. [26] have proved that the STDP plasticity was simpler and superior to other unsupervised learning rules in the same network architectures.

The propagation of synaptic plasticity is closely related to the credit assignment of error signals in SNNs. Zhang et al. have given an overview introduction of several target propagation methods, such as error propagation, symbol propagation, and label propagation [27], where the reward propagation can propagate the reward (instead of the traditional error signals) directly to all hidden layers (instead of the traditional layer-to-layer backpropagation). We think this plasticity is biologically-plausible and will also use it as the main credit assignment of SNNs in our NRR-SNN algorithm. Zhao et al. have proposed a similar method, where global random feedback alignment is combined with STDP for efficient credit assignment [28].

Besides the plasticity, network structures have played important roles in network learning. Until now, the network structures in SNNs are similar to their counterpart DNNs [29, 30], depending on the requirement of different spatial or temporal tasks. For example, feedforward-type architectures are usually used during the spatial information processing (e.g., the image classification on the MNIST dataset) [16, 31], and recurrent-type architectures are more constructed for the sequential information processing (e.g., the auditory sequence recognition on the TIDigits dataset) [32, 33].

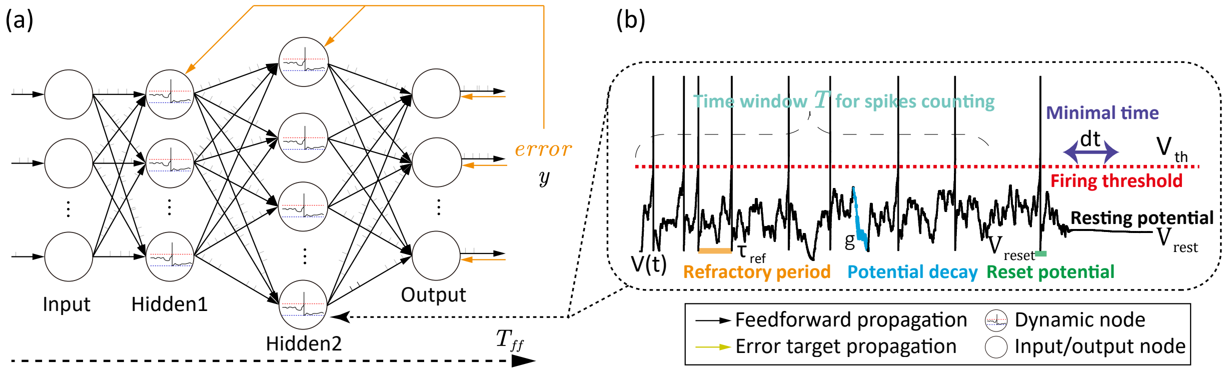


Figure 1. A schematic diagram depicting the SNN with dynamic neurons, feedforward spike propagation, and feedback error propagation. (a) The feedforward propagation and error target propagation in the standard SNN, containing dynamic neurons at spiking or resting states. (b) The dynamic LIF neuron with spikes and subthreshold membrane potential.

3 BACKGROUND

3.1 Dynamic spiking neurons

The dynamic spiking neurons in SNNs are not continuous in the real number field, which is different from the artificial neurons such as the Sigmoid activation function, Tanh activation function, and Rectified linear unit (ReLU). The standard LIF neuron is shown as follows:

$$\begin{cases} C\frac{dV_i(t)}{dt} = g(V_i(t) - V_{rest}) + \sum_{j=1}^{N} W_{i,j}X_j(t) \\ V_i(t) = V_{reset} \quad if(V_i(t) = V_{th}, t - t_{spike} > \tau_{ref}) \end{cases}, \tag{1}$$

where $V_i(t)$ is membrane potential, V_{th} is firing threshold, V_{reset} is reset membrane potential (also generating a spike at the same time) and V_{rest} is the resting potential. τ_{ref} is the refractory time period, where the $V_i(t)$ will not increase towards the V_{th} at time t only if it is still during the period of τ_{ref}. $X_j(t)$ is the receiving LIF neuron input from the presynaptic neuron j. One schematic diagram of dynamic LIF neuron is shown in Fig. 1(b).

3.2 Feedforward propagation in SNN

Fig. 1(a) shows the sequential spike trains in the feedforward propagation (labeled as period T_{ff}) of SNNs for each epoch. For example, as a speech, it is spitted as N frames, and each frame is encoded as a normally-distributed spike train. Then these spike trains are sequentially inputted into the feedforward procedure of SNN. The information encoding in each LIF neuron with spikes is shown as follows:

$$\begin{cases} C\frac{dV_i^f(t)}{dt} = g(V_i(t) - V_{rest})(1 - S^f) + \sum_{j=1}^{N} W_{i,j}^f X_j(t) \\ V_i^f(t) = V_{reset}, S^f = 1 \quad if\left(V_i^f(t) = V_{th}\right) \\ S^f = 1 \quad if\left(t - t_{spike^f} < \tau_{ref}, t \in (1, T_1)\right) \end{cases}, \tag{2}$$

where $V_i^f(t)$ is the feedforward membrane potential with historically integrated states, S is a spike flag for the neuron, which indicates the number of spikes when the $V_i(t)$ (where $V_i^f(t)$ is part of $V_i(t)$) reaches V_{th}. The S also controls the refractory time period τ_{ref} by resetting the historical membrane potential $g(V_i(t) - V_{rest})$ instead of blocking the $V_i^f(t)$ directly.

3.3 Standard target propagation

The standard backpropagation (BP) [34] uses the gradient descent algorithm to modify the synaptic weights layer-by-layer with the differential chain rule. However, the derivative of activation functions is usually less than 1, making the backpropagated gradient vanish in some deeper layers.

This paper wants to modify all synaptic weights parallelly without worrying about the gradient vanishing problem, especially for dynamic LIF neurons. Hence, we will pay more attention to the target propagation [27], as shown in Fig. 1(a), where the error or other reward-like signals are directly propagated from the output layer to all hidden layers parallelly without losing accuracy.

4 METHOD

Here we will give a detailed introduction about NRR-SNN, including three main parts: the neuronal plasticity with 2-channel dynamic firing threshold; the recurrent connections with different proportions of sparseness; the reward propagation with the direct tuning of synaptic weights with loaded labels, as shown in Fig. 2.

4.1 Neuronal plasticity

The neuronal plasticity is different from traditional synaptic plasticity, where more dynamic characteristics within neurons are discussed for better spatially-temporal information processing. Here we set an adaptive threshold with an ordinary differential equation (ODE). This is an ingenious effort to make a dynamic firing threshold with an attractor in ODE, instead of directly setting that as a predefined static value, as shown in the following equation:

$$\frac{da_i(t)}{dt} = (\alpha - 1)a_i(t) + \beta(S^f + S^r), \tag{3}$$

where $a_i(t)$ is a dynamic threshold with an equilibrium point of 0 without input spikes, or with another equilibrium point of $-\frac{\beta}{\alpha-1}$ given input spikes $S^f + S^r$ from feedforward and recurrent channels. Hence, the ODE of membrane potential for LIF neuron is updated as follows:

$$C\frac{dV_i(t)}{dt} = g\left(V_i(t) - V_{rest}\right)\left(1 - S^f - S^r\right) + \sum_{j=1}^{N} W_{i,j} X_j(t) - \gamma a_i(t), \tag{4}$$

where during the period from the resetting to the firing of membrane potential, the dynamic threshold parameter $a_i(t)$ is accumulated gradually and eventually reached a relatively stable value. Because of the

$-\gamma a_i(t)$, the firing threshold is increased into $V_{th} + \gamma a_i(t)$. For the $a_i(t)$, we can solve the stable value $a_i^* = \frac{\beta}{1-\alpha}(S^f + S^r)$.

In this paper, we give $\alpha = 0.9$, $\beta = 0.1$ and $\gamma = 1$, therefore the stable $a^* = 0$ for no spikes, $a^* = 1$ for one spike, and $a^* = 2$ for spikes from two channels (i.e., the feedforward and recurrent channels). when $a_i(t) < (S^f + S^r)$, $a_i(t)$ will increase and the threshold increase, otherwise, they will both decrease. It can be considered that the threshold will be changed dynamically with neurons' discharge. The adaptive threshold will also be increased or decreased when the firing frequency is higher or lower. Here we use it as the main controlling part of neuronal plasticity.

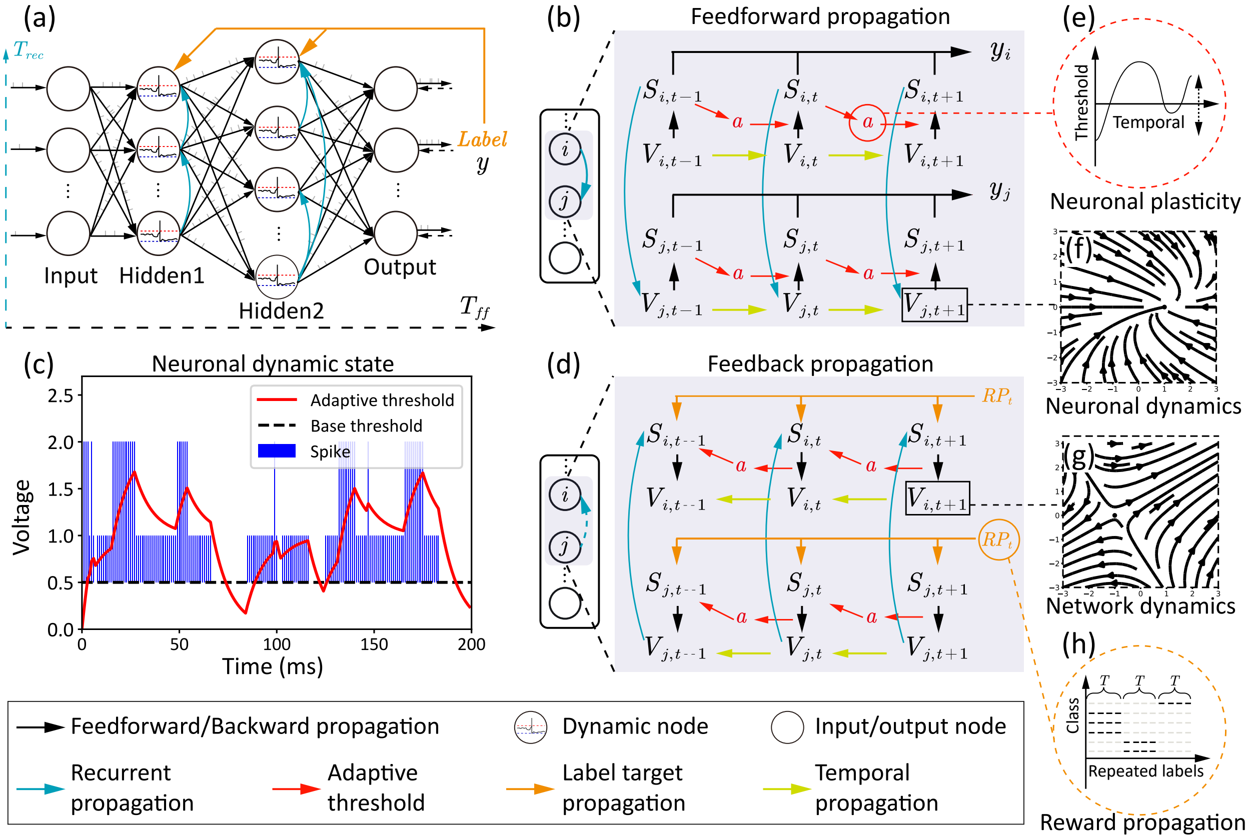


Figure 2. The architecture, two phases of information propagations, and multi-scale dynamics in NRR-SNN. (a) The SNN architecture with the feedforward period T_{ff}, the recurrent period T_{rec}, and the reward propagation with labels. (b) The feedforward information propagation from input neurons $V_{i,t}$ to network output $y_{j,t}$. (c,e) The two-channel neuronal plasticity related to spike trains. (d) The feedback information propagation from label RP_t to hidden neurons $V_{i,t}$, $V_{i,t-k}$ where $k \in T$. (f,g) The vector field examples of dynamic membrane potentials. (h) A diagram depicting the reward propagation with teaching signals of repeated labels.

4.2 Architecture with sparse loops

Recurrent connections show the dynamics at network scale, as shown in Fig. 2(a), where neurons are connected within the inner hidden layers with defined or learnable connections. Hence, two types of membrane potentials are combined in the dynamic neurons. One is the recurrent membrane potential $V_i^r(t)$, and the other is the feedforward membrane potential $V_i^f(t)$. The definitions of these two types of membrane potential can be considered as two channels with the following equations:

$$\begin{cases} V_i^f(t) = V_{reset}, S^f = 1 \quad if(V_i^f(t) = V_{th}) \\ V_i^r(t) = V_{reset}, S^r = 1 \quad if(V_i^r(t) = V_{th}) \\ S^f = 1 \quad if(t - t_{sf} < \tau_{ref}, t \in (1, T_1)) \\ S^r = 1 \quad if(t - t_{s^r} < \tau_{ref}, t \in (1, T_2)) \end{cases}, \tag{5}$$

where two spike flags (S^f and S^r) are defined separately. The dynamic membrane potential of $V_i^r(t)$ and $V_i^f(t)$ are then integrated together, defined as follows:

$$\begin{cases} C\frac{dV_i^f(t)}{dt} = g(V_i(t) - V_{rest})(1 - S) + \sum_{j=1}^N W_{i,j}^f X_j(t) \\ C\frac{dV_i^r(t)}{dt} = \sum_{j=1}^N W_{i,j}^r S \\ V_i(t) = V_i^f(t) + V_i^r(t) \\ S = S^f + S^r \end{cases}, \tag{6}$$

where feedforward T_{ff} and recurrent period T_{rec} are integrated together at memrbane potential $V_i(t) = V_i^f(t) + V_i^r(t)$ and firing flag $S = S^f + S^r$. The $V_i^r(t)$ saves the historical membrane potentials of the adjacent neurons. Furthermore, the recurrent SNN is designed with network dynamics from different scales, as shown in Fig. 2(a), where sparse or dense connections are given to the neurons in the same hidden layer.

4.3 Global Reward Propagation

Different from standard target propagations (the detailed description of them is shown in Section 3.3), the reward propagation uses labels instead of errors as the teaching signals for the tuning of synaptic weights in the hidden layers, as shown in Fig. 2(a,h).

The reward propagation has been reported in our previous work, where only feedforward connections are introduced [13]. Here we update it to satisfy the criteria of both feedforward and recurrent propagations in NRR-SNN architecture. The main idea is also trying to get the state differences from the propagated target gradient, which is defined as follows:

$$\begin{cases} Grad_{RP} = B_{rand}^{f,l} * RP_t - h^{f,l} \\ \Delta W_t^{f,l} = -\eta^f (Grad_{RP}) \\ \Delta W_t^{r,l} = -\eta^r \left(Grad_{t+1} + Grad_{RP}\right) \end{cases}, \tag{7}$$

where $Grad_{RP}$ is the gradient of reward propagation, $B_{rand}^{f,l}$ is a predefined random matrix for the dimension conversion from output layer to the hidden layer l, $h^{f,l}$ is the current layer state, RP_t is the spike train repeated with one-hot labels, $W^{f,l}$ is the synaptic weight at feedforward procedure at the layer l, $W_t^{r,l}$ is the recurrent synaptic weight at layer l, $Grad_{t+1}$ is the gradient calculated from the time $t + 1$.

4.4 Local gradient propagation with pseudo-BP

Here we use pseudo-BP to make the membrane potential differentiable especially for that at the firing time. During the process of the torch.autograd in toolbox PyTorch, we set some “functional hook”, to storage the spike signals and synaptic weight values generated from the feedforward procedure. This hook will then be automatically triggered as a backpropagate function for the pseudo-BP approximation in the feedback procedure.

The $Grad_{local}$ is used to represent the local gradient from hidden membrane potentials to synaptic weights. In this procedure, the non-differential part is only the period of $V_i(t)$ at $V_i(t) = V_{th}$. Hence, the $Grad_{local}$ of the neuron i is revised as follows:

$$Grad_{local} = \frac{\partial S_i(t)}{\partial V_i(t)} = \begin{cases} 1 & if\ (|V_i(t) - V_{th}| < V_{window}) \\ 0 & \text{else} \end{cases}, \tag{8}$$

where only the differential parts are calculated, or else are omitted. Then, the weight gradient of the full connection and loop connection will be calculated by the automatic derivation mechanism of PyTorch.

4.5 The learning procedure of NRR-SNN

After integrating these three main parts, i.e., the neuronal plasticity, recurrent architecture, and reward propagation, we will get the integrated NRR-SNN.

The feedforward and feed-back propagations are shown in Fig. 2(b,d), where the $S_{i,t}$ and $S_{j,t}$ are the neuron-firing states, $V_{i,t}$ and $V_{j,t}$ are the membrane potentials, a is the neuronal plasticity with adaptive threshold. This model has two-time scales, containing T_{ff} for the feedforward propagation and T_{rec} for the recurrent propagation. The feedforward propagation connects a neuron's state at spatial scales, while the recurrent propagation links them at temporal scales. The neuronal plasticity has played important roles on the dynamic information propagation from the previous spike $S_{i,t-1}$ to the next-step membrane potential $V_{i,t}$ in feedforward procedure in Fig. 2(b), and also the gradient propagation from $V_{i,t}$ back to $S_{i,t-1}$ in feedback procedure in Fig. 2(d).

The vector field of the simplified dynamic LIF neuron is shown in Fig. 2(f,g), where Fig. 2(f) shows an attractor at $(1, 0)$, which means membrane potentials would move towards this stable point no matter where the initial point was, Fig. 2(g) shows a saddle point at $(-1, 0)$, which means that the point on the plane would move towards this point on one direction, but keep away from this point on another direction. The trend of these two directions would influence the other points on the plain.

An example of the relationship between neuronal plasticity with dynamic thresholds and spike trains is shown in Fig. 2(c,e), including the neuronal dynamics during learning TIDigits dataset. The blue bar represents the sum of the S^f and S^r, which the S^f means the neuron firing state on the feedforward propagation, the S^r means the neuron firing state on the recurrent propagation. Therefore, the $(S = S^f + S^r) \in \{0, 1, 2\}$. When the dynamic adaptive threshold $a(t) < Spike$, it would like to increase. When $a(t) > S$, it would have a negative attractor that could cause a decrease of $a(t)$. The dynamic adaptive thresholds of different neurons would contribute to the feature learning during training, which would be further introduced in the following experiments.

The encoding of the NRR-SNN contains two parts: the network-input part and the inner-network part. For the first part, in order to retain the original data information as much as possible, we only resize the spectrum data by a scalar variable and then feed it directly into the network. For the second part, we encode information as the spike by comparing the signal to a threshold V_{th}, where the signal above the threshold is set as 1, or else 0.

5 EXPERIMENTS

5.1 Dataset introduction

The TIDigits [35] and TIMIT [36] were selected out as the two benchmark datasets for their sequential characteristics. The TIDigits dataset contains 4,144 spoken digits from number zero to number nine. Each

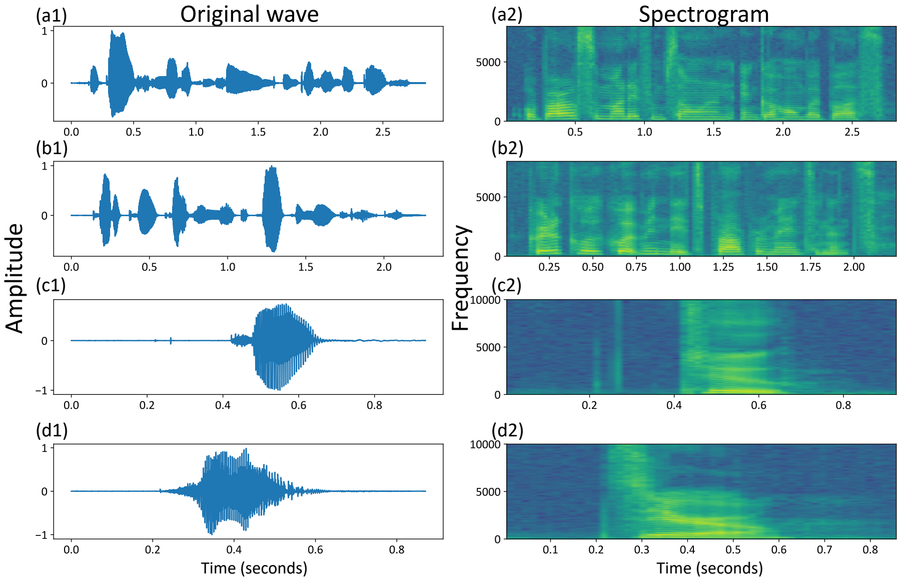


Figure 3. Speech waveforms and spectrograms of some samples, e.g., the temporal and spatial representations of spoken numbers, for example, “Or borrow some money from someone and go home by bus?” (a1, a2), “Critical equipment needs proper maintenance.” (b1, b2), “Two” (c1, c2) and “Zero” (d1, d2).

sample in it was sampled as 20K Hz during 1 second and processed after fast Fourier transform (FFT) with 39 frames and 39 bands. TIMIT contains 630 American speakers, with ten sentences for each person. Each sample was sampled as 16K Hz and processed after MFCC (short for Mel frequency cepstrum coefficient) with different frames and 39 bands. The frames were changed according to voice length, and the maximum was 780 frames.

For an easier description of the two benchmark datasets, Fig. 3 shows the speech waveform of some selected samples, including the spoken word waves from the TIMIT dataset in Fig. 3(a1, b1) and the spoken numbers from the TIDigits dataset in Fig. 3(c1, d1). The waveforms of speeches were in line with our intuition, where the amplitude of the voice waveform would increase for voice signals. However, it was not easy to extract all of the high-dimensional information from the original waves directly.

In the time domain, the speech waves were converted into the frequency domain, called the speech frequency spectrum, to get more valuable speech information at high dimensions. Fig. 3 (a2 to d2) show four spectrograms of four same examples from original waves from Fig. 3 (a1 to d1), respectively.

These two types of datasets covered most of the commonly used spoken words and numbers. From the temporal waves, we could find out that the spoken speeches in Fig. 3 (a1, b1) were more complicated than spoken numbers in Fig. 3 (c1, d1). Similar conclusions could also be found out from the spatial spectrograms, where more dynamics occurred in different voice bands of spoken speeches (with sentences) than spoken numbers (with simple words or numbers), with the MFCC parameters [37].

In our experiments, the accuracy of TIDigits is defined as the number of correct identifying samples dividing by the number of all samples. Different from it, the accuracy of TIMIT is defined as the number of correct identifying phonemes dividing by the number of all phonemes, for the consideration of the multiphonemes in the same sample.

Table 1. NRR-SNN parameters for the two benchmark temporal tasks, where “RFC” is short for recurrent feedforward connection, and “FC” is short for feedforward connection.

Tasks	Topology	Learning rate	g	V_{th}	τ_{ref}	T	V_{window}
TIMIT	RFC500-FC10	Step wise from 1e-3	$0.2nS$	$0.5mV$	$1ms$	10-100ms	$0.5mV$
TIDigits	RFC500-FC39	1e-4					

5.2 Parameters of the NRR-SNN

The key parameters of NRR-SNN for different tasks are shown in Table 1 from the scale of dynamic neurons to networks. In the table, g is conductance, V_{th} is the firing threshold of neurons, τ_{ref} is the refractory period, T is the time window for the simulation of dynamic neurons. Furthermore, the capacitance of membrane potential was $C = 1\mu F/cm^2$, the reset value of membrane potential was $V_{reset} = 0mV$. For the reward propagation network, the loss function was selected as the mean square error (MSE), the optimizer was Adam, and the batch size was set as 50.

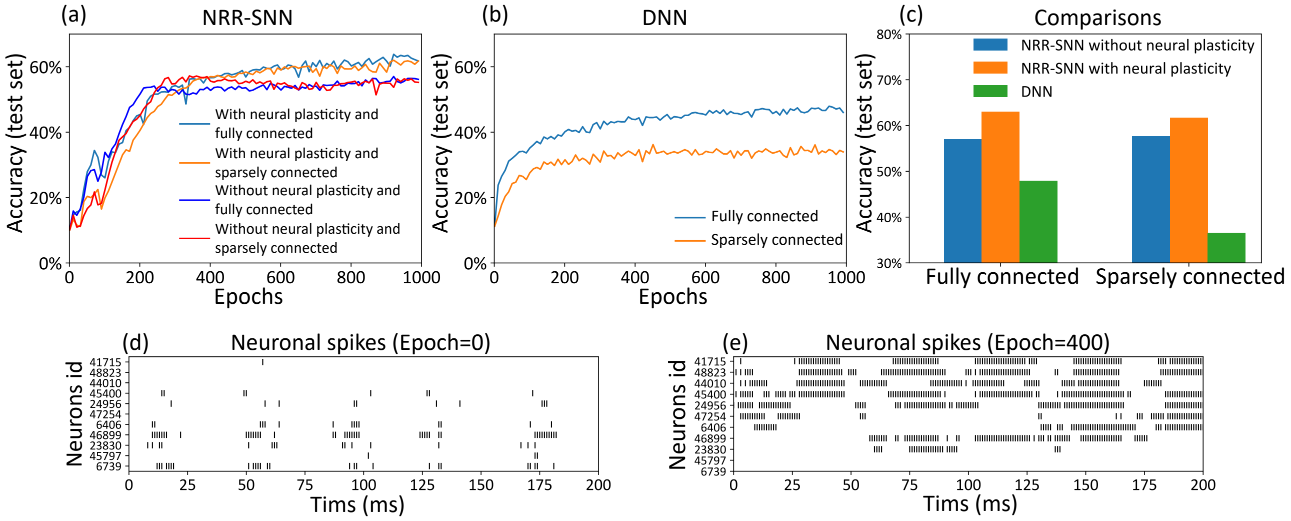


Figure 4. The neuronal plasticity and sparse connection for improving network learning. (a) The test accuracy of NRR-SNN with or without neural plasticity. (b) The performances of DNNs. (c) The performance comparisons of NRR-SNN and DNN, containing 100% connections and 50% sparse connections, with or without neural plasticity. (d,e) The neuronal spikes at different learning epochs, from epoch 1 to epoch 400.

5.3 Neuronal plasticity with adaptive threshold

We tested the NRR-SNN and DNN together, with or without neuronal plasticity (and 50% uniformly-distributed random connections), to better analyze the contribution of neural plasticity to the network learning. The results are shown in Fig. 4, where the neuronal plasticity has played a more important role in improving test performance than that in BP based recurrent SNN (BP-RSNN).

The network of NRR-SNN with 50% sparseness connections had similar performance compared with that with 100% connections in Fig. 4(a). In other words, the sparse connections of neurons reduced power consumption without compromising performance. Fig. 4(b) shows that the sparse connections could largely reduce the accuracy of speech recognition of the DNNs. Furthermore, Fig. 4(c) shows that network using neuronal plasticity could largely increase the test’s accuracy. Considering that it took energy to pass information between neurons, the network’s full connection would consume more computation resources during training. Therefore, the sparse connections of neurons would result in less consumption of computational cost.

Another hypothesis was that the sparse connections between neurons would decrease the network's complexity, but on the contrary, the additional adaptive threshold method of neurons would increase neurons' complexity. NRR-SNN was staying at a proper complexity for the efficient processing of information. This characteristic showed a good balance between neuronal complexity and network complexity.

During training, we also recorded the firing states of different dynamic neurons. Fig. 4(d,e) show the neuron firing states from the beginning of training (e.g., epoch=1) to the end of the learning (e.g., epoch=400). For each epoch, the duration of signal propagation is 200 ms. Some neurons randomly selected from the NRR-SNN network are shown in the figure with x-coordinate as simulation time (ms) and y-coordinate as neuron index (id). The spikes for most neurons were sparser, and the spike count or fire rate was smaller at the beginning of learning (epoch=1) compared to that at the end of learning (epoch=400). Neurons also reached stable learning states with obvious periodic firing. Besides, some neurons had more confidence for the judgment of firing (e.g., the neuron with id 41715) by responding stronger and quicker to the input stimulus, while some other neurons were tuned to be with a weaker response to the same input (e.g., the neuron with id 6739).

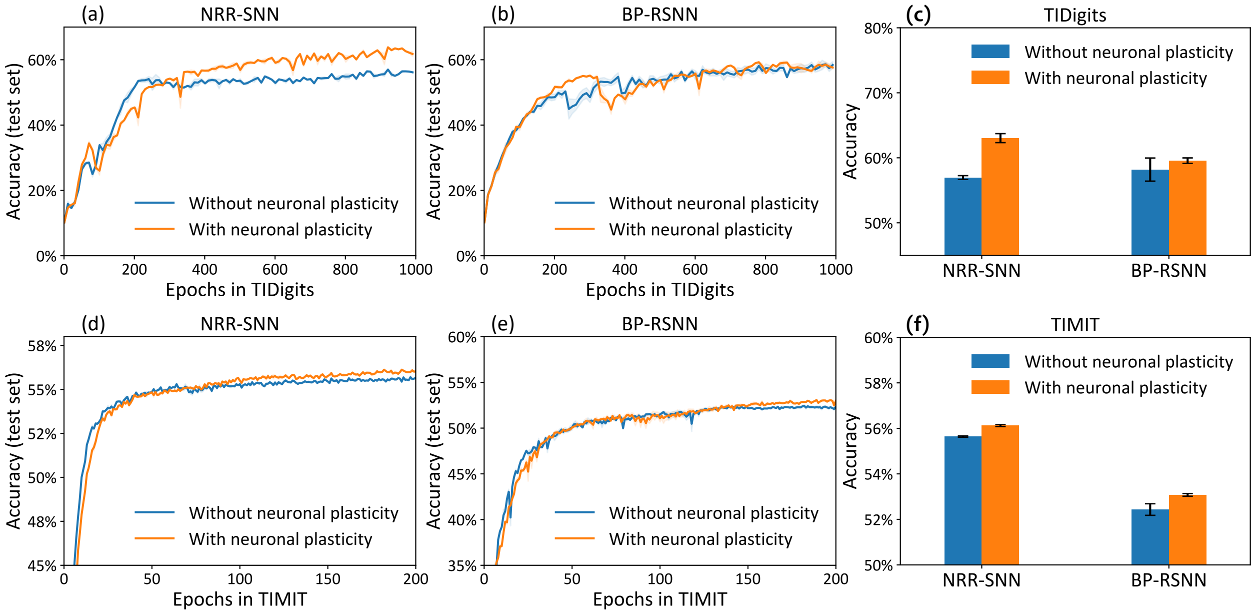
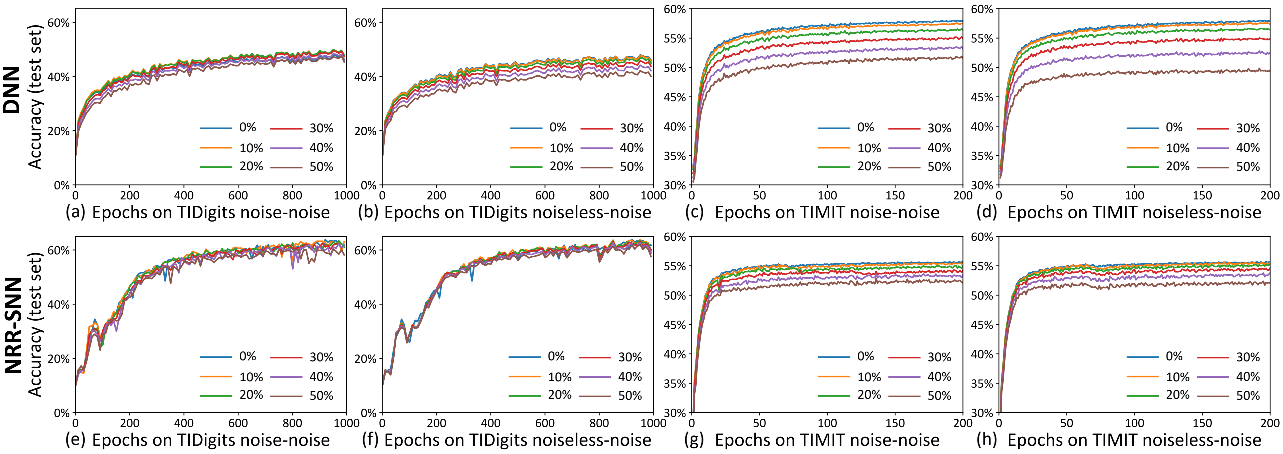


Figure 5. Accuracy on the test set for the models (i.e., NRR-SNN and BP-SNN) that with or without neuronal plasticity. (a-c), The performance on the two models on the TIDigits dataset. (d-f), The performance on the two models on the TIMIT dataset.

5.4 Reward propagation contributed to the neuronal dynamics

The differences between the NRR-SNN and BP-RSNN (recurrent SNN trained with pseudo-BP) were with or without reward propagations. The proposed NRR-SNNs were convergent during the training of TIDigits in Fig. 5(a) and TIMIT in Fig. 5(d). Besides, the models with adaptive thresholds showed higher test accuracies. The standard BP-RSNN models were also tested on these two benchmark datasets in Fig. 5(b) and Fig. 5(e), and showed a smaller difference between that with or without neuronal plasticity. This result shows that NRR-SNN architecture could cooperate better with neuronal plasticity to some extent. Fig. 5(c) showed the maximal test accuracies on the TIDigits dataset. The NRR-SNN and BP-RSNN reached 56.96% and 58.19%, respectively, without neuronal plasticity. After given neuron plasticity, the performance of NRR-SNN was increased higher than BP-RSNN, towards 63.03%, which was higher than BP-RSNN with 59.57%. The similar higher performance of NRR-SNN also reached on TIMIT dataset in Fig. 5(f), where NRR-SNN reached 56.12% accuracy and BP-SNN reached only 53.08% accuracy with neuronal plasticity. These experimental results showed that reward propagation contributed to the neuronal plasticity towards the higher SNNs' performance.



(a) Epochs on TIDigits noise-noise (b) Epochs on TIDigits noiseless-noise (c) Epochs on TIMIT noise-noise (d) Epochs on TIMIT noiseless-noise

(e) Epochs on TIDigits noise-noise (f) Epochs on TIDigits noiseless-noise (g) Epochs on TIMIT noise-noise (h) Epochs on TIMIT noiseless-noise

Figure 6. The comparisons of DNNs and NRR-SNNs for the robust computation on the samples containing noises. The "noise-noise" means that we added the noise both into the training dataset and test dataset. The noiseless-noise meant that we only added the noise to the test dataset without giving that to the training dataset.

5.5 Robust computation with sparse and recurrent connections

The NRR-SNN contained tunable recurrent connections inner the hidden layers that would contribute to the recognition performance, especially for the samples with noise (uniformly-distributed random noise).

Fig. 6(a-d) showed the test accuracy of traditional DNNs, where the performances decayed quickly with the increasing proportions of the noises. Unlike DNNs, the NRR-SNNs performed better towards the robust computation, where the performances nearly were not changed with different proportions of noises on the TIDigits dataset and only a little effected for that on the TIMIT dataset, as shown in Fig. 6 (e-h). Obviously, the recurrent connections in SNNs were the key to keeping a robust classification of sequential information.

Furthermore, we used another standard indicator named accuracy-noise ratio to describe the performances of the robust computation, represented as $RobustRatio = \frac{Acc_{noiseless,noise}}{Acc_{noise,noise}}$, where $Acc_{noiseless,noise}$ meaned "accuracy of noiseless data set for train and noise data set for test", and $Acc_{noise,noise}$ meaned "accuracy of noise data set for train and noise data set for test".

The performance of robust ratio is shown in Fig. 7, where even for the models trained with noise-free train data, accuracy has been maintained when the noise ratio of the test data reached 50%. While for DNNs, they were sensitive to noise, and their recognition accuracy was significantly reduced with the increase of noise proportions.

5.6 The comparison of NRR-SNN with other SNN models

In Table 2, we compared the performance of our NRR-SNNs (with bold marker) with other SNNs. An ablation study was further given, especially on the adaptive threshold, sparse loop, reward propagation, and shallow or deep architectures on SNNs.

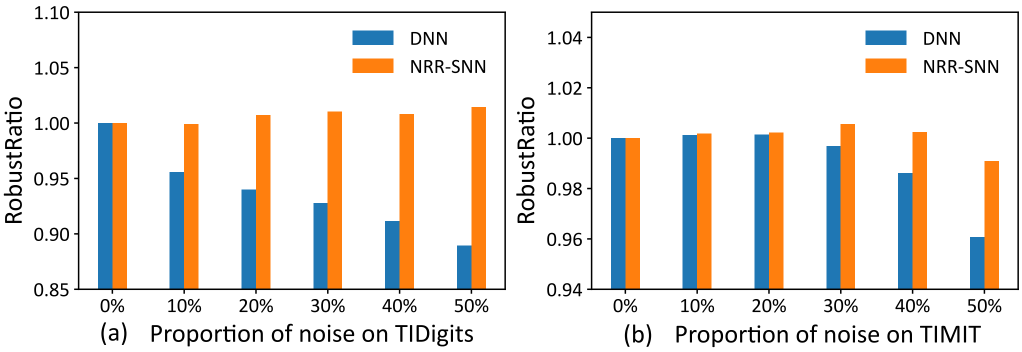


Figure 7. The comparison of robust ratios between DNNs and NRR-SNNs. The robust ratios of NRR-SNN decrease slowly compared to that of DNN on both sequential TIDigits (a) and TIMIT datasets (b).

It was obvious that our NRR-SNN reached the best performance on the TIDigits dataset. The pure feedforward SNN with three layers reached 36.25% tuned with Pseudo-BP. Then SNN with an additional adaptive threshold reached 66.05% accuracy, while that with additional sparse loops reached 60.86% accuracy. We also tested NRR-SNN with different configurations. The NRR-SNN with shallow architecture (three layers, with only one feedforward hidden layer with recurrent loops) got 63.03% accuracy, while a deeper one (five layers, containing input layer, convolution layer, feedforward layer with recurrent loops, feedforward layer, and output layer) got 97.40% accuracy, higher than some other SNNs, such as that based on self-organizing map (SOM) [38] or liquid state machine (LSM) [39].

For the TIMIT dataset, the shallow feedforward SNN reached 52.97% by Pseudo-BP. Then accuracies increased to 53.42% after adding the adaptive threshold and to 55.67% after adding sparse loops. It was reported that the accuracy of SNNs reached 73.06% for that with recurrent connections (RSNN) [40, 41], and 65.40% for that with LSTM-based (long short-term memory) spiking neural networks (LSNN) [42].

We founded out that our NRR-SNN reached 56.12%, which was lower than the previous RSNN and LSNN. However, we also noticed that the accuracy of NRR-SNN was still higher than that after replacing RP with BP (only 53.08%). We thought this was already a good indicator to show the performance of NRR-SNN, since the lower accuracy compared to other SOTA methods was more from the different sample lengths of TIMIT, where all of the samples in the same patch were normalized as the same length by padding zero to short samples. However, these problems had already out of the scope of this paper.

6 CONCLUSION

Most of the research related to SNNs focuses on synaptic plasticity, including the STDP, STP, and other biologically-inspired plasticity rules. However, inner neurons' plasticity also plays important roles in the neural network dynamics but is seldom introduced. This paper's important motivation is to improve the performance of SNNs towards higher classification accuracy and more robust computation for processing temporal information with noises. A special Neuronal-plasticity and Reward-propagation improved Recurrent SNN (NRR-SNN) have been proposed for reaching these goals:

- The historically-related adaptive threshold with two channels is highlighted as important neuronal plasticity for increasing the neuronal dynamics.
- Instead of errors, global labels are used as a reward for the paralleling gradient propagation.
- Dynamic neurons are then connected in a recurrent loop with proper sparseness for the robust computation.

Table 2. The performance comparison of our NRR-SNN model with other spiking models

Task	Architecture	Training type	Learning rule	Performance
TIDigits	SOM-SNN [38]	Rate+Spike	SOM+BP	97.40%
	Liquid state machine [39]	Spike-based	BP	92.30%
	Pure feedforward SNN	**Spike-based**	**Pseudo-BP**	**36.25%**
	Feedforward with adaptive threshold	**Spike-based**	**Pseudo-BP**	**66.05%**
	Feedforward with sparse loop	**Spike-based**	**Pseudo-BP**	**60.86%**
	Shallow NRR-SNN (three layers)	**Spike-based**	**RP**	**63.03%**
	Deep NRR-SNN (five layers)	**Spike-based**	**RP**	**98.34%**
TIMIT	Recurrent-SNN [40, 41]	Spike-based	BPTT	73.60%
	LSNN [42]	Spike-based	E-prop	65.40%
	Pure feedforward SNN	**Spike-based**	**Pseudo-BP**	**52.97%**
	Feedforward with adaptive threshold	**Spike-based**	**Pseudo-BP**	**53.42%**
	Feedforward with sparse loop	**Spike-based**	**Pseudo-BP**	**55.67%**
	NRR-SNN	**Spike-based**	**RP**	**56.12%**
	BP-SNN	**Spike-based**	**RP**	**53.08%**

The experimental results have shown the proposed NRR-SNN's efficiency compared to the standard DNNs and other SNNs.

CONFLICT OF INTEREST STATEMENT

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

T.Z and B.X gave the idea; T.Z., S.J., X.C. and H.L. made the mathematical analyses and experiments. They wrote the paper together.

FUNDING

This study is supported by the National Key R&D Program of China (Grant No. 2020AAA0104305), the National Natural Science Foundation of China (Grant No. 61806195), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB32070100, XDA27010404), and the Beijing Brain Science Project (Z181100001518006).

DATA AVAILABILITY STATEMENT

The TIMIT dataset can be downloaded from https://catalog.ldc.upenn.edu/LDC93S1. The TIDigits dataset can be downloaded from https://catalog.ldc.upenn.edu/LDC93S10.

CODE AVAILABILITY STATEMENT

The source code can be downloaded from Github after the acceptance of the paper.

REFERENCES

[1] Ciregan D, Meier U, Schmidhuber J. Multi-column deep neural networks for image classification. *2012 IEEE conference on computer vision and pattern recognition* (IEEE) (2012), 3642–3649.

[2] Nguyen A, Yosinski J, Clune J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *computer vision and pattern recognition* (2015), 427–436.

[3] He K, Gkioxari G, Dollár P, Girshick R. Mask r-cnn. *Proceedings of the IEEE international conference on computer vision* (2017), 2961–2969.

[4] Kim J, Zeng H, Ghadiyaram D, Lee S, Zhang L, Bovik AC. Deep convolutional neural models for picture-quality prediction: Challenges and solutions to data-driven image quality assessment. *IEEE Signal processing magazine* **34** (2017) 130–141.

[5] Dan Y, Poo Mm. Spike timing-dependent plasticity of neural circuits. *Neuron* **44** (2004) 23–30.

[6] Zucker RS. Short-term synaptic plasticity. *Annual Review of Neuroscience* **12** (1989) 13–31.

[7] Zhang T, Zeng Y, Shi M, Zhao D. A plasticity-centric approach to train the non-differential spiking neural networks. *Thirty-Second AAAI Conference on Artificial Intelligence* (2018), 620–628.

[8] Zhang T, Zeng Y, Zhao D, Xu B. Brain-inspired balanced tuning for spiking neural networks. *International Joint Conference on Artificial Intelligence* (2018), 1653–1659.

[9] Zeng Y, Zhang T, Xu B. Improving multi-layer spiking neural networks by incorporating brain-inspired rules. *Science China Information Sciences* **60** (2017) 052201.

[10] Ito M. Long-term depression. *Annual Review of Neuroscience* **12** (1989) 85–102.

[11] Teyler TJ, DiScenna P. Long-term potentiation. *Annual Review of Neuroscience* **10** (1987) 131–161.

[12] Bengio Y, Mesnard T, Fischer A, Zhang S, Wu Y. Stdp as presynaptic activity times rate of change of postsynaptic activity approximates back-propagation. *Neural Computation* **10** (2017).

[13] [Dataset] Zhang T, Jia S, Cheng X, Xu B. Tuning convolutional spiking neural network with biologically-plausible reward propagation (2020).

[14] Bi G, Poo M. Synaptic modification by correlated activity: Hebb's postulate revisited. *Annu Rev Neurosci* **24** (2001) 139–66. doi:10.1146/annurev.neuro.24.1.139.

[15] Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-inspired artificial intelligence. *Neuron* **95** (2017) 245–258.

[16] Zhang M, Wang J, Zhang Z, Belatreche A, Wu J, Chua Y, et al. Spike-timing-dependent back propagation in deep spiking neural networks. *arXiv preprint arXiv:2003.11837* (2020).

[17] Hodgkin AL, Huxley AF. Action potentials recorded from inside a nerve fibre. *Nature* **144** (1939) 710–711.

[18] Hodgkin AL, Huxley AF. Resting and action potentials in single nerve fibres. *The Journal of physiology* **104** (1945) 176.

[19] Noble D. A modification of the hodgkin—huxley equations applicable to purkinje fibre action and pacemaker potentials. *The Journal of physiology* **160** (1962) 317.

[20] Gerstner W, Kistler WM, Naud R, Paninski L. *Neuronal dynamics: From single neurons to networks and models of cognition* (Cambridge University Press) (2014).

[21] Gerstner W, Ritz R, Van Hemmen JL. Why spikes? hebbian learning and retrieval of time-resolved excitation patterns. *Biological cybernetics* **69** (1993) 503–515.

[22] Gerstner W. Spike-response model. *Scholarpedia* **3** (2008) 1343.

[23] Izhikevich EM. Simple model of spiking neurons. *IEEE Transactions on neural networks* **14** (2003) 1569–1572.

[24] Yu Q, Li H, Tan KC. Spike timing or rate? neurons learn to make decisions for both through threshold-driven plasticity. *IEEE transactions on cybernetics* **49** (2018) 2178–2189.

[25] Zenke F, Ganguli S. Superspike: Supervised learning in multilayer spiking neural networks. *Neural computation* **30** (2018) 1514–1541.

[26] Kheradpisheh SR, Ganjtabesh M, Thorpe SJ, Masquelier e Timothé. Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks* **99** (2018) 56–67.

[27] Frenkel C, Lefebvre M, Bol D. Learning without feedback: Direct random target projection as a feedback-alignment algorithm with layerwise feedforward training. *arXiv: Machine Learning* (2019).

[28] Zhao D, Zeng Y, Zhang T, Shi M, Zhao F. Glsnn: A multi-layer spiking neural network based on global feedback alignment and local stdp plasticity. *Frontiers in Computational Neuroscience* **14** (2020).

[29] Lee JH, Delbruck T, Pfeiffer M. Training deep spiking neural networks using backpropagation. *Front Neurosci* **10** (2016) 508. doi:10.3389/fnins.2016.00508.

[30] Wu J, Xu C, Zhou D, Li H, Tan KC. Progressive tandem learning for pattern recognition with deep spiking neural networks. *arXiv preprint arXiv:2007.01204* (2020).

[31] Diehl PU, Cook M. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in computational neuroscience* **9** (2015) 99.

[32] Pan Z, Wu J, Zhang M, Li H, Chua Y. Neural population coding for effective temporal classification. *2019 International Joint Conference on Neural Networks (IJCNN)* (IEEE) (2019), 1–8.

[33] Dong M, Huang X, Xu B. Unsupervised speech recognition through spike-timing-dependent plasticity in a convolutional spiking neural network. *PloS one* **13** (2018) e0204596.

[34] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. *nature* **323** (1986) 533–536.

[35] Leonard RG, Doddington G. Tidigits ldc93s10. *Web Download. Philadelphia: Linguistic Data Consortium* (1993).

[36] Garofolo JS. Timit acoustic phonetic continuous speech corpus. *Linguistic Data Consortium, 1993* (1993).

[37] Maesa A, Garzia F, Scarpiniti M, Cusani R, et al. Text independent automatic speaker recognition system using mel-frequency cepstrum coefficient and gaussian mixture models. *Journal of Information Security* **3** (2012) 335.

[38] Wu J, Chua Y, Zhang M, Li H, Tan KC. A spiking neural network framework for robust sound classification. *Frontiers in neuroscience* **12** (2018) 836.

[39] Zhang Y, Li P, Jin Y, Choe Y. A digital liquid state machine with biologically inspired learning and its application to speech recognition. *IEEE transactions on neural networks and learning systems* **26** (2015) 2635–2649.

[40] Wu J, Yılmaz E, Zhang M, Li H, Tan KC. Deep spiking neural networks for large vocabulary automatic speech recognition. *Frontiers in neuroscience* **14** (2020) 199. Publisher: Frontiers.

[41] Bellec G, Scherr F, Subramoney A, Hajek E, Salaj D, Legenstein R, et al. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature communications* **11** (2020) 1–15.

[42] Bellec G, Scherr F, Hajek E, Salaj D, Subramoney A, Legenstein R, et al. Eligibility traces provide a data-inspired alternative to backpropagation through time (2019).