UNIVERSITY OF
LINCOLN

Deep Visual Learning with Spike-timing Dependent Plasticity

Daqi Liu

Doctor of Philosophy

2017

Deep Visual Learning with Spike-timing Dependent Plasticity

Daqi Liu

School of Computer Science
University of Lincoln

A dissertation submitted to the School of Computer Science University of Lincoln in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computing

March 2017

Abstract

For most animal species, reliable and fast visual pattern recognition is vital for their survival. Ventral stream, a primary pathway within visual cortex, plays an important role in object representation and form recognition. It is a hierarchical system consisting of various visual areas, in which each visual area extracts different level of abstractions. It is known that the neurons within ventral stream use spikes to represent these abstractions. To increase the level of realism in a neural simulation, spiking neural network (SNN) is often used as the neural network model. From SNN point of view, the analog output values generated by traditional artificial neural network (ANN) can be considered as the average spiking firing rates. Unlike traditional ANN, SNN can not only use spiking rates but also specific spiking timing sequences to represent the structural information of the input visual stimuli, which greatly increases the distinguishability.

To simulate the learning procedure of the ventral stream, various research questions need to be resolved. In most cases, traditional methods use winner-take-all strategy to distinguish different classes. However, such strategy works not well for overlapped classes within decision space. Moreover, neurons within ventral stream tends to recognize new input visual stimuli in a limited time window, which requires a fast learning procedure. Furthermore, within ventral stream, neurons receive continuous input visual stimuli and can only access local information during the learning procedure. However, most traditional methods use separated visual stimuli as the input and incorporate global information within the learning period. Finally, to verify the universality of the proposed SNN framework, it is necessary to investigate its classification performance for complex real world tasks such as video-based face disguise recognition.

To address the above problems, a novel classification method inspired by the soft

winner-take-all strategy has been proposed firstly, in which each associated class will be assigned with a possibility and the input visual stimulus will be classified as the class with the highest possibility. Moreover, to achieve a fast learning procedure, a novel feed-forward SNN framework equipped with an unsupervised spike-timing dependent plasticity (STDP) learning rule has been proposed. Furthermore, an event-driven continuous STDP (ECS) learning method has been proposed, in which two novel continuous input mechanisms have been used to generate a continuous input visual stimuli and a new event-driven STDP learning rule based on the local information has been applied within the training procedure. Finally, such methodologies have also been extended to the video-based disguise face recognition (VDFR) task in which human identities are recognized not just on a few images but the sequences of video stream showing facial muscle movements while speaking.

Declaration

I declare the work presented in this thesis is all my own work unless it is mentioned otherwise and referenced in the text.

Acknowledgements

Here, I want to firstly thank my supervisor, Prof.Shigang Yue, for his insightful suggestions in the academic area , as well as his kind helps in the daily life. When I firstly arrived in Lincoln, Prof.Yue gave me a warm welcome and introduced the colleagues within CIL group to me. He introduced several related state-of-art bio-inspired areas and asked me to choose the most interested one. After I chosen the spiking neural network as my main research area, he even bought me a classic introduction book. Since then, we have lots of face-to-face or skype discussions about the related projects and researches. From our discussions, I noticed Prof.Yue, as a true scholar, always can see the nature of the problem and use relatively easier concept to explain his idea. This characteristic is truly helpful for the students to improve their academic levels. Meanwhile, I also want to thank my former second supervisor, Dr.Oscar Martinez Mozos, and my current second supervisor, Dr.Nicola Bellotto, for their helpful advices and kind helps within my PhD period.

Secondly, I want to thank all the colleagues within CIL group for their kind helps during the whole PhD period. They are always eager to help me when I have difficulties in research or daily life. During my PhD period, we have lots of CIL workshops and everyone is quite active for those events. By exchanging opinions and proposing problems during the CIL workshops, I truly learned a lot from them. Finally, I appreciate the financial support from School of Computer Science at University of Lincoln and EU FP7 projects EYE2E (269118) and LIVCODE (295151).

List of Publications

- **D. Liu** and S. Yue (2014) Spiking Neural Network for Visual Pattern Recognition, *2014 International Conference on Multisensor Fusion and Information Integration for Intelligent System MFI*, pages 1-5.

- **D. Liu** and S. Yue (2016) Visual Pattern Recognition using Unsupervised Spike Timing Dependent Plasticity Learning, *2016 International Joint Conference on Neural Networks (IJCNN 2016)*.

- **D. Liu** and S. Yue (2016) Fast Unsupervised Learning for Visual Pattern Recognition using Spike Timing Dependent Plasticity, Submitted to *Neurocomputing* and finished the first revision.

- **D. Liu** and S. Yue (2017) Event-driven Continuous STDP Learning using HMAX Model for Visual Pattern Recognition, Ready to submit to *IEEE transactions on cybernetics*.

- **D. Liu** and S. Yue (2017) Spiking Neural Network for Video-based Disguise Face Recognition, Prepare to submit to *IEEE transactions on Neural Networks and Learning Systems*.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In computer science, an "intelligent" machine/robot is often refer to a flexible rational agent that can perceive its environment and take actions to achieve certain goal based on their learned "knowledge". Such intelligence exhibited by machine/robot is called artificial intelligence (AI). For instance, a robot is aimed at recognizing the human hand gestures and perform certain simple tasks accordingly: moving forward or backward, executing certain programs, starting or pausing turning around and so on. There are two core ingredients for the robot to achieve such goal: 1) The ability to make the robot see the gesture being performed. 2) The ability to understand the gesture being performed.

Generally speaking, machine learning is often considered as the answer to achieve the above requirements. Specifically, computer vision, one example of machine learning, plays an important role in solving the above task. Besides providing high level information about the environment so that the robot can "see" what is going on, the computer vision methods can also recognize different given patterns so that the robot can "understand" what was going on. However, the intelligence of a robot using traditional machine learning methods remains far behind the human beings. An mammalian brain may contain more than 10 billion densely packed neurons that are connected to an intricate network with numerous spikes are emitted in each millisecond [1]. It has extremely complex network structure, as shown in Figure 1.1, yet still can generate impressive processing speed. Research shows an mammalian brain can process complicated real-life visual pattern recognition scenarios at milliseconds scale [2].

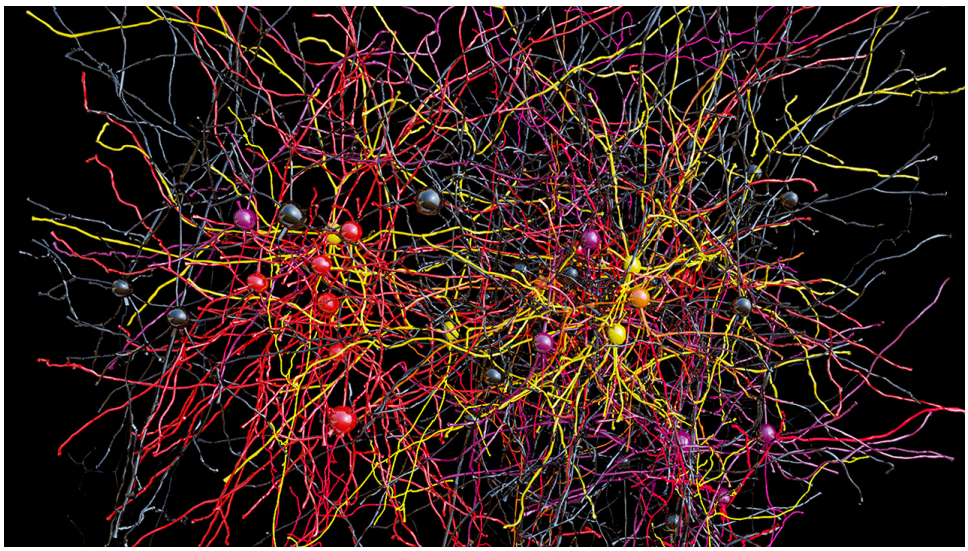


Figure 1.1: Spiking neurons and their synaptic connections within the brain [5].

1.1 Inspiration

Visual cortex within the brain is the part of the cerebral cortex responsible for processing visual information, which is critical for reliable and fast visual pattern recognition tasks. For most animal species, such visual pattern recognition capability is vital for their survival. In order to learn the input visual pattern, visual cortex uses primary visual pathway to transmit and learn the information. There are two primary pathways within the visual cortex: dorsal stream and ventral stream [3], [4]. The dorsal stream, or "where pathway", is involved with processing the object's spatial location relative to the viewer and with speech repetition. The ventral stream, also known as "what pathway", plays an important role in form recognition and object representation.

Ventral stream consists of various visual areas, which gets its main input from the primary visual cortex V1, and goes through V2 and V4 to areas of the inferior temporal lobe. Each visual area contains a full representation of the visual space. That is, it contains neurons whose receptive fields together represent the entire visual filed. Figure 1.2 shows schematic framework of the ventral stream. It is known that the ventral stream has a hierarchical structure with several layers capable of extracting different level of abstractions. Like the Figure 1.2 shown, the higher the layer located, the more abstract features it generated. For instance, by integrating several faces with different positions within the inferior temporal (IT) layer and extracting the most significant structure information, frontal faces with different classes have been generated. Within

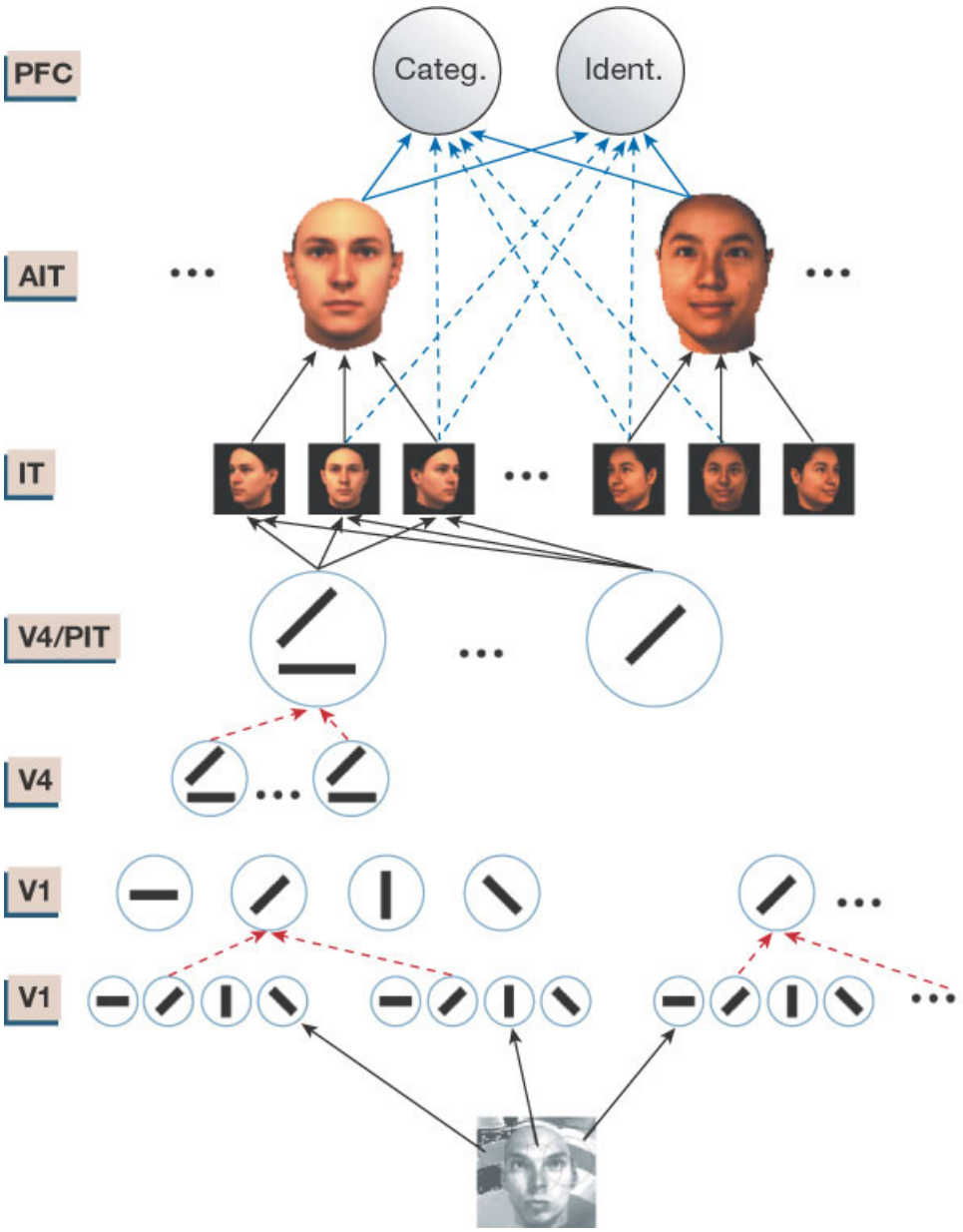


Figure 1.2: Schematic framework of the ventral stream [6]. Here, V1, V4, PIT, IT, AIT and PFC stand for primary visual cortex, visual area V4, posterior inferotemporal area, inferior temporal cortex, anterior inferotemporal area and prefrontal cortex, respectively. It can be seen that ventral stream is a hierarchical system with various layers, in which each layer extracts different level of abstractions.

ventral stream, in most cases, the new visual pattern has been learned in a short time window to adapt to new environment or changes promptly.

Inspired by this hierarchical processing structure, deep learning [7], [8], [9], [10], [11], [12], [13], one part of a broader family of machine learning methods, attempts to make better representations of input data and create models to learn these representation. Various deep learning architectures such as deep neural networks [14], [15], convolutional neural networks [14], [16], [17], deep belief networks [9], [18] and re-

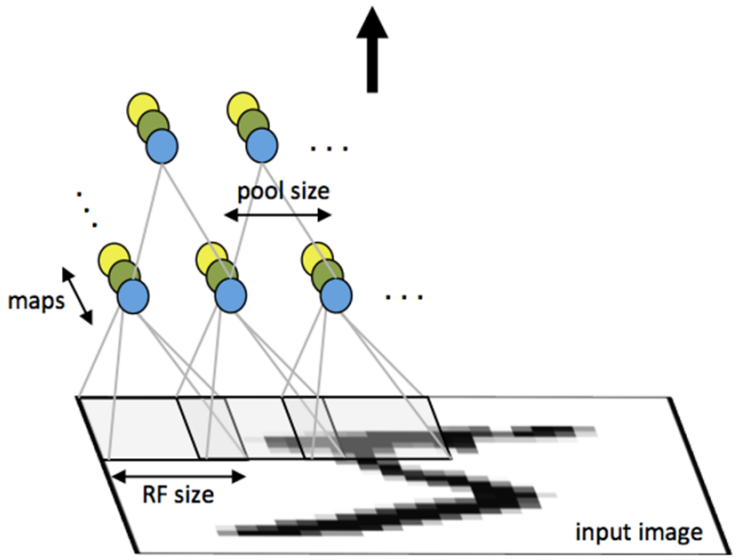


Figure 1.3: The template matching and the max pooling layers within HAMX model adopted in this paper. Units with the same color have tied weights and units of different color represents different filter maps [21]. For an input map, a template matching operation will obtain the convolution of the map by using a specific template (kernel). While a maximum pooling operation, a downsampling technique, will compute the maximum value from a local domain and only use it to represent the whole local domain.

current neural networks [19] have been applied to fields like automatic speech recognition, natural language processing, bioinformatics and so on. For instance, AlphaGo, a computer program developed by Google DeepMind [20], is perhaps the most famous example of deep learning right now, which beats several professional human Go players without handicaps on a full-sized 19×19 board.

Convolutional neural networks (CNNs) [14], [16], [17], compared with other deep learning methods, has shown superior performances in both image and speech applications. CNNs are easier to train than other regular, deep, feed-forward neural networks and have fewer parameters to estimate, making it a highly attractive architecture to use. HMAX [22], a feed-forward CNN model, stands out from other competitors. Such model focuses on the object recognition capabilities of the ventral stream in an "immediate recognition" mode, independent of attention or other top-down effects [22]. To simulate the simple and complex cells within primary visual cortex (V1), HMAX models build an increasingly complex and invariant feature representation by alternating between a template matching and a maximum pooling operation. Figure 1.3 shows the template matching and the max pooling operations used in HMAX model. Specif-

ically, for an input map, a template matching operation will obtain the convolution of the map by using a specific template (kernel). As a downsampling technique, for a local domain, a maximum pooling operation will compute the maximum value from the local domain and only use it to represent the whole local domain.

Traditionally, neurons within classical artificial neural network (ANN) such as CNN use analog values to represent the information. However, with the development of neuroscience, it has been shown that neurons within visual cortex use spikes to represent the information. To increase the level of realism in a neural simulation, spiking neural network (SNN) [23], [24], [25], [26], [27] is often used as the neural network model. By incorporating temporal information into the processing procedure, SNN uses spatiotemporal structural information to represent the input visual stimuli. From the SNN point of view, the output values of traditional ANN can be considered as the average firing rate of the neurons. Unlike the traditional ANN, SNN cannot only use spiking rate but also specific spiking timing sequence (spiking pattern) to represent the information, which greatly improves the distinguishability. As shown in Figure 1.4, from traditional ANN point of view, the two spiking patterns have the same output values $6/t$ and thus should be identical. However, if using specific spiking timing sequences to represent the information, these two spiking patterns are clearly different. SNN can distinguish different input spiking patterns even they have the same spiking rates and thus, compared with traditional ANN, the distinguishability of SNN has been significantly improved.

To specify a SNN, three core factors need to be included: architecture, neural model and learning rule. Those core factors have complementary relationships and have their own objectives. For example, architecture depicts what variables are involved in the network and their topological relationships, neural model defines how the activities of the neurons change in response to each other and learning rule specifies the way in which the neural network's weights change with time.

Inspired by ventral stream, lots of architectures used in SNNs apply hierarchical layers to define the variables involving in the network and their topological relationships. From the information theory point of view, SNNs need spiking coding schemes

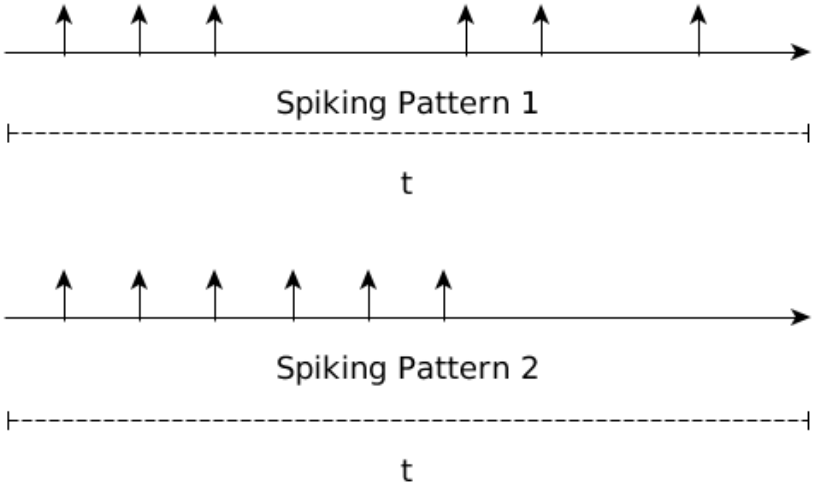


Figure 1.4: The comparison between SNN and traditional ANN. Here, each short vertical arrow represents a specific spiking timing. Based on the traditional ANN, spiking pattern 1 and 2 have the same spiking rates $6/t$ and thus should be the same. However, from SNN point of view, those two spiking patterns are clearly different. Instead of just using spiking rate like ANN, SNN can also use specific spiking timing sequences to represent the structural information.

to transform the input analog features to real spike trains so that they can be further processed within the following layers. Various coding methods such as rate coding [28], [29], temporal coding [30], [31], [32], population coding [33], [34] and sparse coding [35] have been widely applied in the related fields.

To regulate the neuron activity, various neural models such as leaky integrate-and-fire (LIF) model [36], HodgkinHuxley model [37], [38] and exponential integrate-and-fire model [39] have been proposed. Those models describe the relationship between neuron membrane currents and membrane voltage. Unlike the traditional perceptron neurons, after feeding presynaptic spikes, a neuron within SNN fires a postsynaptic spike when and only when its membrane voltage cross the predefined threshold.

For SNNs, spiking learning rules are quite essential as they can "recall" or "recognize" different spiking patterns. In neuroscience, Hebbian theory [40] is probably the most famous learning rule, which argues an increase in synaptic efficacy arises from the presynaptic cell's repeated and persistent stimulation of the postsynaptic cell. It is often summarized as "Cells that fire together, wire together". Spike-timing dependent plasticity (STDP) [41], [42], [43], [44], [45], a temporally asymmetric form of Heb-

bian learning, is widely believed that it underlies learning and information storage in the brain, as well as the development and refinement of neuronal circuits during brain development.

1.2 Research Questions

For face recognition tasks, traditional learning methods such as batch learning rule [23], [24], [46] and on-line learning rule [47], [48], [49] use winner-take-all(WTA) strategy to distinguish different classes. However, overlapped classes within the decision space could worsen the classification performance of WTA strategy. When classifying a sample within the decision space, WTA strategy only relies on the class information of its nearest neighbor cluster. Such classifying strategy only works well if the samples within the same class tend to stay together. However, the samples within different classes, in most cases, often have a distributed layout and tend to overlap with each other. The nearest neighbor cluster of a sample may actually belong to another class. Therefore, it is necessary to find a method to reduce the error classification of WTA strategy.

Besides reducing the error classification, it is also desirable to achieve a fast unsupervised learning. To adapt to a new environment, human brain tends to differentiate various input visual patterns in an unsupervised way within a limited time window. For instance, recent electrophysiological studies [50], [51] show that the neurons within the inferotemporal cortex respond selectively to faces only 80-100 ms after presenting the visual stimulus. In fact, compared with human brain, the learning speed of the current cutting-edge supercomputers still lags behind. For example, the Fujitsu K in Japan [52], a supercomputer with 83,000 processors and 1.4 million GB of RAM, takes about 40 minutes to simulate just one second of human brain activity. Thus, obtaining such fast unsupervised learning while retaining a comparable performance proposes a huge challenge for the researchers.

Within a limited period, ventral stream is capable of adaptively learning the spatiotemporal structures from the input spiking patterns. Traditionally, to learn the spatiotemporal structures, a new input spiking pattern is only allowed to feed into the

learning system when the membrane potential generated by the previous spiking pattern has been reseted. However, in ventral stream, the neurons receive the spiking patterns continuously without any resetting involved. Furthermore, traditional learning methods like spike-timing dependent plasticity (STDP) often incorporate global information within the training procedure. For instance, to update the synaptic weights within STDP learning rule, the neuron is required to remember all the related spiking timings within the learning window. In most cases, it is physiologically unrealistic and not efficient. Therefore, adaptively learning the spatiotemporal structures from the continuous input spiking patterns by only accessing local information has became a challenge for researchers.

Furthermore, to demonstrate the universality of a novel learning methodology, it is necessary to verify its classification performance on complex real world recognition tasks. With the increasingly lower cost of video recording, it is natural to transform the face recognition applications from the image-based to video-based. Various video-based face recognition (VFR) methods have been proposed to recognize the faces within varied video face databases with different variations. The common ground for these methods is they need to generate static feature vector firstly. To obtain such static feature vector, certain areas like eyes, mouse or nose within the face should be visible. However, for video-based disguise face recognition like looking for lost persons in train station or locating terrorists in airports, such requirement is hard to achieve since the subjects will try to cover some parts of their faces. Thus, to address the above problem, a novel video-based disguise face recognition is much needed.

1.3 Contributions

The main contributions of this dissertation can be summarized as the following:

1) Inspired by soft winner-take-all (WTA) strategy, a novel classification method has been proposed, in which each related class will be assigned with a probability and the input stimuli belongs to the class with the highest probability. Unlike the traditional WTA strategy, the proposed method adds certain flexibilities into the classification procedure and the input visual stimuli have the potential to be classified as other classes.

2) To strike a trade-off between speed and performance, a novel feed-forward SNN framework along with its unsupervised learning method have been proposed. The proposed method uses the first two layers of HMAX model to generate the local invariant features and applies unsupervised STDP learning method to train the synaptic weights.

3) Since the neurons within the ventral stream receive continuous spiking patterns and can only access the local information, an event-driven continuous STDP (ECS) learning method has been proposed to adaptively learn the synaptic weights. Unlike the above method, the proposed ECS method uses modified HMAX model with sparsity and intermediate features to generate global invariant features. In addition, two novel continuous input mechanisms, a sequential one with interval between adjacent spiking patterns and another parallel one with two separated spiking pattern groups, have been proposed to obtain the continuous input spiking pattern sequence. Furthermore, within the proposed event-driven STDP learning rule, the learning procedure will be activated when the neuron receive a presynaptic or postsynaptic spike event.

4) Unlike the traditional video-based face recognition (VFR) methods [53], [54], [55], which only works when certain areas (i.e. eyes or nose) of the face are visible, the proposed ECS method has been extended to accomplish the video-based disguise face recognition (VDFR) tasks using the dynamic facial movements and achieved satisfactory correct classification performance on the proposed video disguise face database.

Chapter 2

Literature Review

To understand and interact with the environment, visual pattern recognition is a basic yet vital capability for the "intelligent" machine/robot described in AI. With such capability, the "intelligent" machine/robot can "see" the environment and "recognize" the given pattern which has been shown before. The procedure of obtaining such capability can be considered as a learning process. After this learning process, the "intelligent" machine/robot eventually adapts with the environment and accomplish its predefined working objectives.

Essentially, the visual pattern recognition capability can be considered as a specific learning capability. To obtain such learning capability, hundreds of models and algorithms have been proposed during the last several decades. For a clear and through understanding of the background, I will introduce and argue those related methods through a top-down way - from general machine learning methodologies to more relevant and specific fields such as deep learning and spiking neural networks.

2.1 Machine Learning

In 1959, Arthur Samuel summed the machine learning as such statement: "Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed" [56]. Machine learning plays a significant role in a wide range of critical applications, such as data mining, nature language processing, computer vision and expert systems. It provides potential solutions for the above areas

and has already grew into a major research topic.

Machine learning filed is quite vast and is expanding rapidly, being continually partitioned and sub-partitioned. There are so many algorithms available for machine learning tasks and sometimes it may feel overwhelming. To have a clear and through understanding of those machine learning algorithms, it is necessary to divide them into different categories according to two grouping methods: learning style and similarity.

2.1.1 Grouping by Learning Style

Traditionally, in machine learning, there are three different styles an algorithm can model a problem based on the interaction with the input data: supervised learning, unsupervised learning and semi-supervised learning [57], [58]. Those different styles are essential for researchers as it drives you to think about the roles of input data and the model preparation process. In supervised learning, the input data has a known label or result. A model is prepared through a training process where it is required to make predictions and is corrected when those predictions are wrong. The training process continues until the model achieves a desired level of accuracy on the training data. Unlike supervised learning, the input data of unsupervised learning is not labeled. A model is prepared by deducing structures present in the input data, which may be achieved by extracting general rules or systematically reducing redundancy through a mathematical process. The input data of semi-supervised learning is a mixture of labeled and unlabeled examples. There is a desired prediction problem but the model must learn the structures to organize the data as well as make predictions.

2.1.2 Grouping by Similarity

Compared with grouping algorithms by learning style, it is more intuitive to divide those algorithms by similarity, which can be summarized as following [57], [58]:

- regression algorithms
- regularization algorithms
- decision tree algorithms

- instance-based algorithms
- bayesian algorithms
- dimensionality reduction algorithms
- ensemble algorithms
- clustering algorithms
- artificial neural network algorithms

For those algorithms, some (instance-based algorithms or clustering algorithms) are only focus on unsupervised learning field while some (regression algorithms or regularization algorithms) are only concentrate on supervised learning field. Others can be used in both supervised or unsupervised algorithms. Since there are overlaps among different categories, this grouping strategy is not perfect, but it can still help us to better understand these abundant algorithms.

Instead of focusing on traditional algorithms, this thesis pays more attention on bio-inspired methods. Among all the algorithms, only artificial neural network algorithms use bio-inspired methods to deal with the visual pattern recognition tasks. Thus, we will only highlight the artificial neural network algorithms within all the machine learning methods. Artificial Neural Networks are models that are inspired by the structure and/or function of biological neural networks. They are a class of pattern matching that are commonly used for regression and classification problems but are really an enormous subfield comprised of hundreds of algorithms and variations for all manner of problem types. The most commonly used artificial neural network algorithms include: perceptron, back propagation (BP) network, Hopfield network, radial basis function (RBF) network and deep learning.

2.1.3 Traditional Artificial Neural Networks

From [57], [58], we know that if the classes are linearly separable (i.e. they can be separate by an hyperplane in the n-dimensional space defined by your input of length = n), a perceptron is enough for the classification task. Otherwise, at least

one hidden layer should be added into the perceptron so that multilayer perceptrons have been generated. In most case, back propagation method should be used within multilayer perceptrons (MLP). Basically, MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable. Within machine learning, the perceptron is an algorithm for supervised learning of binary classifiers. Back propagation requires a known, desired output for each input value in order to calculate the loss function gradient. Thus, in most cases, it is often considered as a supervised learning method, although it is also used in some unsupervised networks such as autoencoders.

In machine learning, MLP is a feed-forward neural network while Hopfield network is a recurrent neural network. Both of them are deterministic networks: For MLP, it can be shown that the it can estimate the conditional average on the target data; For Hopfield network, once the initial state is given, its dynamics evolves following a Lyapunov function. Unlike the stochastic networks such as Helmholtz and Boltzmann machines, given an input, the state of the Hopfield network does not converge to an ensemble distribution, but to a unique state.

Unlike the MLP, RBF network only has one hidden layer. Besides, it differs from a MLP via its activation and combination functions. Commonly-used types of neural networks such as the MLP and Hopfield network have less limitations and they can add appropriate hidden layers into the networks. However, they are highly vulnerable to adversarial noise and can make very wrong predictions when fed such examples as their inputs. This is not the case in RBF networks which seems to be due to their non-linear nature of these networks. By taking the trade-off between accuracy and robustness, RBF networks bring much more robustness into the predictions.

Arguably, MLP is not really different from deep learning, but just one type of deep learning. Theoretically, back propagation method used in MLP can train a network with many layers. However, few researchers have widespread success training neural networks with more than 2 layers. This was mostly because of vanishing and/or exploding gradients. Specifically, MLP was typically initialized using random numbers such as the gradient of the network's parameters and uses the network's error to adjust

the parameters to better values in each training iteration. In back propagation method, to evaluate this gradient involves the chain rule and you must multiply each layer's parameters and gradients together across all the layers. This is a lot of multiplication, especially for networks with more than 2 layers. Unlike the back propagation network, deep learning proposed a new initialization strategy: use a series of single layer networks - which do not suffer from vanishing/exploding gradients - to find the initial parameters for a deep MLP.

2.1.4 Deep Learning

Deep learning is a modern update to artificial neural network that based on a set of algorithms that attempt to model high-level abstractions in data by using a deep graph with multiple processing layers, composed of multiple linear and non-linear transformations [7], [8], [9], [10], [11], [12], [13]. Specifically, a deep neural network is an artificial neural network with multiple hidden layers of units between in the input and output layers, which has often been used to model complex non-linear relationships.

Various deep learning architectures such as deep neural networks, deep convolutional neural networks, deep belief networks and recurrent neural networks have been applied and achieved state-of-art results on various fields like computer vision, automatic speech recognition, natural language processing, audio recognition and bioinformatics. Research in this area attempts to make better representations and create models to learn these representations from large-scale unlabeled data.

Deep learning algorithms are based on distributed representations. The underlying assumption behind distributed representations is that observed data are generated by the interactions of factors organized in layers. Deep learning adds the assumption that these layers of factors correspond to levels of abstraction or composition. Since its resurgence, deep learning has become part of many state-of-the-art systems in various disciplines, particularly computer vision and automatic speech recognition (ASR).

The first functional Deep Learning networks with many layers, also known as the Group Method of Data Handling (GMDH), were published by Alexey Grigorevich Ivakhnenko and V. G. Lapa in 1965 [59]. GMDH features fully automatic struc-

tural and parametric optimization of models. The activation functions of the network nodes are Kolmogorov-Gabor polynomials that permit additions and multiplications. In 1971, Ivakhnenko proposed a novel method describing the learning of a deep feed-forward multilayer perceptron with eight layers, already much deeper than many later networks [60]. The supervised learning network is grown layer by layer, where each layer is trained by regression analysis. From time to time useless neurons are detected using a validation set, and pruned through regularization. Several related deep learning working architectures inspired by artificial neural network will be elaborated in the following subsections.

2.1.4.1 Deep Neural Network

Inspired by artificial neural network, the neocognitron [16], a hierarchical and multilayered artificial neural network, was proposed by Kunihiko Fukushima in 1980. It has been used for handwritten character recognition and other pattern recognition tasks, and served as the inspiration for convolutional neural networks. In 1989, Yann LeCun et al. were able to apply the standard back propagation algorithm into a deep neural network with the purpose of recognizing handwritten ZIP codes on mail. However, the processing time can extend to approximately 3 days, which is obviously impractical for general use [14]. In 1995, Brendan Frey demonstrated that it was possible to train a network containing six fully connected layers and several hundred hidden units using the wake-sleep algorithm, which was co-developed with Peter Dayan and Geoffrey Hinton [15]. However, training took two days.

Due to the added layers of abstraction, deep neural networks (DNNs) are prone to overfitting, which may generate the long computation time. To reduce the overfitting, several regularization methods such as weight decay or sparsity [60] or dropout regularization [61] have been applied within DNNs. To train the structures of the DNNs, error-correction training methods such as back propagation with gradient descent have often been used. Compared with other methods, such method is easy to implement and tends to converge to better local optima. However, for DNNs, these methods can be computational expensive, which, in most cases, need different GPU techniques to

speedup the training procedure.

2.1.4.2 Convolutional Neural Network

A convolutional neural network (CNN) is composed of one or more convolutional layers with fully connected layers on top, which often uses tied weights and pooling layers. It is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex. Within Fukushima's convolutional architecture [16], max-pooling operation [62], the most important step within CNN, has been proposed to take advantage of the 2D structure of input data. Compared with other regular, deep, feed-forward neural networks, CNN is easier to train and has much fewer parameters to estimate, which has become the method of choice for processing visual and other two-dimensional data [14].

To simulate the behavior of a visual cortex, unlike the MLP, CNN exploits the strong spatially local correlation present within the nature images. Specifically, the layers of a CNN have neurons arranged in 3 dimensions: width, height and depth, also known as 3D volumes of neurons. The neurons inside a layer are only connected to a small region of the layer before it, called a receptive field. Furthermore, CNN exploits spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers so that the architecture can ensure that the learnt "filters" produce the strongest response to a spatially local input pattern, which can be summarized as local connectivity. Besides, within CNN, each filter is replicated across the entire visual field so that these replicated units share the same parameterization (weight vector and bias) and form a feature map. by sharing weights, all the neurons in a given convolutional layer detect exactly the same feature.

2.1.4.3 Neural History Compressor

In 1992, an early generative model called the neural history compressor, implemented as an unsupervised stack of recurrent neural networks (RNNs) [19], has been proposed to tackle the vanishing gradient problem of the back propagation method used in neural networks. The RNN at the input level learns to predict its next input

from the previous input history. Only unpredictable inputs of some RNN in the hierarchy become inputs to the next higher level RNN which therefore recomputes its internal state only rarely. Each higher level RNN thus learns a compressed representation of the information in the RNN below. By doing this, the input sequence can be precisely reconstructed from the sequence representation at the highest level and the system effectively minimizes the description length or the negative logarithm of the probability of the data [63].

2.1.4.4 Deep Belief Network

A deep belief network (DBN) [9] ,[18] is a probabilistic, generative model made up of multiple layers of hidden units. Generally speaking, DBNs are generative neural networks that stack Restricted Boltzmann Machines (RBMs) in which each RBM can be considered as a generative auto-encoder. Specifically, auto-encoder uses deterministic units while RBM uses stochastic units with particular (usually binary of Gaussian) distribution. Learning procedure consists of several steps of Gibbs sampling and adjusting the weights to minimize reconstruction error. In theory, DBNs should be the best models but it is very hard to estimate joint probabilities accurately at the moment. In current literature on benchmark computer vision datasets such as MNIST, CNNS have performed better than DBNs by themselves.

2.1.4.5 Deep Boltzmann Machine

Although Deep Belief Networks (DBNs) and Deep Boltzmann Machines (DBMs) [64], [65] diagrammatically look very similar, they are actually qualitatively very different. Basically, DBNs are directed and DBMs are undirected. If fitting them into the broader multilayer picture, DBNs can be considered as sigmoid belief networks with many densely connected layers of latent variables while DBMs are markov random fields with many densely connected layers of latent variables.

Even though DBNs and DBMs are qualitatively very different, they still share some similarities: both DBNs and the original DBMs use initialization schemes based on greedy layerwise training of RBMs and they both feature layers of latent variables

which are densely connected to the layers above and below, but have no intra-layer connections.

2.2 Spiking Neural Network

Inspired by the central nervous systems of animals, in machine learning, artificial neural network (ANN) has been proposed to estimate functions that can depend on a large number of inputs that are generally unknown. McCulloch-Pitts threshold neurons have been used in the first generation of ANN, which only generate digital output. Instead of using a step- or threshold function, neurons within the second generation use a continuous activation function like the sigmoid or hyperbolic tangent to compute their output signals, which are suitable for analog in- and output. Based on the advancement of biological realism, spiking neural networks (SNNs) has been proposed. Unlike the last two generations, SNN incorporates spatiotemporal information in communication and computation, like the real neurons do. These neurons use pulse coding instead of classic rate coding. Specifically, from the SNN point of view, the output values of traditional ANN can be considered as the average firing rate of the neurons. Besides applying spiking rate, SNN can also use specific spiking timing sequence (spiking pattern) to represent the information, which greatly improves the distinguishability.

Typically, to specify a SNN, four basic parts should be included in the proposed method: framework, neuron model, coding scheme and learning rule. Specifically, framework specifies the topological relationship of the involved variables while neural model depicts how the activities of the neurons change in response to each other. Coding scheme has been used to transform the input visual stimuli into spikes and such spikes will be trained by the learning rule so that the network can have selectivities (synaptic weights) for the input visual stimuli. In this section, I will investigate the SNN from those respects mentioned above.

2.2.1 Framework

Based on the data flow direction, there are two main frameworks within spiking neural networks: feed-forward neural network and recurrent neural network. Feed-forward SNN is the first and most simple type of SNN devised. The information within this network moves only in forward direction. In contrast, recurrent SNN are models with bi-directional data flow, which means there are cycles or loops within this kind of SNN.

It is known that the human brain is a recurrent neural network: a network of neurons with feedback connections. Compared with feed-forward neural networks, such recurrent neural networks are computational more powerful and biologically more plausible. In paper [66], the authors propose a recurrent spiking model capable of learning episodes featuring missing and noisy data. The presented topology provides a means of recalling previously encoded patterns where inhibition is of the high frequency variety aiming to promote stability of the network. By simulating the network response in a moving-dot blanking experiment, the paper [67] resolves how anisotropic connectivity patterns that consider the tuning properties of neurons efficiently predict the trajectory of a disappearing moving stimulus.

However, the recurrent SNN is still in its initial stage and it lacks enough support from the available biologically plausible models. On the contrary, the feed-forward SNN is more intuitive and lots of available biologically plausible models can be used to accomplish the feed-forward SNN. Lots of works [25], [26], [27], [68], [69], [70], [71], [72], [73], [74], [75] have been investigated using such feed-froward frameworks.

2.2.2 Neuron Models

Within visual cortex, neurons use neural spikes to represent/transmit information. Neuron models define how the activities of the neurons change in response to each other. A spiking neuron model is a mathematical model of the electrical properties of neuronal action potentials, which are sharp changes in the electrical potential across the cell membrane that last for about one millisecond.

According to the physical units of the interface of the model, neuron models can

be divided into two categories: Electrical inputoutput membrane voltage models and nature/pharmacological input neuron models. The former one produces a prediction for membrane output voltage as function of electrical stimulation at the input stage while the latter one was inspired from experiments involving either natural or pharmacological stimulation. Electrical inputoutput membrane voltage models describe the relationship between neuronal membrane currents at the input stage, and membrane voltage at the output stage. Due to its efficiency and simplicity, the first category has been chosen as the neuron model in this thesis, as discussed below.

Integrate-and-fire Model: In 1907, the earliest neuron model, named integrate-and-fire (IF) model, has been proposed by Louis Lapicque [76]. Basically, when an input current is applied, the membrane voltage increases with time until it reaches a constant threshold V_{th}, at which point a delta function spike occurs and the voltage is reset to its resting potential, after which the model continues to run. However, the main drawback of this model is that it implements no time-dependent memory. Unlike the observed neuronal behavior, if the model receives a below-threshold signal at some time, it will retain that voltage boost forever until it fires again.

Leaky Integrate-and-fire Model: To resolve the memory problem of the integrate-and-fire model, leaky integrate-and-fire (LIF) model has been proposed by adding a "leak" term into the membrane potential, which indicates the diffusion of ions that occurs through the membrane when some equilibrium is not reached in the cell [36]. A postsynaptic neuron will fire a spike if and only if there are enough related presynaptic spikes fired within a short time window. Otherwise, the potential will gradually decease to zero if no presynaptic spikes fired. By doing this, this model emphasizes the causality of related spikes.

Hodgkin-Huxley Model: The Hodgkin-Huxley model [37], [38] describes the relationship between ion currents crossing the neuronal cell membrane and the membrane voltage [77], [78]. This model is based on the concept of membrane ion channels and experiments that allowed to force membrane voltage using an intra-cellular pipette. It is a very successful model as the authors even won the Nobel Prize for their work. However, such model is too complex to simulate as it requires much more parameters

than other neuron models.

Galves-Locherbach Model: Based on the LIF model, Antonio Galves and Eva Locherbach [79] proposed the Galves-Locherbach model, which is inherently stochastic. Given the model specifications, one can obtain the probability of a given neuron i spikes in a time period t. Essentially, such model can be considered as a specific development of LIF model.

Exponential Integrate-and-fire Model: The exponential integrate-and-fire (EIF) model, proposed by Nicolas Fourcaud-Trocm, David Hansel, Carl van Vreeswijk and Nicolas Brunel [39], is a simple modification of the classical integrate-and-fire model describing how neurons produce action potentials. In the EIF, the threshold for spike initiation is replaced by a depolarizing non-linearity.

Other Models: To describe "regenerative self-excitation" by a nonlinear positive-feedback membrane voltage and recovery by a linear negative-feedback gate voltage, FitzHugh and Nagumo [80] propose a sweeping simplifications to HodgkinHuxley model. In 1981, Morris and Lecar [36] combined HodgkinHuxley and FitzHugh-Nagumo into a voltage-gated calcium channel model with a delayed-rectifier potassium channel, named as MorrisLecar model. Furthermore, based on the FitzHughNagumo model, in 1984, Hindmarsh and Rose proposed the HindmarshRose model described by three coupled first order differential equations.

2.2.3 Coding Scheme

By generating characteristic action potentials, neurons are remarkable in their ability to propagate signals rapidly over large distances. Although action potentials can vary somewhat in duration, amplitude and shape, they are typically treated as identical stereotyped events in neural coding studies. Such identical stereotyped events are also known as neural spikes.

Based on the information theory, the coding scheme transforms the information format from analog features into spike trains. Without such transformation, it is impossible for SNN to process the input information. Various coding schemes such as rate coding, temporal coding or population coding have been proposed to represent and

transmit the structural information existed within the input visual stimuli, as shown below.

Rate Coding: Rate coding assumes that most, if not all, information about the stimuli is contained in the firing rate of the neuron. Thus, such scheme states that as the intensity of a stimulus increases, the frequency or rate of action potentials increases. In most sensory systems, the firing rate increases, generally non-linearly, with increasing stimulus intensity [28].

There are two ways to represent the spiking rate within this coding scheme: spike-count rate and time-dependent firing rate. The former one is obtained by counting the number of spikes that appear during a trial and dividing by the duration of trial while the latter one is defined as the average number of spikes (averaged over trials) appearing during a short interval between times t and $t + \triangle t$, divided by the duration of the interval.

However, it is difficult for rate-based SNN to generate meaningful rate within a short time widow. For instance, research showed that mammalian brain use only millisecond scale time window to process complicated real life visual recognition tasks [2]. Moreover, if the input visual stimuli has been incorporated with the background noise with the same spiking rate, it is impossible for rate-based SNN to generate the selectivity for the input visual stimuli [29].

Temporal Coding: Instead of using spiking rate as the representation of the input stimuli, temporal coding schemes use specific spiking timings of the fired neurons to represent the input stimuli. Thus, spiking patterns will play an important role within the learning procedure. Various potential coding strategies based on spiking timing have been proposed and widely applied, such as time-to-first-spike, phase, correlations and synchrony.

Within time-to-first-spike coding scheme, each neuron only needs to fire a single spike to transmit information, which, clearly, is an idealization. Thorpe et al. [30] argues that the brain does no have time evaluate more than one spike from each neuron per processing step. Therefore, the first spike wave should contain most of the relevant information. Several groups have shown that most of the information about a new

stimuli is indeed conveyed during the first 20 or 50 ms after the onset of the neuronal response [31], [32], [81], [82].

In the olfactory system or hippocampus, oscillation of some global variable (for instance, the population activity) are quite common. These oscillations could serve as an internal reference signal and neural spikes could then then encode information in the phase of a pulse with the respect to the background oscillation [83], [84], [85]. Furthermore, spikes from other neurons can be considered as the reference signal for a spike code. Specifically, the synchrony between a pair or many neurons could signify special events and convey information which is not contained in the firing rate of the neurons [86], [87].

Population Coding: Population coding is a method to represent stimuli by using the joint activities of a number of neurons. In population coding, each neuron has a distribution of responses over some set of inputs, and the responses of many neurons may be combined to determine some value about the inputs. Compared with other coding schemes, population coding is one of a few mathematically well-formulated problems in neuroscience, which grasps the essential features of neural coding and yet is simple enough for theoretic analysis [88]. Several studies [33], [34] have shown that this coding strategy is widely used in the sensor and motor areas of the brain.

Sparse Coding: The sparse code is when each item is encoded by the strong activation of a relatively small set of neurons. For each item to be encoded, this is a different subset of all available neurons. Sparseness within this coding scheme focus either on temporal sparseness or on the sparseness in an activated population of neurons. The capacity of sparse codes may be increased by simultaneous use of temporal coding, as found in the locust olfactory system [35]. Given a potentially large set of input patterns, sparse coding algorithms attempt to automatically find a small number of representative patterns which, when combined in the right proportions, reproduce the original input patterns. The sparse coding for the input then consists of those representative patterns.

2.2.4 Learning Rule

Hebb's postulate [40], one of the most important theory in neuroscience, tries to explain the adaptation of neurons in the brain during the learning process. It emphasizes the causality between pre- and postsynaptic neurons, which also known as "Cells that fire together, wire together". Specifically, in Hebb's postulate, cell A needs to take part in firing cell B, and such causality can only occur if cell A fires just before, not at the same time as, cell B. Based on this original theory, several learning rules have been proposed to learn the selectivities (synaptic weights) from the the spiking patterns, as shown bellow.

BCM Learning Rule: In 1982, Elie Bienenstock, Leon Cooper, and Paul Munro [89] proposed an novel synaptic modification theory (also known as BCM theory) trying to account for experiments measuring the selectivity of neurons in primary sensory cortex and its dependency on neuronal input. It is characterized by a rule expressing synaptic change as a Hebb-like product of the presynaptic activity and a nonlinear function $\phi(y, \theta_M)$, of postsynaptic activity y. θ_M is the modification threshold. For low values of the postsynaptic activity ($y < \theta_M$), $\phi(y, \theta_M)$ is negative; for $y > \theta_M$, $\phi(y, \theta_M)$ is positive. The rule stabilizes by allowing the modification threshold, θ_M, to vary as a super-linear function of the previous activity of the cell.

STDP Learning Rule: Spike timing dependent plasticity (STDP) [41], [42], [43], [44], [45], a temporally asymmetric form of Hebbian learning [40], has been widely applied during the last decade. It is widely believed that it underlies learning and information storage in the brain, as well as the development and refinement of neuronal circuits during brain development [90], [91]. STDP requires no prior information or teaching signals since it is essentially an unsupervised learning rule.

Within neuroscience, long-term potentiation (LTP) is a persistent strengthening of synapses based on recent patterns of activity, while long-term depression (LTD) is an activity-dependent long-lasting reduction in the efficacy of neural synapses. Based on these concepts, the STDP learning rule can be summarized as: when a presynaptic spike fires slightly earlier than the post-synaptic spike, the associated synaptic efficacy will be potentiated (LTP); While the associated synaptic efficacy will be depressed

(LTD) if the pre-synaptic synaptic spike fires later than the post-synaptic spike.

2.3 State-of-the-art Methods using SNN

Still image based visaul recogntion such as face or charactor recognition has been quite commonly used in the past decades. It has been a natural topic for SNN as well. Delorme and Thorpe proposed the SpikeNet [23] architecture to model large networks of integrate-and-fire neurons in 1999. Inspired by this hierarchical architecture, they further proposed a feed-forward spiking neural network with a corresponding batch learning rule [24], [46] and the experimental results on ORL face database shown the proposed method can achieve a quite satisfactory performance even incorporating certain degrees of image degradations into input images. However, this method needs to prior know the number of samples within each class and only works well if there are no obvious overlaps between classes.

To overcome the above drawbacks, Wysoski, Benuskova and kasabov proposed an on-line learning method with structural adaptation [47], [48], [49]. It can adaptively divides the samples within each class into several sub-classes using on-line learning and achieves satisfying experimental results. Actually, batch learning rule can be seen as a special instance of on-line learning rule. However, it still considers one single specific pattern for the classification task. In other words, this on-line learning method assumes the training sample belongs to its nearest single relevant class without any consideration of other related classes.

Reliable and fast visual pattern recognition is vital for most animal species. Without such capability, the specie will be abandoned by the law of nature. In most cases, new visual pattern should be learned in a limited time window to adapt to new environments or changes promptly. Traditionally, within artificial neural networks (ANNs), the standard training method is back propagation. Given the input visual stimuli, each neuron receives its specific error information which is used to update the synaptic efficiency matrix. However, research shows this neuron-specific error information is impossible to obtain within the brain [92]. Compared with back propagation method, the unsupervised methods like STDP are more biologically plausible and thus attract

more and more researchers.

To successfully accomplish the visual pattern recognition task, the learning method should strike a relatively balanced state between biologically plausibility and recognition performance. Even we can design a system with higher recognition performance without any consideration of biologically plausibility, the processing mechanism of the neurons within the visual cortex is still largely unknown. Similarly, without any consideration of recognition performance, it is difficult to know which mechanisms are necessary for the processing procedure of the visual cortex even we are able to design functional systems.

For most visual pattern recognition applications, it is desirable to use more biologically plausible models to build the system without any considerable performance reduction. To achieve the above requirement, two popular approaches have been applied within the related fields: One popular approach is to still rely on back propagation training but afterwards converting the ANN into a SNN [70], [71], [72], [74], [93]. The other approach uses different variants of models of STDP learning methods trying to simulate the biological learning procedure of the visual cortex [17], [25], [26], [27], [68], [69], [75], [94]. For the sake of clarity, those two approaches will be introduced and argued in the following subsections, respectively.

2.3.1 Methods using Back propagation

Methods [70], [71], [72], [74], [93] within this category are still rely on back propagation training but afterwards converting the ANNs into SNNs. Even some of them achieve very high performance on classic recognition tasks like MNIST database with back propagation, such methods are not very biologically plausible or are at least very much abstracted from the biological mechanism. It is also need to mention that the conditions assumed by these classical back propagation methods are often hard to meet in real world applications. For instance, the neuron-specific error information required by back propagation methods to update the synaptic efficiency matrix is often impossible to obtain within the brain [92].

2.3.2 Methods using STDP Variants

Lots of researches [17], [25], [26], [27], [68], [69], [75], [94] have been investigated the visual pattern recognition tasks using different SNN frameworks and variant STDP learning methods. From the spiking encoding point of view, those methods can be divided into two main categories: spiking time-based methods and spiking rate-based methods.

For spiking rate-based methods [25], [26], [27], [75], [94], there are two main drawbacks: a) To generate meaningful spiking rate, the processing time window should be long enough. In other words, those rate-based methods need to run the database several time (also known as several runs), which is obvious time-consuming and inefficient. b) Most of those methods are not biologically plausible as they lack direct biological supports for their simulation of the ventral stream. Thus, in this thesis, we mainly focus on the spiking time-based methods.

In paper [17], Thorpe et. al use their SNN architecture to simulate the processing procedure of HMAX model. However, the simplified STDP learning methods applied in the synaptic connections between $C1$ and $S2$ (the second and the third layers of HMAX model) are only used for local feature extraction. They are not used for global pattern recognition. Inspired by convolutional neural network (CNN), in the paper [68], the authors proposed a novel SNN with supervised learning rule and temporal coding scheme to generate the spike pattern. Such SNN system and its supervised learning rule achieved relatively good classification rate when cross-validate the MNIST database. Such supervised learning rule needs prior knowledge before learning - in many cases, this prior knowledge is hard to obtain. Furthermore, the authors in papers [69] use a modified tempotron rule to learn the selectivities (synaptic weights) from the input visual stimuli. However, from the deep learning point of view, the simplified HMAX model (with only $S1$ and modified $C1$ layer, here, $S1$ and $C1$ layers represent the first and the second layer of HMAX model) employed in their method is not enough to obtain balanced high level features, it will generate lots of intra-class variance within the modified $C1$ layer. Besides, the supervisory signal used in tempotron rule has no strong experimental confirmation, which means it is less

biologically plausible than the STDP learning rule.

2.4 Video-based Disguise Face Recognition

To recognize a human, such as a friend has not been seen for a long time or a colleague with complete changed makeups, we human beings often need time to process the incoming "video sequence" to boost the confidence level on recognizing accuracy. Similarly, for machine learning methods, since it is easy to capture video sequences nowadays, the shift from processing still images only to video streams becomes normal.

There are two main video-based face recognition (VFR) methods: sequence-based methods [53], [55] and set-based methods [54], [95] (with reviews on lots of related methods). Basically, the sequence-based methods use the temporal dynamic information of the face among the adjacent video frames while the set-based methods does not use this temporal dynamic information, considering the video as image set of the separated video frames. Essentially, the common grounds of these two kinds of methods are using different feature vectors to represent the video frames. The former one tries to extract the temporal dynamic information from the feature vector patterns while the latter one models the feature vector patterns, also known as set, and use different correlation methods to compute the set-to-set distance.

Normally, to obtain the feature vectors using these methods, certain areas like eyes or nose of the face within the video frames need to be visible. For instance, in paper [55], within the proposed individual expression recognition (IER) block, to obtain the behavioral map (BM) containing facial evolutions of microexpressions in each frame, at least an eye, a brow or a cheek need to be detected within the video frames. In [53], the authors use genetically-inspired learning method to select meaningful facial features obtaining from five local areas, such as eyes, nose and mouth. These algorithms are essentially sequence-based methods. Similar situations can also be extended into the set-based methods. However, within video-based disguise face recognition (VDFR) applications, such critical requirement cannot be satisfied and thus these methods are not suitable for this specific scenario. Moreover, most of these methods

use SVM or multi-perceptron as the classification algorithms, which are clearly not biologically plausible.

However, the above problems can be successfully addressed if applying our proposed ECS methodologies on the changes of the facial movements. Here, the **changes** of the facial movements, refer to as differential frames in the following chapters, could be the sole or most important inputs to the ventral stream model for recognition. If the only inputs are differential images, as inspired by the computational models [96], [97], the background noise or light condition will not be a problem. Furthermore, for our proposed ECS methodologies, processing video stream is a natural choice since temporal information within different video frames can be treated as intra-class variances, for instance, face image at time t and face image at time $t + 1$ are similar to a handwritten character 1 and another handwritten character 1 , as shown in MNIST database. In this case, even the subject has different makeups, it can still be recognized with such methodologies.

The above proposed methodologies can successfully accomplish the VDFR tasks in which human identities are recognized not just on a few images but the sequences of video stream showing facial muscle movements while speaking. Our experiments on the proposed video disguise face database (VD Face DB) demonstrated the proposed VDFR methods are reliable and efficient.

Chapter 3

Spiking Neural Network for Face Recognition

In artificial intelligence, face recognition is a common yet significant task for the intelligent robots. According to recent electrophysiological studies [50], [51], the neurons within the inferotemporal cortex (IT) respond selectively to faces only 80-100 ms after presenting the visual stimulus. It is known that the neurons within visual cortex use spikes to represent the structural information of the input visual stimuli. To achieve a realistic neural simulation, spiking neural network (SNN) is often used as the neural network model. Specifically, SpikeNET [23], a simple yet powerful spiking neural network architecture, is often used to accomplish the face recognition tasks.

Based on the SpikeNet architecture, two state-of-the-art methods, batch learning rule [23], [24], [46] and on-line learning rule [47], [48], [49], have been proposed to differentiate the input faces. For each class, batch learning rule [23], [24], [46] computes the average cluster of the training samples and uses it to represent the whole training samples. However, it requires prior information (the number of samples within each class) to obtain average cluster and cannot represent well if the class has distributed samples. Unlike the batch learning rule, for each class, on-line learning rule [47], [48], [49] adaptively divides the training samples into various sub-clusters and uses these sub-clusters to represent the whole training samples. Both methods incorporate winner-take-all strategy within the classification procedure and work not well for overlapped classes within the decision space. Specifically, the batch learning rule

only uses average cluster while the on-line learning rule just relies on the nearest associated sub-cluster.

To overcome the above issues, based on the SpikeNET architecture, a novel learning rule inspired by the soft winner-take-all strategy has been proposed. Basically, it will assign a probability for each related class and the testing sample will be classified to the class with the highest probability. The proposed method dose not need to know the number of samples within each class and can represent well if the class has distributed samples. More importantly, since it incorporates soft winner-take-all strategy, the proposed learning rule adds certain flexibilities into the classification procedure and the input visual stimuli have the potential to be classified as other classes. Compared with the two state-of-the-art methods, experimental results on the ORL face database show the proposed method can achieve a better performance.

3.1 Background

Within traditional artificial neural network (ANN), neurons use different activation functions to transform the weighted sum of the input visual stimuli to output values. Specifically, the first generation of neural network model uses step function to generate the binary outputs while the second generation incorporates sigmoid function to obtain the analog values. However, with the development of the neuroscience, it is known that the neurons within the visual cortex use action potentials or spikes to represent the information. Spiking neural network (SNN), the third generation of neural network model, is often used to achieve a realistic neural simulation.

A typical neuron within SNN consists of three functional parts: dendrites, soma and axon [1], as shown in 3.1. Basically, as a "input device", the dendrites collects signals from other neurons and transmits them to the soma. The soma, also known as "central processing unit", performs a significant nonlinear processing step. It will generate an output signal if the total input signal exceeds a specific threshold. The axon is the "output device" that receives the output signal and delivers the signal to other neurons [1]. The neuronal signals consist of short electrical pulses and can be observed by placing a fine electrode close to the soma or axon of a neuron. The pulses,

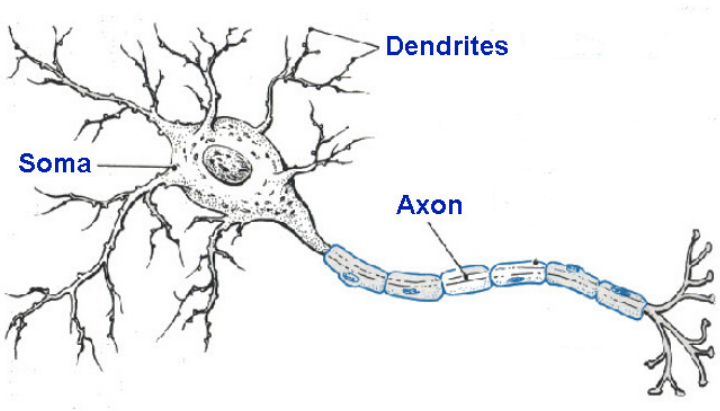


Figure 3.1: A typical neuron within SNN consists of three functional parts: dendrites, soma and axon [1]. As a "input device", the dendrites collects signals from other neurons and transmits them to the soma. The soma, also known as "central processing unit", performs a significant nonlinear processing step. It will generate an output signal if the total input signal exceeds a specific threshold. The axon is the "output device" that receives the output signal and delivers the signal to other neurons [1].

also known as action potentials or spikes, have an amplitude of about 100 mV and a typical duration of 1-2 ms [1].

A sequence of action potentials emitted by a single neuron is often called a spike train, which can also be considered as a chain of stereotyped events occur at regular or irregular intervals. Essentially, the form of the spike does not convey any information as all spikes of a given neuron look alike. Rather, it is the number and timing of spikes which matter [1]. It is impossible to generate a second spike during or immediately after a first one even with very strong input. The minimal distance between two spikes is called the absolute refractory period of the neuron. Thus, within a spike train, action potentials are usually well separated [1].

A neuron sends the spike trains to another neuron through a synapse. The sending/receiving neuron is often called as presynaptic/postsynaptic neuron. The effect of a spike on the postsynaptic neuron can be recorded with an intracellular electrode which measures the potential difference between the interior of the neuron and its surroundings. This potential difference is also known as the membrane potential. Without any spike input, the neuron will remain at the resting potential (a constant membrane potential). If the neuron receives a spike, its membrane potential will change and finally decay back to the resting potential. If the change is positive, the synapse is said to be excitatory. Otherwise, the synapse is inhibitory.

Based on the above phenomenons, SNN incorporates the temporal information within the processing procedure. Besides spiking rate, SNN can also use the specific spiking timing sequence, also known as spiking pattern, to represent the information. Such spiking pattern conveys spatiotemporal structural information of the input visual stimuli. From the SNN point of view, the output value of the second generation of neural network model can be considered as the average spiking firing rate of the neuron. However, it is possible to generate almost countless spiking patterns with the identical spiking rate. Furthermore, the specific spiking pattern of the neuron may convey more significant structural information than the spiking rate itself. Compared with the traditional ANN, the distinguishability of SNN has been greatly improved since it cannot only use spiking rate but also specific spiking patterns.

To investigate the distinguishability of a SNN, it is much needed to specify its framework. The framework defines the variables involved within the SNN and its topological structure. Electrophysiological studies [50], [51] indicate the neurons within the inferotemporal cortex (IT) respond selectively to faces only 80-100 ms after presenting the visual stimulus. It is known that the visual cortex is a hierarchical system consisting of several layers. The retina receives the input visual stimuli and transmits them to cortical visual areas V1, V2 and V4 before reaching higher visual areas in the anterior and posterior inferotemporal cortex. To simulate this hierarchical architecture, SpikeNET framework has been proposed and successfully applied to accomplish the face recognition tasks. It is a feed-forward SNN and includes three layers of retinotopic maps containing relatively simple integrate-and-fire neurons, with a first layer corresponding to the retina, the second one for V1 and the last one for V4-IT. Moreover, within the SpikeNET, neurons cannot spike more than once through the whole network.

Within SpikeNET architecture, code scheme plays an important role in transforming the input visual stimuli to the first spiking pattern. Normally, the input visual stimuli can be coded in either rate-coding scheme or time-coding scheme. The fundamental difference between them is the way it connects the visual saliency and the spike asynchrony. Traditionally, rate-coding scheme considers higher firing rate of spikes means

higher visual saliency. However, it ignores one important fact that this firing rate would only be reliable after a relatively long time window, which is unrealistic for the neurons within the inferotemporal cortex. Thus, rank order coding (ROC) [24], [98], [99], a simple yet powerful time-coding scheme, has been applied in this chapter. Within ROC scheme, for each neuron, it can fire at most one spike within a step time window. Such scheme considers the first spike wave conveys enough structural information for further processing.

Based on the SpikeNET architecture and ROC scheme, Delorme and Thorpe [23], [24], [46] proposed a novel batch learning rule and achieved satisfactory experimental results on ORL face database even incorporating certain degrees of image degradations into input images. Within the batch learning rule, the average cluster of the training samples within each class has been computed and used to represent the class. However, without accessing the number of samples within each class before the experiment, the average cluster cannot be obtained. Moreover, for classes with distributed samples, the average clusters cannot well represent the classes. To resolve the above issues, on-line learning rule [47], [48], [49] has been proposed. For each class, it adaptively divides the training samples into several sub-clusters and use them to represent the class. Within the classification procedure, both methods incorporate winner-take-all strategy. Specifically, to classify a testing sample, the batch learning rule just relies on the average cluster while on-line learning rule only considers the nearest relevant single sub-cluster. However, for overlapped classes within decision space, the testing sample will be surrounded by various associated classes and a interference cluster may be closest to the testing sample. Winner-take-all strategy will classify the testing sample to the wrong class and it has no error correction mechanism for this scenario. Therefore, for overlapped classes within decision space, both state-of-the-art learning rules cannot accurately classify the testing samples.

To address the above problem, based on the SpikeNET framework, a novel learning method inspired by the soft winner-take-all strategy has been proposed. Basically, for overlapped classes within decision space, it will assign a probability for each related class around the testing sample and this testing sample will be classified to the specific

class with the highest probability. Compared with the two state-of-the-art learning rules mentioned above, the proposed method achieved better experimental results on ORL face database.

3.2 Neural Model and Rank Order Coding Scheme

3.2.1 Neural Model

Neural model is essential for SNN as it defines a conductance principle for the neurons. As mentioned in the above section, the feed-forward SpikeNET [23] architecture includes three layers of retinotopic maps containing integrate-and-fire neurons. Since the proposed SNN framework is based on SpikeNET architecture, integrate-and-fire (IF) will still be used as the neural model in this chapter. Basically, each neuron within the SNN acts as a coincidence detection unit. It integrates afferent spikes until it reaches a threshold and then fires once.

A neuron integrates its inputs over time until it reaches a threshold, and fires a single action potential. The neuron is then reset and, after a certain refractory period, starts integrating information again. The latency of discharge of output neuron depends upon the relative order of firing of its afferent neurons. To simulate the above neuronal dynamics, the postsynaptic potential for neuron i at a time t can be calculated as:

$$PSP(i,t) = \Sigma mod^{order(a_j)} w_{j,i} \tag{3.1}$$

where $mod \in (0,1)$ is the the modulation factor, a_j represents the synaptic connection from afferent neuron j to neuron i and $w_{j,i}$ depicts the corresponding synaptic weight. $order(a_j)$ represents the firing rank from afferent neuron j to neuron i. By convention, $order(a_j) = +\infty$ if neuron j has not fire at time t, thus the corresponding term in the above sum equals to zero. Each time the neuron receives a spike, the efficiency of spike integration is divided by the factor mod. Neuron i will fire at time t if and only if:

$$PSP(i,t) \geq Threshold(i) \tag{3.2}$$

where $Threshold(i)$ represents the predefined threshold of the neuron i. This fast shunting inhibition mechanism leads to two key properties: First, the activation of the neuron is highest when the order of afferent discharges matches the pattern of weights. Second, the most strongly activated neurons fired first based on this kind of spike integration.

3.2.2 Rank Order Coding Scheme

For SNN, code scheme is essential since it transforms the analog values to real spikes. Traditionally, spiking rate is considered to represent most, if not all, information of the input visual stimuli. Basically, such scheme states that as the intensity of a stimulus increases, the frequency or rate of action potentials (spikes) increases. However, electrophysiological studies [50], [51] indicate the neurons within the inferotemporal cortex (IT) respond selectively to faces only 80-100 ms after presenting the visual stimulus. Such short time window is not enough to generate a meaningful spiking rate. Furthermore, the specific spiking timing sequence may convey much more important information than the spiking rate itself. Therefore, compared with rate coding scheme, temporal coding scheme is more universal.

Within SNN, when a postsynaptic neuron receives a presynaptic spike, its membrane potential will increase accordingly. The higher the intensity of the input visual stimulus, the more the neuron will be activated. After integrating various presynaptic spikes within a limited time window, the postsynaptic neuron will eventually fires a postsynaptic spike if its membrane potential reaches the threshold. The more the postsynaptic neuron activates, the less the latency of firing a spike will be.

Inspired by the above phenomenon, rank order coding (ROC), a time-to-first-spike temporal coding scheme, has been proposed. Within ROC, the higher the intensity of the input visual stimulus, the less the latency of firing a spike will be, as shown in Figure 3.2. For instance, the neuron received the input stimulus with the highest intensity will generate a spike with the lowest latency. Besides, such coding scheme only uses relative firing orders to represent the input visual stimuli. Furthermore, for each neuron within SNN, it will only be allowed to fire at most once. ROC scheme considers the

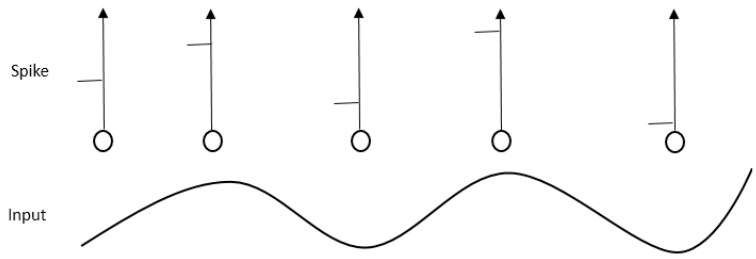


Figure 3.2: Rank order coding scheme diagram. The short horizontal line within the spike part represents the latency of firing a spike. It can be seen that the higher the intensity of the input visual stimulus, the less the latency of firing a spike will be.

first spike wave conveys enough information for further visual processing.

3.3 Proposed SNN Framework and its Learning Rule

It is know that the visual cortex is a hierarchical system consisting of various layers. These layers correspond to different visual areas within visual cortex. Specifically, the retina receive the input visual stimuli and then transmit them to visual areas V1, V2 and V4 before entering higher visual areas in the anterior and posterior inferotemporal cortex.

To simulate this hierarchical system, a feed-forward SpikeNET [23] architecture has been proposed. It includes three layers of retinotopic maps containing relatively simple integrate-and-fire neurons, with a first layer corresponding to the retina, the second one for V1 and the last one for V4-IT (inferotemporal cortex). With the development of neuroscience, it has been shown that the neurons can use DoG (difference of Gaussians) filters to simulate the ganglion cells within retina and Gabor filters to mimic the orientation selective cell within V1. However, for overlapped classes within the decision space, the winner-take-all strategy used in the third layer of SpikeNET architecture has no error correction mechanism. If a interference class is closest to the testing sample, this testing sample will be classified as the interference class. To reduce this error classification, the framework of the proposed SNN and its learning rule will be introduced in the following subsections, respectively.

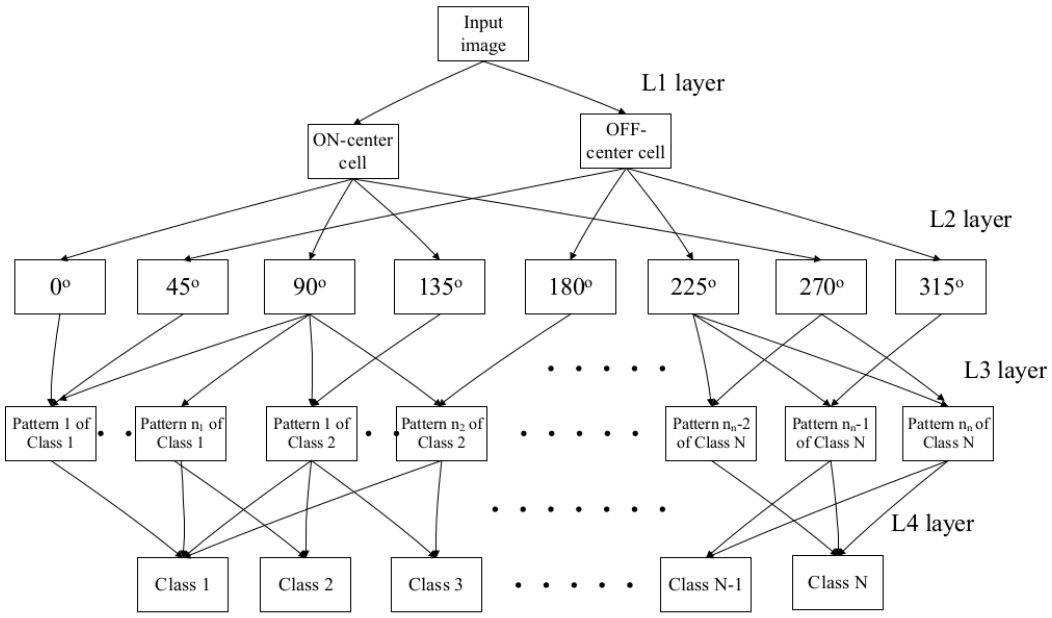


Figure 3.3: The framework of the proposed SNN. For simplicity, the lateral inhibition connections of the last two layers have not been included.

3.3.1 Framework of the proposed SNN

Inspired by the SpikeNET [23] architecture, a novel feed-forward SNN framework has been proposed by keeping the first two layers of SpikeNET architecture and replacing the last layer with two new layers. Thus, in total, the proposed SNN framework includes four layers and there are the lateral inhibition connections existed within the last two layers. Moreover, the proposed SNN is a fully connected network.

Within the proposed SNN framework, each layer contains a number of retinotopic maps and each map consists of various neurons. Specifically, within the first two layers, each neuron within each map corresponds to a specific pixel within the input image and the number of neurons within each map is the same as the number of pixels within the input image. Furthermore, for the third layer, only one neuron exists within each map. To reduce the intra-class variance, for each input image, the center of the receptive field of this neuron corresponds to the center of the face and such center is manually preselected. The center of the face is defined as the isobaric center of the nose and two eyes. Finally, for each map within the last layer, there is only one neuron located in the center of the map. For the sake of the simplicity, Figure 3.3 shows the framework of the proposed spiking neural network with only one scale.

The neurons within the first layer represent the ON and OFF-center ganglion cells

within the retina. To simulate these ganglion cells, difference of Gaussians (DoG) filters have been used. DoG is a feature enhancement algorithm that involves the subtraction of one blurred version of an original image from another. Within DoG, the input image is first smoothed by convolution with Gaussian kernel of certain scale σ_1 and the first smoothed image $g_1(x, y)$ can be computed:

$$g_1(x, y) = G_{\sigma_1}(x, y) * f(x, y) \tag{3.3}$$

where $*$ represents convolution operation and $f(x, y)$ is the input image, x and y represent the location of the pixel within the input image, $G_{\sigma_1}(x, y)$ is the Gaussian kernel with scale σ_1 and can be computed as following:

$$G_{\sigma_1}(x, y) = \frac{1}{\sqrt{2\pi{\sigma_1}^2}} exp(-\frac{x^2 + y^2}{2{\sigma_1}^2}) \tag{3.4}$$

With a different scale σ_2, a second smoothed image $g_2(x, y)$ can be obtained:

$$g_2(x, y) = G_{\sigma_2}(x, y) * f(x, y) \tag{3.5}$$

where $G_{\sigma_2}(x, y)$ is the Gaussian kernel with scale σ_2. The difference of these two smoothed images can be used to detect the edges in the input image.

$$g_1(x, y) - g_2(x, y) = (G_{\sigma_1} - G_{\sigma_2}) * f(x, y) = DoG * f(x, y) \tag{3.6}$$

Thus the DoG as a convolution kernel is defined as:

$$DoG = G_{\sigma_1} - G_{\sigma_2} = \frac{1}{2\pi}(\frac{e^{-(x^2+y^2)/2\sigma_1^2}}{\sigma_1} - \frac{e^{-(x^2+y^2)/2\sigma_2^2}}{\sigma_2}) \tag{3.7}$$

where the sum of the mask elements within the DoG kernel will be normalized to zero. The DoG used in this chapter has two scales. σ_1 can take two values of 0.9 and 1.5. Specifically, for each σ_1, σ_2 takes the value of $2\sigma_1$. Furthermore, the maximum and minimum convolution values will be normalized to $[-1, +1]$ so that the convolution results with different scales share a same range.

Within the second layer (L2), the neurons within each map try to mimic the orientation selective cells within primary visual cortex V1. It has been shown the simple orientation selective cells within V1 can be modeled by Gabor filters [100], [101]. In image processing, frequency and orientation representations of Gabor filters have been found to be particularly appropriate for texture representation and discrimination [100], [101]. The Gabor response $F^{\sigma,\theta}_{(x,y)}$ can be computed by the following equation:

$$\begin{aligned} F^{\sigma,\theta}_{(x,y)} &= exp(-\frac{(x'^2 + \gamma^2 y'^2}{2\sigma^2})cos(2\pi\frac{x'}{\lambda} + \varphi) \\ x' &= xcos(\theta) + ysin(\theta) \\ y' &= -xsin(\theta) + ycos(\theta) \end{aligned} \tag{3.8}$$

where θ is the orientation, x and y represent the location of the pixel within the input image, x' and y' describe the location of the same pixel after rotating θ, φ the phase offset, λ the wavelength, σ the standard deviation of the Gaussian factor and γ the aspect ratio. As mentioned in the above paragraph, there are two scales within the first layer. Inspired by the SpikeNET architecture [23], for each scale, eight orientation maps have been used and each one being selective to different direction every other 45^o. The Gabor filters are normalized globally so that neurons having direction selective cells as inputs can have PSP varying in the same range $[0, PSP_{max}]$.

For each input image, a new map has been created within the third layer and only one neuron exists within this map. To reduce the intra-class variance, the center of the receptive field of this neuron is manually preselected and it is defined as the isobaric center of the nose and two eyes. There are lateral inhibition connections existed in this layer so that only k neurons fired eventually.

Given the number of classes is N, there are N maps within the last layer and each map corresponds to a specific class. Only one neuron exists in each map and it is fully connected to the previous layer. There are lateral inhibition connections existed in this layer so that only one neuron fired eventually. The input image will be classified to the class with a spike firstly fired within the associated map.

3.3.2 Learning Rule

Based on the SpikeNET architecture, two state-of-the-art methods, batch learning rule and on-line learning rule, have been proposed to accomplish the face recognition tasks. For the classification procedure, these two methods share a common methodology: winner-take-all strategy. Specifically, the class is represented using the average cluster within the batch learning rule while the on-line learning rule use various sub-clusters to represent the class. For overlapped classes within the decision space, the input images will not be correctly classified with such winner-take-all strategy.

To address this issue, a novel learning rule inspired by the soft winner-take-all strategy has been proposed. Specifically, given a input image, the proposed learning rule will assign a probability for each associated class and the input image will be classified to the class with the highest probability. Normally, for overlapped classes within the decision space, the testing sample tends to be surrounded by clusters belonging to different classes. In this case, for each associated class, the number of clusters surrounded the testing sample and their distances from the testing sample play an important role in the classification procedure. Basically, the less the distance, the earlier the firing of a spike; The more the number of clusters, the more the neuron activated. Therefore, within the last layer, the probability $P(i,t)$ for the associated class i at the time t can be calculated as:

$$P(i,t) = \Sigma mod_p^{dis(a_j)} w_{j,i}^p \tag{3.9}$$

where $mod_p \in (0,1)$ is the the modulation factor, a_j represents the synaptic connection from afferent neuron j to neuron i and $w_{j,i}^p$ depicts the corresponding synaptic weight. $dis(a_j)$ represents the firing rank from afferent neuron j to neuron i. By convention, $dis(a_j) = +\infty$ if neuron j has not fire at time t, thus the corresponding term in the above sum equals to zero. Each time the neuron receives a spike, the efficiency of spike integration is divided by the factor mod_p. Neuron i will fire at time t if and only if:

$$P(i,t) \geq Thr(i) \tag{3.10}$$

where $Thr(i)$ represents the predefined threshold of the neuron i.

When the order of the afferent discharges matches the pattern of weights, the activation of the neuron is highest and thus it will fire the first spike. Therefore, the whole learning procedure can be described as follows:

1. Propagate a sample k of class K for training into L1 (ganglion cells) and L2 (orientation selective cells).

2. Create a new map in L3 for sample k and train the synaptic weights using the following equation:

$$\triangle w_{j,i} = mod^{order(a_j)} \tag{3.11}$$

where $\triangle w_{j,i}$ is the change of the synaptic weights between neuron j of the L2 layer and neuron i of the L3 layer. a_j represents the synaptic connection from afferent neuron j to neuron i and $order(a_j)$ is the order of arrival spike from neuron j to neuron i. mod represents the modulation factor. Only one neuron exists within each map. To reduce the intra-class variance, the center of the receptive field of this neuron is manually preselected and it is defined as the isobaric center of the nose and two eyes. Within the third layer, only k neurons are allowed to fire spikes. At time t, if the neuron has not received any spike from the synaptic connection a_j, the corresponding $order(a_j)$ will be set to $+\infty$.

3. Create a new map for each class in L4 layer and train the synaptic weights using the following equation:

$$\triangle w^p_{j,i} = mod_p^{dis(a_j)} \tag{3.12}$$

where $\triangle w^p_{j,i}$ is the change of the synaptic weights between neuron j of the L2 layer and neuron i of the L3 layer. a_j is the synaptic connection from afferent neuron j to neuron i and $dis(a_j)$ represents the order of arrival spike from neuron j to neuron i. mod_p is the modulation factor. Within the layer L4, there is only one neuron within each map and it is fully connected to previous layer. At time t, if the neuron has not received any spike from the synaptic connection a_j, the corresponding $dis(a_j)$ will be set to $+\infty$. Moreover, if the neuron receive a spike from a synaptic connection a_j originating from a map belonging to other classes, the corresponding $dis(a_j)$ will be set to $+\infty$.

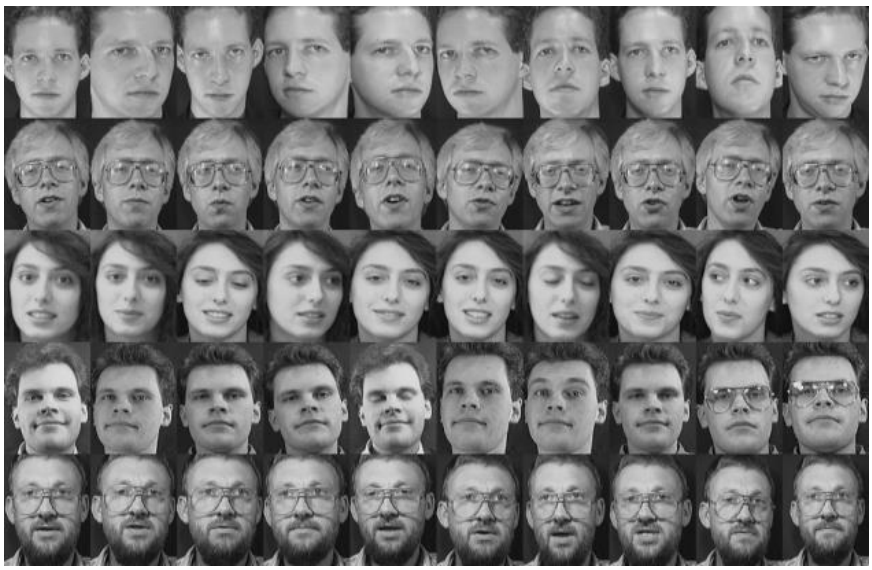


Figure 3.4: 10 views of random 5 persons within the ORL face database.

4. Stop the learning procedure when all the training samples have been fed into the SNN.

3.4 Experiments

To verify the proposed SNN and its learning rule, this chapter uses the AT&T Cambridge Laboratories face database [102] (also known as the ORL face database) to test the accurate classification performance. Figure 3.4 shows 10 views of random 5 persons within the ORL face database. This image database include 400 faces and each of them has been resized to 28×23 pixels. It includes 10 views of 40 persons, which includes both sexes, from different origins, with or without various characteristics such as beard, glasses or mustache. Views were frontal($\pm 30^o$).

3.4.1 Parameter Settings

Within the proposed SNN framework, for fair comparison with the batch learning rule and the on-line learning rule, the parameters used in the first three layers take the same value as SpikeNET architecture [23]. To obtain the best classification performance, the parameter settings demonstrated in Table 3.1 have been used in this chapter. Specifically, k and mod_p are optimized to achieve the best classification performance on the ORL face database while others take the same values as [23].

Table 3.1: Parameter settings of the proposed SNN.

Parameter	Description	Value
σ_1	scale of first Gaussian filter [1]	$0.9, 1.5$
σ_2	scale of second Gaussian filter [1]	$2\sigma_1$
θ	orientations of Gabor filter [1]	every other 45^o
φ	phase offset of Gabor filter [1]	0
λ	wavelength of Gabor filter [1]	5.0
σ	standard deviation of Gabor filter [1]	2.5
γ	aspect radio of Gabor filter [1]	1 ms
k	number of neurons fired in L3 layer [2]	8
mod	model factor of first three layers [1]	0.995
mod_p	model factor of last layer [2]	0.8

[1] Take the same value as [23].
[2] Optimized to achieve the best classification performance.

3.4.2 Experimental results and Analysis

Like batch learning rule [23], [24], [46] and on-line learning rule [47], [48], [49], for each person, we will randomly choose 8 views from 10 views as the training samples and use the remaining 2 views as the testing samples. Specifically, in this chapter, we will use 5-fold cross validation experiments on ORL face database to compare different methods.

Table 3.2 demonstrates the correct classification results of each fold and its average values. To achieve the best performance, on-line learning rule uses $c = 0$ and $sim_{thr} = 1$, where c represents a measure of similarity between maps and sim_{thr} is a chosen threshold. If $c = 1$, then only the training sample evokes the output spike [47], [48], [49]. If the similarity of the two maps reaches the threshold sim_{thr}, they will be merged into a single map [47], [48], [49]. Figure 3.5 shows the 5-fold cross-validation results using three different methods. For each learning method, each point represents each fold result of the 5-fold cross-validation experiment. The trend line is drawn through the average of the results for each learning method and the error bars indicate the standard deviation.

Table 3.2: 5-fold cross-validation classification performance by using three different methods.

Fold	Method		
	Proposed method	On-line [47], [48], [49]	Batch [23], [24], [46]
1	0.8375	0.8125	0.6375
2	0.8125	0.8125	0.6
3	0.825	0.8	0.6
4	0.8375	0.8375	0.625
5	0.8125	0.8125	0.5625
Average	0.825	0.815	0.605

- Note: 0.8375 in this table means 83.75% correct classification rate. To achieve the best performance, in this experiment, on-line learning rule uses $c = 0$ and $sim_{thr} = 1$, where c represents a measure of similarity between maps and sim_{thr} is a chosen threshold. If $c = 1$, then only the training sample evokes the output spike [47], [48], [49]. If the similarity of the two maps reaches the threshold sim_{thr}, they will be merged into a single map [47], [48], [49].

For the distributed samples within a class, the average cluster obtained from batch learning rule may not be a good representation. Moreover, the batch learning rule needs to know the number of samples before the experiments. To address the above problems, for each class, on-line learning rule adaptively divides the training samples into various sub-clusters and uses these sub-clusters to represent the whole training samples. For each class, if all associated maps have been merged into one cluster within the on-line learning rule, batch learning rule can be seen as a special instance of on-line learning rule. However, both methods incorporate winner-take-all strategy within the classification procedure and work not well for overlapped classes within the decision space. Specifically, the batch learning rule only considers average cluster within each class while the on-line learning rule just relies on the nearest single sub-cluster.

Unlike those two state-of-the-art methods, the proposed learning method is inspired by the soft winner-take-all strategy. Basically, it will assign a probability for each related class and the testing sample will be classified to the class with the highest probability. Such probability is computed by collecting the local statistical distribu-

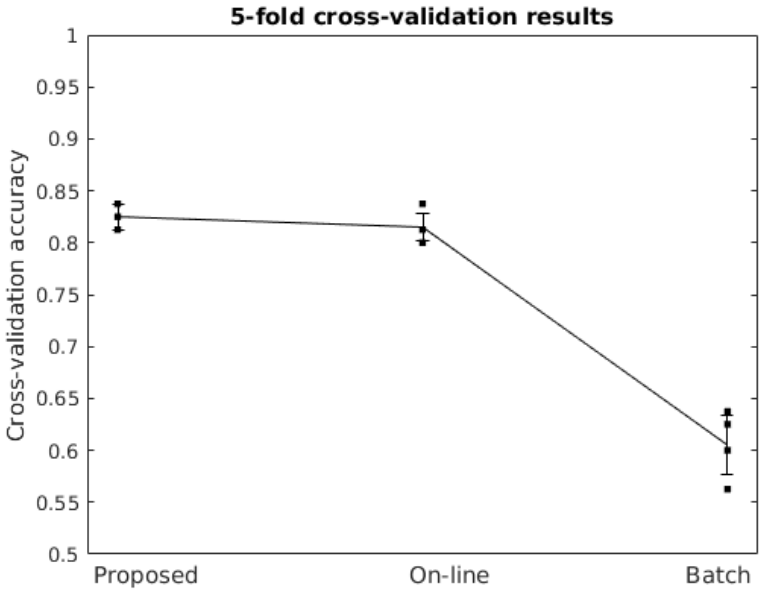


Figure 3.5: 5-fold cross-validation results using three different learning rules. For each learning method, each point represents each fold result of the 5-fold cross-validation experiment. The trend line is drawn through the average of the results for each learning method and the error bars indicate the standard deviation.

tion information around the undetermined sample. If the proposed learning method only cares about the nearest sub-cluster, it can replace the on-line learning rule. From this point of view, on-line learning rule can be considered as a special instance of the proposed learning rule.

3.5 Conclusion

In this chapter, based on the SpikeNET architecture [23], two state-of-the-art learning methods - batch learning rule and on-line learning rule - have been proposed to accomplish the face recognition tasks. Specifically, for each class, batch learning rule computes the average cluster of the samples and on-line learning rule adaptively divides the samples into various sub-clusters. Within the classification procedure, both methods incorporate winner-take-all strategy and then classify the input image according to one single cluster. Batch learning rule just uses average cluster while on-line learning rule only relies on the nearest sub-cluster. However, such winner-take-all strategy works not well for overlapped classes within the decision space. To address the above problem, a novel learning rule inspired by soft winner-take-all strategy has

been proposed. For each input image, it will assign a probability for each associated class and the input image will be classified to the class with the highest probability. Experimental results on ORL face database show our proposed method can provide satisfactory results compared with the above two state-of-the-art methods.

However, those attempts are still in their initial stage as the proposed SNN still needs more balanced features in terms of invariance and differentiation. In other words, the abstract features we used in this paper are still not fully capable for complicated visual pattern recognition tasks. Therefore, extending the framework of the proposed SNN and thus abstracting higher balanced features occupies the priority position in future works.

Chapter 4

Fast Learning for Visual Pattern Recognition using Spike-timing Dependent Plasticity

Based on evolution theory, reliable and fast visual pattern recognition is vital for most animal species. Without such capability, the specie will be selected against by nature. In most cases, new visual pattern should be learned in a limited time window to adapt to new environments or changes promptly. Recent research shows a biological brain can process complicated real-life recognition scenarios at milliseconds scale [2]. With the advancement of the neuroscience, it is known that neurons use spikes to represent information. To achieve a realistic neural simulation, spiking neural network (SNN) is often used as the neural network model. Compared with traditional artificial neural network, SNN cannot only use spiking rate but also specific spiking timing sequence to represent the information, which greatly improves the distinguishability.

To achieve a fast visual pattern recognition, various methods [17], [68] using the spiking timing sequence have been proposed. However, the Tempotron rule used in [68] needs a supervisory error signal to update the synaptic weights and this supervisory error signal, in real scenarios, is hard to obtain. Unlike [68], the authors in [17] use an unsupervised spike-timing dependent plasticity (STDP) method to learn the local intermediate features. It is known that feature extraction is only an important component of pattern recognition. Thus, in [17], the STDP learning method itself is

not enough to accomplish the visual pattern recognition tasks. To address the above issues, in this chapter, a novel feed-forward SNN framework has been proposed. To strike a trade-off between speed and performance, the proposed SNN uses the first two layers of HMAX model to generate the local invariant features and applies an unsupervised STDP learning method to train the synaptic weights. Unlike the above methods, the proposed SNN does not need the supervisory error information and the applied unsupervised STDP learning rule is enough to finish the pattern recognition task. Experimental result on MNIST database shows the proposed method can efficiently achieve an acceptable accuracy within a limited time window.

4.1 Background

In real scenarios, it is often impossible to obtain the whole training database as, in most cases, they are gradually generated over time. Real-time learning method has to learn with limited samples often in real time, without the opportunity to learn the whole training database. Various machine learning applications have the similar learning situation - a rescue robot needs to learn to recognize individuals on the spot, an identification or human recognizing system needs to cope with new criminals in any public areas, an health-care intelligent machine need to learn to cope with new patients quickly with limited information, all of which need to learn in a fast speed in real time.

It is proved that learning visual patterns in real time proposes a huge challenge - the algorithms underlying should process a large volume of visual data in an extremely short period of time. However, for a biological brain, coping with these large volume of visual data in real time is a effortless work. A human brain may contain more than 10 billion densely packed neurons that are connected to an intricate network with numerous spikes are emitted in each millisecond. The mechanism of how these spikes are generated and processed is still an open question. But this has not prevented researchers from proposing biological plausible methods for pattern recognition, as briefed below.

Various spike coding schemes such as rate-based coding [103],[104] and spike timing-based coding [27], [75] have been proposed to transform the input analog fea-

tures into the real spike trains. A human brain can recognize objects in a few tens of milliseconds in a very complicated real-life scenarios. It is almost impossible for rate-based SNN to generate meaningful spiking rate in such a short time window. Recent studies show there are repeating spatiotemporal spiking patterns with millisecond precision existed in vitro and vivo [29]. For rate-based SNN, it is impossible to discriminate the repeating spiking patterns from the distractor with the same spiking rate as the repeating spiking patterns. In contrast, given a appropriate learning rule, the spike timing-based SNN can still extract a repeating pattern within a quite short time window [29]. Furthermore, a spiking pattern itself conveys significant structural information, which cannot be represented by spiking rate alone. Thus, in this chapter, rank order coding (ROC) scheme [24], [98], [99], one type of spike timing-based scheme, will be used to translate the analog features to spiking patterns for further processing. These translated spiking patterns convey unique spatiotemporal structural information corresponds to the inputs images and can be classified via machine learning rules.

To learn these spiking patterns, spike timing dependent plasticity (STDP), one of the most biological plausible learning rule [41], [43], [105], [106], [107], has been applied in this chapter. Like Hebb's postulate [40], it emphasizes the causality of the related spikes and adjusts the efficacy of synaptic connections based on the relative timing of postsynaptic spikes and its input presynaptic spikes. As an unsupervised learning rule, STDP does not need prior information or teaching signal in learning. It will adaptively change the synaptic efficacy and try to extract the most notable spiking pattern.

Several papers using SNN and spike timing-based coding scheme have been proposed for visual pattern recognition tasks [17], [68]. Inspired by HMAX model [108] which consists of four layers (S1-C1-S2-C2) to simulate ventral stream (V1-V2-V4-IT), Thorpe et. al [17] have investigated the learning of C1 to S2 synaptic connections through STDP and suggested that temporal coding may be a key to understand the phenomenal processing speed achieved by the visual system. However, feature extraction is only a significant part of pattern recognition. Thus, in [17], STDP itself was not enough to achieve the spiking pattern recognition. In [68], the authors proposed

a novel SNN with supervised learning rule and temporal coding scheme to generate the spike pattern. Such SNN system and its supervised learning rule achieved relatively good classification rate when conducting simple random sampling experiments on the MNIST database. However, the supervisory error signal used in such supervised learning rule, in real scenarios, is hard to obtain.

To overcome the drawbacks mentioned above, a novel method taking advantages of SNN and spiking timing-based coding scheme has been proposed in this chapter. Unlike [17], in our method, S1 and C1 are only for feature extraction; features are translated to spiking pattern from C1; STDP is used after spiking encoding layer for pattern recognition. To further speed up the leaning process, we only use S1 and C1 to extract visual features. Unsupervised learning rule is employed in the proposed method to make it more practical. Dynamic threshold is introduced to guarantee each training sample can be fully exploited in learning. Fast simple random sampling experiments using MNIST database are carried out to evaluate the efficiency and accuracy of the proposed method.

4.2 Framework of the proposed SNN

In real scenarios, the visual pattern recognition application often involves vast data dimensions and exists significant variability in terms of inter-class and intra-class. Thus, reducing the data dimensionality and obtaining the generalized features is an inevitable choice for most visual pattern recognition applications. Those generalized features should contain the most distinguishable and unchangeable characteristics of the original input image [9], [10]. Furthermore, to transmit the structural information from the input visual stimuli to the proposed SNN, the analog features need to be transformed to spike trains. Rank order coding has been chosen as the coding scheme to achieve the above goals. Based on an appropriate learning strategy, different spiking patterns would be recognized eventually.

The whole framework of the proposed SNN includes three main layers: feature extracting layer, spiking encoding layer and output layer. Figure 4.1 shows the framework of the proposed spike timing-based feed-forward SNN. Within feature extracting

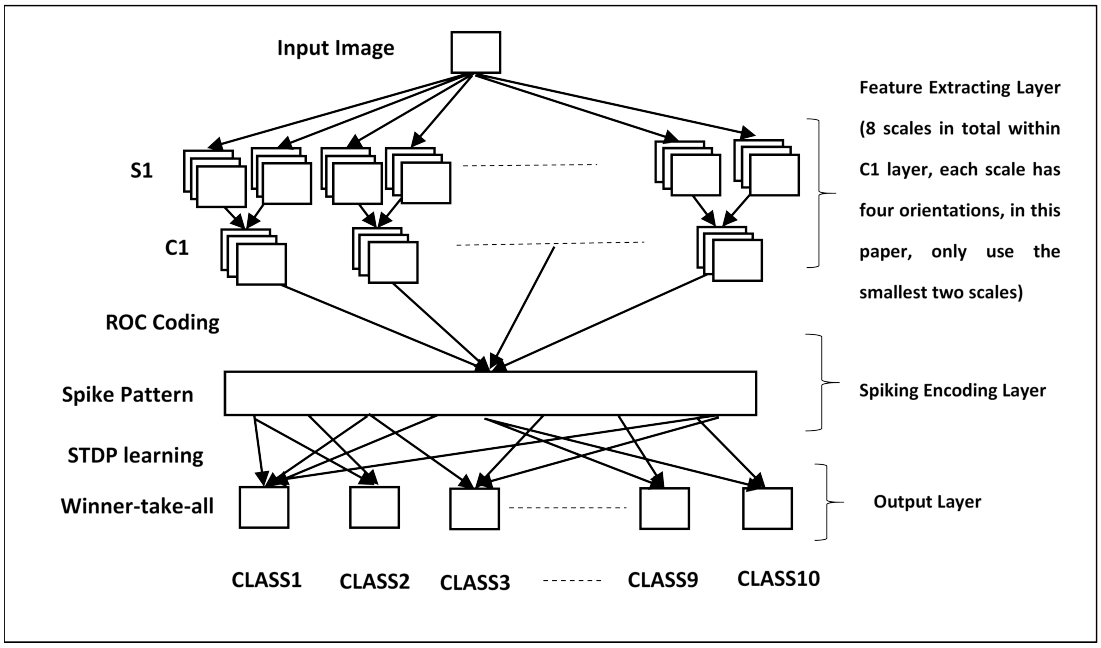


Figure 4.1: The framework of the proposed spike timing-based feed-forward SNN. For simplicity, the lateral inhibition connections of the last layer have not been included.

layer, the input images have been used to generate the corresponding $C1$ features with different scales and directions. Based on the ROC scheme, those $C1$ features have been transformed into spike trains. Each input image has its own corresponding spike pattern after those two layers. The output layer uses STDP learning rule and winner-take-all strategy to train the synaptic efficacy matrix with specific selectivity to the input image. In this chapter, within each output map, there is only one neuron for each specific class. The details of the three layers will be introduced in the following sub-sections.

4.2.1 Feature Extracting Layer

It has been shown that visual processing is hierarchical, aiming to build an invariance to position and scale first and then to viewpoint and other transformations [109]. Inspired by such phenomenon, HMAX model [108], [110], a hierarchical system that closely follows the organization of visual cortex, has been proposed to build an increasingly complex and invariant feature representation by alternating between a template matching and a maximum pooling operation. This hierarchical system includes four layers: $S1$ layer, $C1$ layer, $S2$ layer and $C2$ layer. The simple S units within $S1$ and $S2$

combine their inputs with a bell-shaped tuning function to increase selectivity while the complex C units within $C1$ and $C2$ pool their inputs through a maximum operation, thereby increasing invariance. Basically, for an input map, a template matching operation will obtain the convolution of the map by using a specific template (kernel). As a downsampling technique, for a local domain, a maximum pooling operation will compute the maximum value from the local domain and only use it to represent the whole local domain.

For the sake of efficiency and simplicity, only the first two layers of HMAX model ($S1$ layer and $C1$ layer) have been used to extract the expected features in this chapter. Since simulating the complex cells in $V1$, the features extracted from $C1$ layer convey a relatively local invariance. Specifically, $S1$ features can be generated after applying Gabor filters with vary scales and orientations to the input image, which correspond to the classical simple cells in the primary visual cortex. It has been shown that Gabor response $F^{\sigma,\theta}_{(x,y)}$ can provide a good model of cortical simple cell receptive fields [100], [101], which can be computed according to Equation 3.8. In this chapter, we choose the same parameters settings as the HMAX model [108] that is using a range of sizes from 7×7 pixels to 37×37 form the pyramid of scales, and θ takes four orientations $(0^o, 45^o, 90^o, 135^o)$. Notably, those $S1$ features have been normalized to a predefined range $[-1, 1]$ so that input images with the same contrast will generate same $S1$ features.

$C1$ unit pool over retinotopically organized afferent $S1$ units from the previous $S1$ layer with the same orientation and from the same scale band. It corresponds to the cortical complex cells in $V1$, which convey certain invariance to local transformation. The vital part of $C1$ layer is the max pooling operation, which increases the tolerance to transformation from $S1$ layer to $C1$ layer. Basically, the response $r^{\sigma,\theta}_{(x,y)}$ of a complex $C1$ unit corresponds to the maximum response of its m incoming $\left(F^{\sigma,\theta}_{(x_1,y_1)}, \cdots F^{\sigma,\theta}_{(x_m,y_m)}\right)$ responses from the previous $S1$ layer with two adjacent scales:

$$r^{\sigma,\theta}_{(x,y)} = \underset{j=1\cdots m}{max} F^{\sigma,\theta}_{(x_j,y_j)} \tag{4.1}$$

Unlike the traditional HMAX model [108], to speed up the processing speed, only

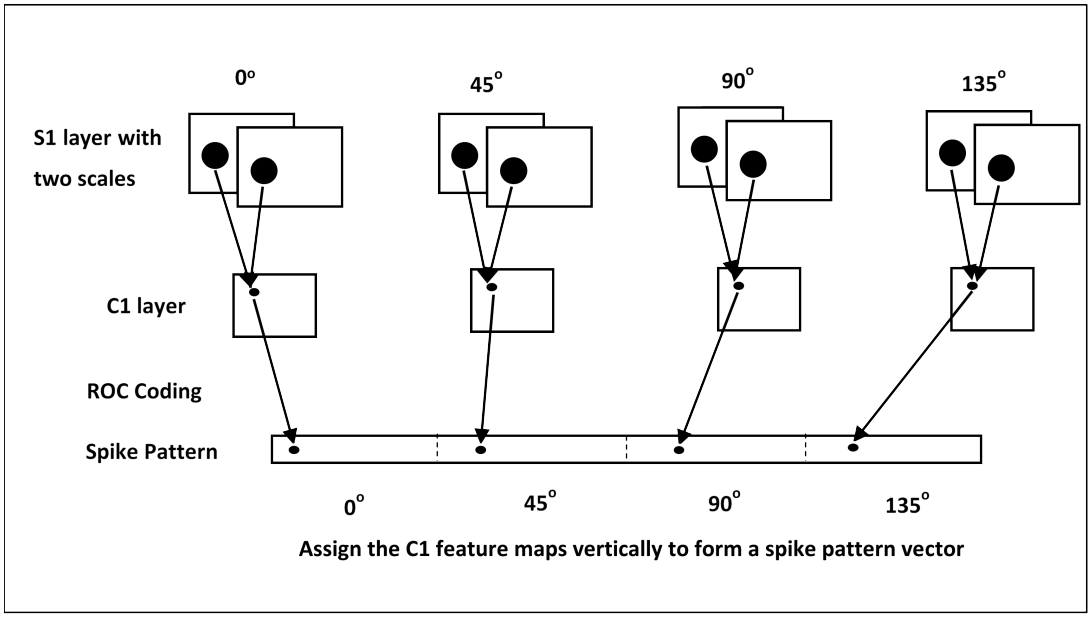


Figure 4.2: The generating procedure of spike pattern with the first smallest scale within C1 layer as example.

the first two smallest scales of $C1$ layer have been used to obtain the spike pattern, as shown in Figure 4.2. The whole generating procedure can be summarized as follows:

1. Create $S1$ maps using the Gabor responses with the same settings as HMAX model [108]. Unlike the HMAX model [108], to speed up the processing speed, only the two smallest scales have been used to generate the $S1$ maps.

2. Downsampling the $S1$ maps using the maximum pooling operation and generate the $C1$ maps. Specifically, for each $S1$ map with the smallest scale, a local sliding window with the size of 8×8 has been applied to all four orientations $(0^o, 45^o, 90^o, 135^o)$ and the maximum value within this sliding window will be computed. This maximum value will be used to represent the whole sliding window. Note, there are overlaps between the adjacent sliding windows and the overlapping size is 4×4. For the second smallest scale, the sliding windows size is 10×10 and the overlapping size is 5×5.

Throughout the feature extracting layer, only the maximum $S1$ feature within the corresponding sliding window has been selected and all others have been discarded. Such max pooling operation can not only ensure the generated $C1$ feature having certain local invariance but also reduce the dimension of the whole data set.

(a) Original image

(b) $S1$ features map

(c) $C1$ features map

Figure 4.3: One input image and its associated $S1$ and $C1$ features maps ($C1$ map has been enlarged for better viewing). (a) is one input image. The $S1$ features map in (b) is intermediate features generated by Gabor filter with a specific orientation and scale. $C1$ features map shown in (c) represents the local invariant features.

Compared with simple cells within $V1$, the cortical complex cells tend to have larger receptive fields [108]. For each orientation, $C1$ unit pool over two $S1$ maps with adjacent filter sizes. Those maps have the same dimensionality but they are the products of different filters. By sub-sampling with local cells with predefined sizes, $C1$ units takes the maximum response from the associated cell grid. Thereby, the dimension has been reduced with this max pooling operation. The bigger the cell grid takes, the lesser the dimensionality of $C1$ maps will be.

In this chapter, the $S1$ features have been normalized to $[-1, 1]$ and the $C1$ features will naturally have the range $[0, 1]$. By doing this, one can easily design the linear transformation strategy used in ROC scheme. For the sake of simplicity, besides showing the input image, Figure 4.3 only shows one associated $S1$ features map and $C1$ features map. Template matching operation used in $S1$ layer generates orientation edge packages with certain selectivity, while max pooling operation in $C1$ layer achieves dimensionality reduction and invariance to local transformation.

4.2.2 Spiking Encoding Layer

Compared with spiking rate, spiking timing sequence may convey more significant information of the input visual stimuli. In this chapter, rank order coding (ROC), a simple yet powerful temporal coding scheme, has been used to generate spikes from the input visual stimuli. As discussed in the section 3.2.2, ROC uses relative firing orders to represent the input visual stimuli.

However, given the same relative order, there are countless spiking pattern combinations but the normal ROC scheme can only distinguish one single pattern (according to the relative order). To address this drawback, we will use specific absolute spiking timings to replace the relative orders. As mentioned in the section 3.2.2, the higher the intensity of the input, the less the latency of firing a spike will be. Therefore, for a specific feature response r within $C1$ layer, the corresponding spiking timing t (with the unit s) can be computed as follows:

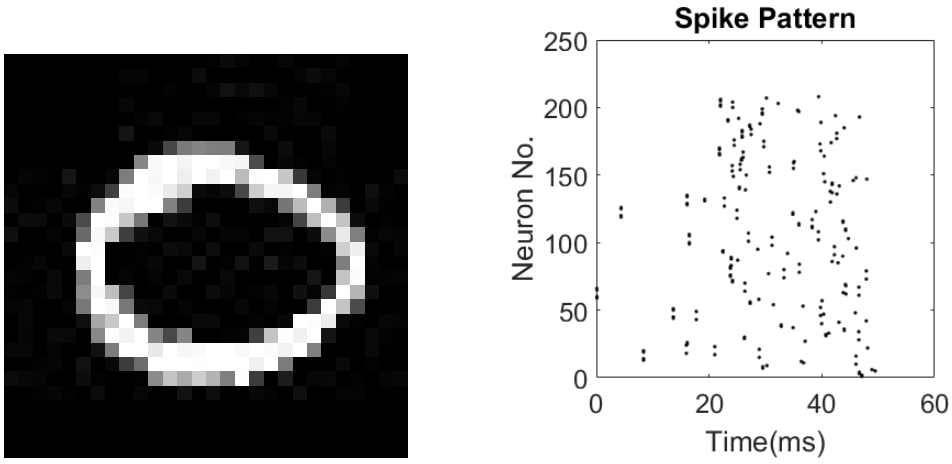
$$t = p(max_r - r) \tag{4.2}$$

where max_r is the maximum value of all related $C1$ features in the receptive filed and p is a positive constant within the range from 0 to 1. Here, p is used to control the length of the processing time window of a specific spiking pattern. As mentioned in the above section, the $C1$ feature response r has been normalized to $[0, 1]$. If p takes 1, then the maximum processing time window of a specific spiking pattern will be 1 s. As Figure 4.2 shown, given the $C1$ maps with all four orientations, the exact spike timing of the corresponding $C1$ feature can be computed using the Equation 4.2. By vertically assigning the $C1$ map, each $C1$ map with certain orientation has been transformed to a horizontal vector with the same orientation. For the sake of simplicity, only one scale of $C1$ layer has been shown in Figure 4.2.

Through the first two layers, the input images will be transformed into spiking patterns with spatiotemporal structural information. Figure 4.4 shows one input image and its spike pattern generated from the first two layers. Specifically, input images belonging to the same class should generate similar spiking patterns with little intra-class variance. In contrast, input images belonging to the different classes should obtain spiking patterns with significant differences.

4.2.3 Output Layer

Within the output layer, there are 10 maps and each map corresponds to a specific class. Note, each map only has one neuron. The neurons within spiking encoding layer and output layer are fully connected so that each output neuron receive synaptic con-



(a) Input image (b) Spike pattern

Figure 4.4: Input image and its spike pattern generated from the first two layers. Here, the spikes are generated by local invariant $C1$ features.

nections from all the neurons within spiking encoding layer. To achieve a competitive learning, there are lateral inhibition connections within the last layer. The output layer uses winner-take-all strategy so that the first fired neuron will strongly depress the rest neurons within the output layer from firing spikes and thus the input image will be considered as the class associated with the fired neuron.

Broadly speaking, the visual pattern recognition application should involve two main tasks: spiking pattern generating and spiking pattern learning. Specifically, the first two layers of the proposed SNN achieve the spiking pattern generating task while the output layer accomplishes the spiking pattern learning task.

4.3 Neuron Model and STDP Learning Rule

To build a successful SNN framework, neuron model and spiking pattern learning rule are two essential building blocks. The former one defines the the conduct principle of the spiking neurons while the latter one provides the specific learning steps for the synaptic connections.

4.3.1 Neuron Model

Within SNN, various neuron models such as leaky integrate-and-fire (LIF) and spike response model (SRM) have been commonly used as the conduct principle of

the spiking neurons [1]. In fact, SRM can be considered as a generalization of LIF model.

Specifically, LIF model model acts as a coincidence detector and the causality between local spikes has been emphasized. When the post-synaptic neuron receives a spike from its presynaptic neuron, the responding post-synaptic potential (PSP) will be generated. One can use certain time course to depict this dynamic PSP change. In leaky integrate-and-fire model, the post-synaptic potential will gradually decrease if no spikes received since last received spike. Therefore, in order to generate a post-synaptic spike, this post-synaptic neuron needs to receive lots of spikes within a relative small time window so that its PSP can reach the predefined threshold.

The dynamic procedure of LIF model can be summarized as follows: when a post-synaptic neuron receives presynaptic spikes, it will generate dynamic synaptic current and this dynamic current will thus produce dynamic synaptic voltage. A postsynaptic spike will fired if the dynamic synaptic voltage reaches the predefined postsynaptic potential threshold. As shown in [1], the dynamic postsynaptic current can be expressed as follows:

$$I_i(t) = \sum_j w_{ij} \sum_f \alpha\left(t - t_j^{(f)}\right) \tag{4.3}$$

where $t_j^{(f)}$ represents the time of the f-th spike of the j-th presynaptic neuron; w_{ij} is the strength of the synaptic efficacy between neuron i and neuron j. $\alpha(t)$ is the time course function, which can be expressed as follows:

$$\alpha(\mathrm{t}) = \alpha \frac{1}{\tau} \exp\left(-\frac{t}{\tau}\right) \Theta(t) \tag{4.4}$$

where Θ is the Heaviside step function with $\Theta(t) = 1$ for $t > 0$ and $\Theta(t) = 0$ else. τ is the time constant. For a given time-varying input current $\mathrm{I}(\mathrm{t})$, the membrane potential $\mathrm{V}(\mathrm{t})$ can be computed as follows:

$$\mathrm{V}(\mathrm{t}) = V_r \exp\left(-\frac{t - t_0}{\tau_m}\right) + \frac{R}{\tau_m} \int_0^{t-t_0} \exp\left(-\frac{s}{\tau_m}\right) I(t - s)\, ds \tag{4.5}$$

where the initial condition $\mathrm{V}(t_0) = V_r$ and τ_m is the membrane time constant. R

represents the resistance. This equation describes the dynamics of the membrane potential between successive spiking events. When the membrane potential reaches the threshold V_{thr},

$$V(t) \geq V_{thr} \tag{4.6}$$

it will fire a spike, followed by the absolute refractory period (the membrane potential is resets to V_r and the absolute refractory time is T_{rf}) and then start to evolve afterwards.

In this chapter, a dynamic post-synaptic potential threshold has been proposed in the training period. Given the input image, for the first run, we collect the generated membrane potentials from the corresponding spiking pattern instead of setting post-synaptic potential threshold. The associated postsynaptic potential threshold has been set to a percentage of the maximum value obtaining from collecting all membrane potentials within the predefined time window, as shown in follows:

$$V_{thr} = k \times \max(V(t)) \tag{4.7}$$

where V_{thr} is the post-synaptic potential threshold and $\mathrm{V}\,(\mathrm{t})$ represents membrane potential. $\max\left(\mathrm{V}\,(\mathrm{t})\right)$ is the maximum value of membrane potential within the predefined spiking time window and k depicts a positive constant within the range $[0, 1]$. By doing this, each input image can be ensured to be trained during the learning procedure. Such scenarios with only a little part of training samples have been actually used (especially those training samples with relatively large intra-class variance) will be avoided. Each input spike pattern will contribute its part to the final learning efficacy matrix with certain selectivity.

Figure 4.5 uses the same input image as Figure 4.4 and shows its spike pattern, dynamic current and membrane potential. It can be seen that the oscillation of dynamic current depends on the closeness of the local spike packages. When the post-synaptic neuron receives lots of spikes from presynaptic neurons in a short time window, the dynamic current will increase dramatically and then gradually decline if no spikes received afterwards. This dynamic current will generate membrane potential in the post-

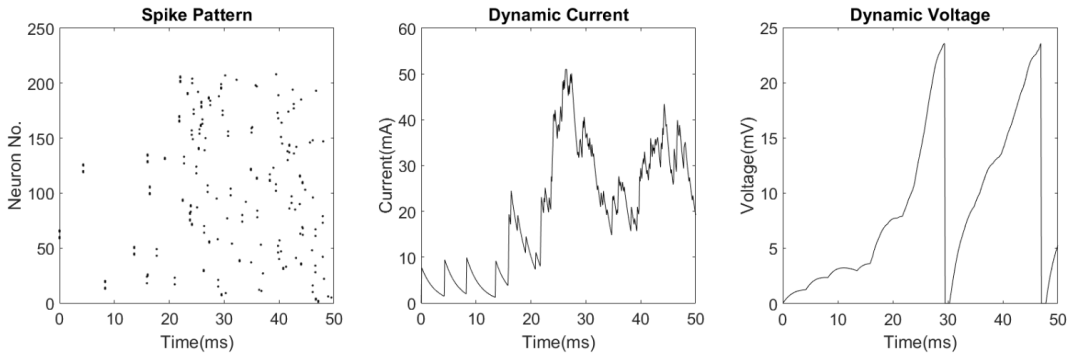


Figure 4.5: Schematic diagram of leaky integrate-and-fire model.

synaptic neuron. When the membrane potential reaches its predefined post-synaptic potential threshold, the post-synaptic neuron will fire a spike, followed by a quite short refractory period (about 1 ms) and then start integrating again.

Note, by using LIF model, only spikes within a short time window can stimulate the postsynaptic neuron to fire a post-synaptic spike. Those presynaptic spikes with much later or earlier have no influence on the procedure of generating a postsynaptic spike.

4.3.2 STDP Learning Rule

Hebb's postulate [40], one of the most important theory in neuroscience, tries to explain the adaptation of neurons in the brain during the learning process. It emphasizes the causality between pre- and postsynaptic neurons, which also known as "Cells that fire together, wire together". Specifically, in Hebb's postulate, cell A needs to take part in firing cell B, and such causality can only occur if cell A fires just before, not at the same time as, cell B.

Like Hebb's postulate, Spike-timing dependent plasticity (STDP) also emphasizes the causality between pre- and postsynaptic neurons [41], [43], [105], [106], [107]. It adjusts the efficacy of synaptic connections based on the relative timing of post-synaptic spike and its input presynaptic spike. In fact, it can be considered as a temporally asymmetric form of Hebb's postulate.

Within neuroscience, long-term potentiation (LTP) is a persistent strengthening of synapses based on recent patterns of activity, while long-term depression (LTD) is an activity-dependent long-lasting reduction in the efficacy of neural synapses. Based on

these concepts, STDP learning rule can be summarized as follows: when a presynaptic spike fires slightly earlier than the post-synaptic spike, the associated synaptic efficacy will be potentiated (LTP); On the other hand, the associated synaptic efficacy will be depressed (LTD) if the presynaptic synaptic spike fires later than the post-synaptic spike. The learning function $\mathrm{W}(\mathrm{t})$ can be expressed as follows (t is the time difference between pre- and postsynaptic spikes):

$$\begin{aligned} \mathrm{W}(\mathrm{t}) &= A_{+} \exp\left(-\frac{t}{\tau_{+}}\right) \quad for \;\; t > 0 \\ \mathrm{W}(\mathrm{t}) &= -A_{-} \exp\left(\frac{t}{\tau_{-}}\right) \quad for \;\; t < 0 \end{aligned} \tag{4.8}$$

where A_{+} and A_{-} represent amplitude of LTP part and LTD part of the *learning window*, respectively. τ_{+} and τ_{-} are time constant for LTP and LTD, respectively.

For biological reasons, it is desirable to keep the synaptic efficacy in a predefined range. Thus, a soft bound strategy [111],[112] has been used to ensure the synaptic efficacy remains in the desired range $w^{min} < w_j < w^{max}$, here, w^{min} and w^{max} represent minimum and maximum value, respectively. The soft bound strategy can be expressed as follows (for the sake of simplicity, the lower bound is set to zero in most models):

$$A_{+}(w_j) = (w^{max} - w_j)\,\eta_{+} \quad and \quad A_{-}(w_j) = w_j \eta_{-} \tag{4.9}$$

where η_{+} and η_{-} are positive constants. Figure 4.6 shows one example of STDP learning window. It has been proven that STDP can reliably find the start of repeating pattern even there are spike jitters or spontaneously activities existed [29]. To achieve stable status for synaptic efficacy, the predefined postsynaptic potential threshold needs to be tuned around its optimum value [1].

Within the proposed SNN, the output layer is the only learning layer. The spiking pattern generated from the first two layers conveys certain selectivity to its input image. Specifically, the spatiotemporal information embedded within the spike pattern plays an important role in defining such selectivity. Spike-timing dependent plasticity learning rule has been applied in the output layer and it will dynamically changes the synaptic efficacy according to the learning window. Eventually, the synaptic efficacy

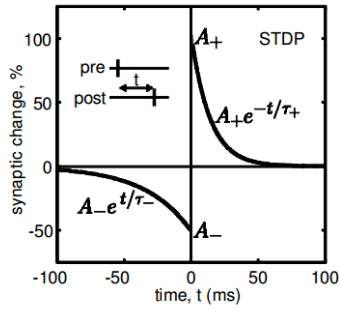


Figure 4.6: One example of STDP learning window. When a presynaptic spike fires slightly earlier than the post-synaptic spike, the associated synaptic efficacy will be potentiated; Otherwise, the associated synaptic efficacy will be depressed.

matrix will be stabilized and thus the selectivity will be emerged after the learning procedure. Unlike [17], in the proposed method, STDP is used after spiking encoding layer for pattern recognition, and $C1$ features are translated to spiking patterns ($S1$ and $C1$ are only for feature extraction).

Note, a new input image should be fed into the feed-forward SNN only until the current input image has been successfully trained or tested. After successfully learning the efficacy weights, the membrane potentials generated during the training procedure will be reset to default values. This learning efficacy weights would be updated each time until the very last training image been feeding into the spiking neural network. Thus, the whole learning procedure can be described as following:

1. Propagate an input image I into the feature extracting layer and obtain its local invariant $C1$ features. Moreover, transform the corresponding 2D local invariant $C1$ features into a 1D vector F, as shown in section 4.2.1.

2. Create a new map for the feature vector F within the spiking encoding layer and obtain its spiking pattern S by using the modified ROC scheme, as shown in section 4.2.2. Note, the number of elements in the local invariant $C1$ feature vector F is identical to the number of neurons within the spiking encoding layer.

3. Create a new map for each class within the output layer and train the incoming spiking pattern S by combing a unsupervised STDP learning rule and a winner-take-all strategy. To achieve the winner-take-all strategy, there are lateral inhibition connections existed within output layer. After the learning the input spiking

pattern S, reset the intermediate variables such as dynamic currents and membrane potentials to their resting values.

4. Stop the learning procedure if all the input training images have been fed into the proposed SNN once. Otherwise, go to Step 1 and continue the learning procedure.

4.4 Experiments

To verify the proposed feed-forward SNN and its unsupervised STDP learning rule, MNIST handwritten digits database has been used as the training and testing database. Specifically, in this section, details about MNIST database and experimental parameter settings will be introduced firstly, followed by different experiments and discussions under various circumstances.

4.4.1 MNIST Database

Within pattern recognition field, MNIST handwritten digits database [113] is often considered as a benchmark database, which contains 60000 training samples and 10000 testing samples (all sample size is 28×28). It includes 10 classes in which each class represents one specific handwritten digit between 0 and 9.

As Figure 4.7 shown, MNIST database has large intra-class variance and thus proposes a quite challenging task for the proposed method. For instance, the digit 1 and 7 in Figure 4.7 have different external shape (the fifth digit in the second row and the sixth digit in the last row have significant different external shape compared with other samples in their class). Sometimes, even human being cannot easily recognize some digits of the database. For example, the fifth digit in the last row could be seen as 4 or 6 and each one can have their own opinion.

4.4.2 Parameter Settings

For the sake of efficiency, only the first two layers of HMAX model have been used in this chapter. For each input image, a local invariant $C1$ feature vector will be

Figure 4.7: Random examples of MNIST database.

Table 4.1: Parameter settings of the proposed SNN.

Parameter	Description	Value
τ	postsynaptic current time constant [2]	2.5 ms
τ_m	membrane time constant [1]	10 ms
τ_+	LTP time constant [1]	0.0168
τ_-	LTD time constant [1]	0.0337
R	the resistance [1]	0.1 $m\Omega$
p	a positive constant for ROC [2]	0.05
T_{rf}	absolute refractory time [1]	1 ms
V_r	resting membrane potential [1]	0 mV
k	a positive constant for V_{thr} [2]	0.8
w^{min}	minimum synaptic weight [1]	0
w^{max}	maximum synaptic weight [1]	1
η_+	a positive constant for A_+ [1]	0.03125
η_-	a positive constant for A_- [1]	0.0265625

[1] Take the same value as [29].
[2] Optimized to achieve the best classification performance.

obtained within the feature extracting layer. In this chapter, we use the same parameter settings as HMAX model [108] to generate this $C1$ feature vector. To simulate the proposed SNN framework and obtain the best classification performance, parameter settings demonstrated in Table 4.1 have been used. As mentioned in section 4.2.2, p controls the time window for a spiking pattern. In this chapter, p is 0.05, which means the time window for a spiking pattern is 50 ms. Note, the time resolution of this experiment is 0.1 ms.

4.4.3 Experiments and Discussions

In many brains areas, temporal aspects of spiking patterns have been found to be highly reproducible [114]. To obtain spiking patterns with no intra-class variance, the generated high level features should remain same for different input images. Otherwise, if the high level features generated from different input images are similar to each other, the spiking patterns will have low intra-class variance; if the high level features vary dramatically from each other for different input images, the spiking patterns will have large intra-class variance.

Moreover, the mean of Pearson's correlation coefficients can be used to describe the level of the intra-class variance. Specifically, given m input images, the feature extracting layer will generate m local invariant $C1$ feature vectors. If we have one local invariant $C1$ vector $\{x_1, \ldots, x_n\}$ containing n elements and another vector $\{y_1, \ldots, y_n\}$ containing n elements then the corresponding Pearson's correlation coefficient r_{xy} can be computed as follows:

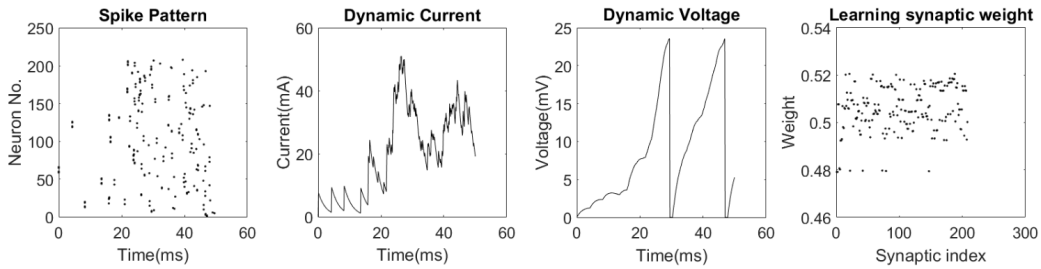
$$r_{xy} = \frac{n\Sigma_{i=1}^{n} x_i y_i - \Sigma_{i=1}^{n} x_i \Sigma_{i=1}^{n} y_i}{\sqrt{n\Sigma_{i=1}^{n} x_i^2 - (\Sigma_{i=1}^{n} x_i)^2}\sqrt{n\Sigma_{i=1}^{n} y_i^2 - (\Sigma_{i=1}^{n} y_i)^2}} \tag{4.10}$$

and the mean of Pearson's correlation coefficients $\bar{r}$ can be computed as follows:

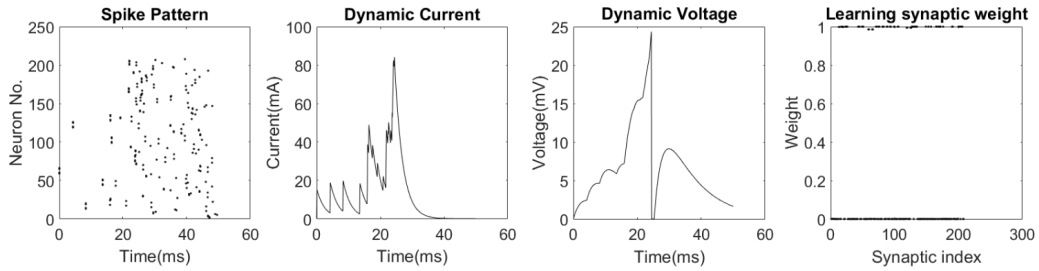
$$\bar{r} = \frac{1}{m^2} \sum_{x=1}^{m} \sum_{y=1}^{m} r_{xy} \tag{4.11}$$

The spiking patterns will have large intra-class variance if $\mid \bar{r} \mid \leq 0.8$. Furthermore, if $0.8 < \mid \bar{r} \mid < 1$, then the spiking patterns will have low intra-class variance. Finally, if $\mid \bar{r} \mid = 1$, then the spiking patterns will have no intra-class variance.

In the theoretical study [114], the authors state that repeated inputs systematically lead to a shaping of the neuron's selectivity, emphasizing its very first input spikes, while steadily decreasing the postsynaptic response latency. However, such statement is only valid for spiking patterns with no or low intra-class variance. Thus, before giving the experimental performance of the proposed method, we will firstly discuss the actual performance of the proposed STDP learning rule under different circumstances (with no intra-class variance and large intra-class variance).



(a) Status of the first iteration of STDP learning



(b) Status of the 200-th iteration of STDP learning

Figure 4.8: Generating selectivity by using unsupervised STDP learning. In this experiment, since repeated inputs are the same images, then $\mid \bar{r} \mid = 1$.

4.4.3.1 STDP Learning with No Intra-class Variance

Figure 4.8 shows dynamic learning procedure of generating selectivity after using unsupervised STDP learning method. Figure 4.8a shows the beginning of the learning procedure. It can be seen that the dynamic synaptic current fluctuates over the whole time window and the synaptic voltage reaches its threshold at about 30 ms and 48 ms. The synaptic efficacy weights are relatively random at this stage. After presenting the same input image (same input image in Figure 4.4) to the SNN system about 200 times, the selectivity finally emerged, just as the Figure 4.8b shows. At this stage, the synaptic current only fluctuates over the first half time window and the synaptic voltage fires the spike at about 24 ms. What's more, the synaptic efficacy matrix has a special status with most weights take 0 and the rest take 1 [114], [115]. Therefore, the selectivity to this specific input image emerges. However, such learning results can be generated only if the intra-class variance of the input images remains at a reasonable level ($\mid \bar{r} \mid > 0.8$).

4.4.3.2 STDP Learning with Relatively Large Intra-class Variance

As Figure 4.8 shown, under the ideal situation that no intra-class variance existed in features, the proposed method can obtain an ideal STDP learning efficacy matrix. In real scenarios, such ideal condition is impossible to achieve. In this experiment, even the input images having high level intra-class variance, as shown in Figure 4.9a, 4.9b, 4.9e, 4.9f, 4.9i, 4.9j, the proposed method can still learn certain selectivity.

In Figure 4.9, according to the level of intra-class variance, 6 input images have been divided into three groups. Note, 4.9a, 4.9e and 4.9i are the same input images. Group 1 ($\mid \bar{r} \mid = 0.9136$) includes 4.9a and 4.9b, 4.9c and 4.9d are the learning synaptic weight of 20-th and 200-th iterations, respectively; Group 2 ($\mid \bar{r} \mid = 0.8461$) consists of 4.9e and 4.9f, 4.9g and 4.9h are dynamic efficacy matrix of 20-th and 200-th iterations, respectively; Group 3 ($\mid \bar{r} \mid = 0.7238$) contains 4.9i and 4.9j, 4.9k and 4.9l are dynamic efficacy matrix of 20-th and 200-th iterations, respectively; Here, one iteration means sequentially feeding the two input images into the proposed SNN once. The learning synaptic weight will be harder to concentrate if increasing the intra-class variance level $\mid \bar{r} \mid$. In other words, from Figure 4.9, one can easily concluded that training samples with more intra-class variance will somehow hard to learn the selectivity.

Given large intra-class variance, Figure 4.10 shows the dynamic learning efficacy matrix with different number of learning iterations. Note, in this experiment, one iteration means sequentially feeding 50 different training samples within a certain class into the proposed SNN framework. It can be seen that the dynamic status only have a very limited changes. However, even the intra-class variance in the experiment remains at a relatively high level, the training samples are not totally independent (e.g. totally random samples), and thus such seemingly random learning efficacy matrix may contains certain selectivity to the input.

To ensure each training sample will be properly learned, a dynamic membrane potential threshold strategy (described in equation 4.7) has been applied in this chapter. Table 4.2 shows the correct classification comparison with the proposed dynamic membrane potential threshold and the predefined voltage threshold. Note, the predefined voltage thresholds have been set within a certain range (10-30 mV) around its

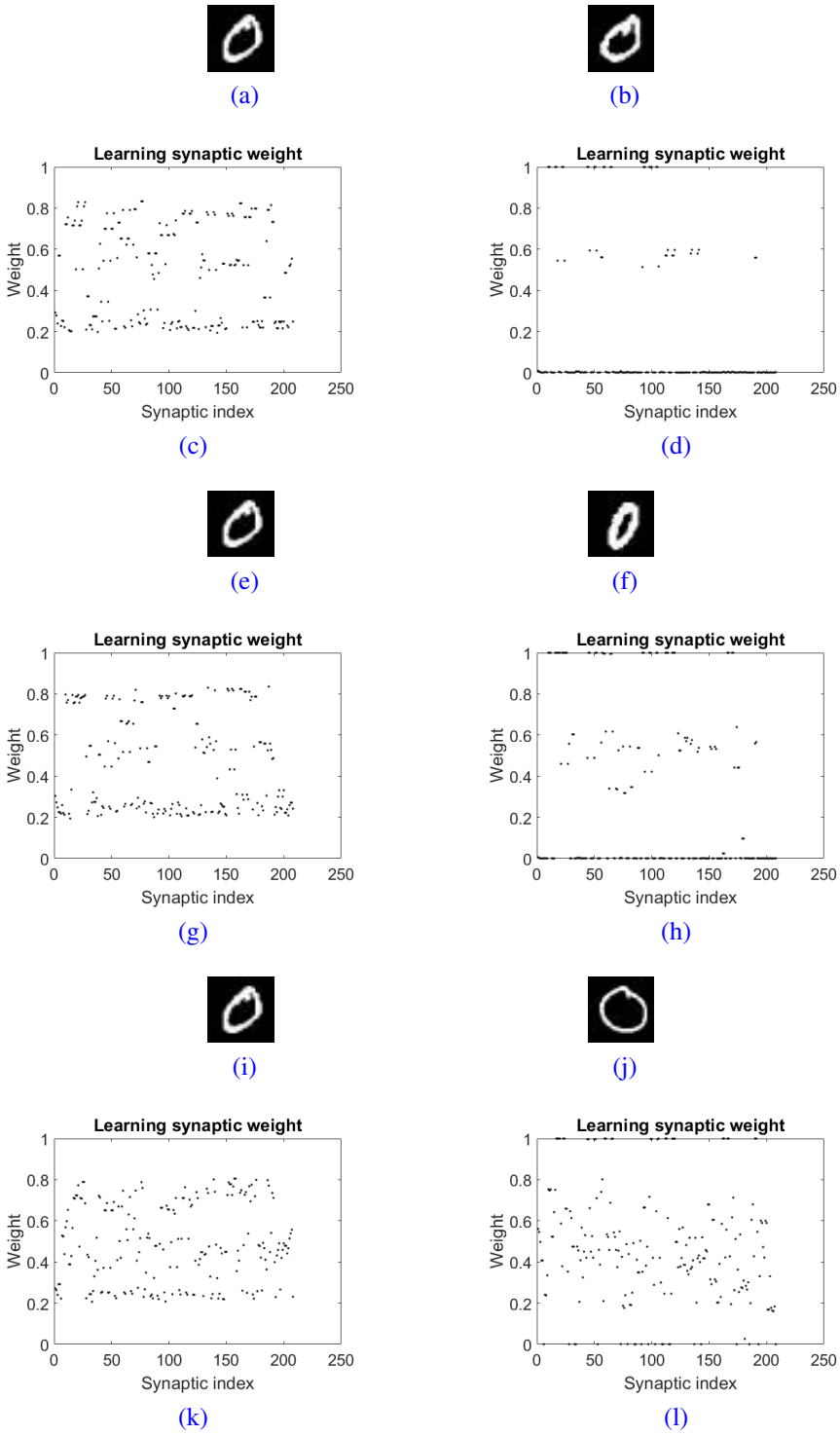
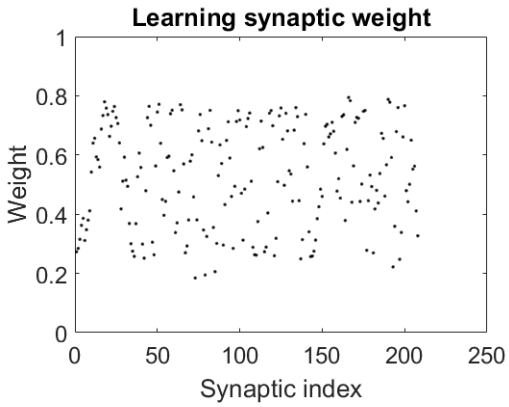
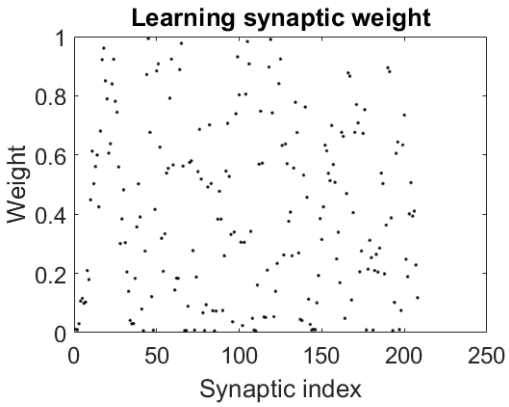


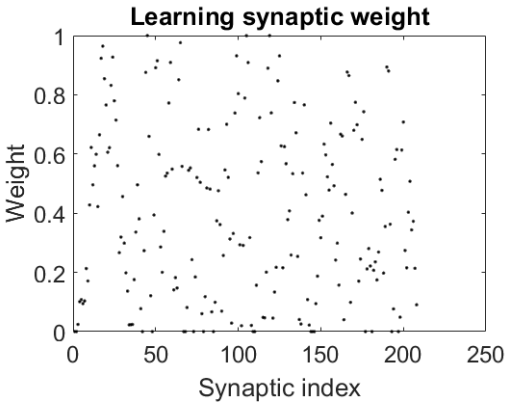
Figure 4.9: Based on the level of intra-class variance, 6 input images have been divided into three groups. Note, (a), (e) and (i) are the same input images. Group 1 ($\mid \bar{r} \mid = 0.9136$) includes (a) and (b), (c) and (d) are the learning synaptic weight of 20-th and 200-th iterations, respectively; Group 2 ($\mid \bar{r} \mid = 0.8461$) consists of (e) and (f), (g) and (h) are dynamic efficacy matrix of 20-th and 200-th iterations, respectively; Group 3 ($\mid \bar{r} \mid = 0.7238$) contains (i) and (j), (k) and (l) are dynamic efficacy matrix of 20-th and 200-th iterations, respectively; Here, one iteration means sequentially feeding the two input images into the proposed SNN once. The learning synaptic weight will be harder to concentrate if increasing the intra-class variance level $\mid \bar{r} \mid$.



(a) The first iteration



(b) The 5-th iteration



(c) The 10-th iteration

Figure 4.10: Learning synaptic weights with large intra-class variance. Here, one iteration means sequentially feeding 50 different training samples within a certain class into the proposed SNN framework once.

Table 4.2: Impact of dynamic membrane potential threshold on recognition rate.

With dynamic threshold	With predifined threshold (mV)		
	10	20	30
0.81	0.73	0.78	0.71

- Note: 0.81 in this table means 81% correct classification rate. Here, classification performances have been obtained by computing the average value of 30 random tests. Furthermore, for method with dynamic threshold and method with 20 mV threshold, we conduct a Wilcoxon signed-rank test on their results and the significance level $p - value$=0.005889. Since it less than 0.05, those two results are statistically different.

optimum value (20 mV). It can be seen that the dynamic membrane potential threshold strategy can not only ensure learning each training sample properly but also generate

Table 4.3: Simple random sampling experiments with different iterations in 10 random tests.

Tests	Number of iterations				
	1	2	3	4	5
1	0.81	0.85	0.81	0.76	0.71
2	0.83	0.78	0.75	0.72	0.74
3	0.81	0.87	0.88	0.87	0.87
4	0.8	0.81	0.81	0.78	0.76
5	0.84	0.8	0.8	0.78	0.77
6	0.8	0.78	0.74	0.73	0.72
7	0.81	0.8	0.78	0.77	0.75
8	0.84	0.81	0.78	0.8	0.79
9	0.81	0.79	0.74	0.74	0.73
10	0.84	0.79	0.78	0.79	0.79
Average	0.819	0.808	0.787	0.774	0.763

- Note: 0.81 in this table means 81% correct classification rate. Here, one iteration means sequentially feeding 50 different training samples within a certain class into the proposed SNN framework once.

the best correct classification performance.

4.4.3.3 Experiments on MNIST Database

There are total 60,000 training samples in MNIST database, as mentioned above, given a real-time learning circumstance, it is hard to fully exploit the whole database with limited time. We will use a simple random sampling method to test the proposed algorithm and to answer the above question. Simple random sampling method, which randomly selects limited samples for training and testing, creates a scenario most similar to a real-time learning situation.

From the MNIST, we randomly choose 50 different training samples for each class and 100 different testing samples to test the correct classification rate. In the following experiments, each test follows the same procedure mentioned above. For fair comparison, each iteration within each test uses the same randomly chosen training samples

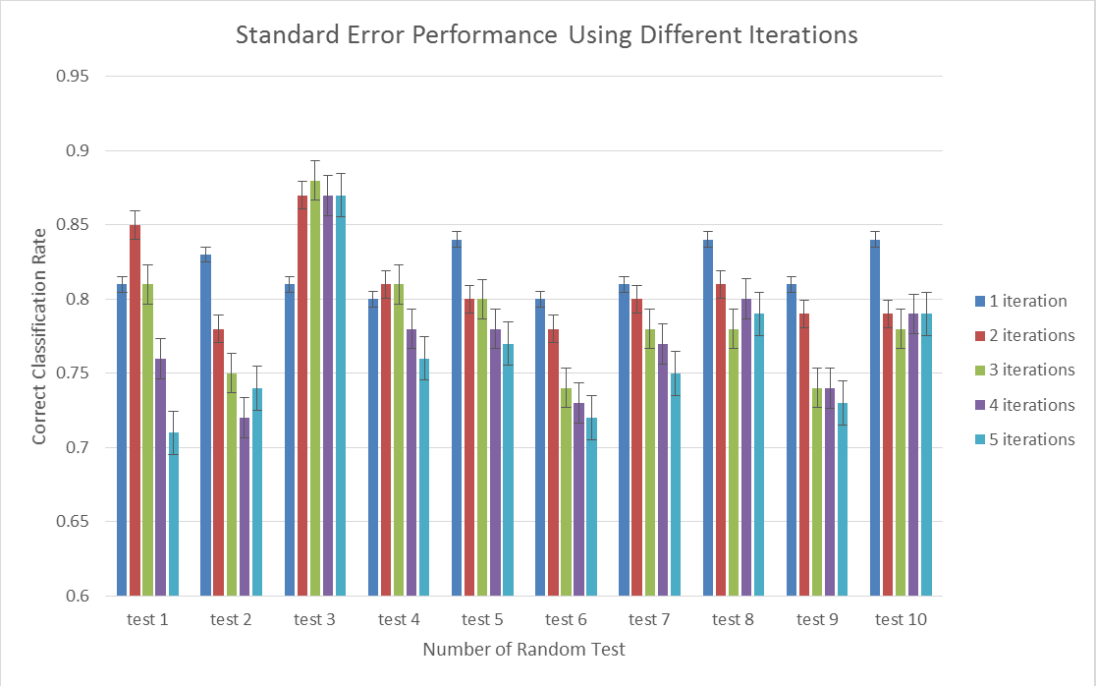


Figure 4.11: Standard error performance using different iterations. Here, one iteration means sequentially feeding 50 different training samples within a certain class into the proposed SNN framework once.

and testing samples.

Table 4.3 shows the corresponding correct classification rate performance when using the experimental conditions mentioned above. Average correct classification rate also has been added in the table. It can be seen that, with one iteration only, almost all the tests achieved the highest performance. This suggests that the proposed learning method is suitable for real-time learning.

Figure 4.11 shows standard error performance using different iterations. It can be seen that, along with increasing of iterations, the correct classification rate gradually decreases. Tests with one iteration only seems to convey the least standard error. Such characteristic indicates the learning methods with one iteration only are more reliable than that with more iterations.

Why more iterations have not led to better performance in this case? This is because, for precisely timed spikes, the synaptic weight saturates close to its maximum value if the presynaptic spikes arrive before the postsynaptic neuron is firing. If the temporal jitter of the pre- and postsynaptic spikes escalated, the weight will take an intermediate value determined by non-Hebbian terms rather than by learning window [116]. Since the generated C1 features within the proposed SNN still contain relatively

Table 4.4: Performance comparison of three methods(%).

Method	Correct rate	Wrong rate	Unknown rate
The proposed method	82 ± 2	18 ± 2	0
Tempotron rule [68]	78.5 ± 1.85	18.35 ± 1.85	3.15 ± 1.64
SVM [68]	79.33 ± 2.03	18.15 ± 1.69	2.53 ± 2.04

- Note: Here, SVM [68] with Gaussian kernel has been used on the local invariant $C1$ feature vectors.

Table 4.5: Running speed tests.

Item	Running time(s)	Equivalent to frames per second
Training	23.32	21.3
Testing	5.63	17.9

- Note: the above results are the mean value of 10 random tests gathered from a laptop with Intel 3rd Gen Core at 2.5 GHz, 8G RAM and 128G SSD. The whole training procedure includes 500 frames/samples represent total 10 classes (50 samples for each class) and the whole testing procedure includes 100 samples.

large intra-class variances (means relatively large temporal jitter of pre- and postsynaptic spikes), increasing the iteration times implies the level of the temporal jitter of pre- and postsynaptic spikes is increased, which may lead to the poor performance with more iterations.

In paper [68], the authors used a supervised temporal learning rule (named Tempotron Rule) to train the MNIST database (almost same experimental conditions as this paper) and achieved 79% correct classification rate in the end. Unlike this state-of-art learning method, the proposed algorithm uses unsupervised STDP learning rule with dynamic post-synaptic potential threshold during the learning procedure.

Table 4.4 and Figure 4.12 show the final classification performance comparison of three different methods. It can be seen that the unknown rate of the proposed method is 0, which means each testing sample would be recognized as one possible class. Compared with Tempotron Rule, the proposed method achieves better correct rate at around 82%, while still remains slightly less wrong rate. Finally, Table 4.5 shows the speed test results for the training and testing periods respectively. It can be seen that

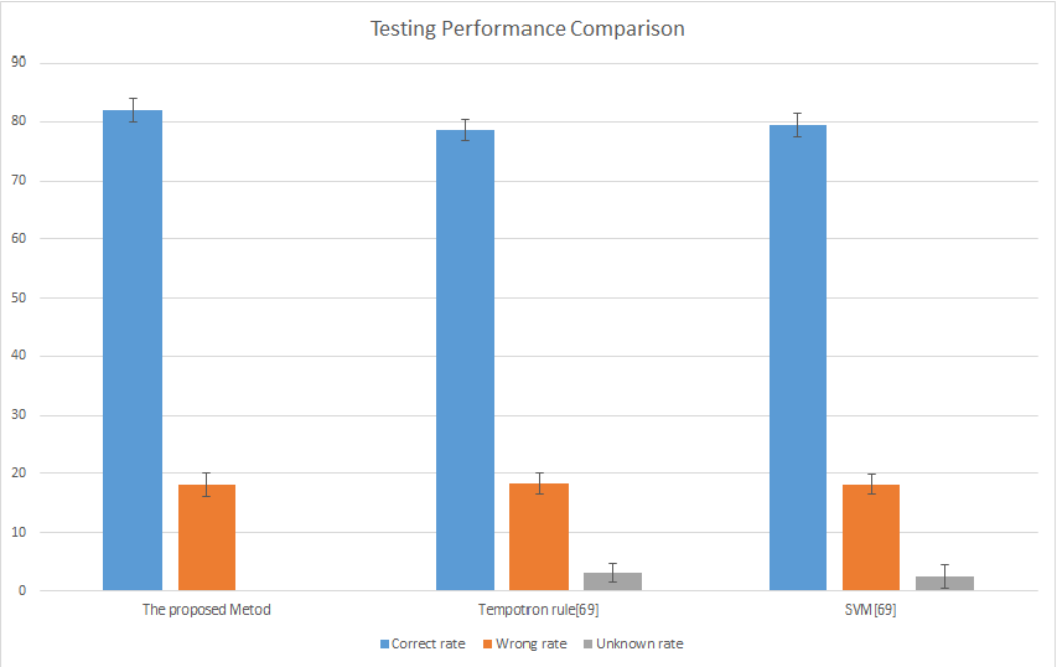


Figure 4.12: Performance comparison of three methods.

the learning and testing speeds are quite impressive - 21.3 fps in training and 17.9 fps in testing, both can be operating in real time.

4.5 Conclusion

Real-time learning needs algorithms operating in a fast speed comparable to human or animal, however this is a huge challenge in processing visual inputs at milliseconds scale. In the above chapters, we proposed a novel real-time learning method by combing the spike timing-based feed-forward spiking neural network (SNN) and the fast unsupervised spike timing dependent plasticity learning method with dynamic post-synaptic thresholds. Fast simple random sampling experiments using MNIST database showed the high efficiency of the proposed method at an acceptable accuracy. Our research may also add to the further understanding of the dynamic processing procedure existed in brain's ventral stream.

Chapter 5

Event-driven Continuous STDP Learning using HMAX Model for Visual Pattern Recognition

Within visual cortex, ventral stream, one type of visual processing pathway, plays an important role in form recognition and object representation. Ventral stream is a hierarchical system in which each layer extracts different level of abstractions [6], but the underlying processing mechanism of the ventral stream is still largely unknown, which proposes a huge challenge for the researchers. Two main categories of methods, spiking rate-based methods [25], [26], [27], [70], [71], [72], [73], [74], [75] and spiking time-based methods [17], [68], [69], have been proposed to address the above issue. However, they suffer various drawbacks: a limited learning time, which often exists in ventral stream, is not enough to generate meaningful spiking rate; A supervisory error signal used to update the synaptic weights, in real scenarios, is hard to obtain.

Furthermore, ventral stream is capable of adaptively learning the spatiotemporal structures from the input spiking patterns. Traditionally, to learn the spatiotemporal structures, a new input spiking pattern is only allowed to feed into the learning system when the membrane potential generated by the previous spiking pattern has been reseted. However, in ventral stream, the neurons receive the spiking patterns continuously without any resetting involved. Moreover, traditional learning methods like

spike-timing dependent plasticity (STDP) often incorporate global information within the training procedure. For instance, to update the synaptic weights within STDP learning rule, the neuron is required to remember all the related spiking timings within the learning window. In most cases, it is physiologically unrealistic and not efficient.

To solve the above problems, in this chapter, an event-driven continuous STDP (ECS) learning method using specific spiking timing sequences has been proposed. Specifically, two novel continuous input mechanisms, a sequential one with interval between adjacent spiking patterns and another parallel one with two separated spiking pattern groups, have been proposed to obtain the continuous input spiking pattern sequence. Furthermore, within the proposed event-driven STDP learning rule, the learning procedure will be activated when the neuron receive a presynaptic or postsynaptic spike event. The simple random sampling and exhaustive experimental results on MNIST database show the proposed method, compared with other state-of-the-art methods, can achieve comparable correct classification performances, but with more biologically plausible supports.

5.1 Background

Within visual cortex, there are two information processing pathways originating in the occipital cortex: ventral stream and dorsal stream, which also known as what (object recognition) and where (spatial vision) pathways. Specifically, the ventral stream plays an important role in form recognition and object representation. Due to the limited understanding of the processing mechanism of the ventral stream, it remains a challenging task for researchers to use recent available biologically plausible mechanisms to simulate the ventral stream.

Specifically, to simulate the ventral stream, several critical issues need to be resolved: 1) what is the architecture of the hierarchical system? 2) how to represent and transmit the information within the system? 3)how to constantly process the input visual information? 4) which learning method is appropriate for learning the selectivities from the input visual stimuli? To accomplish the fundamental functions of the ventral stream, those questions will be discussed and addressed in the following paragraphs.

It is known that ventral stream is a hierarchical system in which each layer extracts different level of abstractions [6]. By transforming the input visual stimuli into generalized abstractions, such hierarchical structure will greatly reduce the data dimensionality. Deep learning, a branch of machine learning, has often been used to accomplish the above scenarios, which is based on a set of algorithms that attempt to model high-level abstractions in data by using complex hierarchical model architectures with multiple non-linear transformations [9]. HMAX [22], [108], [117], a type of convolutional neural networks (CNN) within deep learning methods, has been proposed to provide the much-needed framework for summarizing and integrating input visual stimuli, and thus obtaining the high-level abstractions existed in ventral stream.

In real scenarios, visual cortex uses neural spikes to represent/transmit information. The dynamic interaction procedure within visual cortex can be simulated with spiking neural network (SNN). Within SNN, the spiking patterns with spatiotemporal structural information have often been used as transmission mediums. Neuron models define how the activities of the neurons change in response to each other, which is essential part for building a SNN. In this chapter, conductance-based leaky integrate-and-fire (LIF) neuron model [1] has been used to regulate the behaviors of the neurons within SNN. It emphasizes the causality of related spikes and is more biologically plausible.

According to information theory, the spike coding schemes change the format of information processing, e.g. from analog feature values into spiking patterns. Both spike rate and spike timing can be used to represent the input analog feature values. However, it is difficult for rate-based SNN to generate meaningful rate within a short time widow. For instance, research showed that mammalian brain use only millisecond scale time window to process complicated real life visual recognition tasks [2]. Moreover, if the input visual stimuli has been incorporated with the background noise with the same spiking rate, it is impossible for rate-based SNN to generate the selectivity for the input visual stimuli [29]. In this chapter, rank order coding (ROC) [24], [98], [99], a simple yet powerful spike timing-based coding scheme, has been used to generate the first spike wave. ROC scheme considers the first spike wave conveys enough

significant structural information for further visual pattern recognition application.

In real scenarios, neurons within visual cortex tend to receive input stimulus continuously. Continuous stimulus presentation is a significant feature in generating a versatile and general network [118]. There are two main reasons for choosing continuous input sequence strategy: 1) the visual stimuli can be fed into the SNN without any reseting during each training/testing step; 2) it is more biologically plausible. In this chapter, two different continuous input sequence mechanisms have been proposed in which the first one adds intervals between spiking patterns and updates the synaptic efficiency sequentially, while the second one divides the whole sequence into two subsequences and updates the synaptic efficiency by alternating between those two subsequences. Furthermore, the background neural noise [119], [120] and time jitter (also known as distractor) can be easily added into the input sequence to test the robustness of the proposed method.

To learn the spiking pattern sequence, spike timing dependent plasticity (STDP) [41], [42], [43], [44], [45], a temporally asymmetric form of Hebbian learning [40], has been applied within the SNN. It is widely believed that it underlies learning and information storage in the brain, as well as the development and refinement of neuronal circuits during brain development [90], [91]. Within the traditional STDP learning, to update synaptic weights, neurons need to integrate all the related spikes within the learning window. However, in real scenarios, neurons can only access local information. Moreover, such update procedure is very inefficient. In this chapter, event-driven STDP learning method has been used to overcome those drawbacks by involving two on-line, local learning rules that are applied only in response to occurrences of spike events. It is believed that such event-driven strategy can provide more efficient and biologically plausible learning procedure for the input visual stimuli.

Lots of researches [25], [26], [27], [68], [69], [70], [71], [72], [73], [74], [75] have been investigated the visual pattern recognition tasks using different SNN frameworks and their corresponding learning methods. From the spiking encoding point of view, those methods can be divided into two main categories: spiking time-based methods and spiking rate-based methods. For spiking rate-based methods [25], [26], [27], [70],

[71], [72], [73], [74], [75], there are two main drawbacks: a) To generate meaningful spiking rate, those rate-based methods need to run the database several time (runs), which is obvious inefficient and time-consuming. b) Most of those methods are not biologically plausible as they lack direct biological supports for their simulation of the ventral stream. Thus, in this chapter, we mainly focus on the spiking time-based methods.

In [68], [69], the authors propose a novel SNN framework and use tempotron rule to train the input visual stimuli. However, the supervisory error signal used in tempotron rule, in real scenarios, is hard to obtain. Unlike the above methods, in [17], Thorpe et. al use their SNN architecture to simulate the processing procedure of HMAX model. However, the simplified STDP learning rule used in their method is only for local intermediate feature extracting. It is known that feature extracting is only a significant component of pattern recognition. Thus, the STDP learning rule used in [17] itself is not enough to accomplish the visual pattern recognition tasks.

To address the problems mentioned above, in this chapter, based on the available biologically plausible models, an event-driven continuous spike-timing dependent plasticity (STDP) learning method (ECS) using HMAX model has been proposed. Specifically, based on the modified HMAX model and the proposed spike encoding scheme, the input visual stimuli have been transformed into spiking patterns with spatiotemporal structural information. Within the spiking pattern learning procedure, two different continuous input sequence mechanisms have been applied into the event-driven STDP learning method in which each mechanism has its own specific update procedure. Event-driven strategy and continuous input sequence mechanism are both biologically plausible, while the former one greatly improves the learning efficiency and the latter one ensures the input visual stimuli can be learned without resetting the intermediate variables during each learning step. The simple random sampling and exhaustive experimental results on MNIST database show that the proposed method, compared with other state-of-the-art methods, can achieve acceptable correct classification performances.

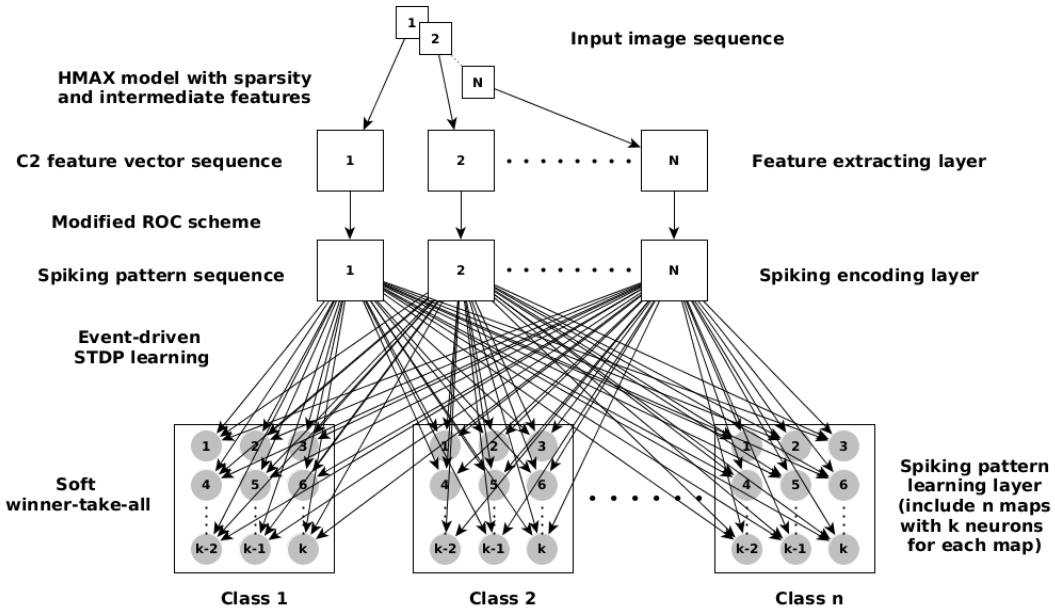


Figure 5.1: The framework of the proposed timing-based feed-forward spiking neural network. For simplicity, the lateral inhibition connections of the last layer have not been included.

5.2 The Proposed SNN and Its ECS Learning Method

To simulate the hierarchical ventral stream, in this chapter, a novel feed-forward SNN framework has been proposed, which includes three layers: feature extracting layer, spiking encoding layer and spiking pattern learning layer. By simulating different visual areas within the ventral stream, each layer within the proposed SNN accomplishes one specific training goal.

Specifically, through the feature extracting layer, the invariant high level features have been extracted by incorporating the modified HMAX model and then transmitted to the spiking encoding layer to obtain the corresponding spatiotemporal spiking pattern sequences. Within the spiking pattern learning layer, by using the proposed continuous event-driven STDP learning method with certain update procedure, the selectivity would be emerged eventually. Figure 5.1 shows the framework of the proposed feed-forward SNN with several keywords explaining the functionality of each layer. The details of each layer will be explained in the following subsections.

5.2.1 Feature Extracting Layer

From the computational model point of view, the visual system is an information processor performing computations on internal symbolic representations of visual information [121]. The computational model plays an important role in obtaining the invariant high level features from the input visual stimuli. According to deep learning theory, the high level features should strike a balance state between invariance and distinguishability [9], which has a large impact on the final classification performance.

Riesenhuber and Poggio [22] proposed a feed-forward processing computational model (named HMAX model) based on the knowledge of visual cortex and achieved promising results on some of the standard classification database. Such model focuses on the object recognition capabilities of the ventral stream in an ”immediate recognition” mode, independent of attention or other top-down effects [22]. It is considered as a starting point for researchers to simulate ventral stream. Inspired by the simple and complex cells within V1 (discovered by Hubel and Wiesel [122]), Serre, Wolf and Poggio [108], [117] extends the original HMAX model and thus built an increasingly complex and invariant feature representation by alternating between a template matching and a maximum pooling operation (demonstrated in section 4.2.1). Increasing the sparsity of basis functions is equals to reduce the capacity of the classifier [123], [124]. Localized intermediate approaches retain some coarsely-coded location information [125] or record the locations of features relative to the object center [126]. After incorporating some additional biologically-motivated properties of the visual cortex, Mutch and Lowe [127] proposed a novel model by adding sparsity and localized intermediate-level features into the model proposed by Serre et al. Such model achieves a significant improvement in final classification performance.

Inspired by the computational model proposed by Mutch and Lowe [127], this chapter tries to build the feature extracting layer based on the base computational model proposed by Mutch and Lowe, as depicted in Figure 5.2. The aim is to build a feature dictionary or feature vector for each input image. The framework used in the feature extracting layer contains five hierarchical layers, besides the input image layer, each built from the previous layer by alternating template matching and max pooling

operations. Note, cortical network simulator (CNS) [128], a GPU-based framework, has been used to simulate the feature extracting layer. Below, we will briefly introduce each layer of the framework used in the feature extracting layer.

5.2.1.1 Input image layer

All input images have been converted to grayscale and scale the shorter edge to 140 pixels while retaining the aspect ratio. An image pyramid with 10 scales has been built, each a factor of $2^{1/4}$ smaller than the last.

5.2.1.2 Gabor filter (S1) layer

Basically, at each possible position and scale, S1 layer applies the Gabor filters with four different orientations $(0^o, 45^o, 90^o, 135^o)$. The Gabor response $F^{\sigma,\theta}_{(x,y)}$ at location (x, y) with the scale σ and the orientation θ can be used to mimic the simple cell within the primary visual cortex V1, which can be computed according to equation 3.8. The Gabor filters are 11×11 in size. Note the components of the each filter are normalized so that their mean is 0 and the sum of their squares is 1. The same size filters have been used for all scales.

5.2.1.3 Local invariance (C1) layer

This C1 layer pool over retinotopically organized afferent S1 units from the previous S1 layer with the same orientation and from the same scale band. Basically, the response $r^{\sigma,\theta}_{(x,y)}$ of a complex C1 unit can be obtained by equation 4.1. Through the maximum pooling operation, the generated C1 units will obtain certain local invariance.

Unlike the traditional HMAX model [108], a lateral inhibition mechanism has been used between S1/C1 units encoding different orientations at the same position and scale. Basically, such mechanism ensures these units are competing to describe the *dominant* orientation (maximally responding C1 unit) at their location. By doing this, those non-dominant orientations will be ignored.

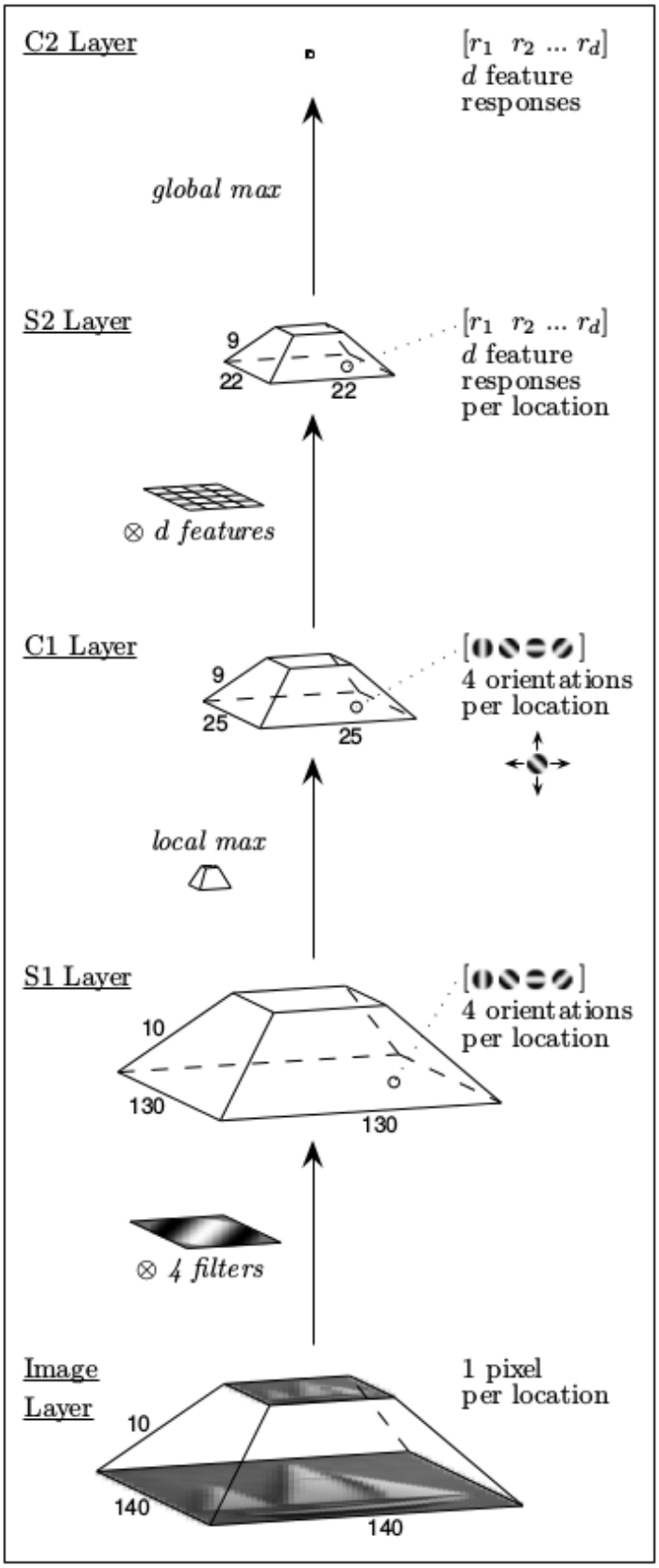


Figure 5.2: The computational base model proposed by Mutch and Lowe [127]. This base model consists of 5 layers, besides the input image layer, each built from the previous layer by alternating template matching and max pooling operations (demonstrated in section 4.2.1). $\otimes$ means the template matching operation. For each input image, a $C2$ feature vector with d elements will be generated.

5.2.1.4 Intermediate feature (S2) layer

Within the computational model proposed by Serre. et al [108], for every position and scale, the template matching operations (demonstrated in section 4.2.1) have been conducted between the patch of C1 units centered at that position/scale and each of d prototype patches. Here, a patch means a set of processing units and the prototype patches can be considered as templates within the template matching operations, as described in [108]. Those prototype patches are randomly sampled from the C1 layers of the training images in an initial feature-learning stage, which represent the intermediate-level features of the base model. During the feature learning stage, sampling is performed by centering a patch of vary sizes at a random position and scale in the C1 layer of a random training image. Therefore, a prototype consists of all the C1 units within the patch. Note, for each position, there are units representing each of the four orientations. A Gaussian radial basis function has been used to compute the response of a patch of C1 units X to a particular S2 prototype P with size of $n \times n$:

$$R(X, P) = exp\left(-\frac{\|X - P\|^2}{2\sigma^2\alpha}\right) \tag{5.1}$$

with X and P have dimensionality $n \times n \times 4$, where $n \in \{4, 8, 12, 16\}$. The standard deviation σ is set to 1 and the parameter α represents a normalizing factor for different patch size.

However, real neurons are likely to be more selective among potential inputs. Therefore, by storing the identity and magnitude of the *dominant* orientation at each of the $n \times n$ positions in the patch, the number of inputs to an S2 feature has been reduced to one per C1 position. By doing this, the dense prototype in the old model has been reduced to sparse prototype, which makes the S2 units less sensitive to local clutter and thus improves the generalization.

5.2.1.5 Global invariance (C2) layer

Within the traditional HMAX model [108], by pooling the maximum response from one of d prototype patches, one element of the d-dimensional vector (C2 features

vector) will be obtained. All position and scale information will be removed if using this mechanism. However, neurons in V4 and IT do not exhibit full invariance and are known to have receptive fileds limited to only a portion of the visual field and range of scales. Therefore, in this chapter, certain limits has been incorporated into the global position/scale invariance mechanism used in [108].

5.2.2 Spiking Encoding Layer

Given the feature extracting layer, one can only obtain the analog high level features. However, within SNN, neurons uses spiking patterns to represent and transmit the spatiotemporal structural information. To generate the spiking patterns from the analog features, spiking encoding layer has been incorporated into the proposed SNN framework. Before elaborating the spike encoding scheme and the proposed continuous input sequence mechanism, the neuron model used in this chapter should be introduced firstly.

5.2.2.1 Neural Model

In this chapter, leaky integrate-and-fire (LIF) model has been chosen as the neuron model as it is biologically plausible and has low computational complexity. The leaking factor within the LIF model ensures that the neurons only fire spikes when there are enough presynaptic spikes fired from its receptive field within a relatively short time window. Thus, such neuron model can be considered as a coincidence detector.

Specifically, instead of using voltage based LIF model, this chapter uses conductance based LIF model to regulate the behaviors of neurons since it increase the level of realism in a neural simulation. Like in [129], [130], the postsynaptic membrane potential of the neuron (V) within the proposed spiking neural network is determined by

$$dV/dt = (g_{ex}(E_{ex} - V) + g_{in}(E_{in} - V) + V_r - V)/\tau_m \tag{5.2}$$

where τ_m is the postsynaptic neuron membrane time constant. E_{ex} and E_{in} represent the membrane potential of excitatory synapse and inhibitory synapse, respectively. V_r

depicts the resting membrane potential. When the postsynaptic membrane potential reaches the threshold V_t,

$$V \geq V_t \tag{5.3}$$

the neuron fires a spike, and then enters the absolute refractory period, in which the membrane potential is resets to V_r and the absolute refractory time is T_{rf}. The excitatory/inhibitory synaptic conductance g_{ex}/g_{in} and its related peak conductance are measured in units of the leakage conductance of the neuron and are thus dimensionless. The excitatory/inhibitory synaptic conductance decays exponentially:

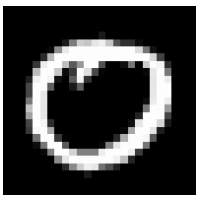
$$\begin{aligned} dg_{ex}/dt &= -g_{ex}/\tau_{ex} \\ dg_{in}/dt &= -g_{in}/\tau_{in} \end{aligned} \tag{5.4}$$

where τ_{ex}/τ_{in} represents the excitatory/inhibitory synaptic conductance time constant.

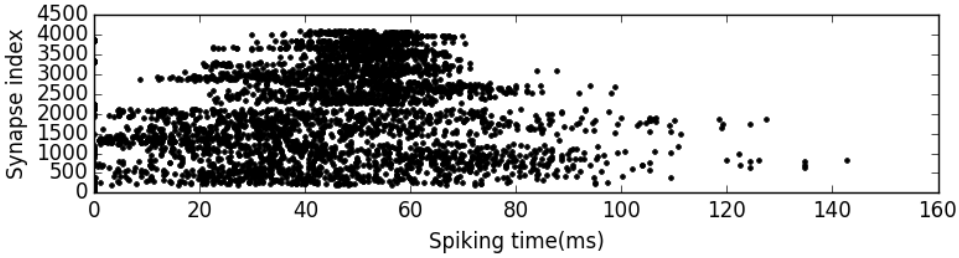
5.2.2.2 Spike encoding scheme

Similar to section 4.2.2, rank order coding (ROC) has been used to generate spiking patterns from the global invariant $C2$ features. For a global invariant $C2$ feature response r, the corresponding spiking timing t can be computed according to equation 4.2. Specifically, p within equation 4.2 controls the length of the processing window of a specific spiking pattern. In this chapter, the processing time window of a specific spiking pattern is set to 0.2 s (p takes 0.2 in equation 4.2).

Figure 5.3 shows one input image and its spiking pattern generated from the first two layers. Note, the C2 feature vector has 4096 (d feature responses in Figure 5.2) elements, and thus there are 4096 neurons to fire spikes. Only those neurons with spiking timing less than 150 millisecond will fire spikes. By feeding 10 input images into the proposed feed-forward SNN, Figure 5.4 shows spiking pattern sequence without interference and with interference, respectively. Specifically, the interference includes time jitter to the input spiking pattern itself and background neural noise between adjacent spiking patterns, which will be discussed in section 5.3.2.2.

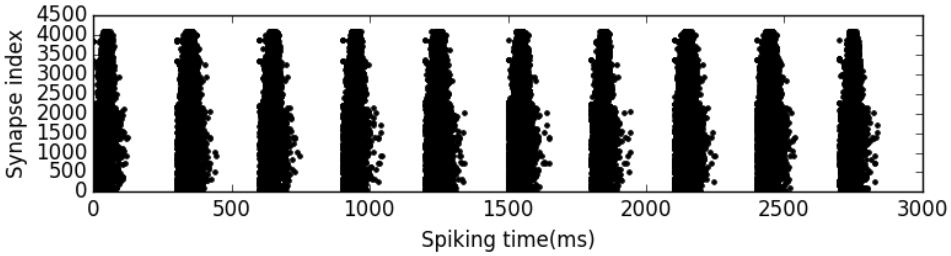


(a) Input image

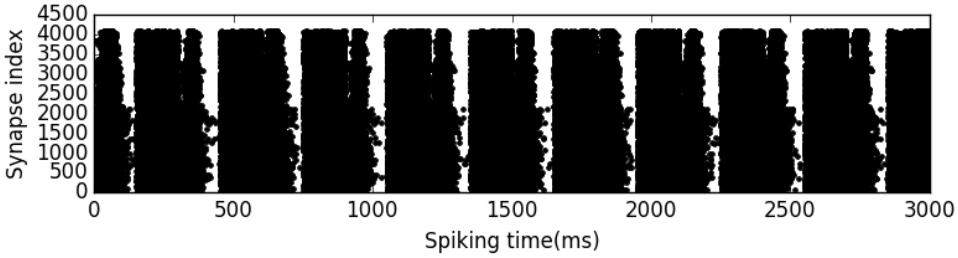


(b) Spiking pattern

Figure 5.3: One input image and its spiking pattern generated from the first two layers. Here, the spikes are generated by global invariant $C2$ features.



(a) Spiking pattern sequence with no interference



(b) Spiking pattern sequence with time jitter and neural noise

Figure 5.4: Spiking pattern sequence (10 input images) without interference and with interference. The interference includes time jitter to the input spiking pattern itself and background neural noise between adjacent spiking patterns.

5.2.2.3 Continuous Input Sequence Mechanism

Traditionally, a new spiking pattern should be fed into SNN only until the current spiking pattern has been successfully trained or tested. Specifically, before starting the next training step, the membrane potentials generated by the current spiking pattern will be reseted to their resting values. Without such procedure, the previous spiking pattern will have a interference with the spiking generating procedure of the current

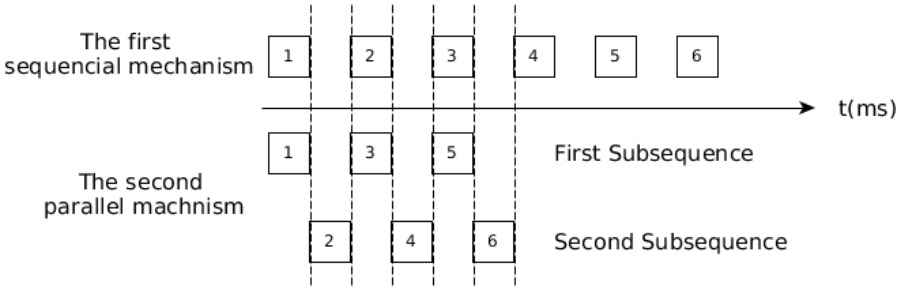


Figure 5.5: Two different continuous input sequence mechanisms. Here, the squares $(1, 2, 3, 4, 5, 6)$ represent spiking patterns. In this chapter, each spiking pattern possesses the same time window and the interval inserted into the sequence or subsequence has an identical time window as each spiking pattern. The first sequential mechanism obtains a single spiking pattern sequence with intervals while the second parallel mechanism generates two subsequences with intervals.

spiking pattern. However, this input mechanism is very inefficient. Moreover, it is not biologically plausible since, in real scenarios, the neurons within the ventral stream tend to receive continuous input visual stimuli.

To address the above drawbacks, two different continuous input sequence mechanisms have been proposed. The first mechanism applies a sequential strategy while the second one follows a parallel fashion. Specifically, the first sequential mechanism obtains a single spiking pattern sequence with intervals while the second parallel mechanism generates two subsequences with intervals, as shown in Figure 5.5. In this chapter, each spiking pattern possesses the same time window (T_s) and the interval inserted into the sequence or subsequence has an identical time window ($T_i = T_s$) as each spiking pattern, as demonstrated in Figure 5.5. Therefore, within the second parallel mechanism, if combined those two sequence into a single sequence, we can obtain a spiking pattern sequence without intervals.

5.2.3 Spiking Pattern Learning Layer

Within the spiking pattern learning layer, there are n maps corresponding to n classes within MNIST database. Each map corresponds to one possible class. The neurons within each map are fully connected to the previous layer. Within each map, there are k neurons corresponding to k possible sub-classes (intra-class variance). Within the

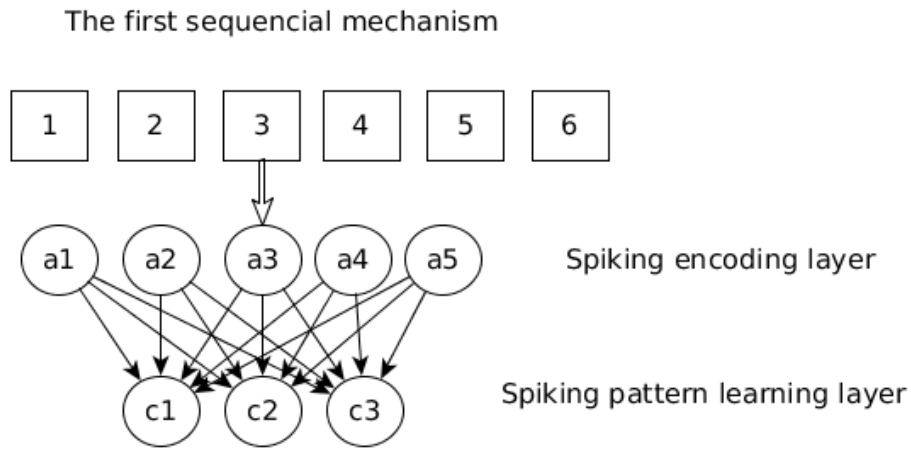


Figure 5.6: Network framework for the first sequential mechanism. Each spiking pattern within the sequence will be sequentially fed into the spiking encoding layer, i.e. (1,2,3,4,5,6)→(a1,a2,a3,a4,a5).

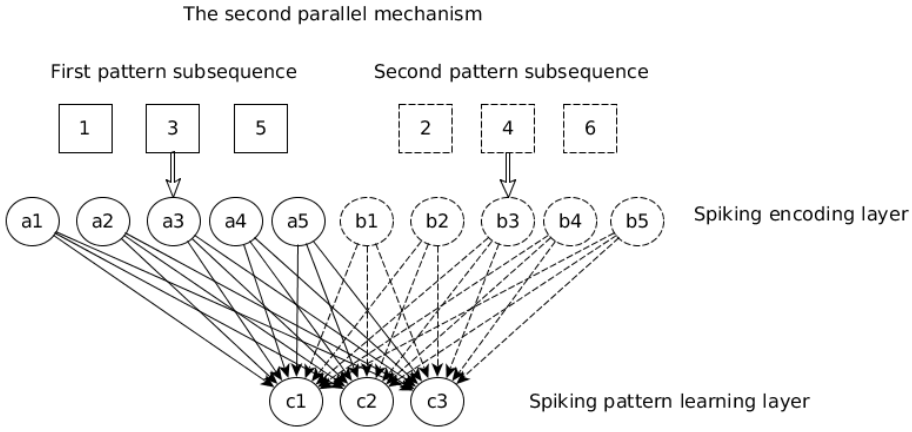


Figure 5.7: Network framework for the second parallel mechanism. The two pattern subsequences can be obtained by Figure 5.5. Each spiking pattern within the corresponding subsequence will be sequentially fed into its corresponding neuron group within the spiking encoding layer, i.e. (1,3,5)→(a1,a2,a3,a4,a5),(2,4,6)→(b1,b2,b3,b4,b5).

last layer, each neuron has lateral inhibition connections to all the other neurons. When the neuron fire a spike, other neurons within the last layer will be strongly inhibited. However, too strong inhibition will let only few neurons fire spikes and too weak inhibition will let all neurons fire spikes. Thus, to achieve a soft winner-take-all strategy, it is desirable to find a relatively balanced inhibition strength by tuning the weights of the inhibition connections. This processing scheme can be considered as a soft winner-take-all strategy. The input image belongs to the map (class) with the largest number of neurons firing the first spikes. Moreover, as mentioned in the above section, there

are two continuous input sequence mechanisms within the spiking encoding layer and different input sequence mechanism has different learning procedure.

Specifically, if choosing the first sequential mechanism, each spiking pattern within the sequence will be sequentially fed into the spiking encoding layer. Figure 5.6 shows the network framework of the first continuous input sequence mechanism. The learning procedure of the first mechanism can be summarized as following: when receiving an incoming spiking pattern, the associated synaptic weights of the neurons within spiking pattern learning layer will be updated. Within the interval between adjacent spiking patterns, the membrane potentials of these neurons will be gradually reduced to their initial values and then these neurons can start to receive the next spiking pattern.

When choosing the second parallel mechanism, the network framework can be depicted as Figure 5.7. Unlike the first mechanism, the number of neurons within the spiking encoding layer has been expanded to twice of the original. There are two neuron groups within the spiking encoding layer and each pattern neuron group corresponds to its associated spiking pattern subsequence. The associated synaptic connections within those two neuron groups share the same synaptic efficiency. For instance, within the Figure 5.7, when the synaptic efficiency W_{a1c1} updates, its associated synaptic efficiency W_{b1c1} within the second neuron group will also be automatically updated to the same value as W_{a1c1}. Such weight sharing strategy plays an important role in the update procedure.

Specifically, the learning procedure of the second mechanism can be summarized as following: when the first neuron group receives the input spiking pattern and starts to update its synaptic efficiencies, like Figure 5.7 shows, the second neuron group will be idle. For the neurons within the second neuron group, the membrane potentials remain at their initial values. Therefore, when receiving the input spiking pattern, the neurons within the second neuron group can use the latest updated synaptic efficiencies and start updating their synaptic efficiencies without any delays. In the meantime, the membrane potentials within the first group will be gradually reduced to their initial values. The first neuron group can start updating immediately after the second

group finishing their updating procedure. The two neuron groups constantly updates the synaptic efficiencies by alternating the above process until no input spiking pattern been fed into the spiking pattern learning layer. To learn the spiking pattern sequence, we will introduce the adaptive thresholding method and event-driven STDP learning rule, respectively.

5.2.3.1 Adaptive Thresholding Method

As an asynchronous neural network, the proposed SNN suffers a major drawback: the neurons fired the first spikes will tend to fire spikes more easier than other neurons. This is because these neurons will immediately start integrating incoming spikes right after firing the first spikes. Hence, the membrane potentials of these neurons will be activated most and then fire spikes earlier than other neurons. The lateral inhibition will further accelerate such situation and this will in turn affects the final classification performance.

To achieve a stable network, we incorporate an adaptive thresholding method [26] into the spiking pattern learning layer. By incorporating such method, the more a neuron fires, the higher will be its membrane threshold. In other words, to fire a spike, the neuron needs to integrate more presynaptic spikes. Specifically, for each neuron, instead of using the predefined membrane threshold V_{thr}, an adaptive membrane threshold V_t will be incorporated within spiking pattern learning layer, which can be computed by the following equation:

$$\tau_v \frac{d}{dt} V_t = V_{thr} - V_t \tag{5.5}$$

where τ_v is the time constant of the adaptive membrane threshold V_t. V_t will increase every time the neuron fires a spike, as shown in the follows:

$$V_t = V_t + V_i \tag{5.6}$$

where V_i represents a predefined increment (unit: mV).

5.2.3.2 Event-driven STDP Learning Rule

To learn selectivities from the spiking pattern sequences, spike timing dependent plasticity (STDP) [41], [42], [43], [44], [45], a temporally asymmetric form of Hebbian learning [40], has been applied within the proposed SNN. It is widely believed that it underlies learning and information storage in the brain, as well as the development and refinement of neuronal circuits during brain development [90], [91]. STDP requires no prior information or teaching signals since it is essentially an unsupervised learning rule.

Similar to section 4.3.2, the traditional STDP function $\mathrm{W}(\mathrm{t})$ can be computed by equation 4.8 (t is the time difference between pre and postsynaptic spikes). For biological reasons, it is desirable to keep the synaptic efficacy in a predefined range. Thus, a hard bound strategy has been used to ensure the synaptic efficacy remains in the desired range $w^{min} \leq w \leq w^{max}$, where w^{min} and w^{max} are minimum and maximum value, respectively. Basically, if w^{min} and w^{max} are specified, values smaller than w^{min} become w^{min}, and values larger than w^{max} become w^{max}, which can be described as:

$$hardbound(w, w^{min}, w^{max}) = \begin{cases} w^{max}, & if \;\; w > w^{max} \\ w^{min}, & if \;\; w < w^{min} \\ w, & if \;\; w^{min} \leq w \leq w^{max} \end{cases} \tag{5.7}$$

Traditionally, if choosing all-to-all spike interaction, STDP needs to sum over all pairs of spikes to update the learning synaptic weight matrix and that is very inefficient. Besides, it would also be physiological unrealistic because the neuron cannot remember all its previous spike times. To overcome such drawbacks, in this chapter, event-driven STDP learning rule has been applied within the proposed SNN by involving two on-line, local learning rules that are applied only in response to occurrences of spike events. Such event-driven STDP learning rule is more biologically plausible and more efficient.

In order to introduce the event-driven STDP learning method, two new variables α_+ and α_- need to be defined firstly. They represent the "traces" of pre and postsynaptic activity. Within the traditional STDP learning procedure, a postsynaptic neuron will

only update its synaptic weights after it fires a postsynaptic spike. However, by tuning the "traces" variables, the neuron can update the synaptic weights whenever it receive a presynaptic spike or firing a postsynaptic spike. These "traces" can be computed by the following equations:

$$\begin{aligned} \tau_+ \frac{d}{dt}\alpha_+ &= -\alpha_+ \\ \tau_- \frac{d}{dt}\alpha_- &= -\alpha_- \end{aligned} \tag{5.8}$$

here, τ_+ and τ_- represent the pre- and post-synaptic activity traces time constant, respectively. When received a presynaptic spike, g_{ex}, A_+ and w will be updated as follows:

$$\begin{aligned} g_{ex} &= g_{ex} + w \\ A_+ &= A_+ + \alpha_+ \\ w &= hardbound(w + A_-, w^{min}, w^{max}) \end{aligned} \tag{5.9}$$

where the $hardbound$ function will make the synaptic weight remains in the desired range $w^{min} \leq w \leq w^{max}$, when fired a post-synaptic spike, A_- and w will be modified according to following equations:

$$\begin{aligned} A_- &= A_- + \alpha_- \\ w &= hardbound(w + A_+, w^{min}, w^{max}) \end{aligned} \tag{5.10}$$

and when received an inhibitory presynaptic spike, g_{in} will be updated as follows:

$$g_{in} = g_{in} + w^{in} \tag{5.11}$$

where w^{in} represents the fixed inhibitory synaptic weight.

In this section, we briefly introduces the framework of the proposed SNN, followed by detailed introduction of each layer. By using the proposed continuous input sequence mechanism, input images have been combined into input image sequence. After feeding these input image sequences into the first two layers, spiking pattern sequences have been generated and further fed into the last layer. The last layer is the only layer involving learning procedure and uses event-driven STDP learning method to train the selectivity for each input visual stimuli. Therefore, the whole learning

procedure can be described as follows:

1. Propagate a sequence of input images $(I_1, I_2, \ldots, I_n)$ into the feature extracting layer, and then obtain a sequence of global invariant $C2$ feature vectors $(F_1, F_2, \ldots, F_n)$ using HMAX model with sparsity and intermediate features, as shown in Figure 5.2. Specifically, for a input image I_j, a global invariant $C2$ feature vector F_j with 4096 (d in Figure 5.2) elements will be obtained, in which $j = 1, 2, \ldots, n$ and n is the number of input images.

2. Create a new map with 4096 neurons within the spiking encoding layer and use two continuous input mechanisms, as demonstrated in Figure 5.5, to generate a continuous spiking pattern sequence $(S_1, S_2, \ldots, S_n)$. Specifically, for a global invariant $C2$ feature vector F_j, we use equation 4.2 to obtain its corresponding spiking pattern S_j, where $j = 1, 2, \ldots, n$.

3. Create a new map with k neurons for each class within the spiking pattern learning layer. The spiking pattern learning layer and the spiking encoding layer are fully connected. To achieve the soft winner-take-all strategy, each neuron within the spiking pattern learning layer has lateral inhibition connections to other neurons within the same layer. Furthermore, initialize the SNN using the parameter settings in Table 5.1 and generate random synaptic weights w.

4. Propagate the continuous spiking pattern sequence $(S_1, S_2, \ldots, S_n)$ into the spiking pattern learning layer and then compute the membrane potential V_j of a neuron j within this layer according to equation 5.2, where $j = 1, 2, \ldots, N$ and N represents the number of neurons within the spiking pattern learning layer. If a neuron j in spiking pattern learning layer receives a presynaptic spike from a neuron i in the previous layer, the synaptic weight w_{ij} will be updated using equation 5.9; If the neuron j reaches the membrane threshold V_t and then fires a postsynaptic spike, the synaptic weight w_{ij} will be modified according to equation 5.10 and the neuron j will inhibit other neurons to fire by using the equation 5.11. Furthermore, when the neuron j fires a postsynaptic spike, its membrane threshold V_t will be adaptively changed as demonstrated in section 5.2.3.1. Dif-

Figure 5.8: Random examples of MNIST database.

ferent continuous input mechanism used in the spiking encoding layer will have different update procedure, as shown in Figure 5.6 and Figure 5.7.

5. Stop the above updating procedure if all spiking patterns within the sequence $(S_1, S_2, \ldots, S_n)$ have been fed into the proposed SNN once.

5.3 Experiments

To verify the proposed ECS methods, the convergence and robustness of the methods are demonstrated first, then two types of experiments are designed. One type of the experiment is using simple random sampling scheme in which only a small part of training/testing samples has been used to learn/test the selectivities. Another one is employing exhaustive scheme in which the whole training/testing samples have been used to learn/test the selectivities. We will use MNIST handwritten digital database in our experiments.

The simple random sampling scheme requires the learning methods to generate acceptable performance based on quite limited learning resources (such as running time or computational capacity). In most real scenarios, the database is expanded over time and cannot be obtained at once. Thus, the simple random sampling scheme is more flexible and is closer to the actual situation. Compared with simple random sampling scheme, exhaustive scheme is quite ideal, which remains in a static state with no new training/testing samples added into the database. Similar to section 4.4.1, MNIST handwritten digital database [113] has been used within this chapter.

5.3.1 Parameter Settings

As mentioned in section 5.2.1, the feature extracting layer is based on the base model proposed in [127]. Thus, within the feature extracting layer, the parameters besides d take the same values as [127]. Normally, to achieve a realistic neural simulation, the parameters often take the values within a limited predefined range. Table 5.1 shows the parameter settings used in the proposed SNN framework. Specifically, some of the parameters used in the proposed SNN framework take the the same values as [129] while the values of other parameters, like [130], are chosen by optimizing the whole classification performance. Note, the time resolution of this experiment is 0.1 ms.

5.3.2 Convergence and Robustness of The Proposed ECS Method

To validate the proposed method, we need to verify the convergence and robustness of the proposed method under different experimental conditions. For convergence, the proposed method should ensure that when the learning procedure enters the final stable state, the number of spiked neurons and their spiking timings will remain at a relatively stable level. For robustness, even incorporating background neural noise and time jitter, the proposed method should be robust enough to ignore such interferences and only concentrate on learning the input visual stimuli.

5.3.2.1 Convergence Analysis

For the proposed ECS learning method, the convergence property is essential since it will guarantee the learning procedure can reach a stable stage. Ideally, when the STDP learning procedure reaches the convergence state, the number of spiked neurons and their spiking timings should remain at a fixed state. Moreover, the synaptic efficacy matrix will be saturated to a special status with most weights take 0 and the rest take 1 [114]. However, in real scenarios, the input images often possess abundant intra-class variance and it is almost impossible for a learning method to cover all the intra-class variance. Therefore, in reality, the number of spiked neurons and their spiking timings will remain at a relatively stable level (not fixed) when reaching the convergence state.

Table 5.1: Parameter settings of the proposed SNN.

Parameter	Description	Value
d	the number of elements in a $C2$ vector [2]	4096
τ_m	membrane time constant [1]	10 ms
τ_v	potential threshold time constant [1]	20 ms
τ_{ex}	excitatory conductance time constant [1]	5 ms
τ_{in}	inhibitory conductance time constant [2]	10 ms
τ_+	presynaptic trace time constant [1]	20 ms
τ_-	postsynaptic trace time constant [1]	20 ms
p	a positive constant for ROC [2]	0.2
T_s	real processing time window [2]	150 ms
T_i	interval time window [2]	150 ms
T_{rf}	absolute refractory time [1]	1 ms
E_{ex}	excitatory membrane potential [1]	0 mV
E_{in}	inhibitory membrane potential [2]	-85 mV
V_r	resting membrane potential [1]	-74 mV
V_{thr}	membrane potential threshold [2]	-45 mV
V_i	increment for adaptive potential threshold [2]	5 mV
w^{min}	minimum synaptic weight [1]	0
w^{max}	maximum synaptic weight [2]	0.01
w^{in}	fixed inhibitory synaptic weight [2]	0.05
α_+	the presynaptic trace [1]	$0.01w^{max}$
α_-	the postsynaptic trace [1]	$-\alpha_+(\tau_+/\tau + -)1.05$

[1] Take the same value as [129].
[2] Optimized to achieve the best classification performance.

Correspondingly, the synaptic efficacy matrix will have weights with the value between 0 and 1.

Thus, quantizing the number of spiked neurons and their spiking timings should be enough to verify whether a learning method is convergent or not. In this chapter, a convergence index f has been proposed by adding the spiking timings of all fired neurons within a spiking pattern period. Specifically, for each separate spiking pattern period (include spiking pattern time window and neural noise time window) within the whole

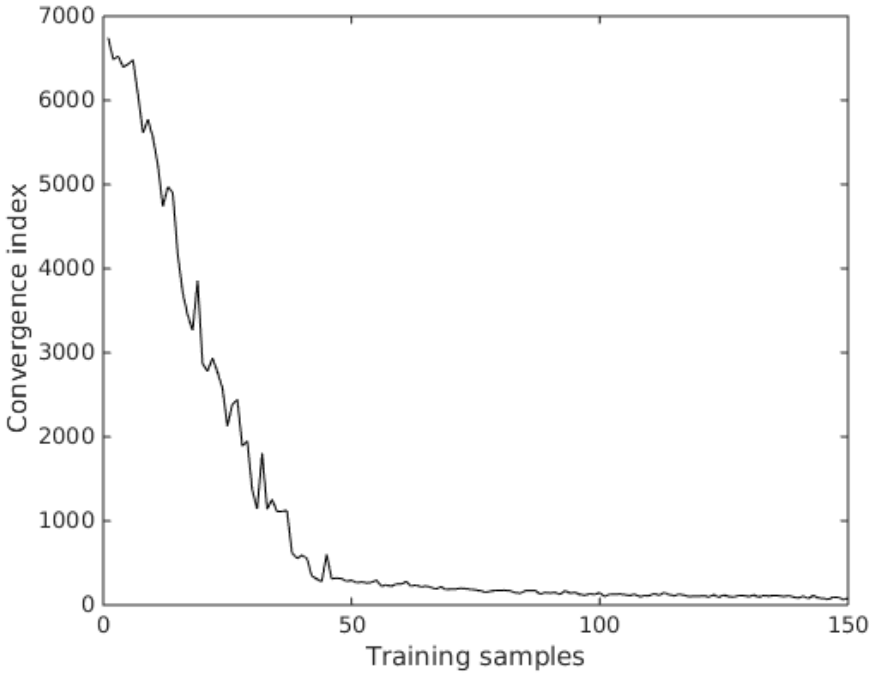


Figure 5.9: Dynamic convergence index using 150 random samples within the same chosen class.

spiking pattern sequence, the convergence index f can be computed as following:

$$f = \sum_{i=1}^{n} (st_i - init) \tag{5.12}$$

where n means the total number of spiked neurons within the spiking pattern learning layer, st_i represents the spiking timing (unit: ms) of that specific fired neuron and $init$ depicts the starting timing of the recent spiking pattern period.

Figure 5.9 shows the dynamic convergence index using 150 random samples within the same chosen class, while Figure 5.10 shows the learned synaptic weight matrix using the same 150 random samples as Figure 5.9. It can be seen from Figure 5.9 that the convergence index sharply decrease until around 50 and then enter a relatively stable stage. The learning weights within Figure 5.10 remains at its extreme values with relatively smaller number of them located at the between.

5.3.2.2 Robustness Analysis

Within ventral stream, there are interferences generated during the information transmission procedure, such as background neural noise and time jitter. To verity the

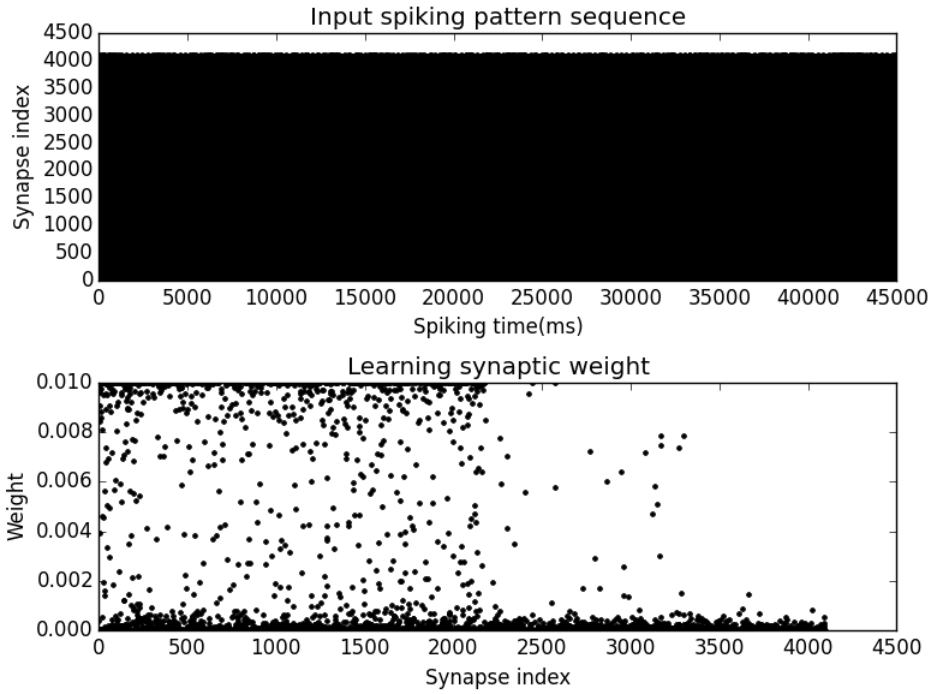


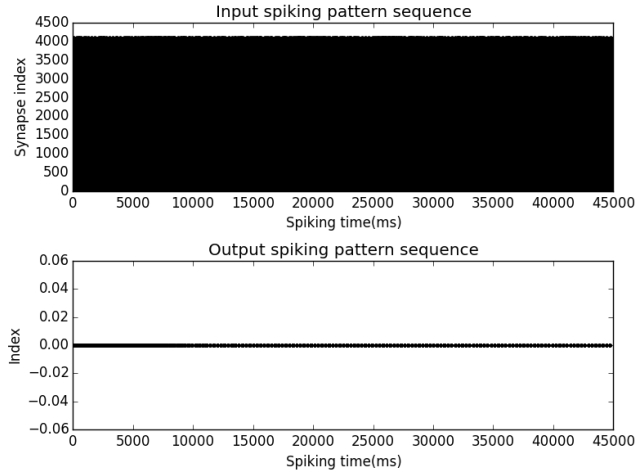
Figure 5.10: Learning synaptic weight using the same 150 random samples as Fig.11.

robustness of the proposed method, we should investigate the dynamic spiking status of the postsynaptic neurons when adding these interferences into the learning procedure. Ideally, the proposed method should be robust enough to ignore such interferences and only concentrate on learning the input visual stimuli.

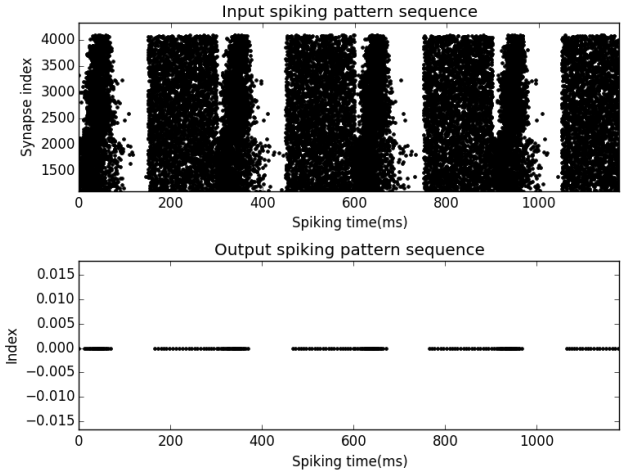
In this experiment, we will use the poisson point process to generate the background neural noise and standard normal distribution to produce the time jitter. Specifically, the time jitter is generated using the standard normal distribution and the uniform poisson point progress with frequency of 7.5 Hz (to simulate a common α brainwave) has been used to produce an background neural noise. The probability density function ($\phi(x)$) of the standard normal distribution can be expressed as follows:

$$\phi(x) = \frac{e^{-\frac{1}{2}x^2}}{\sqrt{2\pi}} \tag{5.13}$$

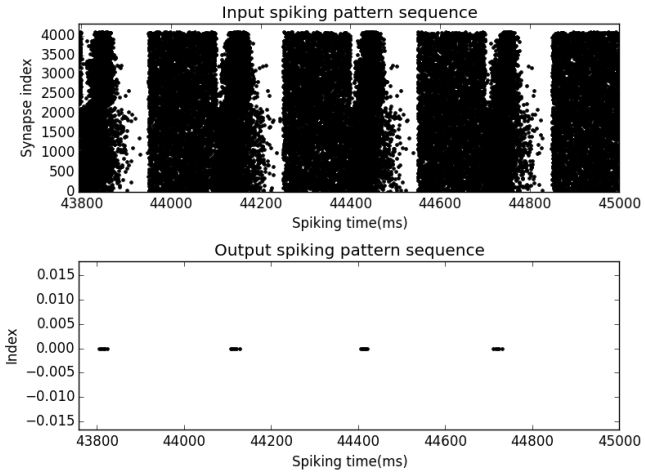
Given a compact set K, a point process X is defined as a mapping from a probability space to configurations of points of K [131]. Note, $N(A)$ is the number of point of a point process X falling in the Borel set A. Let $v(.)$ be a Borel measure on K, then a point process X on K is a poisson point process with intensity $v(.)$ if $N(A)$ is poisson distributed with mean $v(A)$ for every bounded Borel set A included



(a) The whole dynamic learning procedure using 150 random samples within the same class

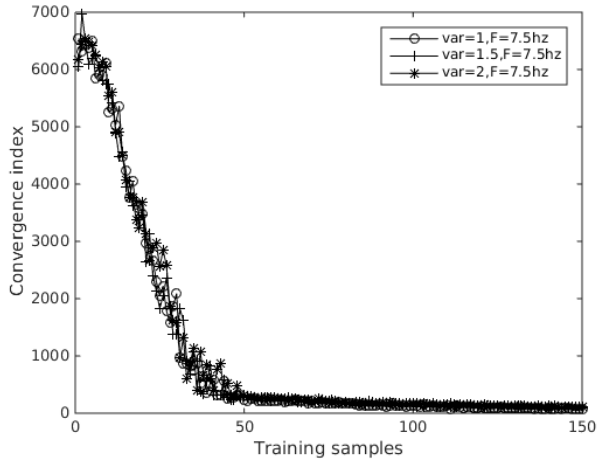


(b) The dynamic learning procedure of the first four spiking pattern period

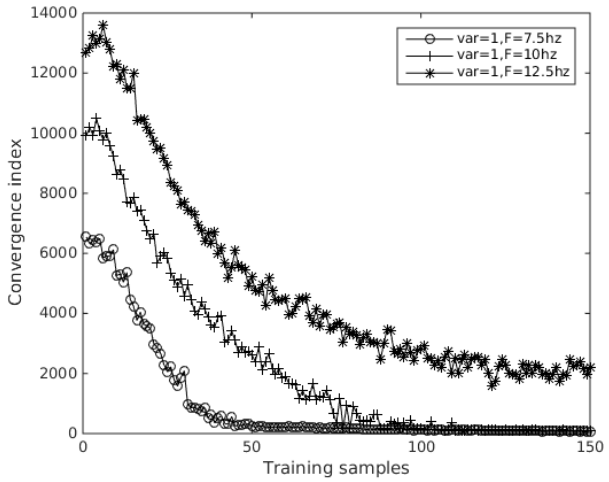


(c) The dynamic learning procedure of the last four spiking pattern period

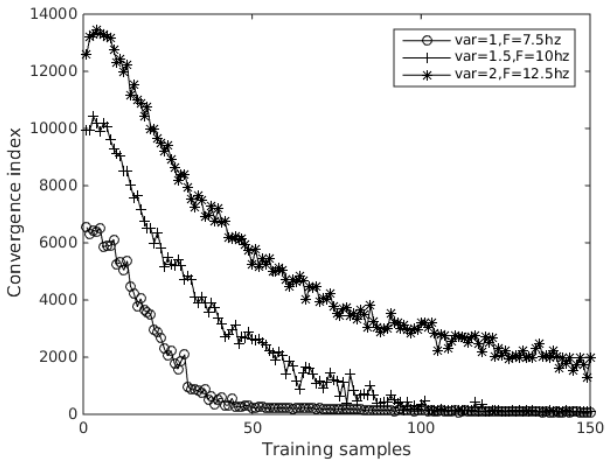
Figure 5.11: Robustness of the proposed method to the interferences.



(a) Same frequency and different variance



(b) Same variance and different frequency



(c) Different variance and different frequency

Figure 5.12: Dynamic convergence index under different interferences settings (var represents the variance of the time jitter and F is the frequency of the background neural noise).

in K and the random variables $N(A_1), ..., N(A_k)$ are independent for any k disjoint bounded Borel set $A_1, ..., A_k$. Uniform poisson point process [132], the most simple poisson point process, is the point process with intensity measure being proportional to the Lebesgue measure on K:

$$v(.) = \beta\lambda_k(.) \tag{5.14}$$

The mean number of points falling into K is then:

$$\mathbb{E}[N(K)] = \beta\lambda_K(K) \tag{5.15}$$

where $\mathbb{E}$ represents expectation symbol and λ_K depicts the frequency of the poisson law. β is a positive constant value. A two steps procedure can be used to generate points in K with the distribution of this point process: Firstly, simulate N according to poisson law with mean given by the latter equation (it gives $N = n$); Secondly, sample each of the n points according to a uniform law on K.

Bascially, the background neural noise will be inserted into the interval between adjacent spiking patterns within the input spiking pattern sequence, while the time jitter will be added into the each spiking pattern itself within the same spiking pattern sequence. By doing so, we can simulate the likely spiking pattern sequence existed within the ventral stream and thus prove the robustness of the proposed method to those interferences.

In Figure 5.11, 5.11a shows the whole dynamic learning procedure using 150 random samples within the same class, 5.11b represents the dynamic learning procedure of the first four spiking pattern period and 5.11c depicts the dynamic learning procedure of the last four spiking pattern period. From 5.11b, it can be seen that there are several output spikes during the background neural noise time window, while in 5.11c, the output spikes only appear within the input spiking pattern time window and there are no spike generated within the background neural noise time window. Such behaviors indicate that those background neural noise will be gradually ignored with the continuous learning.

The possible reason of the above behaviors is that those background neural noise patterns are independent and identically distributed. Specifically, since the the background neural noise is generated from poisson point process, one background neural noise pattern within its corresponding neural noise time window is totally independent to another background neural noise pattern within another neural noise time window. According to probability theory, if two random variables are independent, they are irrelevant. This means the adjacent background neural noise patterns are irrelevant and there are no similarity between them.

Within the proposed event-driven STDP learning method, the integral area of LTD is larger than the integral area of LTP. Therefore, if each adjacent spiking patterns remain independent, the synapses between pre and postsynaptic neurons will be continuously depressed (LTD) and thus no postsynaptic neuron will fire a spike in the end. Otherwise, if the similarity between adjacent spiking patterns remains at a high level, by using the event-driven STDP learning method, the synapses corresponding to the most significant structural information of input visual stimuli will be reinforced to the maximum value and others with less importance will be depressed to minimum value, as shown in Figure 5.10.

To further investigate the influences of those interferences (neural noise and time jitter) to the event-driven STDP learning procedure, it is necessary to analyze the performance of the event-driven STDP learning method under three different interferences settings. Specifically, we will use the dynamic convergence index with different interferences settings to show the influences to the event-driven STDP learning method. Figure 5.12 shows dynamic convergence index under three different interferences settings. With the increase of the variance of the time jitter and the frequency of the background neural noise, more and more training samples are needed for the event-driven STDP learning method to enter the stable state, which also means the turning point is increasingly moving towards larger number of training samples. Specifically, compared with time jitter, the background neural noise influence more to the event-driven STDP learning method.

As discussed in the above subsections, the proposed ECS method can converge to a

stable stage within limited learning steps. Furthermore, it will ignore the interferences (the background neural noise and the time jitter) and only concentrate on learning the input visual stimuli.

5.3.3 Simple Random Sampling Experiments on MNIST Database

In this chapter, simple random sampling experiments on MNIST database have been firstly used to verify the proposed SNN and its ECS learning method. Note, we use 100 random testing samples extracted from MNIST testing database to test the classification performance. As mentioned above, in this chapter, poisson point process with frequency of 7.5 hz has been used to produce an background neural noise and the time jitter is generated using the standard normal distribution. From Figure 5.9, it can be seen that the event-driven STDP learning procedure hits the turning point at around 50 training samples and then enter the relatively stable period.

However, the exact number of training samples needed to generate the optimized classification performance is still unknown. Moreover, since the processing speed is quite important within the ventral stream, it is necessary to find a balanced decision criteria to choose a compromise optimized result between processing time and final classification performance. Ideally, the proposed method should achieve a better classification performance using lesser processing time.

In Table 5.2, HMAX[127], a classical convolution neural network (CNN) method, and the proposed event-driven continuous STDP (ECS) learning method have been used to classify the testing samples using different number of training samples. Note, in the above simple random sampling experiment, we use 100 random testing samples extracted from MNIST testing database to test the classification performance of the above methods. To be fair, for each random test with specific number of training samples, HMAX[127] and ECS use the same random training samples. It can be seen that the proposed ECS method with 100 random training samples strike the balance state between the processing time and final classification performance, which only use 23.3s to achieve 86.5% correct classification performance. Moreover, this performance is achieved when incorporating background neural noise and time jitter

Table 5.2: Correct classification performance using different random training samples with two different methods.

Condition	HMAX[127]:var=0, F=0hz; ECS:var=1, F=7.5hz					
Random test	Number of training samples					
	50		100		150	
	HMAX[127]	ECS	HMAX[127]	ECS	HMAX[127]	ECS
1	0.8	0.86	0.87	0.89	0.82	0.85
2	0.83	0.83	0.86	0.85	0.88	0.86
3	0.83	0.86	0.82	0.84	0.87	0.91
4	0.82	0.86	0.83	0.87	0.88	0.87
5	0.8	0.81	0.85	0.87	0.82	0.87
6	0.84	0.87	0.86	0.88	0.83	0.88
7	0.8	0.87	0.87	0.89	0.87	0.88
8	0.84	0.86	0.85	0.83	0.89	0.84
9	0.81	0.79	0.84	0.85	0.82	0.86
10	0.87	0.8	0.85	0.88	0.85	0.82
cp (average)	0.824	0.841	0.85	0.865	0.853	0.864
rt (average)	22.8s	23.1s	23.1s	23.3s	23.1s	23.4s

- Note: 0.8 in this table means 80% correct classification rate. To be fair, for each random test with specific number of training samples, HMAX[127] and ECS use the same random training samples and the same 100 random testing samples. cp represents the final correct classification performance and rt means the corresponding testing processing time used to generate that specific classification performance.

into the proposed spiking neural network. In other words, compared with HMAX[127] method, the proposed ECS method processes twice as many samples, which include 100 training samples and 100 background neural noise samples. Besides processing more samples, the proposed ECS method will also ignore those interferences along with the continuous learning and gradually focus on learning the real input spiking patterns.

Compared with HMAX[127] method, the proposed ECS method can still achieve a better performance even in quite harsh conditions. In order to generate a fully comparison between HMAX[127] and ECS, we still need to compare their classification per-

Table 5.3: Correct classification performance comparison between HMAX[127] and ECS using 100 training samples without any interferences.

Condition	HMAX[127]:var=0, F=0hz; ECS:var=0, F=0hz	
Random test	HMAX[127]	ECS
1	0.85	0.88
2	0.87	0.86
3	0.83	0.87
4	0.87	0.89
5	0.82	0.90
6	0.9	0.91
7	0.83	0.86
8	0.86	0.95
9	0.8	0.91
10	0.84	0.88
cp (average)	0.847	0.891
rt (average)	23.15s	22.75s

- Note: 0.85 in this table means 85% correct classification rate. To be fair, HMAX[127] and ECS use the same random training samples and the same 100 random testing samples. cp represents the final correct classification performance and rt means the corresponding testing processing time used to generate that specific classification performance.

formance when adding no interferences. Figure 5.13 shows the dynamic convergence index of three random tests using the proposed ECS method without adding adding any interferences (var=0, F=0hz). It shows the proposed ECS method will roughly hit the stable period around 100 training samples. For the sake of simplicity, we only show the classification performance comparison using 100 of training samples within the learning procedure. Table 5.3 shows the correct classification performance using both HMAX[127] and ECS methods when no interferences incorporated into the learning procedure. From Table 5.3, compared with HMAX[127] method, the proposed ECS method uses slightly less time (22.75s) to generate quite higher correct classification performance (89.1%).

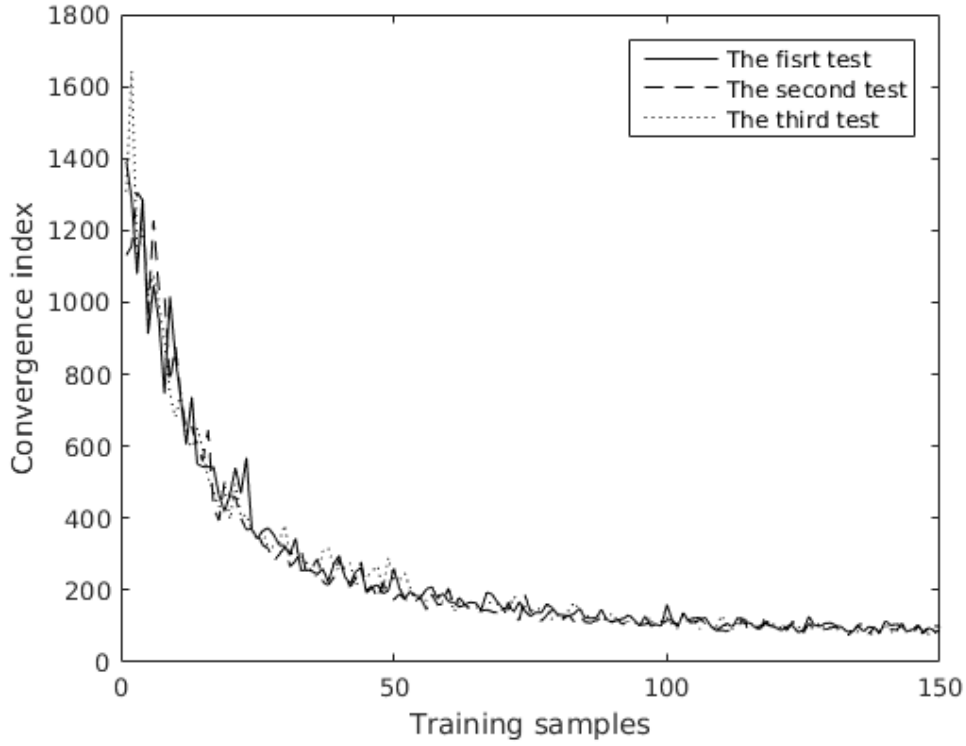


Figure 5.13: Dynamic convergence index of three random tests using the proposed ECS method without adding any interferences (var=0, F=0hz).

5.3.4 Exhaustive experiments on MNIST database

For fair comparison with other state-of-art methods, exhaustive experiments using all training/testing samples have also been conducted in this paper. Unlike the simple random sampling experiments, exhaustive experiments use the whole 60000 training samples to learn the synaptic weights (also known as selectivities) and then distinguish the whole 10000 testing samples based on the above learned synaptic weights.

Table 5.4 shows the simple random sampling and exhaustive classification accuracy performance using the proposed ECS method and different state-of-art learning methods. According to their spiking coding type, the learning methods within Table 5.4 have been divided into two categories: time-based and rate-based. Compared with proposed ECS learning rule, some methods within rate-based category achieves better performance, but they have two main shortages: 1) To achieve the best performance, all those methods need to feed the whole training samples into their SNN frameworks at least several times (several runs) since one run is not enough to generate meaningful spiking rate. 2) Some of the methods are not biologically plausible, for instance, spik-

Table 5.4: Classification accuracy performance using different methods on MNIST database.

Spiking Coding-type	Architecture	Preprocessing	(Un-)supervised	Learning Rule	Performance	
					Simple random sampling [a]	Exhaustive [b]
Time-based	Spiking convolutional neural network	Modified HMAX	Supervised	**ECS(this paper)**	**89%**	**93.0%**
	Two layer network[68]	Simplified HMAX	Supervised	Tempotron rule	79.0%	N/A
	Two layer network[69]	Simplified HMAX	Supervised	Tempotron rule	N/A	91.3%
Rate-based	Dendritic neurons[70]	Thresholding	Supervised	Morphology learning	N/A	90.3% [d]
	Spiking RBM[71]	None	Supervised	Contrastive divergence, linear classifier	N/A	89.0%
	Spiking RBM[72]	Enhanced training set to 120,000 examples	Supervised	Contrastive divergence	N/A	89.0%
	Spiking convolutional neural network[73]	None	Supervised	Backpropagation	N/A	99.1%
	Spiking RBM[74]	Thresholding	Supervised	Contrastive divergence	N/A	92.6% [c]
	Spiking RBM[74]	Thresholding	Supervised	Contrastive divergence	N/A	91.9% [c]
	Two layer network[75]	Edge-detection	Supervised	STDP with calcium variable	N/A	96.5% [e]
	Multi-layer hierarchical neural network[25]	Orientation-detection	Supervised	STDP with calcium variable	N/A	91.6%
	Two layer network[26]	None	Unsupervised	Rectangular STDP	N/A	93.5%
	Two layer network[27]	None	Unsupervised	Exponential STDP	N/A	95.0%

[a] Simple random sampling performance has been generated by averaging 10 random tests using 50 random training samples per class and 100 random testing samples, which is suitable for real-time learning since the whole database is impossible to obtain in most real scenarios.
[b] Exhaustive performance shows the ideal experimental results by using whole 60000 training samples and 10000 testing samples within MNIST database.
[c] The authors only use 1000 testing samples to obtain the performance
[d] The authors only use 5000 testing samples to obtain the performance
[e] The authors use 10000 randomly chosen samples from MNIST database instead of the dedicated testing database

ing convolutional neural network[73] using back propagation (BP) method achieves the best performance, however, in real scenarios, the global error signal used in BP algorithm is hard to obtain and it suffers various drawbacks, such as gradient vanishing/exploding or overfitting.

To overcome those drawbacks, the proposed ECS method uses time-based spiking neural network and it achieves the best performance by only using one run. Besides, we use available biologically plausible models to build the layers of the proposed SNN framework and incorporate the event-driven processing concept into the learning procedure. From Table 5.4, it can be seen that, within the time-based spiking learning methods, the proposed ECS method achieves the best classification performance in both simple random sampling and exhaustive experiments.

5.4 Conclusion

In this chapter, an event-driven continuous STDP learning method (ECS) using HMAX model has been proposed. Instead of feeding single input image separately, this paper uses the proposed continuous input sequence mechanism to generate the input image sequences. After applying the modified HMAX model with sparsity and intermediate variables, the invariant high level features have been extracted from the input image sequences. Through the proposed spiking encoding scheme, the spatiotempo-

ral spiking pattern would be generated and such spatiotemporal information conveys unique and distinguish selectivity to each input visual stimuli. Two novel continuous input sequence mechanisms have been proposed and each mechanism uses different update procedure to learn the synaptic efficiency matrix. After using event-driven STDP learning method, the final selectivity would be emerged with the corresponding update procedure. Even incorporating background neural noise and time jitter into the input visual stimuli, the simple random sampling and exhaustive experimental results on MNIST database using the proposed method still achieve acceptable correct classification performances. Even though, several parts within the proposed method still need to improve for better performance. In order to to generate more than one spike per synapse connection, a more comprehensive spiking encoding scheme is needed. Besides, the feature extracting layer within the proposed method has not been processed using spiking neural networks. Therefore, overcoming those limitations should be the priorities of our future work.

Chapter 6

Spiking Neural Network for Video-based Disguise Face Recognition

Face recognition using still face images has been widely investigated in the last several decades. Compared with the traditional still images-based face recognition, video-based face recognition (VFR) is still in its initial stage. But, with the vast popularity of social media and the increasingly lower cost of the smart devices, the importance of VFR application will increase dramatically over the foreseeable future. Various VFR methods [53], [54], [55], [95], have been proposed to recognize the faces with different kinds of variations within the testing video database, such as pose, expression, lighting, blur and face resolution.

However, none of these testing video databases include the disguise variation. Actually, most of the existed VFR methods [53], [54], [55], [95] cannot deal with such scenarios since the feature vectors within their methods can only be generated when certain areas (eyes, nose, mouth) of the faces are visible within the video database. Therefore, for video-based disguise face recognition (VDFR) application such as looking for lost persons in train station or recognizing the terrorists in airport, the existed VFR methods are not suitable.

To address the above problem, in this chapter, the proposed ECS method has been extended to accomplish the VDFR tasks using the dynamic facial movements. Unlike the traditional VFR methods, the proposed methodology can still work well when certain areas of the face is not visible. Our experiments on the proposed video disguise

face database (VD Face DB) demonstrated the proposed VDFR methods are reliable and efficient.

6.1 Background

Extensively researches have been conducted on the still images-based face recognition applications over the last several decades. Specifically, various methods [133], [134], [135] have been proposed to resolve the disguise detection applications and achieved satisfactory results. Most of these methods usually divide the face area into several patches and use methods such as PCA (Principal component analysis) or ITE (Intensity and Texture Encoder) to generate high level features from these patches. The classification procedure normally uses SVM (Support Vector Machine) or LBP (Local Binary Pattern) to recognize different feature patterns. However, most of these methods use still images as the input and extract the static facial features/patches without any consideration of the temporal relationship between different frames.

However, with the vast popularity of social media and the increasingly lower cost of the smart devices, more and more researchers shifted their attentions from still image-based face recognition (SIFR) to video-based face recognition (VFR). Within real-world VFR application, at least one of query or target needs to be video sequence. Therefore, three different scenarios need to be investigated within VFR: 1) Video-to-Still (V2S); 2) Still-to-Video (S2V); 3) Video-to-Video (V2V) [54].

Various video face databases such as CMU MoBo, Honda/UCSD, YouTube Celebrities and YouTube Faces DB have been built to test the different kinds of VFR methods. Different video face database includes different kinds of variations. Basically, the variations included in these databases can be summarized as varying pose, illumination, expression, resolution, motion blur and walking. Clearly, none of these testing video databases include the disguise variations. However, VDFR plays an important role in scenarios like looking for lost persons in train station or recognizing the terrorists in airport. To resolve the problem, in this chapter, a novel video disguise face (VD Face DB) database has been built.

It has been shown that a set of microexpressions using facial action coding sys-

tems (FACS) cannot be mimicked and can provide authenticity for person identification [136]. it is common sense that changing the subconscious actions such as the movements of facial muscles is an extremely difficult task for human beings. Inspired by this phenomenon, in this chapter, the temporal dynamic information pattern will be obtained from the movements of facial muscles and used to recognize different subjects even when they try to disguise themselves. Various available biologically plausible models will be used to represent the temporal dynamic information patterns and learn the selectivities from these temporal dynamic information patterns.

Specifically, the absolute differences of adjacent video frames have been used to represent the movements of the facial muscles, similar to those differential images fed into motion sensitive neuron models [96], [97]. If there are no moving background, such frame differences can be considered as the representation of the movements of facial muscles by themselves. Apparently, the frame differences would be zero if the subject remains still within the training period. Thus, to avoid such scenarios within the proposed video face database, the subjects will be asked to say a few sentences in front of the camera. However, the frame differences themselves are not enough to represent the video frames within the video face database since there still exists large volumes of the pattern variations. It is known that the ventral stream, a hierarchical system in which each layer extracts different level of abstractions, plays an important role in form recognition and object representation. As discussed in the previous chapter, the framework of the event-driven continuous STDP (ECS) learning method can be used to simulate the ventral stream. Thus, in this chapter, the framework of the ECS method will be used to reduce the pattern variations.

Generally speaking, the traditional VFR methods can be summarized as two main types: sequence-based methods [53], [55] and set-based methods [54], [95]. Basically, the sequence-based methods use the temporal dynamic information of the face among the adjacent video frames while the set-based methods does not use this temporal dynamic information, considering the video as image set of the separated video frames. Essentially, the common grounds of these two kinds of methods are using different feature vectors to represent the video frames. The former one tries to extract the tem-

poral dynamic information from the feature vector patterns while the latter one models the feature vector patterns, also known as set, and use different correlation methods to compute the set-to-set distance.

Normally, to obtain the feature vectors using these methods, certain areas like eyes or nose of the face within the video frames need to be visible. For instance, in paper [55], within the proposed individual expression recognition (IER) block, to obtain the behavioral map (BM) containing facial evolutions of microexpressions in each frame, at least an eye, a brow or a cheek need to be detected within the video frames. In [53], the authors use genetically-inspired learning method to select meaningful facial features obtaining from five local areas, such as eyes, nose and mouth. These algorithms are essentially sequence-based methods. Similar situations can also be extended into the set-based methods. However, within VDFR applications, such critical requirement cannot be satisfied and thus these methods are not suitable for this specific scenario. Moreover, most of these methods use SVM or multi-perceptron as the classification algorithms, which are clearly not biologically plausible.

To address the above problems, based on the ECS learning method, a novel VDFR method using the movements of the facial muscles has been proposed and it has been accomplished using the available biologically plausible models. Specifically, based on the modified HMAX model, the proposed method extracts dynamic features from the input video clips and uses the proposed spike encoding scheme to obtain the spiking patterns. Within the spiking pattern learning procedure, an event-driven continuous spike-timing dependent plasticity (ECS) learning method has been used to learn the selectivities from the spiking pattern sequences, which not only improves the learning efficiency but also ensures the input visual stimuli can be learned without resetting the intermediate variables during each learning step. Experimental results on the proposed testing database shows the proposed method can achieve a quite high correct classification performance.

6.2 Framework of the Proposed SNN

It is known that ventral stream is a hierarchical system in which each layer extracts different level of abstractions [6]. Its aim is to build an invariance to position and scale first, and then to viewpoint and other transformations [109]. To simulate the hierarchical structure of the ventral stream, a novel feed-forward SNN framework has been proposed in this chapter. This framework includes 5 layers: dynamic movements extracting layer, high level feature extracting layer, spiking encoding layer, spiking pattern learning layer and output layer.

Specifically, to obtain the dynamic movements of the facial muscles, the dynamic movements extracting layer tries to compute the absolute difference of adjacent frames. In other words, when fed the input training video, the proposed SNN framework focus on dynamic moving features rather than still frame features. Furthermore, within the high level feature extracting layer, those dynamic movements would be transformed to high level invariant features. The spiking pattern sequence generated from the spiking encoding layer would be trained within spiking pattern learning layer. Finally, the output layer accumulates different judgments from the spiking pattern learning layer and classifies the input sample video into predicted class.

Figure 6.1 shows the framework of the proposed feed-forward SNN. Within the spiking pattern learning layer, for each map, there are k neurons corresponding to k possible sub-classes and each neuron within the map has lateral inhibition connections to all the other neurons of the spiking pattern learning layer. The spiking encoding layer and the spiking pattern learning layer are fully connected. When a neuron fires a spike, other neurons will be strongly inhibited, which also can be considered as a soft winner-take-all strategy. Besides, finding a relatively balanced inhibition strength is extremely significant for this lateral inhibition strategy. Note, the amount of maps within the output layer is the same of the total classes and each map corresponds to a specific related class. There is only one neuron within each map and this neuron only connects with its related neurons within spiking pattern learning layer. The input video belongs to a certain class if the related map fires a spike firstly.

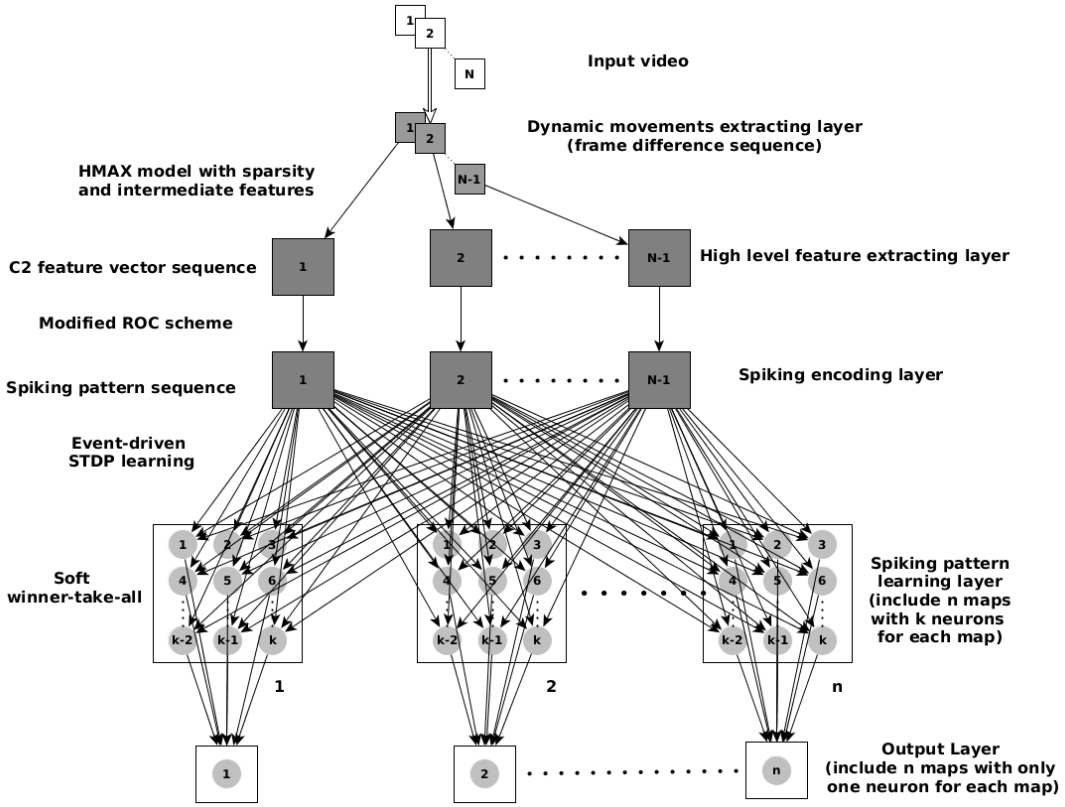


Figure 6.1: Framework of the proposed feed-forward SNN. For simplicity, the lateral inhibition connections of the spiking pattern learning layer have not been included.

6.2.1 Dynamic Movements Extracting Layer

It is known that the traditional face recognition methods cannot easily recognize the faces if the training samples try to disguise themselves. However, for human beings, changing the subconscious actions such as the movements of facial muscles is an extremely difficult task. Essentially, the movements of facial muscles represent the dynamic characteristics of the face. In other words, the traditional facial recognition methods use static features of the subject while the movements of facial muscles use dynamic features of the subject.

Given the video frames as the input, the absolute differences of adjacent video frames contain the dynamic movements of facial muscles. Furthermore, if there are no moving background, such frame differences can be considered as the representation of the movements of facial muscles by themselves. Apparently, the frame differences would be zero if the subject remains still within the training period. Thus, such scenarios should be avoided within this chapter.

6.2.2 High Level Feature Extracting Layer

For the VDFR application, it is not enough to just use dynamic movements as the input of the spiking encoding layer since there are still lots of intra-class variances within the dynamic movements. Thus, the high level features are much needed for further processing. According to deep learning theory, the high level features should strike a balance state between invariance and distinguishability [9], which has a large impact on the final classification performance. Based on the computational model proposed by Mutch and Lowe [127], the high level feature extracting layer can be built according to section 5.2.1.

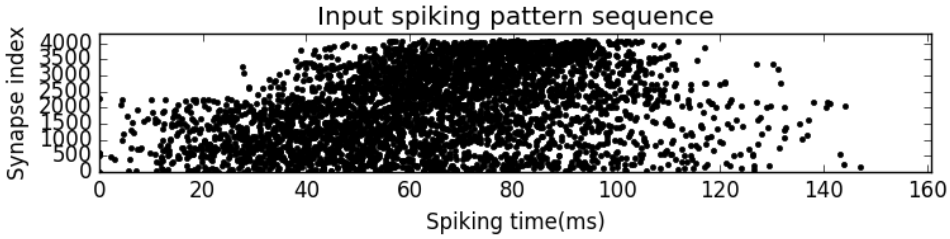
6.2.3 Spiking Encoding Layer

Within visual cortex, neurons use spikes to transmit information to other neurons. Traditionally, spike rate has been considered to contain most, if not all, information of the input visual stimuli. Rate-based SNN assumes that as the intensity of a stimulus increases, the associated spike rate increases. However, such mechanism ignores one significant fact: short processing time window is not enough for rate-based SNN to generate meaningful spiking rate. Recent neuroscience studies show that mammalian brain only use millisecond scale time window to process the complicated real-life visual recognition tasks [2]. Moreover, if the input visual stimuli has been incorporated with the background noise with the same spiking rate, it is impossible for rate-based SNN to generate the selectivity for the input visual stimuli [29].

To resolve the above problems, researchers use the specific spike timings of the fired neurons to represent the information. Compared with spike rate, it is believed such spike timing pattern conveys much more structural information. For instance, given each fired neuron only fires a single spike, the spike timing-based SNN can still distinguish different spiking patterns, which is impossible for spike rate-based SNN. On the other hand, even the spike rate remains unchanged, it is still possible to generate countless spike timing sequences and each of them corresponds to a different input visual stimuli.

Similar to section 5.2.2.2, rank order coding (ROC) scheme has been used within

(a) Input frame difference



(b) Spiking pattern

Figure 6.2: One input frame difference and its spiking pattern generated from the spiking encoding layer.

this chapter. For a global invariant $C2$ feature response r, the corresponding spiking timing t can be computed according to equation 4.2. Like section 5.2.2.2, in this chapter, the processing time window is set to 0.2 s (p takes 0.2 in equation 4.2). Figure 6.2 shows one input frame difference and its spiking pattern generated from the spiking encoding layer. Note, the C2 feature vector has 4096 elements, and thus there are 4096 neurons to fire spikes. Only those neurons with spiking timing less than 150 millisecond will fire spikes.

6.2.4 Spiking Pattern Learning Layer

Spiking pattern learning layer is the only learning layer within the proposed SNN framework. Given the initial state of the synaptic connections, the synaptic efficiency of the neurons within this layer will be adjusted over the learning period and gradually converged to the stable state. The final stable synaptic efficiency matrix represents the selectivity of the input visual stimuli and can be used to distinguish different classes within the testing period.

Within the spiking pattern learning layer, for each map, there are k neurons corre-

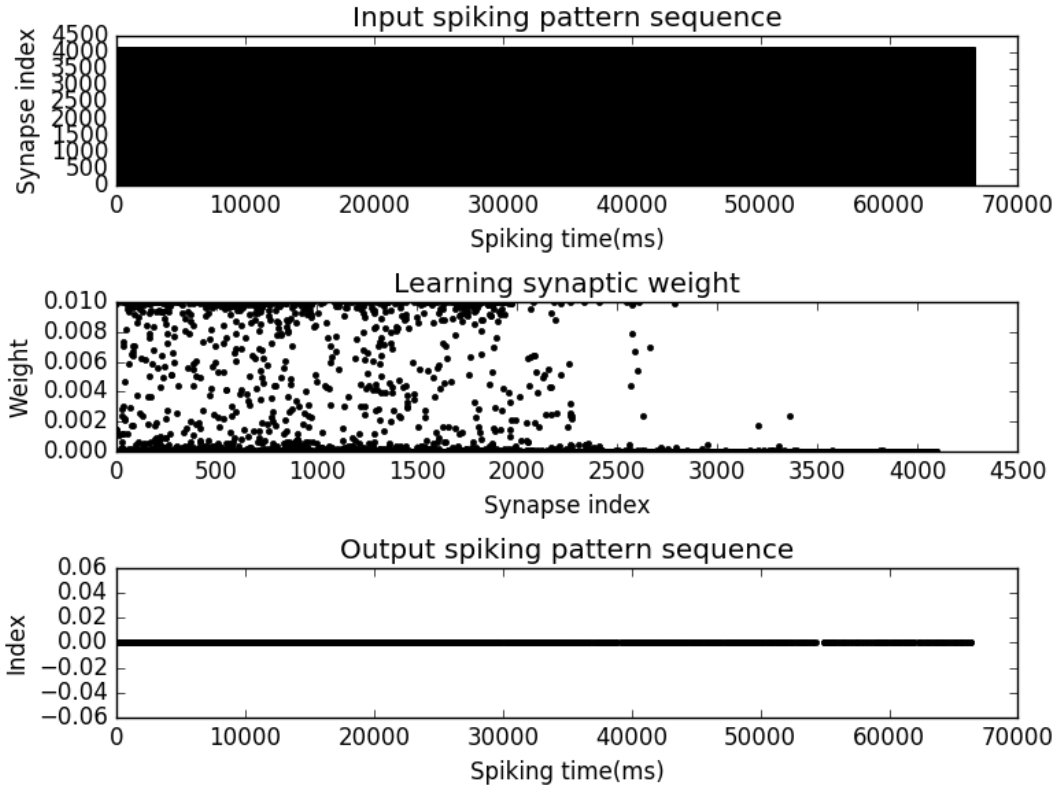


Figure 6.3: One input spiking pattern sequence and its learning results.

sponding to k possible sub-classes and each neuron within the map has lateral inhibition connections to all the other neurons of the spiking pattern learning layer. The spiking encoding layer and the spiking pattern learning layer are fully connected. When a neuron fires a spike, other neurons will be strongly inhibited, which also can be considered as a soft winner-take-all strategy. Besides, finding a relatively balanced inhibition strength is extremely significant for this lateral inhibition strategy. The specific learning method used in this chapter will be introduced in the following section. Figure 6.3 shows one input spiking pattern sequence and its learning results.

6.2.5 Output Layer

The amount of maps within the output layer is the same of the total classes and each map corresponds to a specific related class. There is only one neuron within each map and this neuron only connects with its related neurons within spiking pattern learning layer. The input video belongs to a certain class if the related map fires a spike firstly.

Specifically, during the testing period, for each input frame difference, one neuron

within the spiking pattern learning layer will fire a spike firstly and then this spike transmits to its related neuron within the output layer. Note, within the output layer, the membrane potential of the neuron will increase by a fixed amount when receiving a input spike and remain at the same without any reducing. When the membrane potential crosses the predefined threshold, the neuron will fire a spike. For instance, given a video clip includes 100 frames, it belongs to the class if at least 50 frames are classified as this specific class. Basically, for VDFR task, output layer acts as a information collecting and decision making hub.

6.3 Neuron Model and Learning Method

Within SNN, neurons need neuron model to interact with each other as it can be considered as the conduct principle. On the other hand, learning method plays an important role in training the selectivities from the input spiking patterns. Basically, those two factors are the core building blocks of a complete SNN algorithm and thus will be introduced in the following subsections respectively.

6.3.1 Neuron Model

Neural models define how the activities of the neurons change in response to each other. In this chapter, leaky integrate-and-fire (LIF) model stands out from the competition since it is biologically plausible and has low computational complexity. The leaking factor within the LIF model ensures that the neurons only fire spikes when there are enough presynaptic spikes fired from its receptive field within a relatively short time window. Thus, such neuron model can be considered as a coincidence detector. In this chapter, conductance based LIF model has been used as the neural model for the proposed SNN. Like in [129], [130], the membrane potential of the neuron (V) can be computed according to section 5.2.2.1.

6.3.2 Learning Method

Traditionally, if choosing all-to-all spike interaction, STDP needs to sum over all pairs of spikes to update the learning synaptic weight matrix and that is very inefficient. Besides, it would also be physiological unrealistic because the neuron cannot remember all its previous spike times.

To overcome such drawbacks, in this chapter, event-driven STDP learning rule has been applied within the proposed SNN by involving two on-line, local learning rules that are applied only in response to occurrences of spike events. The details can be found in section 5.2.3.2 . Compared with the traditional STDP, such event-driven STDP learning rule is more biologically plausible and more efficient. Furthermore, similar to section 5.2.3.1, an adaptive thresholding method has been used to achieve a stable network. Therefore, the whole learning procedure can be described as follows:

1. Propagate an input video into the dynamic movements extracting layer and obtain the frame difference sequence.

2. Given a frame difference sequence as the input, the synaptic weights w within the spiking pattern learning layer can be updated using the learning procedure described in section 5.2.3.2.

6.4 Experiments

To verify the proposed VDFR method, a novel video disguise face database (VD Face DB) has been built since the traditional video face databases have no disguise variations. Based on this novel database, various experiments using the proposed VDFR method have been examined under different experimental conditions. Specifically, VD Face DB itself will be introduced in the first subsection, followed by the parameter settings of the proposed method. The experimental performances and analysis will be elaborated in the last subsection.

Figure 6.4: Sample frames without disguise extracted from the VD Face DB.

6.4.1 The Proposed Video Disguise Face Database

Unlike the traditional face recognition methods, the proposed method tries to differentiate different faces based on video-based database instead of still images-based database. To achieve the VDFR, the human identities are recognized not just on a few images but the sequences of video stream showing facial muscle movements while speaking. Compared with other video face databases, VD Face DB introduces the disguise variations to the faces within the video frames.

To simulate the real scenarios, the database consists of two different experimental conditions: subject without any disguise and subject with disguise. In this experiment, to achieve the disguise condition, each subject will be asked to wear different sunglasses, hats and fake beards. Figure 6.4 shows some sample frames of three subjects without any disguise while Figure 6.5 demonstrates the sample frames of the same

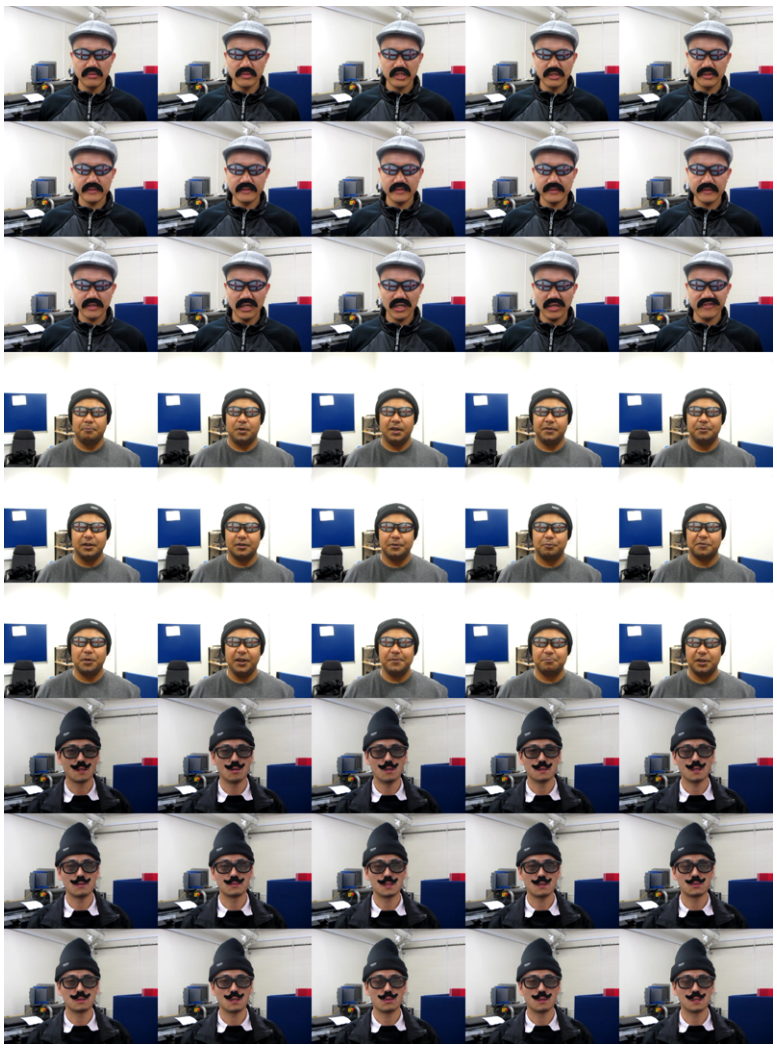


Figure 6.5: Sample frames with disguise extracted from the VD Face DB.

subjects with various disguises. All sample frames are extracted from the proposed VD Face DB.

The database includes 20 subjects totally. There are 5 video clips for each subject and those video clips have been recorded under various experimental conditions. Within each video clip, the subject has been asked to say a few sentences in front of the camera. The frame rate of each video clip is 30 frames/sec and the recording time per subject per time is 3 seconds. The size of each frame within each video clip is set to 320×240. Table 6.1 shows the basic information of different video face database. The proposed video disguise face database (VD Face DB) has been emphasized using bold text. Even though the number of experimental subjects/videos are not comparable with other huge databases like YouTube Faces DB or COX Face DB, the proposed VD Face DB has been incorporated with disguise variations and none of these databases

Table 6.1: Basic information of different video face databases.

Database	Number of subjects/videos	Variations	Scenarios
CMU MoBo	25/150	w	V2V
First Honda/UCSD	20/75	p	V2V
Second Honda/UCSD	15/30	p	V2V
CMU FIA	214/214	p,l,e	V2V
CamFace	100/1400	p,l	V2V
Faces96	152/152	l,r	V2V
VidTIMIT	43/43	p,e	V2V
YouTube Celebrities	47/1910	p,l,e,r,b,w	V2V
MBGC	821/3764	p,l,e,r,b,w	V2S/V2V
ND-Flip-QO	90/14	l,e,r,b	V2V
YouTube Faces DB	1595/3425	p,l,e,r,b,w	V2V
Chokepoint	29/48	p,l,e,r,b	V2V
ScFace	130/910	p,l,r	V2S/V2V
UT Dallas	284/1016	p,l,e,r,b,w	V2S/V2V
UMD Comcast10	16/12	p,l,r,b,w	V2V
PaSC	265/2802	p,l,e,r,b,w	V2S/V2V
Celebrity-1000	1000/7021	p,l,e,r,b,w	V2V
COX Face DB	1000/3000	p,l,e,r,b,w	V2S/S2V/V2V
VD Face DB	**20/100**	**l,e,d**	**V2V**

• Note: Variations include pose(p), illumination(l), expression(e), resolution(r), motion blur(b), walking(w) and disguise(d).

consider such factor.

6.4.2 Parameter Settings

Within the proposed feed-forward SNN framework, besides the postsynaptic potential threshold V_{thr}, the parameters for the first four layers take the same values as Table 5.1. In this experiment, to obtain the best classification performance, the postsynaptic potential threshold V_{thr} is set to -45 mV. Furthermore, as mentioned in section 6.2.5, the membrane potential of the neuron will increase by a fixed amount when re-

ceiving a input spike and remain at the same without any reducing. Thus, given a video includes m frames, it belongs to the class if at least $0.5m$ frames are classified as this specific class. Note, the time resolution of this experiment is 0.1 ms.

6.4.3 Experiments and Discussions

To verify the universality of the proposed VDFR method, various experiments under different experimental conditions need to be explored. In the following subsections, we will test the proposed method by increasing the recognition difficulty.

6.4.3.1 Experiment without any disguise

Before exploring the experiments with disguise, we need to make sure the proposed VDFR method can successfully distinguish different faces without any disguise. This is the basic requirement for the proposed VDFR method. Without such ability, it is impossible for the proposed VDFR method to recognize the faces with disguise.

Specifically, in this subsection, we use different number of video clips without any disguise to train the selectivity matrix and then test the recognition performance on the remaining video clips without any disguise. Note, some subjects within these video clips wear glasses (not the sunglasses within disguise video clips). 10 subjects have been chosen from the database for this specific experiment.

Table 6.2 shows the correct classification performance using different number of training/testing video clips. By using the proposed VDFR method, only 2 training video clips within the total 5 video clips are enough to flawlessly recognize different faces without any disguise. Such high convergence speed is critical for the face recognition applications. In other words, the proposed VDFR method requires less training samples to train the selectivity matrix.

Through this experiment, we demonstrate the proposed VDFR method can flawlessly distinguish different faces without any disguise by only using limited training video clips. Such capability is essential and critical for the following experiments with disguise. However, this is a initial requirement for the proposed method and we still need to verify the proposed method can still work well within not so ideal situation.

Table 6.2: Correct classification performance using different number of training and testing video clips.

Number of training video clips	Number of testing video clips	Performance
1	4	0.925
2	3	1.00
3	2	1.00

• Note: 0.925 in this table means 92.5% correct classification rate.

6.4.3.2 Experiment with disguise

To accomplish the VDFR task, in this subsection, we will feed video clips with disguise within the testing period. According to the above experiment, it has been shown that 2 video clips without any disguise are enough to learn the selectivity. Therefore, we will use 2 video clips without any disguise as the training samples and test the recognition performance using the remaining 3 video clips (with various disguises). All 20 subjects within VD Face DB will be used in this experiment. Table 6.3 shows the average classification performance of the two different methods, CNN [127] and the proposed VDFR method, on testing the disguise video clips in 100 random tests. It can be seen that the proposed VDFR method can averagely achieve 94.7% correct classification performance, while CNN [127] method just obtains 87.3% correct rate. Moreover, we have conducted a Wilcoxon signed-rank test on the correct classification performances by using the above two methods and computed the corresponding significance level $p - value$ (0.005364). Since the significance level $p - value < 0.05$, it clearly shows that the two correct classification performances are statistically different.

Generally speaking, the traditional VFR methods can be summarized as two main types: sequence-based methods and set-based methods. Essentially, the common grounds of these two kinds of methods are using different feature vectors to represent the video frames. The former one tries to extract the temporal dynamic information from the feature vector patterns while the latter one models the feature vector patterns, also known as set, and use different correlation methods to compute the set-to-set distance.

Table 6.3: Classification performances of two different methods on testing video clips with disguise.

Method	Correct rate	Wrong rate	Unknown rate
CNN [127]	87.3%	12.7%	0
Proposed VDFR method	94.7%	5.3%	0

- Note: The correct classification rate has been computed by averaging 100 random tests. Furthermore, we have conducted a Wilcoxon signed-rank test on the correct classification performances by using the above two methods and computed the significance level $p-value$ (0.005364). Such significance level ($p-value < 0.05$) indicates that the two correct classification performances are statistically different.

Normally, to obtain the feature vectors using these methods, certain areas like eyes or nose of the face within the video frames need to be visible. For instance, in paper [55], within the proposed individual expression recognition (IER) block, to obtain the behavioral map (BM) containing facial evolutions of microexpressions in each frame, at least an eye, a brow or a cheek need to be detected within the video frames. In [53], the authors use genetically-inspired learning method to select meaningful facial features obtaining from five local areas, such as eyes, nose and mouth. However, within VDFR application like this experiment, such critical requirement cannot be satisfied. For instance, in Figure 6.5, the sunglasses have covered the eyes of the first and second subjects. Moreover, most of these methods use SVM or multi-perceptron as the classification algorithms, which are clearly not biologically plausible.

6.5 Conclusion

With the vast popularity of social media and the increasingly lower cost of the smart devices, more and more researchers shifted their attentions from SIFR to VFR. Various VFR methods have been proposed to recognize the faces with different kinds of variations. However, none of these testing video databases include the disguise variation. Actually, most of the existed VFR methods cannot deal with such scenarios since the feature vectors within their methods can only be generated when certain areas (eyes, nose, mouth) of the faces are visible within the video databases.

To resolve the above problems, in this chapter, a novel video disguise face database (VD Face DB) has been built and then based on the available biologically plausible models and the event-driven continuous spike-timing dependent plasticity (ECS) learning method, a novel VDFR method using the movements of the facial muscles has been proposed. Specifically, based on the modified HMAX model, the proposed method extracts dynamic features from the input video clips and uses the proposed spike encoding scheme to obtain the spiking patterns. Within the spiking pattern learning procedure, an event-driven continuous spike-timing dependent plasticity (ECS) learning method has been used to learn the selectivities from the spiking pattern sequences, which not only improves the learning efficiency but also ensures the input visual stimuli can be learned without resetting the intermediate variables during each learning step. Experimental results on the proposed testing database shows the proposed method can achieve a quite high correct classification performance. Even though the proposed method achieved a satisfactory performance, it still has several limitations: the feature extracting layer within the proposed method has not been processed using spiking neural networks and it cannot deal with the scenarios with complex moving distractions within the background. Thus, overcoming those limitations should be the priorities of our future work.

Chapter 7

Conclusion

For most animal species, reliable and fast visual pattern recognition is vital for their survival. Within computer science, machine learning is often used to accomplish the visual pattern recognition task. However, recent research shows an mammalian brain can process complicated real-life visual pattern recognition scenarios at milliseconds scale, which proposes a huge challenge for traditional machine learning methods. Thus, using spiking neural network to simulate the visual cortex, especially the ventral stream, has became a natural choice for the researchers. In this thesis, based on the available biologically plausible models, several novel methods have been proposed and successfully applied on still image-based and video-based visual pattern recognition tasks, respectively.

Two main methods, batch learning rule [23], [24], [46] and on-line learning rule [47], [48], [49], have often been used to address the face recognition tasks based on spatiotemporal information extracted from spiking neural network (SNN). However, since they both incorporate winner-take-all strategy within the classification procedure, the batch learning rule only considers the average cluster within the class while the on-line learning rule just relies on the nearest relevant single sub-cluster. Both of them will not work well if there are obvious overlaps between classes. To overcome the above drawback, a novel learning rule inspired by soft winner-take-all strategy has been proposed. Specifically, it will assign a probability for each related class and the testing sample will be classified to the class with the highest probability. Compared with the two state-of-the-art methods mentioned above, experimental results on the

ORL face database show the proposed method can achieve a better performance.

In most cases, new visual pattern should be learned in a limited time window to adapt to new environments or changes promptly. Such learning procedure can be characterized as real-time learning. Recent research shows a biological brain can process complicated real-life recognition scenarios at milliseconds scale [2], which is obviously impossible for the traditional visual pattern recognition methods. Several papers [17], [68] based on SNNs have tried to solve the above problem. However, the method used in paper [68] needs prior knowledge before learning - in many cases, this prior knowledge is hard to obtain. Even the paper [17] incorporated STDP as their learning rule, it is only used as a local feature extractor. More importantly, these methods have not achieved real-timing learning. To overcome these drawbacks, a novel real-time learning method has been proposed by combining the spike timing-based feed-forward spiking neural network (SNN) and the fast unsupervised spike timing dependent plasticity learning method with dynamic post-synaptic thresholds. Experimental result on MNIST database shows the proposed method can efficiently achieve an acceptable accuracy within a limited time window.

The ventral stream, one type of visual processing pathway, plays an important role in form recognition and object representation. But, the underlying processing mechanism of the ventral stream is still largely unknown, which proposes a huge challenge for the researchers. Two main categories of methods, spiking rate-based methods [25], [26], [27], [70], [71], [72], [73], [74], [75] and spiking time-based methods [17], [68], [69], have been proposed to address the above issue. However, spiking rate-based methods [25], [26], [27], [70], [71], [72], [73], [74], [75] suffers two main drawbacks: limited learning time, which often exists in ventral stream, is not enough to generate a meaningful spiking rate and these methods often lack direct biological supports. Unlike the above methods, the authors in paper [17], [68], [69] use spiking time-based methods. But, the supervisory signal used in the first two papers has no strong experimental confirmation and the STDP method used in the last paper is only acted as a local feature extractor. To solve the above problems, based on the available biologically plausible models, an event-driven continuous spike-timing dependent plas-

ticity (STDP) learning method (ECS) using HMAX model has been proposed. The cross-validated and exhaustive experimental results on MNIST database show that the proposed method, compared with other state-of-art methods, can achieve comparable correct classification performances, but with more biologically plausible supports.

So far, the proposed methodologies have only been applied on still image-based visual pattern recognition tasks, but they can also extend to video-based visual pattern recognition tasks. With the vast popularity of social media and the increasingly lower cost of the smart devices, the face recognition tasks have been shifted from still image-based to video-based. Various video-based face recognition (VFR) methods [53], [54], [55], [95], have been proposed to recognize the faces with different kinds of variations within the testing video database, such as pose, expression, lighting, blur and face resolution. However, none of these testing video databases include the disguise variation. Actually, most of the existed VFR methods [53], [54], [55], [95] cannot deal with such scenarios since the feature vectors within their methods can only be generated when certain areas (eyes, nose, mouth) of the faces are visible within the video database. Therefore, for video-based disguise face recognition (VDFR) application such as looking for lost persons in train station or recognizing the terrorists in airport, the existed VFR methods are not suitable. To address the above problem, the proposed ECS method has been extended to accomplish the VDFR tasks using the dynamic facial movements. Unlike the traditional VFR methods, the proposed methodology can still work well when certain areas of the face is not visible. Our experiments on the proposed video disguise face database (VD Face DB) demonstrated the proposed VDFR methods are reliable and efficient.

Even though the proposed methods achieved satisfactory performances, they still have several limitations. Firstly, the HMAX model used to generate the high level features has not been modeled by SNN. Ideally, given a visual input stimulus, SNN should accomplish the whole visual pattern recognition procedure. Specifically, within the feature extracting layer, for an input image, we should use spikes instead of analog values to obtain the high level features. In other words, within an ideal SNN, the input image should be transformed to spikes and then these spikes will be used to accomplish

the visual pattern recognition task.

Moreover, the high level features obtained from HMAX model is still not perfect for further visual processing since they still convey abundant intra-class variance. To further reduce the intra-class variance, a more complicated feature extracting method is much needed. Such feature extracting method should consider incorporating more biological properties of the ventral stream, such as attention mechanism or sparsity. Specifically, attention mechanisms in neural network are loosely based on the visual attention mechanism found in humans. Through attention mechanism, the neural network will only focus on the region of interested and ignore other areas. Even though the structure of the ventral stream is extremely complex and the number of neurons involved is often huge, only a few of them are actually activated within a period. Such sparsity plays an important role in visual information processing within the ventral stream.

Furthermore, in this thesis, we only use rank order coding (ROC) scheme to transform the features into spike patterns. Within ROC, a neuron is only allowed to fire at most one spike. This is certainly an ideal situation. In reality, as long as their membrane potentials reach the threshold, neurons can fire spikes as many as possible. Thus, to generate more than one spike per synapse connection, a more comprehensive spiking encoding scheme is needed.

Finally, we only use feed-forward SNN framework in this thesis. However, it is known that the human brain is a recurrent neural network (RNN): a network of neurons with feedback connections. Specifically, RNN is a class of artificial neural network where connections between units form a directed cycle. This creates an internal state of the network which allows it to exhibit dynamic temporal behavior. Unlike feed-forward neural networks, RNN can use their internal memory to process arbitrary sequences of inputs. This makes them applicable to tasks such as unsegmented connected handwriting recognition or speech recognition. Therefore, a long short-term memory (LSTM), a deep learning RNN, will be investigated in our future work.

Bibliography

[1] W. Gerstner and W. M. Kistler, *Spiking neural models*. Cambridge, MA: Cambridge University Press, 2002.

[2] S. Thorpe, D. Fize, and C. Marlot, "Speed of processing in the human visual system," *Nature*, vol. 381, no. 6582, pp. 520–522, 1996.

[3] L. G. Ungerleider and M. Mishkin, *Analysis of Visual Behavior*. Cambridge, MA: MIT Press, 1982.

[4] M. A. Goodale and A. D. Milner, "Separate visual pathways for perception and action," *Trends in Neurosciences*, vol. 15, pp. 20–25, Jan. 1992.

[5] www.wired.com. Accessed: 2016-06-28.

[6] T. Poggio and E. Bizzi, "Generalization in vision and motor control," *Nature*, vol. 431, pp. 768–774, Oct. 2004.

[7] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning." Book in preparation for MIT Press, 2016.

[8] L. Deng and D. Yu, "Deep Learning: Methods and Applications," *Microsoft Research*, May 2014.

[9] Y. Bengio, "Learning deep architectures for ai," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[10] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[11] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[13] I. Arel, D. C. Rose, and T. P. Karnowski, "Deep Machine Learning - A New Frontier in Artificial Intelligence Research," *IEEE Computational Intelligence Magazine*, vol. 5, pp. 13–18, Nov. 2010.

[14] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, pp. 541–551, Dec. 1989.

[15] G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal, "The "wake-sleep" algorithm for unsupervised neural networks," *Science*, vol. 268, pp. 1158–1161, May 1995.

[16] K. Fukushima, "Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[17] T. Masquelier and S. J. Thorpe, "Unsupervised learning of visual features through spike timing dependent plasticity," *PLoS Computational Biology*, vol. 3, no. 2, p. e31, 2007.

[18] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, pp. 1527–1554, May 2006.

[19] J. Schmidhuber, "Learning Complex, Extended Sequences Using the Principle of History Compression," *Neural Comput.*, vol. 4, pp. 234–242, Mar. 1992.

[20] https://deepmind.com/. Accessed: 2016-06-30.

[21] http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/. Accessed: 2014-03-18.

[22] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, Nov. 1999.

[23] A. Delorme, J. Gautrais, R. van Rullen, and S. Thorpe, "SpikeNET: A simulator for modeling large networks of integrate and fire neurons," *Neurocomputing*, vol. 2627, pp. 989–996, June 1999.

[24] A. Delorme and S. Thorpe, "Face identification using one spike per neuron: resistance to image degradation," *Neural Networks*, vol. 14, no. 6-7, pp. 795–803, 2001.

[25] M. Beyeler, N. D. Dutt, and J. L. Krichmar, "Categorization and decision-making in a neurobiologically plausible spiking network using a STDP-like learning rule," *Neural Networks*, vol. 48, pp. 109–124, Dec. 2013.

[26] D. Querlioz, O. Bichler, P. Dollfus, and C. Gamrat, "Immunity to Device Variations in a Spiking Neural Network With Memristive Nanodevices," *IEEE Transactions on Nanotechnology*, vol. 12, pp. 288–295, May 2013.

[27] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience*, p. 99, 2015.

[28] E. R. Kandel, J. H. Schwartz, and T. M. Jessell, *Principles of neural science*. Elsevier, 1991.

[29] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains," *PLoS ONE*, vol. 3, no. 1, p. e1377, 2008.

[30] S. Thorpe, D. Fize, and C. Marlot, "Speed of processing in the human visual system," *Nature*, vol. 381, pp. 520–522, 1996.

[31] T. W. Kjaer, J. A. Hertz, and B. J. Richmond, "Decoding cortical neuronal signals: network models, information estimation and spatial tuning," *Journal of Computational Neuroscience*, vol. 1, pp. 109–139, June 1994.

[32] M. J. Tovee and E. T. Rolls, "Information encoding in short firing rate epochs by single neurons in the primate temporal visual cortex," *Visual Cognition*, vol. 2, pp. 35–58, Mar. 1995.

[33] E. D. Young and M. B. Sachs, "Representation of steady-state vowels in the temporal aspects of the discharge patterns of populations of auditory-nerve fibers," *The Journal of the Acoustical Society of America*, vol. 66, pp. 1381–1403, Nov. 1979.

[34] J. H. Maunsell and D. C. V. Essen, "Functional properties of neurons in middle temporal visual area of the macaque monkey. I. Selectivity for stimulus direction, speed, and orientation," *Journal of Neurophysiology*, vol. 49, pp. 1127–1147, May 1983.

[35] N. Gupta and M. Stopfer, "A temporal channel for information in sparse sensory coding," *Current biology: CB*, vol. 24, pp. 2247–2256, Oct. 2014.

[36] B. Segee, "Methods in Neuronal Modeling: from Ions to Networks, 2nd Edition," *Computing in Science Engineering*, vol. 1, p. 81, Jan. 1999.

[37] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of Physiology*, vol. 117, pp. 500–544, Aug. 1952.

[38] A. L. Hodgkin, A. F. Huxley, and B. Katz, "Measurement of current-voltage relations in the membrane of the giant axon of Loligo," *The Journal of Physiology*, vol. 116, pp. 424–448, Apr. 1952.

[39] S. Ostojic, N. Brunel, and V. Hakim, "How Connectivity, Background Activity, and Synaptic Properties Shape the Cross-Correlation between Spike Trains," *The Journal of Neuroscience*, vol. 29, pp. 10234–10253, Aug. 2009.

[40] D. O. Hebb, *The Organization of Behavior: a neuropsychological theory*. New York: Wiley, 1949.

[41] W. Gerstner, R. Kempter, J. L. van Hemmen, and H. Wagner, "A neuronal learning rule for sub-millisecond temporal coding," *Nature*, vol. 386, pp. 76–78, 1996.

[42] R. Kempter, W. Gerstner, and J. L. van Hemmen, "Hebbian learning and spiking neurons," *Phys. Rev. E*, vol. 59, pp. 4498–4514, 1999.

[43] H. Markram, J. Lubke, M. Frotscher, and B. Sakmann, "Regulation of synaptic efficacy by coincidence of postsynaptic aps and epsps," *Science*, vol. 275, pp. 213–5, 1997.

[44] G. Q. Bi and M. M. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type," *J Neurosci*, vol. 18, pp. 10464–10472, 1998.

[45] P. J. Sjstrm, G. G. Turrigiano, and S. B. Nelson, "Rate, timing, and cooperativity jointly determine cortical synaptic plasticity," *Neuron*, vol. 32, pp. 1149–1164, 2001.

[46] R. VanRullen and S. J. Thorpe, "Surfing a spike wave down the ventral stream," *Vision Research*, vol. 42, pp. 2593–2615, Oct. 2002.

[47] S. G. Wysoski, L. Benuskova, and N. Kasabov, "On-Line Learning with Structural Adaptation in a Network of Spiking Neurons for Visual Pattern Recognition," in *Artificial Neural Networks ICANN 2006*, no. 4131 in Lecture Notes in Computer Science, pp. 61–70, Springer Berlin Heidelberg, Sept. 2006.

[48] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Fast and adaptive network of spiking neurons for multi-view visual pattern recognition," *Neurocomputing*, vol. 71, pp. 2563–2575, Aug. 2008.

[49] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Evolving spiking neural networks for audiovisual information processing," *Neural Networks*, vol. 23, pp. 819–835, Sept. 2010.

[50] S.-I. Amari and N. Kasabov, eds., *Brain-like Computing and Intelligent Information Systems*. Springer-Verlag Singapore Pte. Limited, 1st ed., 1998.

[51] L. C. Jain, U. Halici, I. Hayashi, S. B. Lee, and S. Tsutsui, *Intelligent Biometric Techniques in Fingerprint and Face Recognition*. CRC Press, June 1999.

[52] http://www.fujitsu.com/global/about/businesspolicy/tech/k/. Accessed: 2016-12-27.

[53] J. Yu, *Super-resolution and Facial Expression for Face Recognition in Video*. PhD thesis, University of California, Riverside, USA, 2007. AAI3298271.

[54] Z. Huang, S. Shan, R. Wang, H. Zhang, S. Lao, A. Kuerban, and X. Chen, "A Benchmark and Comparative Study of Video-Based Face Recognition on COX Face Database," *IEEE Transactions on Image Processing*, vol. 24, pp. 5967–5981, Dec. 2015.

[55] M. Gavrilescu, "Study on using individual differences in facial expressions for a face recognition system immune to spoofing attacks," *IET Biometrics*, vol. 5, no. 3, pp. 236–242, 2016.

[56] P. Simon, "Too big to ignore: The Business Case for Big Data," p. 89, Wiley, Oct. 2014.

[57] I. A. Basheer and M. Hajmeer, "Artificial neural networks: fundamentals, computing, design, and application," *Journal of Microbiological Methods*, vol. 43, pp. 3–31, Dec. 2000.

[58] C. Gallo, "Artificial neural networks: tutorial," *Encyclopedia of Information Science and Technology, 3rd Ed.(10 Volumes), Mehdi Khosrow-Pour Ed. Hershey, PA: IGI Global. doi*, vol. 10, pp. 978–1, 2014.

[59] A. G. Ivakhnenko and V. G. Lapa, "CYBERNETIC PREDICTING DEVICES," *ResearchGate*, p. 250, Apr. 1966.

[60] A. G. Ivakhnenko, "Polynomial Theory of Complex Systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-1, pp. 364–378, Oct. 1971.

[61] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *ICASSP*, pp. 8609–8613, 2013.

[62] J. J. Weng, N. Ahuja, and T. S. Huang, "Learning recognition and segmentation of 3-D objects from 2-D images," in , *Fourth International Conference on Computer Vision, 1993. Proceedings*, pp. 121–128, May 1993.

[63] J. Schmidhuber, "Deep Learning," *Scholarpedia*, vol. 10, no. 11, p. 32832, 2015.

[64] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for boltzmann machines," *Cognitive Science*, vol. 9, pp. 147–169, Jan. 1985.

[65] R. Salakhutdinov and G. E. Hinton, "Deep boltzmann machines.," in *AISTATS*, vol. 1, p. 3, 2009.

[66] K. Dockendorf and N. Srinivasa, "Learning and prospective recall of noisy spike pattern episodes," *Frontiers in Computational Neuroscience*, vol. 7, p. 80, 2013.

[67] B. A. Kaplan, A. Lansner, G. S. Masson, and L. U. Perrinet, "Anisotropic connectivity implements motion-based prediction in a spiking neural network," *Frontiers in Computational Neuroscience*, vol. 7, p. 112, 2013.

[68] Q. Yu, H. Tang, K. Tan, and H. Li, "Rapid feedforward computation by temporal encoding and learning with spiking neurons," *IEEE Trans. Neural Networks and Learning Systems*, vol. 24, no. 10, pp. 1539–1552, 2013.

[69] B. Zhao, R. Ding, S. Chen, B. Linares-Barranco, and H. Tang, "Feedforward Categorization on AER Motion Events Using Cortex-Like Features in a Spiking Neural Network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, pp. 1963–1978, Sept. 2015.

[70] S. Hussain, S. C. Liu, and A. Basu, "Improved margin multi-class classification using dendritic neurons with morphological learning," in *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 2640–2643, June 2014.

[71] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. S. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm," in *2011 IEEE Custom Integrated Circuits Conference (CICC)*, pp. 1–4, Sept. 2011.

[72] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Neuromorphic Engineering*, vol. 7, p. 178, 2013.

[73] P. U. Diehl, D. Neil, J. Binas, M. Cook, S. C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, July 2015.

[74] E. Neftci, S. Das, B. Pedroni, K. Kreutz-Delgado, and G. Cauwenberghs, "Event-driven contrastive divergence for spiking neuromorphic systems," *Neuromorphic Engineering*, vol. 7, p. 272, 2014.

[75] J. M. Brader, W. Senn, and S. Fusi, "Learning Real-World Stimuli in a Neural Network with Spike-Driven Synaptic Dynamics," *Neural Computation*, vol. 19, pp. 2881–2912, Sept. 2007.

[76] L. F. Abbott, "Lapicques introduction of the integrate-and-fire model neuron (1907)," *Brain Research Bulletin*, vol. 50, pp. 303–304, Nov. 1999.

[77] A. L. Hodgkin and A. F. Huxley, "Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo," *The Journal of Physiology*, vol. 116, pp. 449–472, Apr. 1952.

[78] A. L. Hodgkin and A. F. Huxley, "The components of membrane conductance in the giant axon of Loligo," *The Journal of Physiology*, vol. 116, pp. 473–496, Apr. 1952.

[79] A. Galves and E. Lcherbach, "Infinite Systems of Interacting Chains with Memory of Variable LengthA Stochastic Model for Biological Neural Nets," *Journal of Statistical Physics*, vol. 151, pp. 896–921, Mar. 2013.

[80] E. Izhikevich and R. FitzHugh, "FitzHugh-Nagumo model," *Scholarpedia*, vol. 1, no. 9, p. 1349, 2006.

[81] L. M. Optican and B. J. Richmond, "Temporal encoding of two-dimensional patterns by single units in primate inferior temporal cortex. III. Information theoretic analysis," *Journal of Neurophysiology*, vol. 57, pp. 162–178, Jan. 1987.

[82] M. J. Tove, E. T. Rolls, A. Treves, and R. P. Bellis, "Information encoding and the responses of single neurons in the primate temporal visual cortex," *Journal of Neurophysiology*, vol. 70, pp. 640–654, Aug. 1993.

[83] J. J. Hopfield, "Pattern recognition computation using action potential timing for stimulus representation," *Nature*, vol. 376, pp. 33–36, July 1995.

[84] O. Jensen and J. E. Lisman, "Hippocampal CA3 region predicts memory sequences: accounting for the phase precession of place cells," *Learning & Memory (Cold Spring Harbor, N.Y.)*, vol. 3, pp. 279–287, Oct. 1996.

[85] W. Maass, "Lower Bounds for the Computational Power of Networks of Spiking Neurons," *Neural Computation*, vol. 8, pp. 1–40, Jan. 1996.

[86] C. von der Malsburg and J. Buhmann, "Sensory segmentation with coupled neural oscillators," *Biological Cybernetics*, vol. 67, no. 3, pp. 233–242, 1992.

[87] C. M. Gray and W. Singer, "Stimulus-specific neuronal oscillations in orientation columns of cat visual cortex," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 86, pp. 1698–1702, Mar. 1989.

[88] S. Wu, S.-i. Amari, and H. Nakahara, "Population Coding and Decoding in a Neural Field: A Computational Study," *Neural Computation*, vol. 14, pp. 999–1026, May 2002.

[89] E. L. Bienenstock, L. N. Cooper, and P. W. Munro, "Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual

cortex," *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, vol. 2, pp. 32–48, Jan. 1982.

[90] G. Bi and M. Poo, "Synaptic modification by correlated activity: Hebb's postulate revisited," *Annual Review of Neuroscience*, vol. 24, pp. 139–166, 2001.

[91] P. J. Sjstrm, E. A. Rancz, A. Roth, and M. Husser, "Dendritic excitability and synaptic plasticity," *Physiological Reviews*, vol. 88, pp. 769–840, Apr. 2008.

[92] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1," ch. Learning Internal Representations by Error Propagation, pp. 318–362, Cambridge, MA, USA: MIT Press, 1986.

[93] D. Neil and S. C. Liu, "Minitaur, an Event-Driven FPGA-Based Spiking Network Accelerator," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, pp. 2621–2628, Dec. 2014.

[94] S. Habenschuss, J. Bill, and B. Nessler, "Homeostatic plasticity in Bayesian spiking networks as Expectation Maximization with posterior constraints," in *Advances in Neural Information Processing Systems 25*, pp. 773–781, Curran Associates, Inc., 2012.

[95] Z. Huang, R. Wang, S. Shan, and X. Chen, "Learning Euclidean-to-Riemannian Metric for Point-to-Set Classification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1677–1684, June 2014.

[96] S. Yue and F. Rind, "Collision detection in complex dynamic scenes using an LGMD-based visual neural network with feature enhancement," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 705–716, 2006.

[97] S. Yue and F. Rind, "Redundant Neural Vision Systems -Competing for Collision Recognition Roles," *IEEE Transactions on Autonomous Mental Development*, vol. 5, no. 2, pp. 173–186, 2013.

[98] A. Delorme, J. Gautrais, R. van Rullen, and S. Thorpe, "Spikenet: a simulator for modeling large large networks of integrate and fire neurons," *Neurocomputing*, vol. 26, pp. 989–996, 1999.

[99] A. Delorme, L. Perrinet, and S. Thorpe, "Networks of integrate-and-fire neurons using rank order coding," *Neurocomputing*, vol. 38-40, pp. 539–545, 2001.

[100] S. Marcelja, "Mathematical description of the responses of simple cortical cells," *Journal of the Optical Society of America*, vol. 70, pp. 1297–1300, Nov. 1980.

[101] J. G. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," *Journal of the Optical Society of America. A, Optics and Image Science*, vol. 2, pp. 1160–1169, July 1985.

[102] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html. Accessed: 2013-04-25.

[103] F. Rieke, R. Warland, D.de Ruyter van Steveninck, and W. Bialek, *Spikes:Exploring the neural code*. Cambridge, MA: MIT Press, 1996.

[104] M. N. Shadlen and W. T. Newsome, "Noise, neural codes and cortical organization," *Curr. Opin. Neurobial.*, vol. 4, pp. 569–579, 1994.

[105] K. Kitano, H. Cteau, and T. Fukai, "Sustained activity with low firing rate in a recurrent network regulated by spike-timing-dependent plasticity," *Neurocomputing*, vol. 4446, pp. 473–478, June 2002.

[106] F. Henry, E. Dauc, and H. Soula, "Temporal pattern identification using spike-timing dependent plasticity," *Neurocomputing*, vol. 70, pp. 2009–2016, June 2007.

[107] A. Shahim-Aeen and G. Karimi, "Triplet-based spike timing dependent plasticity (TSTDP) modeling using VHDL-AMS," *Neurocomputing*, vol. 149, Part C, pp. 1440–1444, Feb. 2015.

[108] S. Thomas, W. Lior, B. Stanley, R. Maximilian, and P. Tomaso, "Robust object recognition with cortex-like mechanisms," *IEEE Trans. Pattern Anal. Mach. Intelli.*, vol. 29, no. 3, pp. 411–426, 2007.

[109] T. Serre, M. Kouh, C. Cadieu, U. Knoblich, G. Kreiman, and T. Poggio, "A theory of object recognition: Computations and circuits in the feedforward path of the ventral stream in primate visual cortex," 2005.

[110] T. Christian, T. Nicolas, and C. Matthieu, "Extended coding and pooling in the hmax model," *IEEE Trans. Image Processing*, vol. 22, no. 2, pp. 764–777, 2013.

[111] M. C. W. van Rossum, G. Q. Bi, and G. G. Turrigiano, "Stable hebbian learning from spike time dependent plasticity," *Journal of Neuroscience*, vol. 20, no. 88, pp. 12–21, 2000.

[112] J. Rubin, R. Gerkin, G. Bi, and C. Chow, "Calcium time course as a signal for spike-timing-dependent plasticity," *J Neurophysiol*, vol. 93, pp. 2600–13, 2005.

[113] http://yann.lecun.com/exdb/mnist/. Accessed: 2013-09-30.

[114] R. Guyonneau, R. VanRullen, and S. J. Thorpe, "Neurons tune to the earliest spikes through STDP," *Neural Computation*, vol. 17, pp. 859–879, Apr. 2005.

[115] R. Guyonneau, R. Vanrullen, and S. J. Thorpe, "Temporal codes and sparse representations: a key to understanding rapid processing in the visual system," *Journal of Physiology, Paris*, vol. 98, pp. 487–497, Nov. 2004.

[116] W. M. Kistler and J. L. van Hemmen, "Modeling synaptic plasticity in conjuction with the timing of pre- and postsynaptic action potentials," *Neural Computation*, vol. 12, pp. 385–405, Feb. 2000.

[117] T. Serre, G. Kreiman, M. Kouh, C. Cadieu, U. Knoblich, and T. Poggio, "A quantitative theory of immediate visual recognition," *Progress in Brain Research*, vol. 165, pp. 33–56, 2007.

[118] T. Rumbell, S. Denham, and T. Wennekers, "A Spiking Self-Organizing Map Combining STDP, Oscillations, and Continuous Learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 894–907, 2014.

[119] P. S. Swain and A. Longtin, "Noise in genetic and neural networks," *Chaos (Woodbury, N.Y.)*, vol. 16, p. 026101, June 2006.

[120] G. Deco, V. K. Jirsa, and A. R. McIntosh, "Emerging concepts for the dynamical organization of resting-state activity in the brain," *Nature Reviews. Neuroscience*, vol. 12, pp. 43–56, Jan. 2011.

[121] D. . T. P. Marr, "From understanding computation to understanding neural circuitry," *Neurosciences Res. Prog. Bull.*, vol. 15, pp. 470–488, 1977.

[122] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of Physiology*, vol. 148, pp. 574–591, Oct. 1959.

[123] M. Figueiredo, "Adaptive sparseness for supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1150–1159, Sept. 2003.

[124] B. Krishnapuram, L. Carin, M. Figueiredo, and A. Hartemink, "Sparse multinomial logistic regression: fast algorithms and generalization bounds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 957–968, June 2005.

[125] S. Agarwal, A. Awan, and D. Roth, "Learning to detect objects in images via a sparse, part-based representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, pp. 1475–1490, Nov. 2004.

[126] A. Berg, T. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, vol. 1, pp. 26–33, 2005.

[127] J. Mutch and D. G. Lowe, "Object Class Recognition and Localization Using Sparse Features with Limited Receptive Fields," *International Journal of Computer Vision*, vol. 80, pp. 45–57, Jan. 2008.

[128] J. Mutch, U. Knoblich, and T. Poggio, "CNS: a GPU-based framework for simulating cortically-organized networks," Tech. Rep. MIT-CSAIL-TR-2010-013 / CBCL-286, Massachusetts Institute of Technology, Cambridge, MA, February 2010.

[129] S. Song, K. D. Miller, and L. F. Abbott, "Competitive Hebbian learning through spike-timing-dependent synaptic plasticity," *Nature Neuroscience*, vol. 3, pp. 919–926, Sept. 2000.

[130] S. Song and L. F. Abbott, "Cortical development and remapping through spike timing-dependent plasticity," *Neuron*, vol. 32, pp. 339–350, Oct. 2001.

[131] V.-J. D. Daley, D.J, *An Introduction to the Theory of Point Processes*. New York: Springer, 1988.

[132] W. S. K. D. Stoyan and J. Mecke., *Stochastic geometry and its applications*, vol. 2. Wiley, 1988.

[133] R. Singh, M. Vatsa, and A. Noore, "Face recognition with disguise and single gallery images," *Image and Vision Computing*, vol. 27, pp. 245–257, Feb. 2009.

[134] T. I. Dhamecha, A. Nigam, R. Singh, and M. Vatsa, "Disguise detection and face recognition in visible and thermal spectrums," in *2013 International Conference on Biometrics (ICB)*, pp. 1–8, June 2013.

[135] T. I. Dhamecha, R. Singh, M. Vatsa, and A. Kumar, "Recognizing Disguised Faces: Human and Machine Evaluation," *PLOS ONE*, vol. 9, p. e99212, July 2014.

[136] J. F. Cohn, K. Schmidt, R. Gross, and P. Ekman, "Individual differences in facial expression: stability over time, relation to self-reported emotion, and ability to inform person identification," in *Fourth IEEE International Conference on Multimodal Interfaces, 2002. Proceedings*, pp. 491–496, 2002.