Neurocomputing ■ (■■■■) ■■■–■■■

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A brain-inspired spiking neural network model with temporal encoding and learning

Qiang Yu [a], Huajin Tang [b,c,*], Kay Chen Tan [a], Haoyong Yu [d]

[a] Department of Electrical and Computer Engineering, National University of Singapore, 117576, Singapore
[b] Institute for Infocomm Research, Agency for Science Technology and Research (A*STAR), 138632, Singapore
[c] College of Computer Science, Sichuan University, Chengdu 610065, China
[d] Department of Bioengineering, National University of Singapore, 117576, Singapore

ARTICLE INFO

Article history:
Received 21 August 2012
Received in revised form
18 March 2013
Accepted 27 June 2013

Keywords:
Spiking neural networks (SNNs)
Pattern recognition
Cognitive memory
Temporal encoding
Temporal learning

ABSTRACT

Neural coding and learning are important components in cognitive memory system, by processing the sensory inputs and distinguishing different patterns to allow for higher level brain functions such as memory storage and retrieval. Benefitting from biological relevance, this paper presents a spiking neural network of leaky integrate-and-fire (LIF) neurons for pattern recognition. A biologically plausible supervised synaptic learning rule is used so that neurons can efficiently make a decision. The whole system contains encoding, learning and readout. Utilizing the temporal coding and learning, networks of spiking neurons can effectively and efficiently perform various classification tasks. It can classify complex patterns of activities stored in a vector, as well as the real-world stimuli. Our approach is also benchmarked on the nonlinearly separable Iris dataset. The proposed approach achieves a good generalization, with a classification accuracy of 99.63% for training and 92.55% for testing. In addition, the trained networks demonstrate that the temporal coding is a viable means for fast neural information processing.

1. Introduction

The great computational power of biological systems has drawn increasing attention from researchers. Although the detailed information processing involved in memory is still unclear, observed biological processes have inspired many computational models operating at power efficiencies close to biological systems. Pattern recognition is the ability to identify objects in the environment, and several conventional methods are used to implement it, such as maximum entropy classifier, naive Bayes classifier, decision trees, and support vector machines. As is a necessary first step in all cognitive processes including memory, it is better to consider pattern recognition from brain-inspired models which could potentially provide great computational power.

To approach biological neural networks, the artificial neural networks (ANNs) are developed as simplified approximations in terms of structure and function. Since early neurons of the McCulloch–Pitt neuron in 1940s and the perceptron in 1950s [1], referred as the first generation neuron models, ANNs have been evolving towards more neural-realistic models. Different from the first generation neurons in which step-function threshold is used, the second generation neurons use continuous activation functions (like a sigmoid or radial basis function) as threshold for output determination [2]. The first two generations are referred as traditional neuron models. Studies on biological systems disclose that neurons communicate with each other through action potentials (pulses or spikes). As the third generation neuron model, spiking neurons raise the level of biological realism by utilizing spikes. The spiking neurons dealing with precise timing spikes improve the traditional neural models on both the aspects of accuracy and computational power [3]. There are several kinds of spiking neuron models such as the integrate-and-fire (IF) model [4], the resonate-and-fire model [5], the Hodgkin–Huxley model [6], and the Izhikevich model [7]. Since the IF model is simple and computationally effective [8], it is the most widely used spiking neuron model [9–15], despite other more biologically realistic models.

Encoding is the first step in creating a memory, which considers how information is represented in the brain. Although results remains unclear, there are strong reasons to believe that it is optimal using pulses to encode the information for transmission [16]. The inputs to a spiking neuron are discrete spike times. Rate coding and temporal coding are two basic and widely studied schemes of encoding information in these spikes. In the rate coding the average firing rate within a time window is considered, while for the temporal coding the precise timings of spikes are considered [17]. Neurons, in the retina [18,19], the lateral

* Corresponding author.

http://dx.doi.org/10.1016/j.neucom.2013.06.052
0925-2312/

geniculate nucleus (LGN) [20] and the visual cortex [21] as well as in many other sensory systems, are observed to precisely respond to stimuli on a millisecond timescale [22]. Temporal patterns can carry more information than rate-based patterns [23–25]. A simple example of the temporal encoding is spike latency coding. The capability of encoding information in the timing of single spikes to compute and learn realistic data is demonstrated in [26]. Since this coding utilizes only single spikes to transfer information, it could potentially be beneficial for efficient pulse-stream very large scale integration (VLSI) implementations.

Many algorithms for spiking neural networks (SNNs) have been proposed. Based on arithmetic calculations, the SpikeProp [9,26] was proposed for training SNNs, similar in concept to the back-propagation (BP) algorithm developed for traditional neural networks [27]. Others use bio-inspired algorithms, such as spike timing dependent plasticity (STDP) [28–31], the spike-driven synaptic plasticity [13], and the tempotron rule [14]. Although the arithmetic calculations can easily reveal why and how networks can be trained, the arithmetic-based rules are not a good choice building networks with a biological performance. STDP is found to be able to learn distinct patterns in an unsupervised way [12], and it characterizes synaptic changes solely in terms of the temporal contiguity of presynaptic spikes and postsynaptic potentials or spikes. In the spike-driven synaptic plasticity [13], a rate coding is used. The learning process is supervised and stochastic, in which a teacher signal steers the output neuron to a desired firing rate. Being different with spike-driven synaptic plasticity, the tempotron learning rule [14] is efficient to learn spiking patterns where information is embedded in precise timing spikes.

Although SNNs show promising capability in playing a similar performance as living brains due to their more faithful similarity to biological neural networks, the big challenge of dealing with SNNs is reading data into and out of them, which requires proper encoding and decoding methods [32]. Some existing SNNs for pattern recognition (as in [13,33]) based on the rate coding. Different from these SNNs, we focus more on the temporal coding which could potentially carry the same information efficiently using less number of spikes than the rate coding. This could largely facilitate the computing speed.

In this paper, we build a bio-inspired model of SNNs containing encoding, learning and readout. Neural coding and learning are the main considerations in this paper, since they are important components in cognitive memory system by processing the sensory inputs and distinguishing different patterns to allow for higher level brain functions such as memory storage and retrieval [34]. Inspired by the local receptive fields of biological neurons, the encoding neuron integrates information from its receptive field and represents the encoded information through precise timing of spikes. The timing scale of spikes is on a millisecond level which is consistent with biological experimental observations. The readout part uses a simple binary presentation as proposed in this paper to represent fired or non-fired state of the output neuron. Through the encoding and readout, SNNs can be applied to deal with real data well.

The main contribution of this paper lies in the approaches of designing SNNs for pattern recognition. Pattern recognition helps to identify and sort information for further processing in brain systems. A new coming pattern is recognized upon paying attention and similarity to previously learned patterns which are obtained through weight modification. Recognition memory is formed and stored in synaptic strengths. Inspired by biology, spiking neurons are employed for computation in this paper. This paper is extended from our preliminary work [35] by adding more comparative and analytic studies. The system contains encoding, learning and readout part. We demonstrate that, utilizing the temporal coding and learning, networks of spiking neurons can effectively and efficiently perform various classification tasks. In addition, the results also demonstrate that the temporal coding is a viable means for fast neural information processing and learning on real-world data.

The rest of this paper is organized as follows. Section 2 presents the architecture of the spiking neural network. Section 3 describes the temporal learning rule we used in our approaches. The relationship between this rule and well-studied STDP is also introduced. Section 4 shows the ability of the network to learn different patterns of neural activities (discrete-valued vectors). Section 5 shows the SNN for learning continuous input variables. We use the well-known Iris dataset problem to benchmark our approach against several existing methods. In Section 6, we demonstrate the ability of our spiking network for learning real-world stimuli (images). Finally, we end up with discussions in Section 7, followed by conclusions in the last section.

2. The spiking neural network

In this section, we describe the whole system architecture of spiking neurons for obtaining recognition memory. The system composes 3 functional parts: the encoding part, the learning part and the readout part (see Fig. 1). A stimulus consists of several components. The components are partially connected to encoding neurons to generate encoded spiking information. The encoding neurons are fully connected to learning neurons.

Each part plays a different functional role in the system: the encoding layer generates a set of specific activity patterns that represent various attributes of external stimuli; the learning layer tunes the neurons' weights making sure that particular neurons can respond to certain patterns correctly; the readout part extracts information about the stimulus from a given neural response. Through this architecture, the problem of getting data into and out of the spiking neural network is solved, and the task of pattern recognition could be fulfilled.

2.1. Encoding

The encoding part aims to generate spiking patterns that represent the input stimuli. The temporal encoding is used over rate-based encoding when patterns within the encoding window [17] provide information about the stimulus that cannot be obtained from spike count. The latency code [17] is a simple example of temporal encoding. It encodes information in the timing of response relative to the encoding window, which is usually defined with respect to stimulus onset. The single spike latencies are used to encode stimulus information in our system. Within the encoding window, each input neuron fires only once.

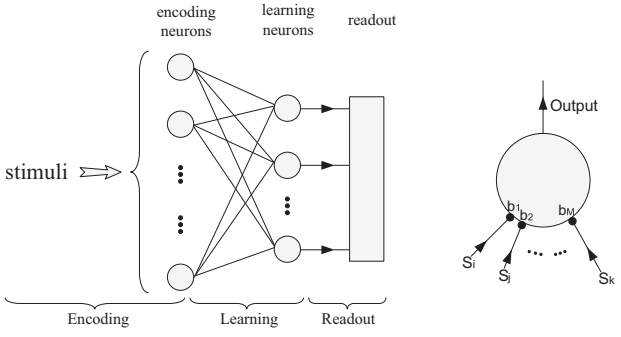

Fig. 1. Architecture for pattern recognition. Left: a schematic of the system architecture. Right: encoding neuron model. It has *M* input points connected to part of the stimulus and one output. It performs a mapping function that converts a value string to a temporal spike.

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

Each encoding neuron has M input points (Fig. 1) which are selected from components of the stimulus. It performs a specific function to convert the input points into latencies within the encoding window. For example, if the stimulus is composed of binary values (0 or 1), the function of the encoding neuron is to convert the binary strings into temporal patterns of discrete spikes. The encoding time window is chosen to be hundreds of milliseconds, consistent with biological observations.

2.2. Learning

The learning part of the network is composed of one layer of tempotrons [14]. The encoding neurons are fully connected to the learning neurons. The number of synapses to a learning neuron is equal to the number of encoding neurons (N_{en}) according to the structure. The tempotron can perform the classification task as long as the load is less than a critical value, approximately 3 [14]. That is to say, the maximum number of randomly generated patterns that a tempotron can learn is roughly 3 times the number of its synapses. Therefore, as long as the number of patterns does not exceed the critical load value, the network can perform the task well. If there are too many patterns, the number of encoding neurons should be increased correspondingly.

The tempotrons comprise the learning neurons. The neurons generate action potentials (or spikes), when the internal neuron state variable crosses a firing threshold value. Depending on the learning rule, the spiking neuron adapts its synaptic efficacies to react at a desired firing state when presented to incoming stimuli.

2.3. Readout

The readout part aims to extract information about the stimulus from responses of the learning neurons. In this part, we can use a binary sequence to represent a certain class of patterns for the reason that each learning neuron can only discriminate two groups. Each learning neuron responds to a stimulus by firing (1) or not firing (0). So, the total N learning neurons as the output can represent a maximum number of 2^N classes of patterns. The number of learning neurons is determined by the number of classes in the recognition task. For example, four readout is sufficient for a group of patterns containing 16 classes.

3. Temporal learning rule

Temporal learning rules aim to deal with information encoded by precise spike timing. One of the most commonly studied rules is spike-timing-dependent plasticity (STDP) which has emerged in recent years as experimentally most studied form of synaptic plasticity (see [28–31,36] for reviews). According to STDP, the plasticity depends on the intervals between pre- and postsynaptic spikes. The basic mechanisms of plasticity found in STDP are the long term potentiation (LTP) and the long term depression (LTD). However, STDP characterizes synaptic changes solely in terms of the temporal contiguity of presynaptic spikes and postsynaptic potentials or spikes. In addition, to get convergence of learning with STDP, a suitable balance of many parameters is needed [31]. In [14], the tempotron learning rule is presented. In this rule, the synaptic plasticity is governed by the temporal contiguity of presynaptic spike and postsynaptic depolarization, and a supervisory signal. The tempotron can make appropriate decision under the supervisory signal by tuning fewer parameters than STDP. Moreover, the tempotron rule also uses mechanisms of LTP and LTD to fulfill synaptic plasticity as in STDP. Because of the discrete nature of spikes, the evaluation of neural dynamics in our study is performed on a time step of $dt=1$ ms.

The neuron model used here is a leaky integrate-and-fire (LIF) neuron driven by exponential decaying synaptic currents generated by its synaptic afferents. The potential of the neuron is a weighted sum of postsynaptic potentials (PSPs) from all incoming spikes:

$$V(t)=\sum_i w_i \sum_{t_i} K(t-t_i)+V_{rest}. \tag{1}$$

Here w_i and t_i are the synaptic efficacy and the firing time of the ith afferent, respectively. V_{rest} is the rest potential of the neuron. K denotes a normalized PSP kernel:

$$K(t-t_i)=V_0\left(\exp\left(\frac{-(t-t_i)}{\tau_m}\right)-\exp\left(\frac{-(t-t_i)}{\tau_s}\right)\right). \tag{2}$$

Here τ_m and τ_s denote decay time constants of membrane integration and synaptic currents. We choose $\tau_m=4\tau_s=15$ ms in following sections. V_0 normalizes PSP so that the maximum value of the kernel is 1. $K(t-t_i)$ is a causal filter that only considers spikes $t_i \le t$. The kernel function is shown in Fig. 2. Each afferent spike will cause a change in the potential of postsynaptic neuron.

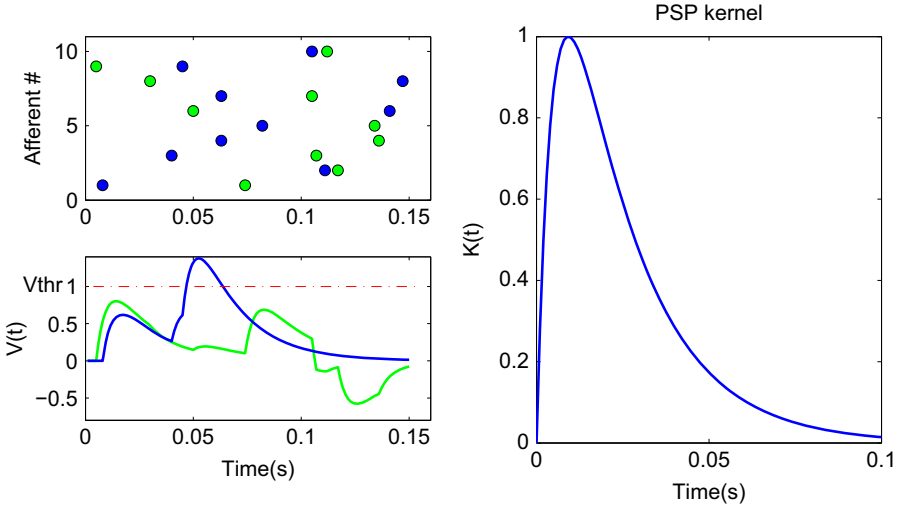

Fig. 2. Dynamic tempotron response. Left top: examples of spiking patterns. There are two patterns (blue and green) and each spike from an input afferent is denoted by a dot. The Y-axis is input identification number. Left bottom: neural potential traces. Each colour of lines corresponds to the same colour patterns on left top. In this neuron model, the potential boundaries at threshold and rest potential are ignored. Right: normalized PSP kernel. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

The height of the PSP is modulated by the synaptic efficacy w_i to get effective postsynaptic potential. The final potential of the postsynaptic neuron is a summation over all afferents.

A neuron is fired when $V(t)$ crosses the firing threshold, after which the potential smoothly decreases to V_{rest} by shunting down all the following spikes from input afferents; the spikes after firing time do not have effect on the postsynaptic voltage. Fig. 2 illustrates the dynamic response of neurons. The blue line indicates that a neuron has fired; the green line indicates a neuron has not fired. In this neuron model, the potential boundaries at threshold and rest potential are ignored.

In the classification task, each input pattern belongs to one of the two classes (which are labeled by P^+ and P^-). One neuron can discriminate these patterns by firing or not. When a P^+ pattern is presented to the neuron, it should fire a spike; when a P^- pattern is presented to the neuron, it should keep silent by not firing. The neuron learns patterns by changing its synaptic efficacies (w_i) whenever there is an error. If the neuron fails to fire in response to a P^+ pattern, this is denoted as a P^+ error. If the neuron erroneously fires a spike in response to a P^- pattern, it is denoted as a P^- error. Depending on the type of error, the learning rule is written as

$$\Delta w_i = \begin{cases} \lambda_+ \sum\limits_{t_i < t_{max}} K(t_{max} - t_i) & \text{if } P^+ \text{ error,} \\ -\lambda_- \sum\limits_{t_i < t_{max}} K(t_{max} - t_i) & \text{if } P^- \text{ error,} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Here t_{max} denotes the time at which the neuron reaches its maximum potential value in the time domain. $\lambda > 0$ is a constant representing the learning rate (we set $\lambda_+ = \lambda_- = \lambda = 0.005$ here). It denotes the maximum change on synaptic efficacies.

The tempotron updates its weights whenever it fails to respond as the same desired state as the instructor. It means that, within the presentation time of a pattern (T), the neuron will perform weight modification as long as its firing state violates the instructor. Such a method requires the supervisory signal to evaluate neuron's responding state at each time step. A trial updating method could also be adopted where the synaptic weights are modified at the end of each pattern presentation. This method only requires the instructor to make evaluation at the end of pattern presentation. Considering efficiency and biological realism, we adopt the dynamic updating method. Whenever an error occurs, the neuron will immediately update its weights. Each spike firing prior to t_{max} will result in a change on the corresponding synapse. The shape of learning window follows kernel K and the changing amount of the weight depends on the time difference between t_i and t_{max}. If we only consider single spike coding and the latest spike updating, the learning rule will be simplified as

$$\Delta w_i = \begin{cases} \lambda_+ K(t_{max} - t_i)\Theta(t_{max} - t_i) & \text{if } P^+ \text{ error,} \\ -\lambda_- K(t_{max} - t_i)\Theta(t_{max} - t_i) & \text{if } P^- \text{ error,} \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

where $\Theta(x)$ is a heaviside function. This simplified rule is used in the rest of this paper.

From a biological perspective, a training algorithm should adapt synaptic weights basing on the states of pre and postsynaptic neurons to keep with Hebbian theory [37]. Normally, a longer time difference will result in a little weight change while a shorter time difference results in a larger change, as like processes in biological systems. In this learning rule, two STDP-like windows are used to adjust the synaptic weights (see Fig. 3). The LTP window is used to increase the synaptic weights and the LTD window is for depressing the weights, whenever the input fails to result in a desired output (teaching signal).

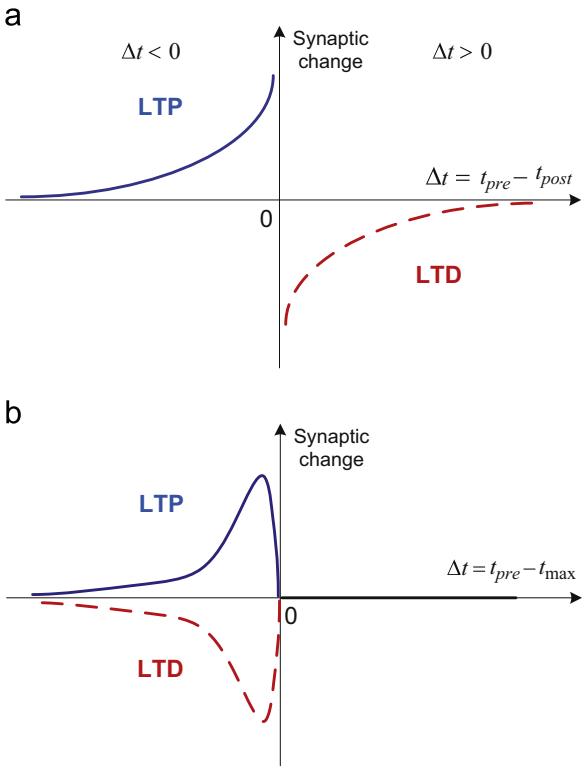

Fig. 3. Learning windows of different rules. The blue lines denote the LTP process and the dashed red ones denote the LTD process. (a) The learning window for STDP; (b) tempotron rule. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

The learning process uses a supervisory signal. Although so far there is no strong experimental confirmation of the supervisory signal, an increasing body of evidence shows that this kind of learning is also exploited by the brain [38]. The most documented evidence for this type of rule comes from studies on the cerebellum and the cerebellar cortex [39,40]. In addition, there is evidence that the supervisory signals are provided to the learning modules by sensory feedback [41] or other supervisory neural structures in the brain [40]. In the tempotron, the supervisory signal is only for determining the polarity of synaptic changes. Classical error feedback is a possible way to implement this control of polarity. A neuromodulator released by the supervisory system can induce the control of adaption. This control occurs for several neuromodulatory pathways, such as dopamine and acetylcholine [14,42,43]. In addition, the gating role of the supervisory signal has strong biological resonance such as voltage-gated calcium channels and NMDA receptors. Their involvements in the induction of long term plasticity is well established [44,45].

4. Learning patterns of neural activities

Many ways of encoding memory patterns in neural networks have been studied. The memory patterns encoded in synaptic weights can be taken to be binary vectors, as well as they can also be taken to be drawn from a distribution with several discrete activity values or from a continuous distribution [34]. In Hopfield network [46], memory patterns are expressed through the activities of neurons, where the states of the neurons have binary values ($+1$ for active neuron and -1 for inactive neuron). In some other networks, non-binary coding schemes [47] are also introduced.

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

In the previous section, the ability of tempotron to separate temporal patterns is introduced. It can classify a number of spatiotemporal patterns due to the learning rule. However, the following questions arise: can this method be used to recognize memory patterns mentioned above in this section? If so, how can it perform the task?

The patterns are n-dimensional vectors and the value of each element in the vector refers to neuron's activity which can be drawn from several discrete values. The coding schemes used here are the same as that in Treves and Rolls [48]. The activity η of each neuron follows a probability distribution function $p(\eta)$:

$$p(\eta)=\begin{cases}(1-c)\delta(\eta-\eta_0)+c\delta(\eta-\eta_1) & \text{(binary)}\\ \left(1-\frac{4c}{3}\right)\delta(\eta-\eta_0)+c\delta(\eta-\eta_1) & \\ +\frac{c}{3}\delta(\eta-\eta_2) & \text{(ternary)}\\ (1-2c)\delta(\eta)+4ce^{-2\eta} & \text{(exponential)}\end{cases} \quad (5)$$

where $\delta(x)$ is the Dirac's function:

$$\delta(x)=\begin{cases}1 & \text{if } x=0\\ 0 & \text{otherwise}\end{cases}$$

and c is the coding level which is defined as the mean level of activity of the network [34,48].

As explorations for the ability of tempotron to classify different patterns of activities, we use binary and ternary patterns as stimuli. The ternary patterns represent a simple non-binary structure. We also use variable coding levels to see the performance. The pattern vectors are generated according to Eq. (5). The activity values we choose for binary patterns are $\eta_0=0$ and $\eta_1=1$, and for ternary patterns are $\eta_0=0$, $\eta_1=1$ and $\eta_2=2$. Some examples of binary and ternary patterns are shown in Fig. 4.

Pattern is stored in an n-dimensional vector with discrete values of activity. We use the system architecture of spiking neurons to classify pattern vectors (see Fig. 1). The layer of encoding neurons performs a function converting the pattern vector into temporal pattern for tempotron to classify. We only use one learning neuron to test the ability of tempotron learning two groups of patterns.

To test the performance, we generate 100 memory patterns with 1024 elements, and assign half of patterns to a same group and others to another group. We also use different coding levels ($c=0.2$ and $c=0.5$) in our simulation. We set the number of input points of the encoding neuron to 8 and 5 for binary and ternary patterns, respectively. Each element of the pattern vector is connected to only one encoding neuron and the connections between the pattern vector and encoding neurons are in order. For example, in binary patterns, the first 8 elements connect to the first encoding neuron and the second 8 elements connect to the second and the last 8 connect to the last encoding neuron. The encoding neuron in this case acts as a converter that translates a binary or ternary string into a spike timing.

From Fig. 5, we can see that the tempotron can successfully learn different patterns of activities that are presented in discrete values. After several iterations of learning, the neuron can correctly classify discrete-valued patterns under different coding levels. Through this approach, we successfully investigate a method for spiking neurons to perform classification on discrete-valued patterns.

5. Learning patterns of continuous input variables

In this section, we conduct experiments with our spiking neural network on classifying patterns with continuous variables. We use the Iris dataset to benchmark our approach against several existing methods.

5.1. Encoding continuous variables into spike times

To encode the continuous variables into spike times on a precision of millisecond level, we employ a similar approach as in [9] based on arrays of receptive fields. As a result, each input variable is represented by a group of neurons with graded and overlapping sensitivity profiles. This approach is a biologically plausible and well studied method for representing real-valued parameters [9,49].

In our experiments, each input dimension is encoded by an array of one-dimensional Gaussian receptive fields. For a variable x in a range $[x_{min}, x_{max}]$, n neurons with different Gaussian receptive fields are used to encode. The center and width of the ith neuron are set to $\mu_i=x_{min}+(2\cdot i-3)/2\cdot(x_{max}-x_{min})/(n-2)$ and $\sigma_i=1/1.5\cdot(x_{max}-x_{min})/(n-2)$, respectively. The activation values of the n neurons encoding the variable x are calculated. Highly activated

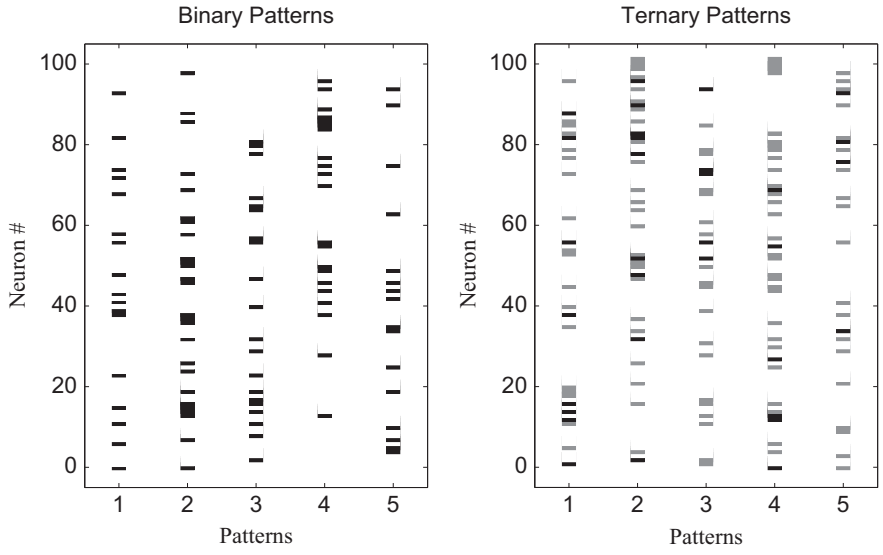

Fig. 4. Examples of binary and ternary patterns with c=0.2. The neural activities are shown in gray scale (the maximal activity value is shown in black, and the minimal activity value is in white). There are 5 patterns in each sub-figure and only the activities of 100 neurons are included.

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

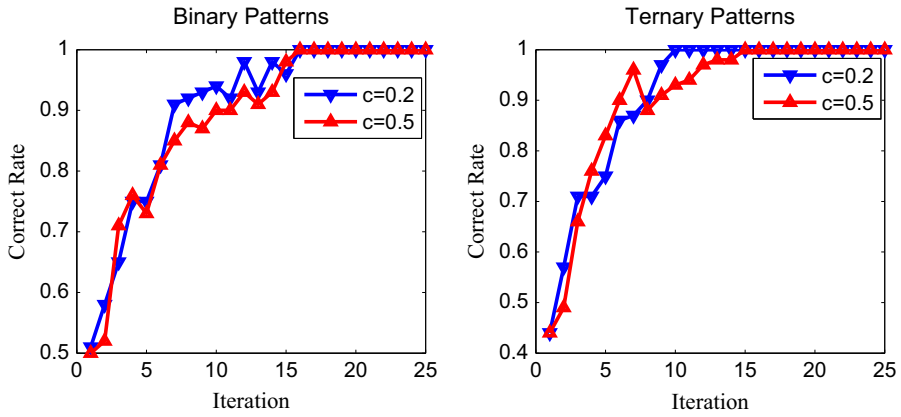

Fig. 5. Results of classification for different patterns of activities. The pattern is 1024-dimensional vector. The total number of patterns is 100 (each class has 50) in each simulation.

neurons will fire early and less activated neurons will fire later or not fire.

Through this temporal encoding approach, two important properties are obtained. Firstly, a sparse coding, allowing for efficient simulation as in [50], is achieved through a small set of significantly activated neurons. Secondly, an optimal number of neurons could be roughly obtained for each independently encoded variable.

5.2. Experiments on the Iris dataset

The three-class Iris dataset is used to benchmark our approach since it is perhaps one of the best known databases to be found in the recognition literature. The different three classes represent the different species of the Iris plant, including Iris Setosa Canadenisis (*Class* 1), Iris Veriscolor (*Class* 2) and Iris Virginica (*Class* 3). The dataset contains 150 samples, 50 for each class. Each sample has 4 input variables: sepal length, sepal width, petal length and petal width. The latter two classes are not linearly separable from each other.

To encode these data, we firstly normalize the 4 variables into a same range. Each input variable is encoded by $n=12$ neurons with Gaussian receptive fields. For each input pattern, 48 activation values between 0 and 1 can be calculated. We ignore activation values below 0.1 since they are too weak to stimulate a spike. These activation values are then linearly converted to delay times, associating $t=0$ with activation value 1 and later times up to $t=100$ ms with lower activation values. The spike times are rounded to $dt=1$ ms precision. The dataset is split into two sets and classified using two-fold cross-validation.

Before the multi-class problem, it is necessary to investigate the performance of a single spiking neuron to classify two classes. Through this process, a proper stopping criteria for the purpose of training the network could be chosen. We simulate one spiking neuron to separate *Class* 2 from the other classes. We set the maximum number of iterations for training to be $Max_{iter}=200$. According to Fig. 6, the neuron can rapidly reach a high accuracy (0.95) within tens of training iterations, and it stabilizes at high accuracy for further training. For balancing between a high simulation speed and a high accuracy, it is reasonable to choose a lower Max_{iter} according to Fig. 6. We set $Max_{iter}=100$ for the multi-class problem. After each training period, the neuron will fall into two cases. We refer *Case* 1 as that the neuron successfully separates all samples in the training set before Max_{iter} reaches, and *Case* 2 as that the neuron still cannot separate all samples at the end of maximum number of training iterations.

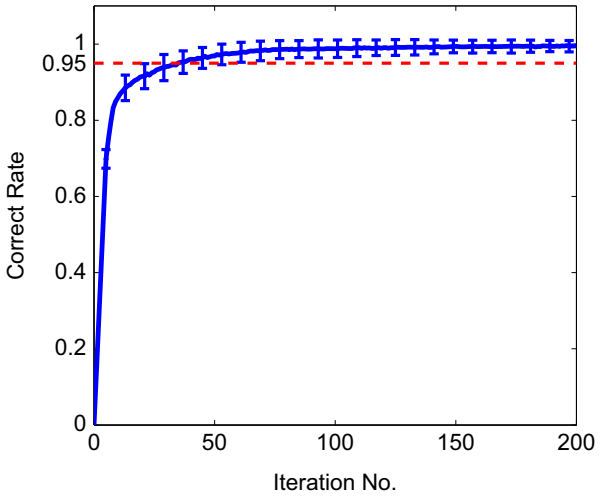

Fig. 6. Correct rate of classifying *Class* 2 from the other classes vs. iterations for training. The plot is averaged over 100 runs.

For the three-class Iris problem, we employ only one neuron per output class, and the output neuron with the strongest activation state represents the class association. The training is stopped either when the Max_{iter} reaches or when the neuron successfully separates all training samples before Max_{iter} reaches. After 100 runs of training and testing, the averaged classification accuracy for the training set is 96.63% and for testing set is 92.55%. Among the 100 runs of training, we find that there are 66 runs in which all the three neurons are trained in *Case* 1, and 34 runs where at less one neuron is trained in *Case* 2. We refer these two situations as Tem*Case* 1 and Tem*Case* 2 respectively. Interestingly, in the case of Tem*Case* 1, the classification accuracy for both the training set and testing set is improved, reaching 100% and 93.09%, respectively.

Table 1 presents the results of our approach against several existing algorithms for the Iris dataset. MatlabBP and MatlabLM, representing traditional artificial neural network, are build-in functions of Matlab that implement the backpropagation and Levenberg–Marquardt training algorithms. SpikeProp [3,9] and SWAT [10], as spiking neural networks applied on Iris dataset, are also used to benchmark our approach. The SpikeProp, as the first supervised training algorithm for SNNs, was an adaptation of gradient-descent-based-error-backpropagation method [9]. An efficient SpikeProp for Iris classification was presented in [3], where less synaptic weights were used.

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

Table 1
Comparison of training algorithm: results for Iris dataset.

Algorithm	Inputs	Hidden	Outputs	Iterations	Training set	Test set
MatlabBP	50	10	3	2.6×10^6	$98.2\% \pm 0.9$	$95.5\% \pm 2.0$
MatlabLM	50	10	3	3750	$99.0\% \pm 0.1$	$95.7\% \pm 0.1$
SpikeProp [9]	50	10	3	1000	$97.4\% \pm 0.1$	$96.1\% \pm 0.1$
SpikeProp [3]	17	8	3	37	$\geq 95\%$	92.7%
SWAT	16	208	3	500	$95.5\% \pm 0.6$	$95.3\% \pm 3.6$
Tem	48	–	3	Less than 100	$99.63\% \pm 0.81$	$92.55\% \pm 3.3$
Tem*Case* 1	48	–	3	Less than 100	100%	$93.09\% \pm 2.94$
Tem*Case* 2	48	–	3	Less than 100	$98.9\% \pm 1.06$	$91.49\% \pm 3.74$

According to Table 1, it shows that the training accuracy of our approach slightly surpasses other approaches, and the testing accuracy is comparable and acceptable. Some of the state-of-the-art approaches such as [51,52] that come from hybrid-system approach, can even result in a higher accuracy (normally over 96%). However, it is extremely time consuming to train if genetic algorithms are used in the hybrid system. Although the classification accuracy of our approach does not beat other approaches at this moment, the ability to use a biologically plausible SNN to do the task is highlighted. With a comparable and acceptable classification accuracy, our approach is more efficient and effective than other methods as listed in Table 1. It can perform the task comparably well with less neurons (without a layer of hidden neurons) and with less number of learning iterations (within 100). This preliminary approach with biologically plausible SNN demonstrates the great computational power inherited from biology. Continued improvements on this approach could be explored to perform better than conventional machine-learning algorithms.

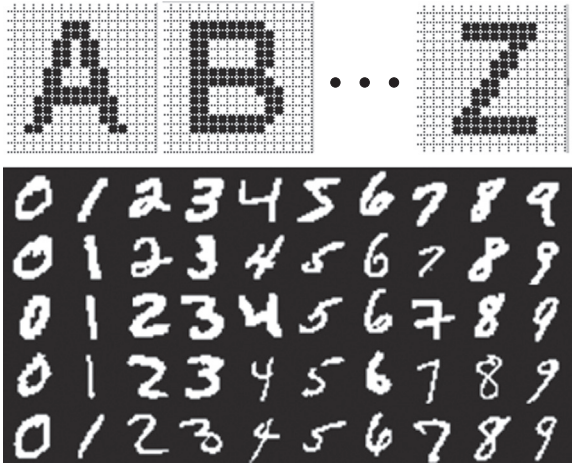

Fig. 7. Examples of data sets. Top: alphabetic letters. Bottom: handwritten digits from MNIST database.

6. Learning real-world stimuli

From the previous sections, we show the SNN has the ability to learn different patterns of activities and continuous input variables. It can separate different vector patterns successfully. To move forward a step, we apply the tempotron to learn some real-world stimuli (images).

In this section, we show the system architecture to perform visual pattern recognition. The image is composed of n pixels. The pixels are partially connected to encoding neurons to generate spiking information.

We implement the learning procedure on two kinds of data sets: the handwritten digits from MNIST database and the 26 alphabetic letters. Both of the data sets are composed of binary images. We use these sets to test the performance of our network on real-world stimuli.

6.1. The data sets and the classification problem

The stimuli from real world typically have a complex statistical structure. It is quite different from idealized case of random patterns often considered. In the real world, the stimuli hold large variability in a given class and have a high level of correlation between members of different classes. There are two data sets we consider here: the MNIST digits and the alphabetic letters (see Fig. 7).

The MNIST set consists of ten classes (digits 0–9) and each example is on a grid of 28×28 pixels. (The MNIST set is available from http://yann.lecun.com/exdb/mnist in which many classification results from different methods are listed). To get the input data from the database, each image with grayscale values is converted to a binary vector. Each element of the 784-element vector of an image is set to 1 if its grayscale value is above 128, otherwise set to 0. The MNIST data set provides a good benchmark for our network performance on classifying images.

The great number of images in the MNIST set are written by different people. As real-world stimuli, images are variable in the same class and highly correlated with other images in different classes. On one side, images from the same class differ from each other more or less; on the other side, images from different classes share some similarities with each other. These properties determine that the real-world stimuli are quite difficult for learning compared with randomly generated patterns in theoretical studies.

The alphabetic letters set is the second data set we consider. There are 26 classes (from A to Z), with one example in each class. This set is a binary representation of 26 alphabetic letters. Each letter is on a grid of 16×16 pixels. Although this set is small, there exist large overlaps between different classes (like "C", "O" and "Q"). This data set is convenient for investigating the performance of the network on real-world pattern recognition.

6.2. Encoding

The encoding focuses on converting images into representations in the form of spikes. A good encoding method should retain adequate information of the original images and facilitate the later learning process.

The input images are 2-dimensional binary pictures. They compose of a number of pixels. Each pixel is either value 0 or 1. The output values of the encoding part are encoded by pulses in the encoding window. The pixels to the encoding neurons are partially connected. Fig. 1 shows the encoding neuron model. It acts as a mapping function that converts a binary string to a decimal value. The number of input points defines the time domain of the encoding window. For example, if there are N input

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

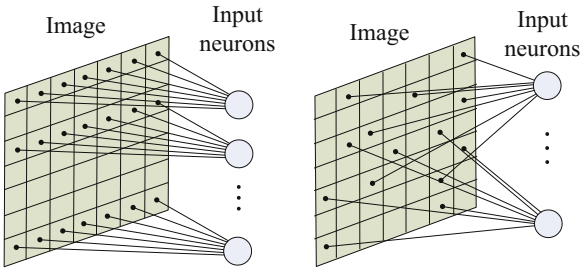

Fig. 8. A schematic of encoding. Left: successive encoding; right: random encoding.

points, the encoding window length is 2^N ms. However, the encoding window could also be flexible by scaling up or down.

The encoding neuron has M input points (Fig. 1) which are selected from the pixels pool. There are two kinds of selecting methods that we considered: the successive one and the random one (see Fig. 8). The successive encoding chooses input pixels from an image in a row-wise manner. The first M pixels for the first encoding neuron, and the next M pixels for the second encoding neuron, and so on. The random encoding chooses input pixels for each encoding neuron in a random manner. After selection, the input points of an encoding neuron combine a binary string which could be used to calculate the output pulse time by converting it to a decimal value. The spiking time is on a millisecond time scale. So, the encoding neurons convert the images to latency patterns in which each neuron only fire once in the time domain.

To simply compare these two encoding ways, we perform the recognition task on the alphabetic data set. We set the number of input points for each encoding neuron to 8. Thus, the encoding window length of the pattern is 256 ms. Fig. 9 illustrates the encoding results of the letter "A". Each dot and cross denote a spike from the ith afferent at time t. Through simulation for recognition, we find that the random selection method is better than the other one both in convergent speed and in correct rate of classification under the same condition (see Fig. 9). This is because many spikes fire at the same moment in the successive encoding, while in random encoding the spikes distribute randomly in the time domain. The distributed spikes make full use of the whole encoding window, which results in a better recognition of the patterns.

6.3. Recognition results

If there are too many patterns that exceed the load of the system, the number of encoding neurons should be increased by randomly selecting 8 pixels from the image for a new encoding neuron.

To see the performance ability of the network on recognition task, we use a small data set from the MNIST. After several iterations of learning, the network can recognize all patterns in this set. Here, we take the recognition results of several digits as an example (Fig. 10). If the potential of the learning neuron crosses the threshold, which is said to fire, the value of this neuron is considered as 1, otherwise it is 0. According to Fig. 10, when image "1" shows up to the network, only neuron 4 fires, so the result is $[0001]_{bin}$. For image "5", the result is $[0101]_{bin}$, and for "8" it's $[1000]_{bin}$. This shows that the spiking neural network we used could successfully perform the recognition task.

7. Discussion

In this section, we discuss several considerations regarding biological relevance that benefit the procession in our approach, and present the future directions.

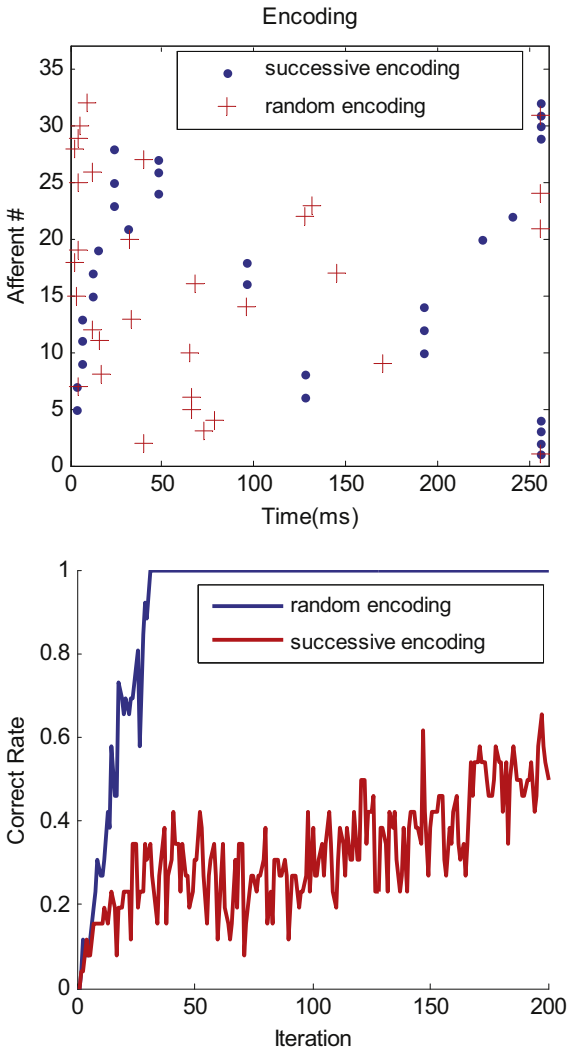

Fig. 9. Comparison between the two encoding ways. The top one is encoding examples of image "A". Each dot and cross denote a spike from the ith afferent at time t. The bottom one is the correct rate of recognition for alphabetic letters during learning.

7.1. Benefits from biological plausibility

The computational power of biological neurons attracts the community to develop models for computation on action potentials. The average firing rate of neuron is generally assumed to be the coding scheme, with the success in neural network modeling and the substantial electrophysiological support. However, there has been increasing number of reports showing that precise timing of action potentials (spikes) carries significant information (e.g., in the retina [18,19], the lateral geniculate nucleus (LGN) [20], and the visual cortex [21]). In addition, it has been demonstrated that the temporal coding with precise timings can carry more information than the rate coding scheme [23–25]. The usage of time-to-first-spike coding facilitates the computational speed in SNNs. For machine learning purposes, efficient implementations of SNNs can be obtained by the event nature of spikes.

The encoding scheme used in this paper is inspired from receptive fields of biological neurons. Each neuron receives a partial information from external stimuli. The encoding window of the temporal patterns is chosen to be on a scale of hundreds of milliseconds, which matches the biological evidence [17,22,24]. Although the encoding window could be flexible by scaling up or down, a choice from biology could make this approach consistent

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

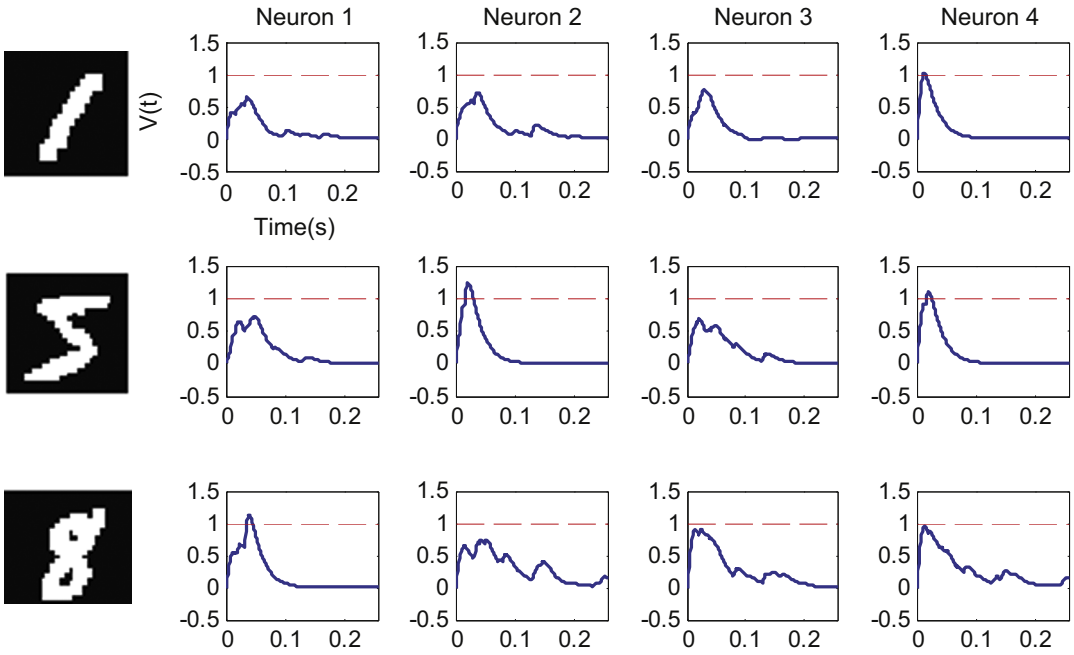

Fig. 10. Recognition results of digits. These show 4 learning neurons (neurons 1–4) and 3 images. The neuron responds to an image by firing (1) or not (0). The results for "1", "5" and "8" are $[0001]_{bin}$, $[0101]_{bin}$ and $[1000]_{bin}$, respectively.

and compatible with other bio-inspired models in the case of combination. In addition, the random encoding plays an important role in classification. Since the real-world stimuli are variable in a same class and highly correlated with patterns in different classes, random encoding focuses on finding the difference by making full use of the encoding window.

In our encoding schemes, the sparse activity is achieved by neurons collecting information from different receptive fields. The sparse output, as is observed in biological agents [53], allows an efficient computation. Relevant to biological observations, the encoding neuron integrates information from its receptive field. Besides the encoding, in the learning for spiking neurons, a biological-like learning window and an injected supervisory teacher signal are used. All these result in an efficient and effective approach for pattern recognition.

Besides biological plausibility, another benefit of a biologically inspired spiking system would be that it offers the possibility of real-time learning systems. Biological neural networks need to respond in real time to real-world stimuli. The needs for fast reactive systems normally shadow classical computing approaches which have mostly focussed on off-line problems. Thus, the responding speed is another reason for the choice of spiking neural networks.

7.2. Future work

In this paper, we have explored the approach of spiking neural network to perform pattern recognition on different tasks. Through a proper choice of encoding scheme, an efficient and effective approach is achieved. Using the SNN architecture, recognition on real-world stimuli is investigated. However, the task of digital images we chose here might be simple compared to majority of complex real-world patterns. In future, we will investigate the SNN for processing more complex visual patterns and auditory patterns.

One possible expansion of the visual processing part would be to explore the role of different receptive fields. The successive and random encoding methods are examined in this paper. The successive encoding would be expected to perform poorly because it is more or less expecting to find features orientated along the scan axis. A random approach could randomly explore possible features. Another possible approach is to use more-biological-plausible model of contrast- or direction-selective receptive fields as applied in [15,54]. As future direction, aiming to improve the use of biologically realistic neural networks for pattern recognition, it is important to investigate a proper encoding scheme. Through layers of encoding neurons, the external stimuli are sparsely and robustly represented by output spikes of neurons. Additionally, research into optimal parameters and training schemes will be another logical step for future investigations. As is shown in Section 4, by implementing a proper 'stop learning' criteria, the performance of the network might be improved.

Another possible future direction is to combine our model with other compatible spiking neural networks such as Liquid State Machine (LSM) [55]. LSM is an implementation of a broader concept known as reservoir computing [55–57], and it acts as a set of filters projecting low-dimensional temporal input into a series of high-dimensional states. These states are normally classified using a variety of conventional methods, such as linear classifier, perceptron, and support vector machine. By replacing these conventional classifiers as the output layer, our approach combined with LSM would make a more biologically plausible approach, and hopefully result in better performance.

8. Conclusion

This paper presents an architecture of spiking neurons to approach pattern recognition on various classification tasks such as recognition of neural activities, continuous input variables and real-world stimuli. The recognition memory is formed through weights modification during learning process. A new pattern could be recognized through matching what the network has learnt. Since the temporal encoding and learning in this paper are believed to be inherited from the biological neural systems, the biological plausibility of the approach is a main aspect in this study considering machine learning as a main target. The whole system contains encoding, learning and readout. Utilizing the temporal encoding and learning, an effective and efficient

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

approach for recognition through spiking neural network is obtained. Our approach is benchmarked using the Iris dataset problem, and the results highlight the capability of our approach to classify nonlinearly separable data effectively and efficiently. It demonstrates that our approach results in a good generalization. The classification accuracy for this data set is 99.63% for training and 92.55% for testing. In addition, the trained networks demonstrate that the temporal coding is a viable means for fast neural information processing.

Acknowledgments

This work was supported by Agency for Science, Technology, and Research (A*STAR), Singapore under SERC Grant 092 157 0130.

References

[1] H. Adeli, S.L. Hung, Machine learning – neural networks, genetic algorithms, and fuzzy sets, John Wiley and Sons, New York, 1995.
[2] W. Maass, Lower bounds for the computational power of networks of spiking neurons, Neural Comput. 8 (1) (1996) 1–40.
[3] S. Ghosh-Dastidar, H. Adeli, Improved spiking neural networks for eeg classification and epilepsy and seizure detection, Integr. Comput.-Aided Eng. 14 (3) (2007) 187–212.
[4] W. Gerstner, W.M. Kistler, Spiking Neuron Models: Single Neurons, Populations, Plasticity, Cambridge University Press, Cambridge, 2002.
[5] E.M. Izhikevich, Resonate-and-fire neurons, Neural Netw. 14 (6-7) (2001) 883–894.
[6] A. Hodgkin, A. Huxley, A quantitative description of membrane current and its application to conduction and excitation in nerve, J. Physiol. 117 (1952) 500–544.
[7] E.M. Izhikevich, Simple model of spiking neurons, IEEE Trans. Neural Netw 14 (6) (2003) 1569–1572.
[8] E.M. Izhikevich, Which model to use for cortical spiking neurons? IEEE Trans. Neural Netw. 15 (5) (2004) 1063–1070.
[9] S.M. Bohte, J.N. Kok, J.A.L. Poutré, Error-backpropagation in temporally encoded networks of spiking neurons, Neurocomputing 48 (1–4) (2002) 17–37.
[10] J.J. Wade, L.J. McDaid, J.A. Santos, H.M. Sayers, SWAT: a spiking neural network training algorithm for classification problems, IEEE Trans. Neural Netw. 21 (11) (2010) 1817–1830.
[11] F. Ponulak, A.J. Kasinski, Supervised learning in spiking neural networks with resume: sequence learning, classification, and spike shifting, Neural Comput. 22 (2) (2010) 467–510.
[12] T. Masquelier, R. Guyonneau, S.J. Thorpe, Competitive stdp-based spike pattern learning, Neural Comput. 21 (5) (2009) 1259–1276.
[13] J.M. Brader, W. Senn, S. Fusi, Learning real-world stimuli in a neural network with spike-driven synaptic dynamics, Neural Comput. 19 (11) (2007) 2881–2912.
[14] R. Gütig, H. Sompolinsky, The tempotron: a neuron that learns spike timing-based decisions, Nat. Neurosci. 9 (3) (2006) 420–428.
[15] S.G. Wysoski, L. Benuskova, N. Kasabov, Fast and adaptive network of spiking neurons for multi-view visual pattern recognition, Neurocomputing 71 (13–15) (2008) 2563–2575.
[16] X. Pan, M. Tsukada, The spatiotemporal learning rule and its efficiency in separating spatiotemporal patterns, Biol. Cybern. 92 (2005) 139–146.
[17] S. Panzeri, N. Brunel, N.K. Logothetis, C. Kayser, Sensory neural codes using multiplexed temporal scales, Trends Neurosci. 33 (3) (2010) 111–120.
[18] V.J. Uzzell, E.J. Chichilnisky, Precision of spike trains in primate retinal ganglion cells, J. Neurophysiol. 92 (2) (2004) 780–789.
[19] T. Gollisch, M. Meister, Rapid neural coding in the retina with relative spike latencies, Science 319 (5866) (2008) 1108–1111.
[20] P. Reinagel, R.C. Reid, Temporal coding of visual information in the thalamus, J. Neurosci. 20 (14) (2000) 5392–5400.
[21] W. Bair, C. Koch, Temporal precision of spike trains in extrastriate cortex of the behaving macaque monkey, Neural Comput. 8 (6) (1996) 1185–1202.
[22] D.A. Butts, C. Weng, J. Jin, C.-I. Yeh, N.A. Lesica, J.-M. Alonso, G.B. Stanley, Temporal precision in the neural code and the timescales of natural vision, Nature 449 (7158) (2007) 92–95.
[23] R. Kempter, W. Gerstner, J. L. van Hemmen, Spike-Based Compared to Rate-Based Hebbian Learning, in: Advances in Neural Information Processing Systems 11, MIT-Press, 1999, pp. 125–131.
[24] A. Borst, F.E. Theunissen, Information theory and neural coding, Nat. Neurosci. 2 (11) (1999) 947–957.
[25] J.J. Hopfield, Pattern recognition computation using action potential timing for stimulus representation, Nature 376 (6535) (1995) 33–36.
[26] S.M. Bohte, E.M. Bohte, H.L. Poutr, J.N. Kok, Unsupervised clustering with spiking neurons by sparse temporal coding and multi-layer RBF networks, IEEE Trans. Neural Netw. 13 (2002) 426–435.
[27] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning Internal Representations by Error Propagation, 1986, pp. 318–362.
[28] R. Kempter, W. Gerstner, J.L. van Hemmen, Hebbian learning and spiking neurons, Phys. Rev. E 59 (4) (1999) 4498–4514.
[29] S. Song, K.D. Miller, L.F. Abbott, Competitive Hebbian learning through spike-timing-dependent synaptic plasticity, Nat. Neurosci. 3 (2000) 919–926.
[30] G.Q. Bi, M.M. Poo, Synaptic modification by correlated activity: Hebb's postulate revisited, Annu. Rev. Neurosci. 24 (2001) 139–166.
[31] R. Legenstein, C. Naeger, W. Maass, What can a neuron learn with spike-timing-dependent plasticity? Neural Comput. 17 (2005) 2337–2382.
[32] C. Johnson, G.K. Venayagamoorthy, Encoding real values into polychronous spiking networks, in: IJCNN, 2010, pp. 1–7.
[33] S. Mitra, S. Fusi, G. Indiveri, Real-time classification of complex patterns using spike-based learning in neuromorphic, VLSI IEEE Transactions on Biomedical Circuits and Systems 3 (1) (2008) 32–42.
[34] H. Tang, H. Li, R. Yan, Memory dynamics in attractor networks with saliency weights, Neural Comput. 22 (7) (2010) 1899–1926.
[35] Q. Yu, K. Tan, H. Tang, Pattern recognition computation in a spiking neural network with temporal encoding and learning, in: 2012 International Joint Conference on Neural Networks (IJCNN), IEEE, 2012, pp. 1–7.
[36] R.C. Froemke, Y. Dan, Spike-timing-dependent synaptic modification induced by natural spike trains, Nature 416 (6879) (2002) 433–438.
[37] D. Hebb, The Organization of Behavior: A Neuropsychological Theory, Taylor and Francis Group, London, 2002.
[38] E.I. Knudsen, Supervised learning in the brain, J. Neurosci. 14 (7) (1994) 3985–3997.
[39] W.T. Thach, On the specific role of the cerebellum in motor learning and cognition: clues from PET activation and lesion studies in man, Behav. Brain Sci. 19 (3) (1996) 411–431.
[40] M. Ito, Mechanisms of motor learning in the cerebellum, Brain Res. 886 (1–2) (2000) 237–245.
[41] M.R. Carey, J.F. Medina, S.G. Lisberger, Instructive signals for motor learning from visual cortical area MT, Nat. Neurosci. 8 (6) (2005) 813–9+.
[42] R.C. Foehring, N.M. Lorenzon, Neuromodulation, development and synaptic plasticity, Can. J. Exp. Psychol./Rev. can. psychol. exp. 53 (1) (1999) 45.
[43] J.K. Seamans, C.R. Yang, et al., The principal features and mechanisms of dopamine modulation in the prefrontal cortex, Prog. Neurobiol. 74 (1) (2004) 1.
[44] M. Randic, M. Jiang, R. Cerne, Long-term potentiation and long-term depression of primary afferent neurotransmission in the rat spinal cord, J. Neurosci. 13 (12) (1993) 5228–5241.
[45] C. Hansel, A. Artola, W. Singer, Relation between dendritic Ca^{2+} levels and the polarity of synaptic long-term modifications in rat visual cortex neurons, Eur. J. Neurosci. 9 (11) (2006) 2309–2322.
[46] J.J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proc. Natl. Acad. Sci. 79 (8) (1982) 2554–2558.
[47] A. Treves, E.T. Rolls, What determines the capacity of autoassociative memories in the brain? Netw. – Comput. Neural Syst. 2 (1991) 371–398.
[48] A. Treves, Graded-response neurons and information encoding in autoassociative memory, Phys. Rev. A 42 (4) (1990) 2418–2430.
[49] C.W. Eurich, S.D. Wilke, Multi-dimensional encoding strategy of spiking neurons, Neural Comput. 12 (2000) 1519–1529.
[50] A. Delorme, J. Gautrais, R. van Rullen, S. Thorpe, SpikeNET: a simulator for modeling large networks of integrate and fire neurons, Neurocomputing 24 (1999) 26–27.
[51] K.C. Tan, E.J. Teoh, Q. Yu, K.C. Goh, A hybrid evolutionary algorithm for attribute selection in data mining, Expert Syst. Appl. 36 (4) (2009) 8616–8630.
[52] M. Fallahnezhad, M.H. Moradi, S. Zaferanlouei, A hybrid higher order neural classifier for handling classification problems, Expert Syst. Appl. 38 (1) (2011) 386–393.
[53] B.A. Olshausen, D.J. Field, Sparse coding with an overcomplete basis set: a strategy employed by V1? Vis. Res. 37 (23) (1997) 3311–3325.
[54] H. Tang, Q. Yu, K.C. Tan, Learning real-world stimuli by single-spike coding and tempotron rule, in: IJCNN'12, 2012, pp. 1–6.
[55] W. Maass, T. Natschläger, H. Markram, Real-time computing without stable states: a new framework for neural computation based on perturbations, Neural Comput. 14 (11) (2002) 2531–2560.
[56] H. Jaeger, The "echo state" approach to analysing and training recurrent neural networks, GMD Report 148, GMD – German National Research Institute for Computer Science, 2001.
[57] M. Lukoševičius, H. Jaeger, Reservoir computing approaches to recurrent neural network training, Comput. Sci. Rev. 3 (3) (2009) 127–149.

Qiang Yu received his B.Eng. degree in Electrical Engineering and Automation from Harbin Institute of Technology, Harbin, China, in 2010. He is a Ph.D. student in Department of Electrical and Computer Engineering, National University of Singapore. His research interests include learning algorithms in spiking neural networks, neural encoding and cognitive computation. He is currently working on neural circuit theories for cognitive computing.

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052

Huajin Tang received the B.Eng. degree from Zhejiang University, Hangzhou, China, the M.Eng. degree from Shanghai Jiao Tong University, Shanghai, China, and the Ph.D. degree in Electrical and Computer Engineering from the National University of Singapore, Singapore, in 1998, 2001, and 2005, respectively. He was a System Engineer with STMicroelectronics, Singapore, from 2004 to 2006. From 2006 to 2008, he was a Postdoctoral Fellow with Queensland Brain Institute, University of Queensland, Australia. He is currently a Research Scientist and leading the cognitive computing group at the Institute for Infocomm Research, Singapore. He has published one monograph (Springer-Verlag, 2007) and over 20 international journal papers. His current research interests include neural computation, machine learning, neuromorphic cognitive systems, and neuro-cognitive robots. He now serves as an Associate Editor of IEEE Transactions on Neural Networks and Learning Systems.

Kay Chen Tan is an associate professor at National University of Singapore in Department of Electrical and Computer Engineering. He received the B.Eng. degree with First Class Honors in Electronics and Electrical Engineering, and the Ph.D. degree from the University of Glasgow, Scotland, in 1994 and 1997, respectively. He is actively pursuing research in computational and artificial intelligence, with applications to multi-objective optimization, scheduling, automation, data mining, and games. He has published over 100 journal papers, over 100 papers in conference proceedings, co-authored 5 books.

He is currently the Editor-in-Chief of IEEE Computational Intelligence Magazine (CIM). He also serves as an Associate Editor/Editorial Board member of over 15 international journals, such as IEEE Transactions on Evolutionary Computation, IEEE Transactions on Systems, Man and Cybernetics: Part B Cybernetics, IEEE Transactions on Computational Intelligence and AI in Games, Evolutionary Computation (MIT Press), European Journal of Operational Research, Journal of Scheduling, and International Journal of Systems Science. Dr Tan is the awardee of the 2012 IEEE Computational Intelligence Society (CIS) Outstanding Early Career Award for his contributions to evolutionary computation in multi-objective optimization.

Haoyong Yu received the B.S. and M.S. degrees in Mechanical Engineering from Shanghai Jiao Tong University, Shanghai, China, in 1988 and 1991, respectively. He received the Ph.D. degree in Mechanical Engineering from Massachusetts Institute of Technology, Cambridge, Massachusetts, US, in 2002. He is currently an Assistant Professor with the Department of Bioengineering and Principal Investigator of the Singapore Institute of Neurotechnology (SiNAPSE) at National University of Singapore. His areas of research include medical robotics, rehabilitation engineering and assistive technologies, system dynamics and control. He is a member of IEEE.

Please cite this article as: Q. Yu, et al., A brain-inspired spiking neural network model with temporal encoding and learning, Neurocomputing (2014), http://dx.doi.org/10.1016/j.neucom.2013.06.052