CSNN: An Augmented Spiking based Framework with Perceptron-Inception

Qi Xu[1], **Yu Qi**[1], **Hang Yu**[1], **Jiangrong Shen**[1,2], **Huajin Tang**[3] and **Gang Pan**[1,*]
[1] College of Computer Science and Technology, Zhejiang University
[2] Qiushi Academy for Advanced Studies, Zhejiang University
[3] College of Computer Science, Sichuan University
{xuqi123, qiyu}@zju.edu.cn, me@ywuhang.com, jrshen@zju.edu.cn, htang@scu.edu.cn

Abstract

Spiking Neural Networks (SNNs) represent and transmit information in spikes, which is considered more biologically realistic and computationally powerful than the traditional Artificial Neural Networks. The spiking neurons encode useful temporal information and possess highly anti-noise property. The feature extraction ability of typical SNNs is limited by shallow structures. This paper focuses on improving the feature extraction ability of SNNs in virtue of powerful feature extraction ability of Convolutional Neural Networks (CNNs). CNNs can extract abstract features resorting to the structure of the convolutional feature maps. We propose a CNN-SNN (CSNN) model to combine feature learning ability of CNNs with cognition ability of SNNs. The CSNN model learns the encoded spatiotemporal representations of images in an event-driven way. We evaluate the CSNN model on the MNIST and its variants, including learning capabilities, encoding mechanisms, robustness to noisy stimuli and its classification performance. The results show that CSNN behaves well compared to other cognitive models with significantly fewer neurons and training samples. Our work brings more biological realism into modern image classification models, with the hope that these models can inform how the brain performs this high-level vision task.

1 Introduction

There are various conventional methods to implement pattern recognition, such as maximum entropy classifier, naïve Bayes classifier, decision trees and support vector machines (SVMs). However, all of these methods are less biologically plausible compared to spiking based neural networks. Humans can easily discriminate different classes within a short time. Moreover, human brains outperform computers in intelligent information processing tasks. Unlike traditional Artificial Neural Networks (ANNs), biological neural networks communicate via discrete spikes instead of numerical values, forming Spiking Neural Networks (SNNs). In SNNs, a neuron is activated only when it receives an input spike, hence inactive neurons without any input spikes can be put into low-power mode to save power. Due to the temporal dynamics features, SNNs are advantageous to deal with spatio-temporal patterns, through spike-based learning and memory mechanisms [Hu *et al.*, 2016]. However, compared with deep Convolutional Neural Networks (CNNs), typical SNNs are surely at a great disadvantage about feature extraction because they consist of just a fully-connected layer with biologically based neurons. One fully-connected layer cannot detect and capture some deeper and hidden information which is different from deep structures of CNNs.

On the other hand, CNNs have recently enjoyed a great success in many areas of computer vision, especially classification [Krizhevsky *et al.*, 2012]. The great performance of CNNs comes hand-in-hand with high model complexity. It is common for modern CNNs to have tens of millions of parameters. This high complexity allows CNNs to learn complex concepts from training data. This motivates us to investigate computational models for generalized pattern recognition from a biological point of view. Meanwhile improving the performance of SNNs resorts to the feature extraction ability of CNNs.

How information is represented in the brain still remains unclear. However, there is strong evidence to believe that spike trains are an optimal way for transmission and information representation. In human brains, the retina is a functional part and its structures are remarkably well known. It is widely believed that the retina receives the outside stimuli, and extracts the features through visual systems. The ganglion cells (GCs) collect the information from the receptive fields [Hubel and Wiesel, 1968]. Information collected by the GCs from outside stimuli are as the first layer, the complex cells (CCs) following GCs collect information from a local position of GCs, and a MAX operation among these GCs determines the activation value of CC unit.

Inspired by the mechanism of vision formation in biological brain, we propose a brain-inspired Perceptron-Inception based neural network, named CSNN, which consists of a partial CNN and an SNN. We name them the Perceptron and the Inception respectively. The Perceptron consists of convolutional and pooling layers as in CNN and the Inception is a fully-connected-layer SNN. The Perceptron acts as a feature

[*]Corresponding author: gpan@zju.edu.cn

extractor which is similar to the GCs and the CCs in visual systems. The Inception encodes the features generated from Perceptron to spikes, adjusts the synaptic weights and outputs the final classification result. Finally, the Inception will make the decision which is represented by the outside stimuli. By combining the Perceptron and the Inception into the same model, CSNN is able to exploit the powerful feature extraction ability of the CNN to enhance the pattern recognition of the SNN.

We try to build a bridge between a real-world task (image encoding) and the human visual formation (cognitive ability) with the powerful feature extraction ability of CNNs and the generalization ability of SNNs. This paper evaluates CSNN model on three different sizes of benchmark datasets: basic MNIST and two its variational datasets including: background-MNIST, background-random MNIST. Experimental results show that the CSNN is not only capable of recognizing images with a performance comparable to that of current cognitive models, but also having generalization ability about the noise-cancelling. Moreover, our structure suggests a biological plausibility proof for a class of feedforward models of rapid and robust recognition in the brain.

2 Related Work

Typical Convolutional Neural Networks (CNNs) consist of three different kinds of components: convolutional layers, pooling layers and fully-connected layers. Convolutional layers extract features, pooling layers maintain the invariance of fields and reduce the dimension of feature maps, fully-connected layers are a kind of classifier which classify the pattern with respect to the input features. The feature extraction ability of CNNs is powerful, because its deep structure and functional parts such as convolutional and pooling parts.

Spiking Neural Networks (SNNs) are another feedforward computational model for pattern recognition tasks. General SNN model composes three parts: the encoding layer, the learning layer, and the readout layer. The neurons in SNNs are more biological than those of CNNs.

When a neuron in the SNN receives an input spike, its membrane potential either increases or decreases, depending on whether the input spike is from an excitatory or an inhibitory synapse. If its membrane potential reaches the firing threshold, the neuron generates an outgoing spike, which travels down the axon to downstream synapses and neurons. When a spike arrives at a synapse, the pre-synaptic neuron releases neurotransmitters into the synaptic cleft, which in turn bind to dendritic receptors of the postsynaptic neuron, causing a change in the membrane potential of the postsynaptic neuron. Such behavior dynamics can be modeled at different levels of abstraction (neuron models).

There are several kinds of spiking neuron models developed by researchers, such as the resonate-and-fire model [Izhikevich, 2001], the Hodgkin–Huxley (HH) model [Hodgkin and Huxley, 1952], the Izhikevich model [Izhikevich, 2003] and the leaky integrate-and-fire model (LIF) [Hu *et al.*, 2013]. LIF neuron model is adjusted in modeling an SNN because of its strong biology support and effective computation. There are several variants of the Leaky Integrate-and-Fire (LIF) model. The simplest and the most widely used variant is the current-based LIF model, which is much more computationally efficient than the HH model. The membrane potential V of a LIF neuron is governed by the following equations.

$$C_m \frac{dV}{dt} = g_l(E_l - V) + I, \tag{1}$$

$$V = V_{rest}, \;\; if\; V \geq V_{th}, \tag{2}$$

C_m denotes membrane capacitance, g_l denotes conductance (inverse of resistance) of the leakage channels, E_l denotes the equilibrium potential of the leakage channels, and I denotes total input current. The membrane potential $V(t)$ of a LIF neuron is weighted sum of postsynaptic potentials (PSP) from all afferent stimuli.

Generally, the deeper network extracts more important information than shallow one. However, due to the limitation of existed training rules, deep SNNs cannot be trained sufficiently. Hence the feature extraction ability for typical SNNs is limited. In SNNs, encoding layers act as feature extractor which encode the raw images to spikes, forming spatial-temporal patterns. Learning layers tune their synaptic weights via spikes. The accuracy of feature extraction directly affects the performance of SNNs.

Encoding layers take an input image to spikes using specific encoding methods. There exists many encoding mechanisms to encode images, the rate based coding [Peter *et al.*, 2013] is used to encode images into dense spikes, a higher firing rate is defined as high sensory variable which can be represented as the average number of spikes counting within a temporal encoding window. The rate based coding always uses dense spikes (The Poisson spike trains) to represent the neurons firing rate. [Merolla *et al.*, 2011] proposes a novel algorithm which adopted filtered spike train as transition from original images. The sparse coding [Perrinet *et al.*, 2004] clusters a relatively small subset of neurons which have nearly the same firing rate. But studies [Berry and Meister, 1998; Uzzell and Chichilnisky, 2004] have proved that neurons in human retina firing more likely as temporal coding mechanism. Patterns encoded from temporal coding can carry more information in spatiotemporal spikes and consume fewer computational resources than rate based coding.

Although the encoding mechanisms are various, the encoding part is only one layer. The feature extraction ability of typical SNNs is limited. In this paper, the CSNN model focuses on improving the feature extraction ability with the help of CNNs. Compared to typical SNNs, our model can detect deeper and more important information.

3 CSNN Model

This section will introduce the CSNN model which is a mixture of a partial CNN and an SNN as shown in Figure 1. We name the partial CNN as Perceptron, which consists of convolutional and pooling layers and name the SNN part as Inception, respectively. The Perceptron acts as the V1 of retina in human brain, which is an interesting sensory area to study neural information processing, since its functional organization and structure are well known. It is also widely believed

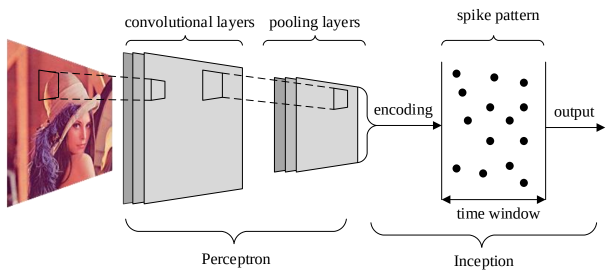


Figure 1: CSNN model

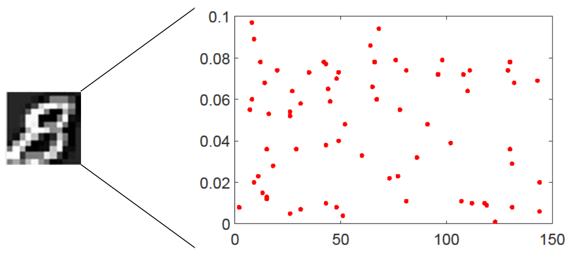


Figure 2: A feature map is encoded to spikes

that information transmitted from retina to brain codes the visual stimuli at each specific receptive field. The Inception is a classifier which takes input features to spikes from the Perceptron and is trained by SNN learning rules. The CSNN model is a unified systematic model with feature extraction, consistent encoding, learning and readout parts.

3.1 Perceptron Filters Based on CNN

Focusing on simulating the information processing in visual cortex, we use a partial model (CNN) for feature extractor. This CNN in CSNN acts as the Perceptron which is a hierarchical system.

In CSNN, the image information is transmitted to the Perceptron within convolutional and pooling parts. Convolutional layers in CSNN model act as similar to the GCs parts in human brain, since the filters in convolutional layers are believed to mimic how neural processing in the human retina of the eyes extract the important information from external stimuli [Krizhevsky *et al.*, 2012]. In the cortex, the GCs are used as the first layer to collect information from outside stimuli, after that, the CCs extract the features from the local regions in a whole image produced by GCs.

There is a similarity between the roles the pooling layers in CSNN plays and the CCs layer. A pooling layer applies a nonlinear max pooling operation to its input to achieve invariance. Max pooling over different directions, different scales and different local positions offers contrast reverse invariance, scale invariance and position invariance, respectively. [Yu *et al.*, 2002] has proposed biophysically plausible implementations of the MAX operation. Biological evidence [Lampl *et al.*, 2004] of neurons performing MAX-like operation have been found in a subclass of CCs in V1 and V4.

Following the MAX operation, the activation function would trigger the value of the feature maps produced by the pooling layer. The pixel would more easily be activated if its value is larger, whereas numerical small ones would be activated weakly. In this paper, we choose sigmod function as Perceptron's activation function.

3.2 Temporal Encoding for CSNN

In CSNN, we consider the actual values of neurons' activations to generate spikes. In particular, we choose temporal encoding as our fundamental mechanism. Each image is presented to the encoding layer and converted into spatiotemporal pattern. The encoding rule is important as a bridge between numerical values and spikes.

The Perceptron produces a set of feature maps (analog values), corresponding to the activation levels of pooling layers. The strongly activated value would fire earlier, the weakly activated figure would fire later or cannot fire anymore. The spike latencies are linearly mapped into a time window.

The activation values are linearly mapped to delay times, for example, t=0 with activation value 1, with lower activation value, later times up to time window. Eq. (3) illustrates the spiking time transferred from the image. T_{spike} is the firing time which are calculated from time window T and activation value A of row pixel.

$$T_{spike} = T - T * A, \tag{3}$$

Furthermore, the neurons with activation value of 0 would not fire because this numerical number is too low to induce neurons firing.

CSNN is capable to extract the basic information from an original image and encode it to a spatiotemporal spiking pattern. A sparse representation is finally obtained through the whole encoding structure as shown in Figure 2. The x-axis denotes the timewindow and the y-axis represents the firing time of spikes. This spatiotemporal representation can transfer image information to sparse spiking pattern, to some extent, is compatible with the biological observations in the retina.

3.3 Inception Classification Based on SNN

In the Inception, one neuron can discriminate the pattern by firing or not. The synaptic weights of the Inception are adjusted through the following Tempotron rule. This rule can make appropriate output under the supervisory signal by tuning parameters and modifies the synaptic weights. Such that the trained neuron will produce a spike when it is presented with a pattern corresponding to one class (A) through the specific neuron or the neuron group, while no spike is presented with another class (B) through its corresponding neuron or the neuron group.

The LIF neuron model is driven by postsynaptic potential (PSP) produced by its afferent synapses. The membrane potential V of the neuron is a weighted sum from all incoming spikes and the kernel function $K(t-t_i)$, t_i is the firing time of the i_{th} afferent.

$$V(t) = \sum_i W_i \sum_{t_i} K(t-t_i) + V_{rest}, \tag{4}$$

$$K(t-t_i) = \begin{cases} V_{base}(e^{-\frac{t-t_i}{\tau_m}} - e^{-\frac{t-t_i}{\tau_s}}) & if\ (t-t_i) > 0 \\ 0 & if\ (t-t_i) \le 0 \end{cases}, \tag{5}$$

The kernel function can be described as Eq. (5), where τ_m and τ_s denote time constants of the membrane potential and synaptic currents. The maximum of the kernel function is normalized by the V_{base}. Meanwhile, $K(t)$ is also a filter which only consider that the firing time t_i less than or equal to the current time t during the time window. With this kernel function normalization, the effect of PSP amplitudes on the neuron membrane potential is adjusted by the synaptic efficacies.

In the Inception of CSNN model, we specify the following parameters. In this LIF neuron model, V_{rest} is 0, V_{base} is the normalization value that limits the maximum of $K(t)$ to 1, the time constant τ_m, τ_s, are 0.01s and 0.0025s respectively. The threshold is set to 1. When a neuron emits a spike, its membrane potential is reset to V_{rest} (0) and is held there for a refractory period (0.003s).

In a two-class pattern recognition task, the input pattern is presented to the neuron belongs to which category (the patterns are labeled by A and B). One neuron makes a final decision through firing or not. When a pattern A is present to the specific neuron, this neuron would fire. Meanwhile, a pattern B is present to this neuron, and this neuron would keep silent. The Tempotron learning rule tunes the synaptic weight W_i whenever there is an error. This rule behaves like the gradient-descent rule that minimizes a cost function as follows:

$$C = \begin{cases} V_{thr} - V_{t_{max}} & if\ A\ error \\ V_{t_{max} - V_{thr}} & if\ B\ error \end{cases}, \tag{6}$$

Where $V_{t_{max}}$ represents the maximum value of the PSP of V. V_{thr} is the threshold, neurons behave differently when they meet different patterns.

Applying the gradient-decent method to minimize the cost forms to the Tempotron rule:

$$\Delta W_i = \begin{cases} \lambda \sum_{t_i < t_{max}} K(t_{max} - t_i) & if\ A\ error \\ -\lambda \sum_{t_i < t_{max}} K(t_{max} - t_i) & if\ B\ error \\ 0 & otherwise \end{cases}, \tag{7}$$

Where t_{max} denotes the time when the neuron reaches its maximum potential value in the time window. λ is a constant about the learning rate. It controls the maximum change on the synaptic efficacies. A error means, the corresponding neuron should have fired but it did not, B error denotes that the neuron should not fire but it did. In this kernel shape, the efficacies of afferents that the spikes change more near to t_{max} than those far away from that time. t_{max} is a label for adjusting the synaptic weights.

The readout layer is the classification layer in the Inception. This part aims to extract useful information about the stimuli from responses of the learning neurons. When the spikes through learning layers arrive, the output neurons will emit spikes according to the afferent synaptic weights. In this paper, we use winner-take-all (WTA) [Oster *et al.*, 2005] as decision making strategy.

4 Experimental Results

In this section, we evaluate CSNN model on three benchmark datasets, basic MNIST [Lécun *et al.*, 1998], background MNIST [Larochelle *et al.*, 2007], background-random MNIST [Larochelle *et al.*, 2007] as shown in Figure 3. Each MNIST dataset consists of 28x28 grayscale images of handwritten digits from 0 to 9, the dataset is divided into two parts, 50000 training samples and 10000 test samples.

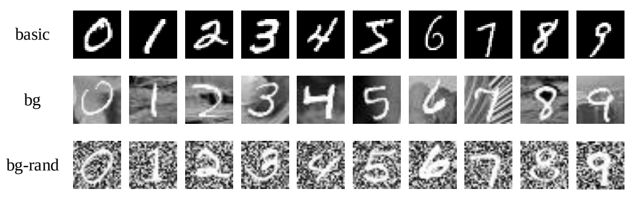


Figure 3: MNIST and its variations

Furthermore, in order to verify that CSNN can achieve better performance on small-size training sets than other cognitive models, the size of datasets is divided into different amount.

The learning algorithm of CSNN follows two steps: Firstly, training a full CNN with the Stochastic Gradient Descent (SGD) algorithm. After the training phase of the full CNN, the convolutional layers and pooling layers of the trained CNN are kept and used as the Perceptron; while the fully-connected layers are discarded. Then, training the Inception according to the Tempotron rule. During the training phase of the Inception, parameters of the Perceptron are fixed.

4.1 Experimental Settings

The experiments are run on a windows server equipped with two-processor Intel Xeon(R) Core CPU, and 64 GB main memory. The operating system is windows server 2012 R2 Standard. We use Matlab for training and testing the CSNN.

For MNIST and its two variational datasets, we trained a CNN which is a variant of LeNet-5 as the Perceptron of CSNN. The CSNN network architecture is (CNN)6C5@28x28-P2-(SNN)F200-F100. In this CSNN network, the Perceptron consists of a partial CNN including convolutional and pooling layers. The Inception composes 2 fully-connected layers (F200 and F100), the convolutional layer of the Perceptron has 6 5x5 filters, followed by a max-pooling layer, which is followed by the Inception. The Inception adopts the LIF as the neuron model, the first FCL F200 is the encoding layer which encodes feature maps produced by the Perceptron to spikes, the second one F100 is the readout layer. The readout layer consists of 100 LIF neurons, each 10 neurons represents the same class. The final output decision is made through the maximum summation of spikes that come from the specific neuron cluster.

4.2 Influence of Three Channel Selection Methods

The Perceptron consists of convolutional and pooling layers, multi-channels of convolutional kernels will produce various specific feature maps. The size of the Inception is adjusted by the encoding mechanisms. In this section, we design and implement three channel selection methods to show the performance of CSNN:

- **First Channel:** Choosing the feature maps produced by the first channel of the Perceptron

- **Avg. Channel:** Taking average of the feature maps produced by all channels of the Perceptron
- **All Channels:** Using all feature maps produced by channels from the Perceptron.

First, only the feature maps from the first channel are kept. Second, using the feature maps which are taken average of all feature maps. Third, using feature maps from all channels, each channel of feature maps corresponds to a group of encoding neurons, thus, the amount of encoding and output neurons are multiplied depending on the number of channels.

As shown in Figure 4, for both three channel selection methods, we train three datasets respectively, and test the performance on each corresponding test sets. We observe that when the size of the training set is limited, First Channel performs better than Avg. Channel. This is expected since the network which includes fewer parameters can be trained sufficiently with the feature maps produced by the first channel. Compared to First Channel, Avg. Channel takes average of feature maps from all channels, when the size of training set is limited, the feature maps from average of all channels may loose the useful information through mixing the feature maps which are at different degree of chromatism. With the training samples increasing, Avg. Channel behaves slightly better than First Channel. It is reasonable that the average feature maps can represent more useful information with sufficient training samples. As for All Channels, it performs nearly best no matter what size training sets have. In All Channels, the scale of the Inception is multiplied according to the number of channels. Generally, if the capacity of the network is larger, it can behave better than small one. But All Channels brings a new issue that the neurons and parameters are increased sharply compared to First Channel and Avg. Channel, the network may not be trained sufficiently if the size of training set is limited.

The test accuracies are different if the training sets are diverse. In Fig. 4(a), the test accuracies can achieve 86% when the training samples are enough. As for bg and bg rand MNIST, CSNN behaves worse than on basic set by all three methods, because the variational sets are too noisy. And the observation from Fig. 4(b) and Fig. 4(c) clearly suggests that the role which channel selection methods play is the same as Fig. 4(a) shows.

4.3 Comparison of Feature Extraction Ability Between CSNN and S1C1-SNN

To show the feature extraction ability, we compare CSNN model with S1C1-SNN [Yu *et al.*, 2013] which chooses gabor filters and max operation as feature extractor. Since S1C1-SNN only selects one averaged feature map, we choose $Avg.\ Channel$ as channel selection method. Training sets and test sets are mixed. For example, the CSNN is trained on Standard MNIST training set and tested on test sets from Standard MNIST, Background MNIST and Background random MNIST. These experimental conditions are expected to test the different degree of their generalization ability. For each of the datasets, we train three datasets, each corresponding to three test sets to test the performance, so there are 9 cases: training set is basic MNIST, test sets are basic, bg and bg-rand, training set is bg, test sets are basic, bg and bg-rand, and training set is bg-rand, test sets are basic, bg and bg-rand respectively. The perceptron of CSNN is trained with batch size 10 for 100 training set and batch size 100 for the rest sizes of the training sets.

Test and training accuracies of all 9 conditional cases are shown in Table 1. The left results in the grid come from S1C1-SNN, the right are produced by CSNN. From Table 1, we observe that when training set is basic, test is basic, CSNN achieves significantly better test accuracies than S1C1-SNN. It reaches 87% when the training samples are 1000. And the test accuracy is decreased with the training size increasing, because the scale of SNN part in CSNN and S1C1-SNN is small, which consist of only 200 learning neurons and 100 output neurons. The network capacity limits the performance.

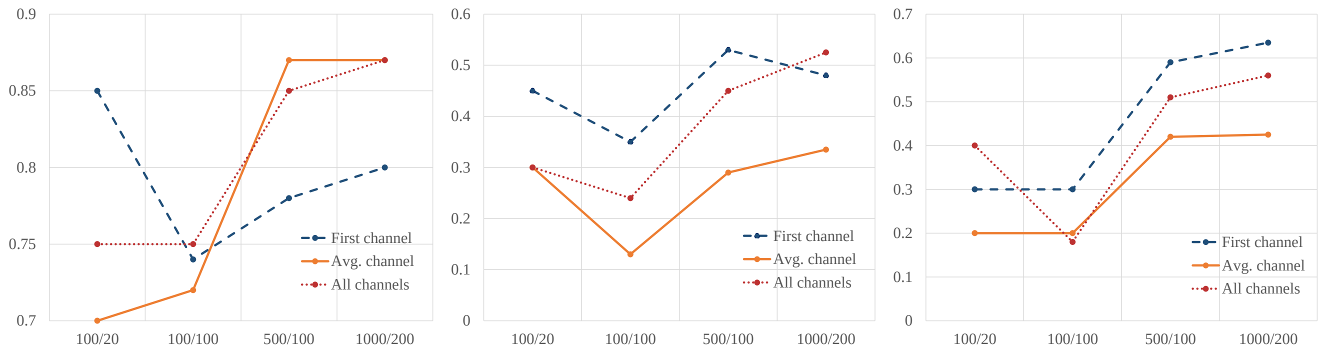
Furthermore, CSNN behaves better in almost all cases than S1C1-SNN, especial the rightmost four columns in $test\ accuracies\ of\ 9\ cases$ grids, when the training sets are bg, bg-rand, bg-rand, bg-rand, the test sets are bg-rand, basic, bg, bg-rand. As the classical cognitive model, S1C1-SNN also behaves better than CSNN on the following cases when the training sets are basic, bg and the test sets are bg and basic respectively. All test accuracies are still low except the results from using basic for training and basic for test.

From above experimental results, we observe that although CSNN does not perform better than S1C1-SNN within all experimental conditionals, CSNN can extract more useful information than S1C1-SNN to promote training of SNN part through the CNN based Perceptron compared to S1C1 part in S1C1-SNN, especially when the training sets are noisy with background images.

The three rightmost columns of Table 1 give information about training accuracies on S1C1-SNN and CSNN respectively. We observe that all of the training accuracies produced by CSNN are better than results from S1C1-SNN. Furthermore, when the training size is small, the training accuracy is easier to achieve higher, which is normal, because the network would be overfitting easily on small size training sets. With training samples increasing, the training accuracies are decreased with the small scale SNN part (200 encoding neurons and 100 output neurons). When the training sets are bg and bg-rand, the gap of accuracies are widening. The accuracy of CSNN reaches 70.67% when the training set includes 1000 bg-rand samples, compared to 16.10% of S1C1-SNN. In particularly, the training accuracies are low when the training sets are bg and bg-rand within 5000, 10000, 40000 training samples compared to basic datasets based training sets. The training accuracies are also limited by the amount of network parameters.

4.4 Performance Comparison with Other Methods

The presented CSNN achieves good classification performance on the basic MNIST with the Perceptron made of feature extractor and the Inception made of biologically plausible components under limited neurons. A comparison of SNN based models for benchmark is shown in Table 2. We compare CSNN to four state of the art cognitive models: S1C1-SNN, Dendritic Neurons [Hussain *et al.*, 2014], Spiking RBM [Merolla *et al.*, 2011] and Unsupervised STD-



(a) Training set: basic Test set: basic   (b) Training set: bg Test set: bg   (c) Training set: bg_rand Test set: bg_rand

Figure 4: Test accuracy (%) of basic, bg and bg-rand MNIST with 3 different channel selection methods

	Test accuracies of 9 cases									Training accuracies of 3 datasets		
Traing	basic	basic	basic	bg	bg	bg	bg-rand	bg-rand	bg-rand	basic	bg	bg-rand
Test	basic	bg	bg-rand	basic	bg	bg-rand	basic	bg	bg-rand	basic	bg	bg-rand
100	**70.00/75.00**	31.00/21.00	12.00/22.00	31.00/20.00	**27.00/22.00**	20.00/26.00	21.00/27.00	23.00/17.00	**30.00/18.00**	100/100	100/100	100/100
500	**75.00/81.00**	30.00/27.00	19.00/38.00	45.00/43.00	**28.00/60.00**	15.00/64.00	21.00/,9.00	20.00/44.00	**20.00/62.00**	94.60/99.65	46.60/93.04	71.60/99.96
1000	**78.50/87.00**	29.00/26.00	14.50/23.50	58.00/47.00	**34.00/33.50**	15.50/38.50	19.00/24.00	21.00/34.50	**15.50/42.50**	84.97/94.50	34.27/70.23	16.10/70.67
5000	**77.30/84.70**	32.40/23.50	15.60/25.90	64.50/52.20	**25.60/39.80**	12.20/40.60	10.90/27.30	14.80/29.40	**12.20/39.00**	76.98/86.44	22.40/47.26	11.51/44.12
10000	**77.40/86.05**	31.30/22.25	12.25/24.00	60.20/52.90	**19.80/41.75**	10.80/39.85	11.35/23.25	15.35/27.20	**12.40/39.15**	75.65/85.04	20.47/42.49	8.12/41.12
40000	**75.99/83.78**	29.23/34.68	14.25/24.38	59.20/52.63	**20.20/39.63**	11.34/41.58	9.78/21.48	13.13/26.28	**11.56/37.03**	75.28/83.50	20.89/40.86	11.60/38.35

Table 1: Comparison of feature extraction ability between S1C1-SNN and CSNN.

P [Diehl and Cook, 2015]. There models are all spiking based systems.

The size of each model is various, for example the S1C1-SNN only has 300 neurons, but the Dendritic Neurons, Spiking RBM and Unsupervised STDP has 5010, 7480 and 13584 neurons respectively. The training samples are adjusted according to the network capacity. Table 2 shows the test accuracies of basic MNIST with different cognitive models. We observe that with the limited neurons (300), CSNN achieves 81% and 87% with the 500 training samples and 1000 training samples. Compared to CSNN, S1C1-SNN only gets 76.89% with more training samples (5000) and test samples (1000).

As for the other three networks, we choose $All\ Channels$ (using feature maps from all channels) mentioned in section 4.3 to increase the capacity of CSNN. CSNN has 800 neurons to receive spikes produced by the Perceptron and 400 neurons to learn and output the final classification. The test accuracy of CSNN (1200 neurons) is 88.00% with so limited training samples (500). Meanwhile, the test accuracy of Dendritic Neurons is 90.26% with 10,000 training samples, the figure of Spiking RBM is 94.09% with 12,000 training samples and 95.00% of Unsupervised STDP with the largest number of training samples (60,000).

Although the CSNN model cannot behave best compared to these models, it has the smallest size which only use 300 or 1200 neurons to construct the model to recognize the digits. Furthermore, with the restrict of small size structure, CSNN can get a well performance on fewer training samples, which brings a more efficient way to apply it on resource constrained Neuromorphic devices.

Network type	Neurons	Training samples	Test samples	Accuracy
S1C1-SNN	200+100	5000	1000	76.98
CSNN	200+100	500	100	81.00
CSNN	**200+100**	**1000**	**200**	**87.00**
CSNN	**800+400**	**500**	**100**	**88.00**
Dendritic Neurons	5000+10	10,000	5000	90.26
Spiking RBM	6470+1010	12,000	10,000	94.09
Unsupervised STDP	784+6400+6400	60,000	10,000	95.00

Table 2: Test accuracies (%) of basic MNIST with different cognitive models.

5 Conclusion

In this paper, a brain-inspired cognitive model CSNN is proposed. CSNN combines feature extraction ability of CNNs and biological plausibility of SNNs. The spatiotemporal representations encoded from external stimuli by the Inception have properties of selectivity and invariance. We show the performance of the system applied to MNIST and its variations is comparable to that of the cognitive models: S1C1-SNN, Dendritic Neurons, Spiking RBM and unsupervised STDP with significantly fewer neurons and training samples. Using this structure, it would potentially be beneficial for implementations of neuromorphic chips [Ma *et al.*, 2017] and VLSI. This work brings more biological realism into modern image classification models, with the hope that these models can inform how the brain performs this high-level vision task. In the future, we will further investigate how to combine C-SNN with other regularization algorithm [Xu and Pan, 2017], the basic idea of the CSNN may be extended to other type of network [Shen *et al.*, 2017] and medical application [Zhou *et al.*, 2018].

Acknowledgments

This work was partly supported by National Key Research and Development Program of China (2017YFB1002503), Natural Science Foundation of China (No.61772460), and Zhejiang Provincial Natural Science Foundation of China (LR15F020001).

References

[Berry and Meister, 1998] Michael J Berry and Markus Meister. Refractoriness and neural precision. *Journal of Neuroscience*, 18(6):2200, 1998.

[Diehl and Cook, 2015] Peter U. Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, 9:99, 2015.

[Hodgkin and Huxley, 1952] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Bulletin of Mathematical Biology*, 117(4):500–544, 1952.

[Hu *et al.*, 2013] Jun Hu, Huajin Tang, Kay Chen. Tan, Haizhou Li, and Luping Shi. A spike-timing-based integrated model for pattern recognition. *Neural Computation*, 25(2):450–472, 2013.

[Hu *et al.*, 2016] Jun Hu, Huajin Tang, Kay Chen Tan, and Haizhou Li. How the brain formulates memory: A spatio-temporal model research frontier. *IEEE Computational Intelligence Magazine*, 11(2):56–68, 2016.

[Hubel and Wiesel, 1968] David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture of monkey striate cortex. *Journal of Physiology*, 195(1):215–243, 1968.

[Hussain *et al.*, 2014] S Hussain, Shih Chii Liu, and A Basu. Improved margin multi-class classification using dendritic neurons with morphological learning. In *IEEE International Symposium on Circuits and Systems*, pages 2640–2643, 2014.

[Izhikevich, 2001] Eugene M. Izhikevich. Resonate-and-fire neurons. *Neural Networks*, 14(6–7):883–894, 2001.

[Izhikevich, 2003] Eugene M Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569, 2003.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105, 2012.

[Lampl *et al.*, 2004] Ilan Lampl, David Ferster, Tomaso Poggio, and Maximilian Riesenhuber. Intracellular measurements of spatial integration and the MAX operation in complex cells of the cat primary visual cortex. *Journal of Neurophysiology*, 92(5):2704, 2004.

[Larochelle *et al.*, 2007] Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *International Conference on Machine Learning*, pages 473–480, 2007.

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

[Ma *et al.*, 2017] De Ma, Juncheng Shen, Zonghua Gu, Ming Zhang, Xiaolei Zhu, Xiaoqiang Xu, Qi Xu, Yangjing Shen, and Gang Pan. Darwin: A neuromorphic hardware co-processor based on spiking neural networks. *Journal of Systems Architecture*, 2017.

[Merolla *et al.*, 2011] Paul Merolla, John Arthur, Filipp Akopyan, Nabil Imam, Rajit Manohar, and Dharmendra S. Modha. A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm. In *Custom Integrated Circuits Conference*, pages 1–4, 2011.

[Oster *et al.*, 2005] Matthias Oster, Rodney J Douglas, and Shihchii Liu. Computation with spikes in a winner-take-all network. *Neural Computation*, 21(9):2437, 2005.

[Perrinet *et al.*, 2004] Laurent Perrinet, Manuel Samuelides, and Simon Thorpe. Sparse spike coding in an asynchronous feed-forward multi-layer neural network using matching pursuit. *Neurocomputing*, 57:125–134, 2004.

[Peter *et al.*, 2013] O'Connor Peter, Neil Daniel, Shih Chii Liu, Delbruck Tobi, and Pfeiffer Michael. Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in Neuroscience*, 7:178, 2013.

[Shen *et al.*, 2017] Jiangrong Shen, Kang Lin, Yueming Wang, and Gang Pan. Character recognition from trajectory by recurrent spiking neural networks. In *International Conference of the IEEE Engineering in Medicine and Biology Society*, page 2900, 2017.

[Uzzell and Chichilnisky, 2004] V. J. Uzzell and E. J. Chichilnisky. Precision of spike trains in primate retinal ganglion cells. *Journal of Neurophysiology*, 92(2):780–9, 2004.

[Xu and Pan, 2017] Qi Xu and Gang Pan. SparseConnect: regularising CNNs on fully connected layers. *Electronics Letters*, 53(18):1246–1248, 2017.

[Yu *et al.*, 2002] Angela J Yu, Martin A Giese, and Tomaso Poggio. Biophysiologically plausible implementations of the maximum operation. *Neural Computation*, 14(12):2857, 2002.

[Yu *et al.*, 2013] Qiang Yu, Huajin Tang, Kay Chen Tan, and Haizhou Li. Rapid feedforward computation by temporal encoding and learning with spiking neurons. *IEEE Transaction on Neural Networks and Learning Systems*, 24(10):1539–1552, 2013.

[Zhou *et al.*, 2018] Zhen Zhou, Jian Bao Wang, Yu Feng Zang, and Gang Pan. PAIR comparison between two within-group conditions of resting-state fMRI improves classification accuracy. *Frontiers in Neuroscience*, 11:740, 2018.