LISNN: Improving Spiking Neural Networks with Lateral Interactions for Robust Object Recognition

Xiang Cheng[1,2,3] , **Yunzhe Hao**[1,2,3*] , **Jiaming Xu**[1,2†] and **Bo Xu**[1,2,3,4†]
[1]Institute of Automation, Chinese Academy of Sciences (CASIA). Beijing, China
[2]Research Center for Brain-inspired Intelligence, CASIA
[3]University of Chinese Academy of Sciences
[4]Center for Excellence in Brain Science and Intelligence Technology, CAS. China
{chengxiang2018, haoyunzhe2017, jiaming.xu, xubo}@ia.ac.cn

Abstract

Spiking Neural Network (SNN) is considered more biologically plausible and energy-efficient on emerging neuromorphic hardware. Recently backpropagation algorithm has been utilized for training SNN, which allows SNN to go deeper and achieve higher performance. However, most existing SNN models for object recognition are mainly convolutional structures or fully-connected structures, which only have inter-layer connections, but no intra-layer connections. Inspired by Lateral Interactions in neuroscience, we propose a high-performance and noise-robust Spiking Neural Network (dubbed LISNN). Based on the convolutional SNN, we model the lateral interactions between spatially adjacent neurons and integrate it into the spiking neuron membrane potential formula, then build a multi-layer SNN on a popular deep learning framework, i. e., PyTorch. We utilize the pseudo-derivative method to solve the non-differentiable problem when applying backpropagation to train LISNN and test LISNN on multiple standard datasets. Experimental results demonstrate that the proposed model can achieve competitive or better performance compared to current state-of-the-art spiking neural networks on MNIST, Fashion-MNIST, and N-MNIST datasets. Besides, thanks to lateral interactions, our model processes stronger noise-robustness than other SNN. Our work brings a biologically plausible mechanism into SNN, hoping that it can help us understand the visual information processing in the brain.

1 Introduction

Now Artificial Neural Network (ANN) has been widely utilized on variant tasks, such as image classification, speech recognition, and natural language processing [LeCun *et al.*, 2015]. However, ANN only imitates brain structures in several ways abstractly and roughly, e. g., neurons use synapses to integrate input information and interconnect with each other, networks consist of layered sub-structure [Hinton and Salakhutdinov, 2006]. The brain has more information processing mechanisms. Spiking Neural Network (SNN) utilizes discrete action potentials (or spikes) to represent and process information, which is similar to how the brain works. The exploration of SNN can help us understand the working mechanism of the mind and the intelligence [Ghosh-Dastidar and Adeli, 2009]. Besides, in most instances, the neurons in SNN do not excite until they receive input spikes. Thus SNN is potentially energy-efficient and may have a wide range of application scenarios with emerging neuromorphic hardware [Pfeiffer and Pfeil, 2018].

However, due to the complex neural dynamics and non-differential nature of SNN, it is challenging to construct an efficient SNN model. Researchers have made many efforts in model design. In terms of learning rule, Diehl and Cook [2015] utilized a biologically plausible learning rule, i. e., Spike-Timing Dependent Plasticity (STDP) [Bi and Poo, 1998], to train a two-layer SNN with lateral inhibitions in an unsupervised learning style. Hao *et al.* [2020] trained a three-layer SNN using symmetric STDP rule, which was a spiking version of the Hebbian rule summarised as “Cells that fire together wire together” [Shatz, 1992; Hebb, 1962]. Some researchers made changes to STDP and proposed STDP variants learning rule [Hu *et al.*, 2017]. The Leaky Integrate-and-Fire (LIF) model was converted into an iterative unit, and a pseudo-derivative method was proposed to overcome the non-differentiable problem. Then deeper SNN could be trained using the backpropagation (BP) algorithm successfully [Woźniak *et al.*, 2018; Wu *et al.*, 2018]. There are two categories of the encoding scheme in SNN, rate coding and temporal coding. The temporal coding scheme encodes information with the relative timing of individual spikes [Comsa *et al.*, 2019; Zhang *et al.*, 2019]. More models use rate coding, which uses the rate of spike train in a long time window to encode information [Jin *et al.*, 2018; Wu *et al.*, 2019]. As for the network structure design, Xu *et al.* [2018] proposed CSNN, which uses Convolutional Neural Networks (CNN) as the front module to improve the feature extraction ability of SNN. Wu *et al.* [2019] utilized spiking neurons that have a convolutional receptive field to build the network. Hu *et al.* [2018] imitated the residual net-

[*] The first two authors contributed equally.

[†] Corresponding author

work in deep learning and proposed a spiking residual network. Bellec *et al.* [2018] got inspired by adapting neurons of different time constants in the brain, proposed a framework named LSNN which consists of two populations with different neuron type, i. e., non-adapting neuron or adapting neuron. Zhang and Li [2019] and Woźniak *et al.* [2018] trained SNN with recurrent connection layer.

With proposed BP-like learning algorithms, SNN can develop to larger scales and achieve better performance [Wu *et al.*, 2019; Woźniak *et al.*, 2018]. Existing SNN for object recognition has the convolutional structure in general, which can extract features of objects effectively [Xu *et al.*, 2018; Wu *et al.*, 2019; Woźniak *et al.*, 2018]. Most of them have inter-layer connections, such as convolution kernel and feedforward connections. However, no model focuses on the intra-layer connections, which is an essential mechanism of a biological neural system for object recognition. Neuroscientists discover that lateral interactions in the retina neurons can enhance the visual object edge [Ratliff *et al.*, 1974]. In the computational neuroscience and cognitive science, Dynamic Neuronal Field (DNF) is a popular recurrent neural network with attractor dynamics, where a neuron excites neurons nearby and inhibits further neurons [Schöner and Spencer, 2016; Sandamirskaya, 2014]. This mechanism can enhance the significant input regions, suppress noisy areas, and retain the information in the neuron population [Evanusa *et al.*, 2019]. Inspired by lateral interactions, we supposed that local lateral interaction connection can process spike train information efficiently. Thus we proposed a new lateral interaction based SNN named LISNN. We test the LISNN model in visual object recognition tasks, i. e., MNIST, N-MNIST, and Fashion-MNIST datasets [LeCun *et al.*, 1998; Orchard *et al.*, 2015; Xiao *et al.*, 2017]. The experiment results show that lateral interactions help our model achieve high performance in MNIST and N-MNIST datasets, and achieve state-of-the-art in Fashion-MNIST dataset. Besides, LISNN has stronger noise-robustness than others. Our main contributions are three-fold:

1. To our best knowledge, this is the first attempt to apply a lateral interaction mechanism in biology to convolutional SNN. Thus lateral interactions play such an essential role in the nervous system and human intelligence, we believe that this study of combination between the computational model and lateral interactions may be helpful to further exploration of the secret of cognition.
2. We improve the anti-noise capability of SNN for object recognition. The performance of deep learning algorithms depends on the completeness of the training dataset. As a result, the inevitable noise produced during data collection, transmission, and processing can significantly impact the inference performance of networks. Our model shows a stronger noise-robustness than other SNN.
3. Our model achieves the best performance among existing SNN on the Fashion-MNIST dataset, which is recognized as a harder version of the widely-used MNIST.

Our research indicates that lateral interactions have a positive effect on improving the performance and robustness of the network. We hope our research can help neuroscientists understand the advantages of lateral interactions from the perspective of computational models, and help us explore the working mechanism of lateral interactions in the brain.

We organize the rest of this paper as follows. Related work is briefly discussed in Section 2. The proposed LISNN model is introduced in detail in Section 3. Experimental evaluations are presented in Section 4. Finally, Section 5 concludes this paper.

2 Related Work

Our goal is to combine the lateral interaction mechanism and the convolution structure. Therefore, we mainly focus on related work of convolution structure and lateral interaction mechanism on the SNN in recent years.

2.1 Convolutional Structure SNN

Xu *et al.* [2018] proposed an augmented spiking based framework named CSNN, which combines the feature extraction capability of CNN and the biological rationality of SNN. Although CSNN achieved comparable performances to other cognitive models with significantly fewer neurons and training samples, in the strict sense, it is not the SNN of convolutional structure. Wu *et al.* [2019] proposed a directly-training algorithm to train deeper and more complex SNN with an explicitly iterative version of LIF model in the PyTorch[1], a popular machine learning framework. Their model consists of different modules, such as convolutional structure, mean pooling, and multi-layer fully-connected structure. Kheradpisheh *et al.* [2018] built a three-layer convolutional spiking neural network for object recognition. Their model utilized sparse spikes to encode and process information, which gained remarkable processing speed and low energy consumption. But their model is trained by unsupervised learning rule, STDP. Thus it needs extra classifier to complete classification task.

2.2 Lateral Interaction

Lateral interaction connections have been used in many computing models and have achieved excellent results. Diehl and Cook [2015] use fixed lateral inhibition in the hidden layer to form a winner-take-all mechanism, helping the network to build different receptive fields and improve recognition performance. The lateral interaction mechanism helps the network to suppress interference, even if the significance of the noise has changed drastically, and it can even reserve the object position in the field of view when the object stopped. [Evanusa *et al.*, 2019].

Recurrent Spiking Neural Network (RSNN) is a fundamental class of SNN and especially suitable for processing time-series data. Liquid State Machine (LSM), a special RSNN, contains an input module, a recurrent reservoir layer, and a readout module [Maass *et al.*, 2002]. Generally, the internal connection of the reservoir is fixed or modified by local unsupervised learning rules, e. g., STDP, with only a readout module trained by the supervised signal. Lacking global

[1] https://pytorch.org/

supervision information, the performance of LSM is limited. Recently backpropagation is used to train SNN. Thus some models with trained recurrent connections were proposed. Woźniak *et al.* [2018] introduced an alternative perspective on spiking neurons as a particular ANN block called Spiking Neural Unit (SNU). The recurrent SNU (rSNU), i. e., SNU variant model with a recurrent connection like Long Shot-Time Memory (LSTM) and Gated Recurrent Units (GRU), surpassed the state-of-the-art ANN without dropout on the language modeling task using Penn Treebank (PTB) dataset. Zhang and Li [2019] proposed Spike-Train level RSNN backpropagation (ST-RSBP) algorithm to train deep RSNN with multiple feedforward and recurrent layers. However, its network structure is just simply fully-connected and not optimized explicitly for image tasks.

Although lateral interaction is similar to the recurrent structure in ANN, i. e., Recurrent Neural Networks (RNN) and RSNN, we suppose that recurrent structure in ANN is an abstraction and a particular case of lateral interaction in biology. Besides, the scope of lateral interaction in biology is generally local, but the scope of the recurrent structure of existing SNN is the entire layer. Thus SNN with local lateral interaction is more biologically plausible.

3 Model

In this section, we introduce the implemented details of our model, including (1) how to encode visual features into spiking train; (2) how to achieve lateral interactions in SNN structure; (3) how to train our model by BP algorithm.

3.1 Network Structure

There are three styles of layers in our network, convolution layer, pooling layer, and fully-connected layer. Like most CNN, convolution layers are applied in the forepart of our network, which work as the receptor and afferent nerve where external information is collected and extracted preliminarily. Meanwhile, fully-connected layers are used in the posterior of the network, which function as cortex where advanced cognitive takes place. We suppose lateral interactions make more effort in the feature extraction process. Thus we apply lateral interactions in convolution layers, and no lateral interactions are applied to fully-connected layers and pooling layers during the following experiments. Different from conventional CNN, we use mean-pooling to replace max-pooling, since max-pooling leads to tremendous information loss in spiking trains. A detailed structure of our model is shown in Figure1.

3.2 Input Encoding

To encode inputs into spiking trains, which is suitable for the inherent dynamic of SNN, we use several methods to preprocess different kinds of datasets. In terms of frame-based data, firstly, we normalize the input and generate random numbers from 0 to 1 evenly distributed, which has the same size as input at every time step. Then we compare every generated random number with its corresponding original data. If the generated random number is greater than its original data, it is set to 0. Otherwise, it is set to 1. By this means, we transform a floating-point input into a spiking train, which also ensures positions that have greater original data have higher firing frequency. In terms of event-based data, we set a time window that accumulates information in a period. For each time window, we record the position where events are triggered. If multiple triggered events happen in one position, they are not counted repeatedly and considered to be a single spike in a time window. By dynamically choosing the length of a time window, event-based data can be transformed into available spiking train.

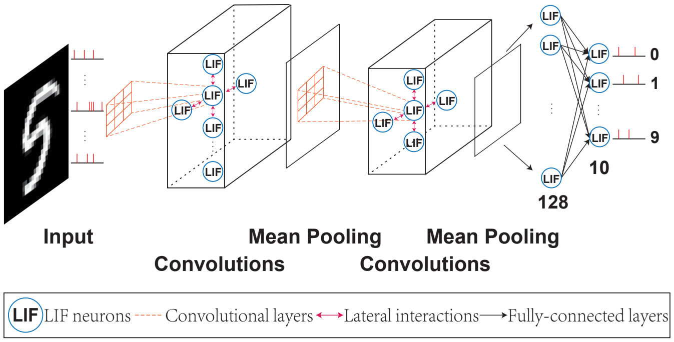

Figure 1: The architecture of LISNN.

3.3 Iterable SNN

The dynamic of SNN is based on studies of membrane potential. A most commonly used mathematic model to simulate the physiological process is the Leaky Integrate-and-Fire (LIF) model. LIF model can be described by:

$$\tau \frac{dV(t)}{dt} = -V(t) + RI(t). \tag{1}$$

Here τ is the time constant which equals the product of the resistance R and the capacitance C of the neuron soma. $I(t)$ is the input current collecting from synapses and is integrated into $V(t)$, which is the membrane potential. If $V(t)$ is greater than a certain threshold V_{th}, neuron emits an output spike. Then $V(t)$ resets to V_{rest} and starts to accumulate again.

However, the LIF model has a complex internal structure that is incompatible to nowadays deep learning framework. By discretizing and transforming the ordinary differential equation of the LIF model, SNN can be applied in deep learning framework:

$$V(t) = (1 - \frac{1}{\tau})V(t-1) + \frac{1}{C}I(t). \tag{2}$$

3.4 Lateral Interactions

Lateral interaction mechanism was first brought up by Mach in the 1860s to explain an optical illusion called Mach bands. When edges of the slightly differing shades of gray contact one another, it triggers edge-detection in the human visual system and exaggerates the contrast. Beyond Mach bands effect, Ratliff *et al.* [1974] found biological evidence of lateral interactions during his research in the receptor units of the compound eye of *limulus* and came up with a mathematic model to quantitatively describe lateral interactions in the 1950s. Nowadays, lateral interactions are confirmed not only in the visual system but also in other sensory systems like the auditory system. Due to its ability to sharpen edges and highlight the contour, lateral interactions have been applied in many areas. Weights of lateral interactions in many models are preset and fixed according to the mathematical description of activity patterns in brain cortices, such as Mexican hat function, which defines weights of lateral interactions as a function of the distance between neurons. One of the successful applications of lateral interactions, DNF [Schöner and Spencer, 2016], raises an equation to simulate the physiological process:

$$\tau\dot{u}(x,t) = -\,u(x,t) + h + \int f(u(x',t))\omega(x,x')\,\mathrm{d}x' + S(x,t). \quad (3)$$

Here, $u(x,t)$ is the activation function of DNF at time t, which is defined over a parameter space x describing the state of the system. h is a negative item that decreases $u(x,\ t)$ to resting level when external input $S(x,\ t)$ is absent. Finally, $f(u)$ is a non-linear function shaping the output of the DNF.

3.5 Applications of Lateral Interactions in SNN

Our model follows the basic structure of many other SNN, which can be trained with backpropagation. We use discretized time steps and divide the forward propagation inside spiking neurons into two phases: update of membrane potential and firing. Membrane potential and spiking state are decided by:

$$V(t) = lV(t-1)(1-s(t-1)) + b + I(t) + (s_{neigh}(t-1) * \omega). \quad (4)$$

$$s(t) = \begin{cases} 1 & V(t) > V_{th}; \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Here $V(t)$ is decided by four factors. The first factor is membrane potential at the previous time step, which is multiplied by a fixed leaky item l and a reset item $1 - s(t-1)$ (we set the V_{rest} as 0). The second factor b is a trainable bias vector. $I(t)$ is the input from synapses at the same time step. And the last factor is lateral interactions taking place between neighbor neurons. s_{neigh} is the spiking state of the neighbor. ω is a interaction kernel that are trainable in our model.

Accordingly, SNN without lateral interactions updates membrane potential following:

$$V(t) = lV(t-1)(1-s(t-1)) + b + I(t). \quad (6)$$

Except this, the during following experiments, SNN has exactly the same structure and hyperparameter as its corresponding LISNN in order to control variables.

We test LISNN on image datasets for their convenience to correspond the neurons to pixels in the input image and define the neighbor of a neuron. In this perspective, we take different channels as items describing various features of a pixel and merge them when updating membrane potential.

3.6 Backpropagation for SNN

At last, there is still an obstacle for applying backpropagation in SNN because Eq. 5 is indifferentiable. This problem can be solved by appointing a particular pseudo derivative. Here we use a rectangular function:

$$dx = \begin{cases} 1 & 0 < x < 1; \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

With all these efforts, now it is feasible to use backpropagation to train SNN.

4 Experiment

4.1 Experimental Settings

The experiments are implemented on NVIDIA TITAN Xp. The code is written under the PyTorch framework, thus weights are randomly initialized by the default method of PyTorch. Besides, we use Adam as an optimizer and decay learning rate by epochs. Table 1 lists the hyperparameter used in our experiments, including batch size, learning rate, the structure of the network, and so on. We evaluate the performance of LISNN on three classification tasks: MNIST, Fashion-MNIST, and N-MNIST, and compare it with the current state-of-the-art performances of SNN. We also compare the robustness of LISNN to SNN without lateral interactions. We record accuracy of networks on datasets with added noise of different gradients, which are trained on MNIST and Fashion-MNIST with or without artificially added noise. Every accuracy that we present below is the average of five replications with different random seeds.

Hyperparameters	MNIST	Fashion-MNIST	N-MNIST
total time step	20	20	20
learning rate	0.001	0.001	0.001
batch size	100	100	50
training epoch	100	100	100
leaky item	0.2	0.2	0.2
potential threshold	0.5	0.5	0.5

Table 1: Hyperparameters of LISNN.

4.2 Static Image Dataset

MNIST[2] and Fashion-MNIST[3] are static image datasets, both contain a training set of 60000 images and a testing set of 10000 images. Each image in these two datasets is a 28×28 normalized grey-scale picture. For MNIST, its images belong to ten categories of handwritten numbers from 0 to 9. Fashion-MNIST is designed to be a more difficult dataset than MNIST, and its images belong to ten categories of clothing.

Firstly, we test the effect of interaction kernel size on the MNIST dataset and find that when the interaction kernel size is 5×5, the model achieves the best performance. Detailed results are listed in Table 2. As a result, 5×5 is used as an interaction kernel size in the following experiments. Then

[2] http://yann.lecun.com/exdb/mnist/

[3] https://github.com/zalandoresearch/fashion-mnist

Interaction Kernel Size	Hidden Layers	ACC (%)
3×3	32C3-P2-32C3-P2-128[a]	99.46
5×5	32C3-P2-32C3-P2-128	**99.50**
7×7	32C3-P2-32C3-P2-128	99.49

[a] 32C3 represents convolution layer with 32 channels of the 3×3 filters. P2 indicates pooling layer with 2×2 filters.

Table 2: Effects of different interaction kernel sizes on MNIST.

Algorithms	Hidden Layers	ACC (%)
Spiking CNN [Lee *et al.*, 2016]	20C5-P2-50C5-P2-200-10	99.31
SLAYER [Shrestha and Orchard, 2018]	12C5-P2-64C5-P2	99.41
STBP [Wu *et al.*, 2018]	12C5-P2-40C5-P2-300	99.42
HM-2BP [Jin *et al.*, 2018]	15C5-P2-40C5-P2-300	99.49
ST-RSBP [Zhang and Li, 2019]	15C5-P2-40C5-P2-300	**99.62**
SNN (without LI)	32C3-P2-32C3-P2-128	99.48
LISNN	32C3-P2-32C3-P2-128	99.50

Table 3: Performance of different algorithms on MNIST.

we compare the performance of our model with several other algorithms working on these two tasks. Detailed results are listed in Table 3 and Table 4. It is shown that LISNN produces excellent results on MNIST, which is close to state-of-the-art SNN performance and gains the highest accuracy on Fashion-MNIST among others.

4.3 Dynamic Image Dataset

N-MNIST is a spiking version of the frame-based MNIST dataset. It contains a training set of 60000 sets of events and a testing set of 10000 sets of events derived from the original MNIST dataset. The N-MNIST dataset is captured by moving the sensor on a motorized unit and having it viewing MNIST images on a monitor. N-MNIST images have two channels that record brighter pixels and darker pixels separately. In this case, we partition the triggered pixels into 20-time steps according to its timestamp.

We compare the performance of our model with several other algorithms working on N-MNIST. The results are listed in Table 5. Besides, it is shown that SNN-based models gain competitive accuracy on N-MNIST. One of the possible reasons is that Dynamic Vision Sensors (DVS) create inevitable noise when capturing the object. This increases the difficulty for networks to extract the significative features from input data. However, with lateral interactions, noise can be excluded from objects. Thus it guarantees better convergence and generalization capabilities.

Algorithms	Hidden Layers	ACC (%)
LRA-E [Ororbia and Mali, 2019]	5×256	87.69
HM-2BP [Jin *et al.*, 2018]	400-400	88.99
DL-BP [Agarap, 2018]	3×512	89.06
ST-RSBP [Zhang and Li, 2019]	400-R400	90.13
SNN (without LI)	32C3-P2-32C3-P2-128	91.60
LISNN	32C3-P2-32C3-P2-128	**92.07**

Table 4: Performance of different algorithms on Fashion-MNIST.

Algorithms	Hidden Layers	ACC (%)
Spiking-CNN [Neil and Liu, 2016]	-	95.72
LSTM [Neil *et al.*, 2016]	-	97.05
MLP [Lee *et al.*, 2016]	800	97.80
CNN [Neil and Liu, 2016]	-	98.30
Spiking-MLP [Lee *et al.*, 2016]	800	98.74
STBP [Wu *et al.*, 2018]	800	**99.53**
LISNN	32C3-P2-32C3-P2-128	99.45

Table 5: Performance of different algorithms on N-MNIST.

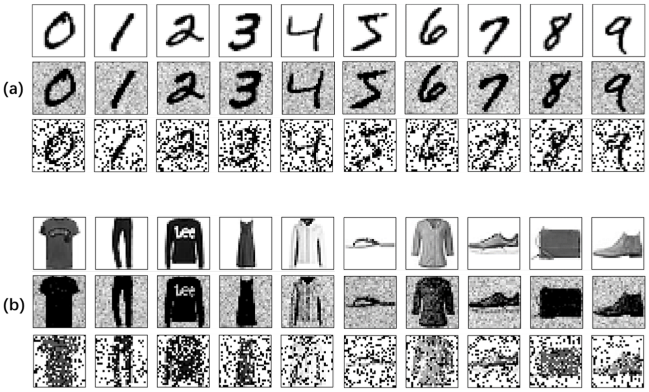

Figure 2: Dataset Demonstration. (a) Pictures in the first row are extracted from ten different categories of MNIST. Pictures in the second row and third row are fabricated by adding 0.4-mean gaussian noise and 0.4-amount impulse noise to original MNIST, respectively. We show Fashion-MNIST samples with or without noise using the same selection and generation method, as shown in (b).

4.4 Noise-added Image Dataset

To exploit the potential of lateral interactions, we train models on training datasets of original MNIST and Fashion-MNIST, then measure their performance on testing datasets, which is artificially appended with two different kinds of common noise, i. e., gaussian noise and impulse noise. We demonstrate several samples from testing datasets with and without noise in Figure 2. Figure 3(a) and Figure 3(b) is the accuracy curves of models trained on original training datasets of MNIST and Fashion-MNIST. The noise level of gaussian is measured by its mean and the noise level of impulse is measured by its amount. Both of them range from 0.05 to 0.5, with an interval of 0.05. In contrast, we also trained models on training datasets of MNIST and Fashion-MNIST with 0.4-mean gaussian noise. Their accuracy curves are demonstrated in Figure 3(c) and Figure 3(d).

From Figure 3(a) and Figure 3(b), we can conclude that SNN with lateral interactions exceeds SNN without lateral interactions at almost all situations, especially when inferencing the unseen noise data. Besides, from Figure 3(c) and Figure 3(d), we can conclude that training on noise datasets helps improving performance on the noisy situation but leads to performance loss on the original image. Thus, directly adding noise to training datasets may not be a better strategy against performance loss caused by noise than improving the model's robustness in all situations.

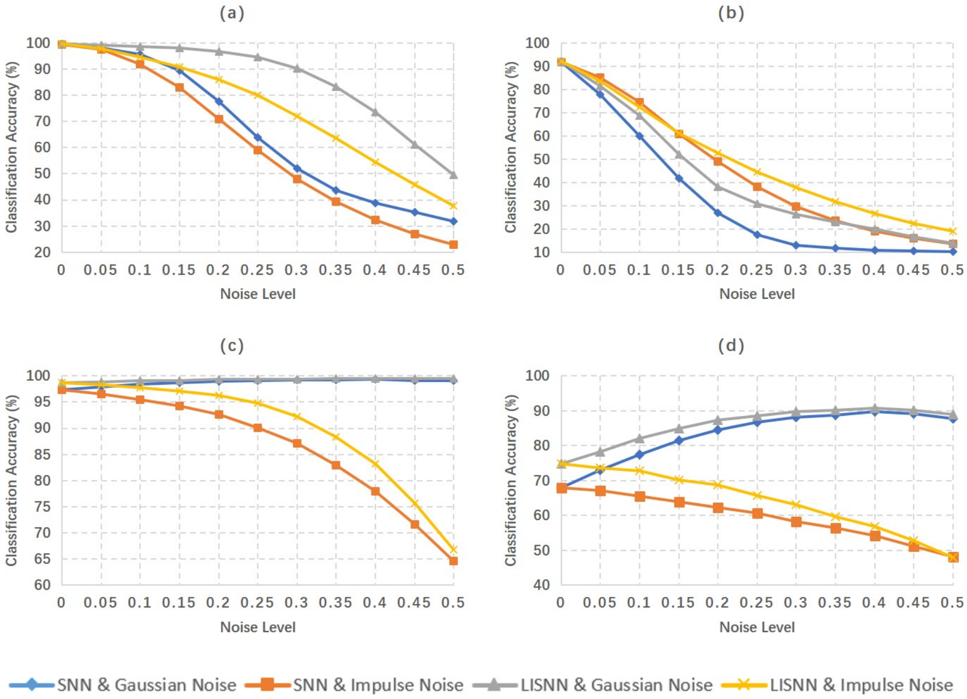

Figure 3: Accuracy Curves. Gaussian Noise/Impulse Noise in legend indicates the kind of noise added to the datasets which are used for testing. (a) Model is trained on original MNIST. (b) Model is trained on original Fashion-MNIST. (c) Model is trained on MNIST added with 0.4-mean gaussian noise. (d) Model is trained on Fashion-MNIST added with 0.4-mean gaussian noise.

5 Conclusion

It is a feasible and challenging way to explore and understand how the brain works by computational modeling using information-processing mechanisms in the brain. In this paper, we get inspiration from the lateral inhibition mechanism in the visual neural system and DNF in computational neuroscience and cognitive science, then introduce lateral interactions into computation models. Finally, we propose a new SNN model for robust visual object recognition, named LISNN. LISNN achieves the best accuracy on Fashion-MNIST and comparable performance on MNIST and N-MNIST, which indicates lateral interactions have positive effects on improving the performance of the model. Besides, comparing to SNN without lateral interaction, LISNN shows stronger noise robustness.

Acknowledgments

This work was supported by the Major Project for New Generation of AI (Grant No. 2018AAA0100400), the Advance Research Program (Grant No. 31511130301), the Beijing Brain Science Project (Grant No. Z181100001518006) and the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB32070000).

References

[Agarap, 2018] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[Bellec *et al.*, 2018] Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *NIPS*, pages 787–797, 2018.

[Bi and Poo, 1998] Guo-qiang Bi and Mu-ming Poo. Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of Neuroscience*, 18(24):10464–10472, 1998.

[Comsa *et al.*, 2019] Iulia M Comsa, Krzysztof Potempa, Luca Versari, Thomas Fischbacher, Andrea Gesmundo, and Jyrki Alakuijala. Temporal coding in spiking neural networks with alpha synaptic function. *arXiv preprint arXiv:1907.13223*, 2019.

[Diehl and Cook, 2015] Peter U. Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, 9, 2015.

[Evanusa *et al.*, 2019] Matthew Evanusa, Yulia Sandamirskaya, et al. Event-based attention and tracking on neuromorphic hardware. In *CVPR Workshops*, 2019.

[Ghosh-Dastidar and Adeli, 2009] Samanwoy Ghosh-Dastidar and Hojjat Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.

[Hao *et al.*, 2020] Yunzhe Hao, Xuhui Huang, Meng Dong, and Bo Xu. A biologically plausible supervised learning method for spiking neural networks using the symmetric stdp rule. *Neural Networks*, 121:387–395, 2020.

[Hebb, 1962] Donald Olding Hebb. *The organization of behavior: a neuropsychological theory*. Science Editions, 1962.

[Hinton and Salakhutdinov, 2006] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[Hu *et al.*, 2017] Zhanhao Hu, Tao Wang, and Xiaolin Hu. An stdp-based supervised learning algorithm for spiking neural networks. In *ICONIP*, pages 92–100. Springer, 2017.

[Hu *et al.*, 2018] Yangfan Hu, Huajin Tang, Yueming Wang, and Gang Pan. Spiking deep residual network, 2018.

[Jin *et al.*, 2018] Yingyezhe Jin, Wenrui Zhang, and Peng Li. Hybrid macro/micro level backpropagation for training deep spiking neural networks. In *NIPS*, pages 7005–7015, 2018.

[Kheradpisheh *et al.*, 2018] Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, Simon J Thorpe, and Timothe Masquelier. Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67, 2018.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[Lee *et al.*, 2016] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 10:508, 2016.

[Maass *et al.*, 2002] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[Neil and Liu, 2016] Daniel Neil and Shih-Chii Liu. Effective sensor fusion with event-based sensors and deep network architectures. In *ISCAS*, pages 2282–2285. IEEE, 2016.

[Neil *et al.*, 2016] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. Phased lstm: Accelerating recurrent network training for long or event-based sequences. In *NIPS*, pages 3882–3890, 2016.

[Orchard *et al.*, 2015] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in neuroscience*, 9:437, 2015.

[Ororbia and Mali, 2019] Alexander G Ororbia and Ankur Mali. Biologically motivated algorithms for propagating local target representations. In *AAAI*, volume 33, pages 4651–4658, 2019.

[Pfeiffer and Pfeil, 2018] Michael Pfeiffer and Thomas Pfeil. Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in Neuroscience*, 12, 2018.

[Ratliff *et al.*, 1974] Floyd Ratliff, H Keffer Hartline, and D Lange. The dynamics of lateral inhibition in the compound eye of limulus. i. page 463, 1974.

[Sandamirskaya, 2014] Yulia Sandamirskaya. Dynamic neural fields as a step toward cognitive neuromorphic architectures. *Frontiers in Neuroscience*, 7:276, 2014.

[Schöner and Spencer, 2016] Gregor Schöner and John Spencer. *Dynamic thinking: A primer on dynamic field theory*. Oxford University Press, 2016.

[Shatz, 1992] Carla J Shatz. The developing brain. *Scientific American*, 267(3):60–67, 1992.

[Shrestha and Orchard, 2018] Sumit Bam Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. In *NIPS*, pages 1412–1421, 2018.

[Woźniak *et al.*, 2018] Stanisław Woźniak, Angeliki Pantazi, Thomas Bohnstingl, and Evangelos Eleftheriou. Deep learning incorporating biologically-inspired neural dynamics. *arXiv preprint arXiv:1812.07040*, 2018.

[Wu *et al.*, 2018] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12, 2018.

[Wu *et al.*, 2019] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. Direct training for spiking neural networks: Faster, larger, better. In *AAAI*, volume 33, pages 1311–1318, 2019.

[Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[Xu *et al.*, 2018] Qi Xu, Yu Qi, Hang Yu, Jiangrong Shen, Huajin Tang, and Gang Pan. Csnn: An augmented spiking based framework with perceptron-inception. In *IJCAI*, pages 1646–1652, 2018.

[Zhang and Li, 2019] Wenrui Zhang and Peng Li. Spike-train level backpropagation for training deep recurrent spiking neural networks. In *NIPS*, pages 7800–7811, 2019.

[Zhang *et al.*, 2019] Lei Zhang, Shengyuan Zhou, Tian Zhi, Zidong Du, and Yunji Chen. Tdsnn: From deep neural networks to deep spike neural networks with temporal-coding. In *AAAI*, volume 33, pages 1319–1326, 2019.