11-7-2016

A Biologically Plausible Supervised Learning Method for Spiking Neurons with Real-world Applications

Lilin Guo
Florida International University, lguo002@fiu.edu

DOI: 10.25148/etd.FIDC001243

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

A BIOLOGICALLY PLAUSIBLE SUPERVISED LEARNING METHOD FOR SPIKING NEURONS WITH REAL-WORLD APPLICATIONS

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Lilin Guo

2016

To: Interim Dean Ranu Jung
College of Engineering and Computing

This dissertation, written by Lilin Guo, and entitled A Biologically Plausible Supervised Learning Method for Spiking Neurons with Real-world Applications, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Armando Barreto

Nezih Pala

Mercedes Cabrerizo

Wei-Chiang Lin

Malek Adjouadi, Major Professor

Date of Defense: November 7, 2016

The dissertation of Lilin Guo is approved.

Interim Dean Ranu Jung
College of Engineering and Computing

Andrés G. Gil
Vice President for Research and Economic Development
and Dean of the University Graduate School

Florida International University, 2016

DEDICATION

I dedicate this dissertation to my supportive husband, Tan Ma, to my lovely daughter, Esther Ma, for making this work meaningful. I also dedicate this work to my beloved parents. Without their patience, understanding, support, and love, the completion of this endeavor would never have been possible.

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my advisor, Dr. Malek Adjouadi, for his excellent guidance, caring, patience, and invaluable insights throughout all the stages of researching and writing this dissertation, as well as for the financial support provided to me for the completion of this dissertation. He provided unserved support during my PhD, patiently corrected my writing and generously paved the way for my development as a researcher. His passion and dedication are incomparable and always inspire me.

Next, I wish to express my great appreciation to my committee members, Dr. Armando Barreto, Dr. Nezih Pala, Dr. Mercedes Cabrerizo and Dr. Wei-Chiang Lin, for their valuable discussions improving the quality of this dissertation.

Furthermore, I would like to acknowledge the research support provided from the Department of Electrical and Computer Engineering at Florida International University, the support of National Science Foundation under grant Numbers: CNS-1532061, CNS-1642193, CNS-0959985, CNS-1042341, HRD-0833093, IIP 1338922, IIP-1230661, and CNS-1429345, and the generous support of the Ware Foundation, which facilitated the research essential to the completion of this dissertation.

I would also like to extent my gratitude to all the lab mates and members at the Center for Advanced Technology and Education for creating an amazing working environment, especially, Zhenzhong Wang, whose encouragement greatly helped me in achieving my research goals; to the FIU community and department staffs, especially Pat Brammer and Oscar Silveira, who have given me great help and support.

ABSTRACT OF THE DISSERTATION

A BIOLOGICALLY PLAUSIBLE SUPERVISED LEARNING METHOD FOR SPIKING NEURONS WITH REAL-WORLD APPLICATIONS

by

Lilin Guo

Florida International University, 2016

Miami, Florida

Professor Malek Adjouadi, Major Professor

Learning is central to infusing intelligence to any biologically inspired system. This study introduces a novel Cross-Correlated Delay Shift (CCDS) learning method for spiking neurons with the ability to learn and reproduce arbitrary spike patterns in a supervised fashion with applicability to spatiotemporal information encoded at the precise timing of spikes. By integrating the cross-correlated term, axonal and synapse delays, the CCDS rule is proven to be both biologically plausible and computationally efficient. The proposed learning algorithm is evaluated in terms of reliability, adaptive learning performance, generality to different neuron models, learning in the presence of noise, effects of its learning parameters and classification performance. The results indicate that the proposed CCDS learning rule greatly improves classification accuracy when compared to the standards reached with the Spike Pattern Association Neuron (SPAN) learning rule and the Tempotron learning rule.

Network structure is the crucial part for any application domain of Artificial Spiking Neural Network (ASNN). Thus, temporal learning rules in multilayer spiking neural networks are investigated. As extensions of single-layer learning rules, the multilayer CCDS (MutCCDS) is also developed. Correlated neurons are connected through fine-tuned weights and delays. In contrast to the multilayer Remote Supervised Method (MutReSuMe) and multilayer tempotron rule (MutTmptr), the newly developed MutCCDS shows better generalization ability and faster convergence. The proposed multilayer rules provide an efficient and biologically plausible mechanism, describing how delays and synapses in the multilayer networks are adjusted to facilitate learning.

Interictal spikes (IS) are morphologically defined brief events observed in electroencephalography (EEG) records from patients with epilepsy. The detection of IS remains an essential task for 3D source localization as well as in developing algorithms for seizure prediction and guided therapy. In this work, we present a new IS detection method using the Wavelet Encoding Device (WED) method together with CCDS learning rule and a specially designed Spiking Neural Network (SNN) structure. The results confirm the ability of such SNN to achieve good performance for automatically detecting such events from multichannel EEG records.

TABLE OF CONTENTS

LIST OF TABLES

FIGURE PAGE

LIST OF FIGURES

ABBREVIATIONS AND ACRONYMS

AER	Address-Event Representation
ANN	Artificial Neural Network
ASNN	Artificial Spiking Neural Network
BCM	Bienenstock-Cooper-Munro
BSA	Bens Spiker Algorithm
BP	Back Propagation
CCDS	Cross-Correlated Delay Shift
EEG	Electroencephalography
ESN	Echo State Network
FPGA	Field-Programmable Gate Array
GPGPU	General Purpose Graphic Process Unit
GRF	Gaussian Receptive Field
HH	Hodgkin and Huxley
HSA	Hough Spiker Algorithm
IB	Intrinsically Bursting
ICA	Independent Component Analysis
IF	Integrate-and-Fire
LIF	Leaky Integrate-and-Fire
LOOCV	Leave-One-Out-Cross-Validation
LSM	Liquid State Machine
LTD	Long-Term Depression

LTP	Long-Term Potentiation
MAT	Multiple-timescale Adaptive Threshold
MuSpiNN	Multi-Spiking Neural Network
MutCCDS	Multilayer Cross-Correlated Delay Shift
MutReSuMe	Multilayer Remote Supervised Method
MutTmptr	Multilayer Tempotron
NEST	NEural Simulation Tool
ODE	Ordinary Differential Equation
PE	Phase Encoding
PBSNLR	Perceptron-Based Spiking Neuron Learning Rule
PSC	Post-Synaptic Current
PSD	Precision-Spike-Driven
PSP	Postsynaptic Potential
RBF	Radial Basis Function
ReSuMe	Remote Supervised Method
ROC	Rank-Order Coding
RS	Regular Spiking
SNN	Spiking Neural Network
SPAN	Spike Pattern Association Neuron
SRM	Spike Response Model
STD	Short-Term Depression
STDP	Spike Timing Dependent Plasticity
STP	Short-Term Potentiation

SWAT	Synaptic Weight Association Training
VLSI	Very-Large-Scale Integration
WBC	Wisconsin Breast Cancer
WED	Wavelet Encoding Device

1. INTRODUCTION

1.1 Background

Many artificial/ biological systems begin with a small set of abilities, and develop new abilities through learning and other types of adaptation as time goes on. Artificial Neural Networks (ANNs) is an important class of machining learning methods inspired by the features of biological neurons and nervous systems. The research on ANNs has achieved a great deal in both theories and engineering applications.

Dubbed as the third generation of ANN, Spiking Neural Networks (SNNs) [1], [2] are considered to be more biologically realistic as compared to the typical rate-coded networks since they can model more closely different types of neurons and their related temporal dynamics. Neurons will send out short pulses of energy (spikes) as signals if they have received enough input from other neurons. Based on this mechanism, spiking neurons are developed with the same capability of processing spikes as would be expected from biological neurons. Such biologically plausible construction ensures SNNs a greater processing power in manipulating temporal signals, and better robustness in complex patterns learning. Most of the attention of SNNs has been focus on: 1) the design of spiking neuron model [3], [4]; 2) information encoding method [5]–[7]; 3) training algorithm [8]–[10]; 4) applications [11]–[13] and 5) hardware implementation [14]–[16].

The learning of ANNs indicates the rules to change the synapse weights, and consequently modifies the firing pattern of output neurons. The time dependent features of SNNs learning and recall procedure could naturally satisfy the requirements of the temporal biological data processing and could also provide greater computing power than would be expected from conventional ANNs learning using smaller networks sizes. SNNs have been extensively studied in recent years, but questions of how information is represented by spikes and how the neurons process these spikes remain unclear.

SNNs exhibit interesting properties that make them particularly suitable for applications that require fast and efficient computation and where the timing of input/output signals carries important information. Firstly, SNNs are not only adaptive, but also computationally powerful. Second, the representation of signals transmitted through and produced by SNNs resembles those required to stimulate the nerves. Moreover, SNNs possess the ability to learn from examples and are expected to inherit generalization properties, common to most classes of ANN. SNNs have three main functional components: encoding, learning and decoding. The choices in terms of encoding method and learning method depend on the application. Strong evidence suggests that supervised learning occurs in the cerebellum and the cerebellar cortex [17]. In order to take advantage of the properties of SNNs, an efficient learning algorithm is desired.

Applications of ANNs have gained increasing interest over the past two decades. Various tasks has been done successfully, such as pattern and sequence recognition [18]; data processing [19]; decision making [20]; system identification and control [21]; medical diagnoses [13]; image/object recognition [22]; autonomous robotics [23]; optical character/handwriting recognition [22]; real-time embedded control [24], and so on. As

SNN get closer to biological examples, it become possible to emulate part of function of the biological nervous system to process the information which normally happen in the brain, but still not clear understood. For instance, utilize SNN into the study of processes such as: the study of mental illnesses by emulating brain disorders and testing how drugs affect the brain. When the SNNs is used to analyze brain signals, supervised learning is required for the SNNs to work as a classifier or a pattern recognizer. The final goal of studying involving SNN is to understand how the human brain works. However, the limitations in the existing encoding method and learning rules have thus far restricted the applications of SNNs. Therefore, improvements on these aspects as proposed here could broaden the practical implications of SNNs-based systems and their consequential application in signal processing.

1.2 Research Purpose

This dissertation focuses on developing a biologically plausible information processing system using SNNs, and its applications on biomedical signals. The specific goal of this research is fivefold:

1. Developing an integrated consistent system of spiking neurons to perform various recognition tasks, where the encoding, the learning, and the readout are considered from a systematic level.

2. Developing a new temporal learning algorithm that is both computational efficient and also biologically plausible.

3. Investigating various properties of the proposed learning algorithm, such as learning in the presence of noise, generality to different neuron model, adaptive learning performance, effects of parameters, etc.

4. Investigating the temporal learning in multilayer spiking neural networks.

5. Investigating the ability of the proposed learning rule and neural network structure for different cognitive tasks, such as interictal spike detection, Iris classification and Wisconsin Breast Cancer classification problem, etc.

1.3 Methodology

In this work, we applied analytical and computational methods for the following tasks:

1. Our approach is to present a systemic understanding of how pattern encodings might happen in the nervous system and a learning displays technical capacity. An analytical method was used to develop what is termed here as a Cross-Correlated Delay Shift (CCDS) learning rule based on the design principles of the ReSuMe rule. Axonal delays and synapse delays were integrated to the CCDS rule as an extension of the ReSuMe learning rule. The delay shift method adapts the actual delay values of the connections between neurons during training. Input spike patterns close to the synaptic vector will make the neuron emit an output spike. The proposed learning method is a heuristic method, which helps the neuron generate output spikes at desired instances and also tries to remove undesired output spikes. During learning, the output neuron postsynaptic potential (PSP) is increased at desired instances to hit the firing threshold

level and produce the desired output. In addition, the total PSP is reduced below the firing threshold at the instances of undesired output spikes to remove the undesired spikes. Meanwhile, the cross-correlated term is introduced to speed up the learning process. The CCDS learning rule was tested with random input temporal spike patterns, and the output accuracy and computation efficiency is compared to those of the ReSuMe learning rule as stimulated by the same input.

2. We expanded the structure of weight matrix, axonal delays and synapse delays with the idea that combining those variables and mapping them into a single higher dimensional matrix. The proof of concept was done by performing the gradient descent on such special structured spiking neural network and minimizing network error function similar to back-propagation of ANN. It combines the quality of SpikeProp with the flexibility and efficiency of CCDS, i.e., it can be used with multi spikes and different neuron model in multiple layers in the efficient training process. It improves the capability of CCDS on classification of nonlinear problems when networks without hidden layers cannot perform nonlinear operations.

3. We extracted features from EEG records which could help seizure detection by simulating a specially designed SNN structure with customized neuron models. The SNNs with several input neurons and one output neuron was implemented to analyze EEG data with interictal spikes. Each channel of the EEG signals was encoded into spike trains using recently proposed Wavelet Encoding Device (WED) [7] encoding method and fed to the corresponding input neurons. After training, similar spikes will

be generated at the output neuron when epileptic seizures occur. An assessment was also be conducted on the interictal spikes detection by the template matching algorithm [25], the feature extraction method using Walsh transform [26], the ReSuMe rule [27] and the proposed learning method for validation purposes. The entire simulation was done in the Python interface of the SNN simulation platform together with Matlab environment.

1.4 Original Contribution of This Dissertation

Neurons in the nervous systems transmit information through spikes (action potential). It is still unknown that how neurons with spiking features give rise to cognitive functions of the brain. This dissertation presents detailed investigation on information processing and cognitive computing in SNNs, trying to reveal how the biological systems might operate under a temporal framework. The main contribution of this study is threefold:

1. Developed a novel biologically plausible learning rule which has improved learning accuracy and learning speed in processing and memorizing spatiotemporal patterns. It reliable in its deployment in the supervised training phase. The learning rule has the following features:

   (i) It enables efficient and effect training of SNN to store and precisely reproduce spatiotemporal patterns of spikes. Competitive efficiency compared to other supervised learning rule.

(ii) It exhibits generalization properties, i.e., the proposed rule is independent on the spiking neural models and can be effectively applied to different class of spiking neurons.

(iii) Robustness against noisy conditions. The functions of delay and noise in neuron connection is well tested.

(iv) Adaptive to variant spatiotemporal patterns of spikes.

2. Applied the single layer temporal learning rule to the multi-layer network. Comparing with the multilayer Remote Supervised Method (MutReSuMe) [28] and multilayer tempotron rule (MutTmptr) [29], Multilayer CCDS (MutCCDS) shows better generalization ability and faster convergence. The proposed multilayer rules provide an efficient and biologically plausible mechanism, describing how delays and synapses in the multilayer networks are adjusted to facilitate the learning.

3. Developed an integrated consistent system of spiking neurons, and applied SNN to biomedical signal processing. Specifically in this work, we used the spike time encoding method to extract important features from electroencephalogram (EEG) records, then utilized the proposed supervised learning rule to analysis and detect the interictal spikes for patients with epilepsy. By integrating with encoding and learning function parts, it achieved a reasonably high detection accuracy, i.e., it identified 69 spikes out of 82 spikes, or 84% detection rate. Simulation results show that the applied CCDS rule outperforms ReSuMe for identifying these spikes.

1.5 Dissertation Organization

This dissertation introduces a novel learning algorithm for implementing ASNN and applying it to biomedical signal processing. The dissertation is structured into seven chapters, starting from the current chapter that outlines the research background and purpose.

Chapter 2 introduces multiple existing models of spiking neurons and synapses, and the selection criterion for a good neuron model is discussed. The SNN architecture are overviewed. Existing rate encoding and temporal encoding approaches are also reviewed in this chapter to provide a retrospective on SNN encoding schemes. Various methods of unsupervised and supervised learning in SNN are presented at the end of this chapter.

In Chapter 3, a novel temporal learning rule, named Cross-Correlated Delay Shift (CCDS) learning rule, is developed for learning association of spatiotemporal spike patterns. The formal definitions of the CCDS learning rule and network architecture for CCDS is described. Various properties and learning performance are investigated through extensive experiments. The CCDS rule is able to perform the classification task, but also can memorize patterns by firing desired spikes at precise time in a faster way than existing supervised learning methods. It is simple, efficient, yet biologically plausible.

In Chapter 4, the learning in multilayer spiking neural networks is investigated. The comparison with other multilayer learning rules, such as multilayer Tempotron and multilayer ReSuMe is given. Several tasks are used to analyze the learning performance of the multilayer network.

Chapter 5 elaborates on the learning rule and network structure, the application of the CCDS rule to the interictal spike detection in epilepsy patients from EEG records are presented. A preprocessing unit for the Leaky Integrate-and-Fire (LIF) spiking neurons is utilized to decompose the input EEG signal into the wavelet spectrum, then further encode the spectrum amplitude into the delay amount between output spikes and the clock signals. Empirical results of Phase Encoding (PE) of EEG signals are provided first. The system architecture and parameters for detection task are described in this chapter, followed by the detection results and discussions.

Chapter 6 summarizes and discusses the main results of this dissertation, and provides possible directions for the future work.

2. SPIKING NEURAL NETWORKS

Spiking neural network (SNNs), termed as the third generation of ANNs, transfer information in the form of precisely time events called spikes. Spiking neural networks are biologically-inspired networks that model the behavior of neurons in the brain. Such networks have a number of advanced properties compared to the traditional rate-based artificial neural network, the main difference coming from the information transmitted by time. The basic idea is biologically well found: the more intensive the input, the earlier the spike transmission, as in visual systems [27]. Discrete spikes and propagation delays are used to identify temporal patterns. Since the basic principle underlying SNNs is different, much of the work on traditional neural networks, such as learning rules and theoretical results of neuron models, has to be adapted, or even has to be rethought. Most popular neuron models and how to select the appropriate neuron model for a specific application are described first, then models of synapse and network architecture are summarized. This chapter also reviews the literature of spiking neurons on existing encoding methods in determining input signals for SNNs, and temporal learning method for spiking neural networks.

2.1 Models of Neurons

Neuron models are the elementary units which determine the performance of a SNN. It is usually a mathematical model or electronic unit, which characterizes the membrane potential dynamics of a neuron cell. A lot of spiking neuron models have been emergent in literature. We only summarize the most popular ones in this section.

Hodgkin-Huxley (HH) model [30]: In 1952, Hodgkin and Huxley modeled the electro-chemical information transmission of natural neurons with electrical circuits: u_m is the membrane potential, C_m is the capacitance of the membrane, g_i stand for the conductance parameter for a specific ion channel (sodium (Na), potassium (K), etc.) and E_i is the corresponding equilibrium potential. The variable m, h and n describe the opening and closing of the voltage dependent channels. The dynamic of membrane potential is governed by the following ODE:

$$C_m \frac{du_m}{dt} = I_{all}(t) - g_{Na}(m,n)(u_m - E_{Na}) - g_k(h)(u_m - E_k) - g_L(u_m - E_{Na}) . \quad (2.1)$$

Hodgkin and Huxley model was based on the well-known voltage clamp experiment on the giant axon neuron found in squid. By analyzing the dynamics of these gating parameters, the functions g_{Na} and g_{k} are fitted to polynomial functions:

$$\begin{aligned}
\frac{dm}{dt} &= \alpha_m(u_{\mathrm{m}})(1-m) - \beta_m(u_{\mathrm{m}})m \\
\frac{dn}{dt} &= \alpha_n(u_{\mathrm{m}})(1-n) - \beta_n(u_{\mathrm{m}})n \\
\frac{dh}{dt} &= \alpha_h(u_{\mathrm{m}})(1-h) - \beta_h(u_{\mathrm{m}})h \\
g_{\mathrm{Na}} &= \bar{g}_{\mathrm{Na}} m^3 h \\
g_{\mathrm{K}}(n) &= \bar{g}_{\mathrm{K}} n^4
\end{aligned} \quad (2.2)$$

where $\bar{g}_{\mathrm{Na}}$ and $\bar{g}_{\mathrm{K}}$ are the constants of maximum conductance for sodium and potassium channels. Parameters α_m, α_n, and α_h are the changing speeds of gates related to m, n, and h

from open state to close state, while β_m, β_n, and β_h are the changing speed in the opposite direction. All these changing speeds are unitless univariate functions that depend solely on u_m, with outcome ranges between zero and one. The functions fitted by Hodgkin and Huxley are:

$$\begin{aligned}
\alpha_m &= \frac{0.1(u_m+25)}{\exp((u_m+25)/10)-1} & \beta_m &= 4\exp(u_m/18) \\
\alpha_n &= \frac{0.01(u_m+10)}{\exp((u_m+10)/10)-1} & \beta_n &= 0.125\exp(u_m/80) \\
\alpha_h &= 0.07\exp(u_m/20) & \beta_h &= 1/(\exp((u_m+30)/10)+1)
\end{aligned} \tag{2.3}$$

Equations (2.1), (2.2), and (2.3) constitute the original HH model. It is by far the most detailed and complex neuron model. However, the HH model is less suitable for simulations of large networks and its applications in ASNN are still rare, due to its large computational cost.

Integrate-and-Fire (IF) model [31]: and its variants, such as Leaky integrate-and-fire (LIF) [32], quadratic integrate-and-fire (QIF) [32], exponential integrate-and-fire (EIF) [33] and generalized integrate-and-fire (gLIF) [34] are simpler than the Hodgkin-Huxley neuron model and much more computationally tractable. The most important simplification in the LIF neuron implies that the shape of the action potentials is neglected and every spike is considered as a uniform event defined only by the time of its appearance. The electrical circuit equivalent for a LIF neuron consists of a capacitor C in parallel with a resistor R driven by an input current $I(t)$. The dynamics of the membrane potential in the LIF are described by a single first-order linear differential equation:

$$C\frac{dV}{dt} = -\frac{1}{R}(V(t) - V_{rest}) + I(t)\,, \tag{2.4}$$

where V is the membrane potential, spike firing time $t^{(f)}$ is defined by $V(t^{(f)}) = \upsilon$ with $V^{'}(t^{(f)}) > 0$. When V is not differentiable, $V^{'}$ corresponds to the left derivative.

The LIF model has been further improved by introducing other biologically plausible features, such as nonlinear leakage term [35] and moving thresholds [33], [36]. The variability of thresholds in the moving threshold models equipped the LIF model with a refractory period, which is argued to be very important in the cognition process of spiking neural network (SNN) [31], [37]. The multi timescale adaptive threshold (MAT) model proposed in [4] is preeminent in its accuracy of reproducing the neuron behaviors, which won the competition of neuronal activity challenge launched by the International Neuroinformatics Coordinating Facility in 2009 [38].

Izhikevich neuron model: By applying bifurcation methodologies to the HH model, Izhikevich proposed a two dimensional model [3], [39] recently. It able to reproduce many realistic neural behaviors, like bursting or single spiking, with different parameter values in simple equations:

$$\begin{aligned} \frac{du_{\mathrm{m}}}{dt} &= a(bv - u_{\mathrm{m}}) \\ \frac{dv}{dt} &= 0.04v^2 + 5v + 140 - u + I_{all} \end{aligned} \tag{2.5}$$

with after-fire resetting:

$$\text{if } v \geq 30 \text{ mV, then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} . \tag{2.6}$$

The Izhikevich model successfully reproduces different types of neuronal dynamics, although the conductance and current changes of the ion-channels are not fully described. It was widely acknowledged by researchers working on large-scale neural network simulation [40], [41].

Spike Response Model (SRM): as defined by Gerstner et al. [32] in a different way, expresses the membrane potential V_i of neuron N_i as a time integral over the past, including a model of refractoriness. The SRM is a phenomenological model of neuron, based on the occurrence of spike firings.

Let $F_i = \{t_i^{(f)} : 1 \leq f \leq n\} = \{t \mid u_i(t) = \upsilon \wedge u_i^{'}(t) > 0\}$ denote the set of all firing times of neuron N_i, and $\Gamma_i = \{ j \mid N_j$ is presynaptic to $N_i \}$ define its set of presynaptic neurons. The state $V_i(t)$ of neuron N_i at time t is given by

$$V_i(t) = \sum_{t_i^{(f)} \in F_i} \eta_i(t - t_i^{(f)}) + \sum_{j \in \Gamma_j} \sum_{t_i^{(f)} \in F_i} w_{ij} \varepsilon_{ij}(t - t_j^{(f)}) + \underbrace{\int_0^\infty \kappa_i(r) I(t-r) dr}_{\text{if external input current}}, \tag{2.7}$$

where the kernel functions η_i, ε_{ij} and κ_i respectively describe the potential reset, the response to a presynaptic spike and the response to an external current.

E. Izhikevich provided a nice table [39] in 2004 that compares different neuron models in terms of biophysical meaning, the types of biological neuron behavior that the model is able to replicate and the number of floating-point operations required for each step of simulation during a 1ms time span. These neuron models vary from the most complicated ones fitted to mimic real biological neurons, to the simplest ones, which only abstract the most important electrophysiology features. However, the challenge remains in selecting a proper model for the SNN because of the difficulty in balancing accuracy and complexity of the mathematical model while attempting to reproduce the dynamic behavior of a neuron model. The HH model and the Izhikevich model have been successfully used in simulating functional blocks of a biological nervous systems [42], [43] due to their ability to simulate complicated single neuron activities. The model of a single neuron for building large-scale brain must be: 1) capable of producing rich firing patterns exhibited by real biological neurons, yet, 2) computational simple, that is, simple neuron models with few parameters. However, most applications require large-scale SNN implementation, making phenomenal models the preferred ones for their simplicity in structure and efficiency in the simulation process [1]. The LIF model with its plausible biological features has been proven to work well in biological SNN behavior analysis and computer-aided recognition and classification tasks [13], [33], [44], [45]. When the computational requirements are substantial, as in the case of large-scale SNN implementation, the LIF models or even simper IF models will be preferable [46]–[51].

2.2 Models of Synapse

In the nervous system, synapse is a structure or interaction between two neurons with electrical or chemical signal transmitted from one to the other. Synaptic interaction is a more complex phenomenon than the neuron dynamics themselves. Thus, few detailed biophysical synaptic model exist. The most widely used synapse model in computational neuroscience are the phenomenon model, in which synaptic interactions are modeled by interaction kernels which sum up linearly over different synapse and time. The total impact of all synapses can be expressed in the following equation:

$$f_{syn}(t) = \sum_{synapses\,k} \sum_{spikes\,s} w_k \varepsilon_k (t - t_s), \tag{2.8}$$

where w_k is the weight of the *k*th synapse and ε_k denotes its synaptic interaction kernel. The interaction kernel can be an arbitrary shape, but it is constrained by the biophysics of synaptic interaction in most models. A widely used phenomenological model is as the following function:

$$\varepsilon(t) = A \cdot H(t) \frac{1}{\tau_{rise} - \tau_{fall}} \left[\exp(-\frac{t}{\tau_{rise}}) - \exp(-\frac{t}{\tau_{fall}}) \right], \tag{2.9}$$

where the two exponential functions model the arrival and leaving of neurotransmitters at the postsynaptic site, governed by time instance τ_{rise} and τ_{fall}, respectively. $H(t)$ is the Heaviside step function.

2.3 Network Architectures

Within the class of computationally oriented spiking neural networks, two main directions are distinguished. First, use the network structure equivalent to traditional neural networks. Feed-forward network is the simplest and mostly used network structure for spiking neurons[28], [52], [53]. Due to the complex dynamic of the spiking neurons, recurrent structure of spiking neural network is rare investigate[54]. Second, there are uniquely network structure for networks of spiking neurons. Most famous two network structures are the Echo State Network (ESN) and Liquid State Machine (LSM), which also called reservoir computing. Typically an input signal is fed into a fixed dynamical system called reservoir and the dynamics of the reservoir map the input to a higher dimension. Then a simple readout mechanism is performed to read the state of the reservoir and map it to the desired output. For reservoir computing, the training is only performed at the readout stage and the reservoir is invariant.

Echo State Network [55]: proposed by Jaeger in 2001, was intend to learning time series $\mathbf{u}(1), \mathbf{d}(1), \cdots, \mathbf{u}(T), \mathbf{d}(T)$ with recurrent networks. The internal states of the reservoir are supposed to reflect the concurrent effect of a new teacher input $u(t+1)$ and the desired output $d(t)$, related to the previous time. Therefore, there are the backward connections from the output layer toward the reservoir (Fig. 2.1) and dynamics of the network is governed by the following equation:

$$\mathbf{x}(t+1) = f(W^{in}\mathbf{u}(t+1) + W\mathbf{x}(t) + W^{back}\mathbf{d}(t)) \tag{2.10}$$

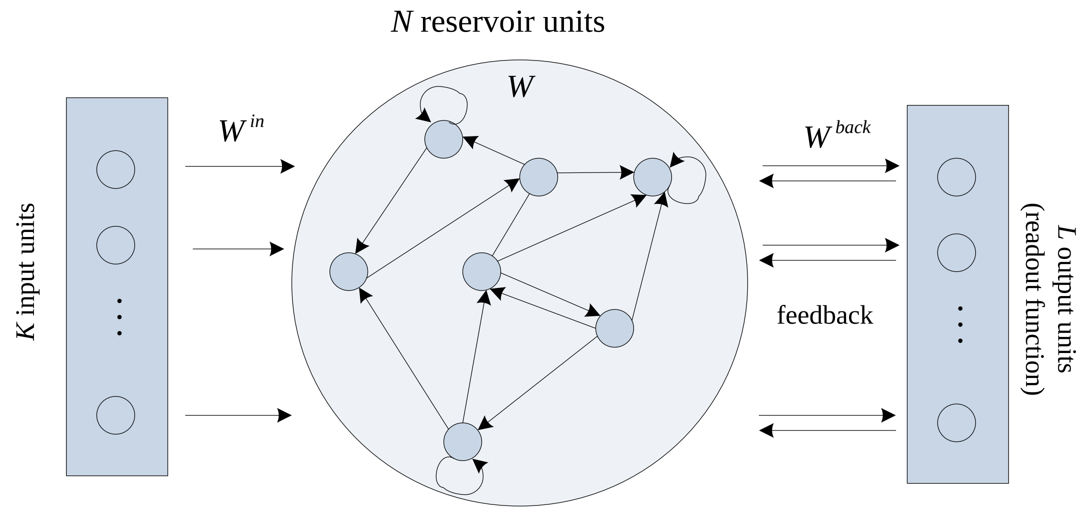


Figure 2.1 Architecture of ESN

where $\mathbf{x}(t+1)$ is the new state of the reservoir, W^{in}, W, and W^{back} are the input weight matrix, the matrix of weights in the reservoir and the matrix of feedback weights from the output of the reservoir, respectively.

ESNs have been successfully applied in many experiments, with 20-400 internal units in the network. Jaeger introduced spiking neurons model, LIF model, in the ESNs, mastered the benchmark task of learning the Mackey-Glass chaotic attractor [55]. Results improve substantially over the ESNs using sigmoid units. Verstraeten et al. [56] compared several measures for the reservoir dynamics with different neuron models.

Liquid State Machine [32]: proposed by Maass, was to explain how a continuous stream of input $u(\cdot)$ from the changing environment can be processed in real time by recurrent connections of IF neurons, as shown in Fig. 2.2. The reservoir here works as a liquid filter

L^M, operates similarly to water undertaking the transformation from the low-dimensional space of a set of stimulating surface into a higher dimensional space of waves in parallel. The liquid states $x^M(t)$ are transformed by a readout map f^M to generate an output $y(t)$.

LSM can be written as the following equations:

$$x^M(t) = (L^M(u))(t) \tag{2.11}$$

$$y(t) = f^M(x^M(t)) \tag{2.12}$$

where L^M is an operator that maps input functions $u(\cdot)$ onto functions $x^M(t)$. The L^M operator can be implemented by a randomly connected recurrent neural network. f^M is the memoryless readout map that transforms the current liquid state into the machine output. The readout is usually implemented by one or several IF neurons that trained to perform a specific task using supervised learning rule.

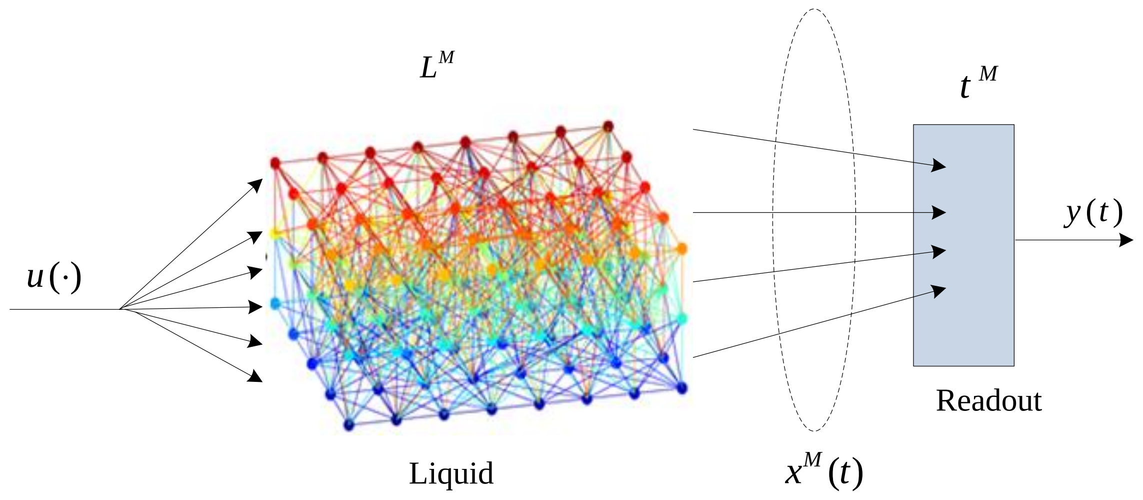


Figure 2.2 Architecture of LSM

LSM has been widely applied to nonlinear classification problems, such as XOR[57] etc. It is convenient for capturing most temporal features of spiking neuron processing, especially for time series prediction and for temporal pattern recognition.

2.4 Encoding Scheme

In biological nervous systems, a neuron transmits information to others via spike trains with specific frequency and amplitude. The inputs and output of traditional artificial neuron models such as threshold perceptions and sigmoidal neurons consider only rate encoding (coded by frequency) and will result in a loss of information expressed in the form of precise firing times of spikes. The data from the real-world is extremely dynamic, that everything changes continuously over time. Understanding the representation of external stimuli in the brain directly determines what kind of information mechanism should be utilized in the neural network. The most significant difference between SNN and traditional neural networks is that information in SNN is represented by spike trains which are a series of pulses with timings of interests. There are mainly two kinds of interpretations developed about how information is related to spike trains: 1) the rate encoding, which assumes that the information is encoded by the counts of spikes in a short time window; and 2) the spike time encoding which considers information carried at the precise time of each action potential in the spike train. Although the mechanisms for data representation and analysis using biologically-inspired neural networks is still under development, there are evidences show that spike time encoding might be more reliable in explaining experiments on the biology of nervous systems [58], [59]. Both rate encoding and spike time encoding essential in SNN applications. The followings further present a detailed overview of the rate code and the temporal code.

2.4.1 Rate Code

Rate code is a traditional coding scheme, which assuming that information about the stimulus is contained in the firing rate of the neuron. Before encoding external information, precise calculation of the firing rate is required. Thus neuronal responses are treated statistically or probabilistically. Mostly, rate encoding consider the spike count within an encoding window. Any information possibly encoded in the temporal structure of the spike train is ignored.

The simplest rate encoding is feed an analog signal to a Poisson neuron, which fires output spikes at probability proportional to its membrane potential. Such an encoding method has been adopted by Sprekeler *et al*. [60]. and Keer *et al*. [61] to analyze the recurrent ASNN behaviors. Although Poisson neuron model is easy for theoretical analysis, it was rarely implemented in real-world applications due to its inaccuracy in mapping analog signals to spike trains. Another rate encoding method, termed Hough Spiker Algorithm (HSA), was introduced by De Garis *et al*. [62] in 2000. It de-convolves the input signal into its individual spike responses, so that the post synaptic potential of the encoded spike train could be quite similar compared to the original signal. Schrauwen et al. [6] improved this encoding method by optimizing the de-convolution threshold, introduced a new encoding method called Bens Spiker Algorithm (BSA). BSA has been used widely as a rate encoding method for SNN applications, especially for electroencephalogram data [63]–[65].

Address-Event Representation (AER) is an asynchronous protocol designed for analog neural system simulation platforms [66]. It is also referred to as an encoding method by some researchers [67]–[69]. It encounters of “ON” and “OFF” events in the input signals are registered by AER to generate corresponding output spikes. The “ON” and “OFF” events in AER indicate the time when a change in the input signal either exceeds a positive threshold or fall behind a negative threshold. Under such definition, AER could be treated as a rate encoding method with regards to the derivatives of the input signal. The major problem of rate encoding methods is that an averaging time window is required for each sampling of the input signal, which as a consequence limits the temporal resolution of the encoded signals.

2.4.2 Temporal Code

Temporal coding is a straightforward method for translating a vector of real numbers into a spike train, for instance, for simulating traditional connectionist models using SNNs. Precise spike timing as a means to encode information in neural networks is biologically supported. The temporal encoding is outperform rate-based encoding when patterns within the encoding window provide the information about the input stimulus that cannot obtained from spike count. For example, there are evidences show that the populations of neurons in the primary auditory cortex coordinate the relative timing of their action potentials by grouping the neighboring spikes in short bursts, without changing the number of firings per second [70]. Another evidence for precise spike timing coding paradigm is required in artificial systems is: neuroprostheses for movement control where the precise timing of impulses applied to the paralyses muscles is critical for generating the desired, smooth movement trajectories [71]. Thus, the temporal patterns in spatiotemporal spikes can carry more information than rate encoding especially in the system where the processing speed is required to be high. For these reasons, much more attention has been focused on the development of learning rules for spiking neural networks that utilize a temporal coding scheme. Potential temporal coding strategies based on the precise timing of spikes are summarized in the following list:

Time to first spike: under this coding scheme, information is assumed to be encoded in the latency between the beginning of stimulus and the time of the first spike in the response neuron.

Rank-Order Coding (ROC): the information is encoded by the order of spikes in the activities of a neural population according to this coding method.

Latency code: the information is encoded by the relative latency between the neighboring spikes.

Resonant burst coding: downstream neurons affected by resonance are determined by the frequency of a burst.

Coding by synchrony: this scheme is assumed the neurons that encode different bit of information on the same object fir synchronously.

Phase encoding: Neuronal spike trains could encode information in the phase of a pulse with respect to the background oscillation. A simple implementation of phase encoding could be realized by linearly mapping the input signal to the delay of spikes within each synchronizing period [37].

Population encoding: Temporal receptive fields (Fig. 2.3) and the Cosine squared function method are the most famous population encoding schemes could also be utilized for phase encoding to improve the encoding resolution [72], [73]. To be more biologically plausible, Rumbell *et al*. [74] introduced a synchronizing method which considered spiking neurons as phase encoding units instead of performing linear mapping between analog values and spike delays. As shown in Fig. 2.3, input data a is encoded into temporal spike-time patterns for the input-neurons encoding this input-variable, using multiple local receptive fields like Radial Basis Functions. The translation of inputs into relative firing times is straightforward: the highest stimulated neuron, neuron 2, fires at a time close to $t=0$, whereas less-stimulated neurons, as for instance neuron 3, fire at increasingly later times. For a data range $[I_{\min}^{n},\cdots,I_{\max}^{n}]$ of a variable n,m neurons were used with Gaussian receptive files. For a neuron i its center was set to $I_{\min}^{n}+(2i-3)/2\cdot\{I_{\max}^{n}-I_{\min}^{n}\}/(m-2)$ and width $\sigma=1/\beta\{I_{\max}^{n}-I_{\min}^{n}\}/(m-2)$. The learning parameter β is chosen by trial and error. With such encoding, spiking neural networks ware shown to be effective for clustering tasks [75].

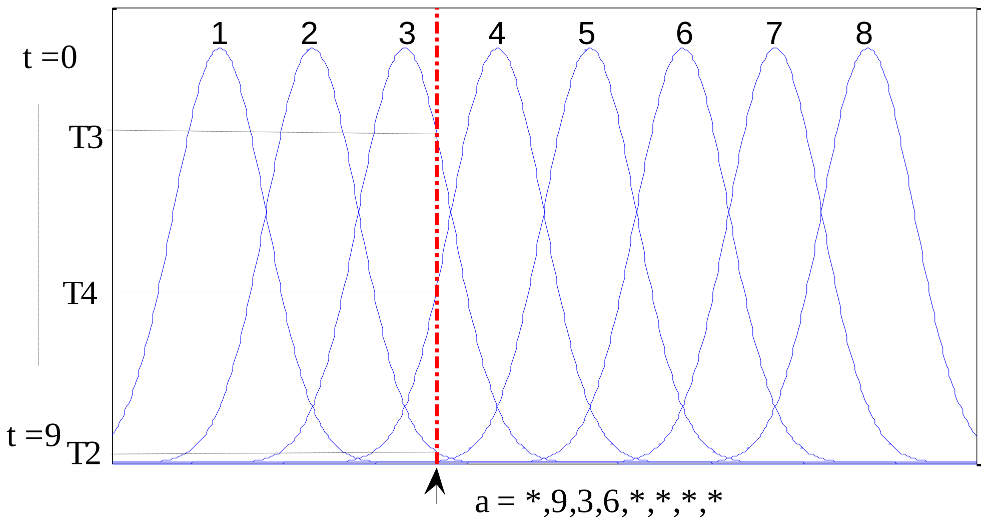


Figure 2.3 Population encoding with 8 overlapping Gaussian receptive fields

2.5 Temporal Learning in SNN

Traditional neural networks have been applied to pattern recognition in various guises. The best-known learning rules for such networks are of course the class of error-back-propagation rules for supervised learning and unsupervised learning rues such as Hebbian learning, or Kohonen self-organizing maps. By substituting traditional neurons with spiking neuron models, two main directions of learning rules for computationally oriented spiking neural networks have been developed. One is the development of learning methods equivalent to or similar to those developed for traditional neural networks. The other one is development of computational learning algorithms unique for networks of spiking neurons. Both supervised learning and unsupervised learning rules in the temporal framework are described in this section.

2.5.1 Synaptic Plasticity

Synaptic plasticity, also called unsupervised learning, refers to the adjustments of synapses between neurons in the brain. From the biological aspect of neurons, the changes of synaptic weights with effects lasting seconds or minutes, are called Short-Term Potentiation (STP) if the weights are strengthened, while Short-Term Depression (STD) if the weight values are decreased. On the scale of several hours or even more, the weight changes are referred to Long-Term Potentiation (LTP) and Long-Term Depression (LTD). A good review of the main synaptic plasticity mechanisms for regulating levels of activity in conjunction with Hebbian synaptic modification is given in [76]. There are neurobiological evidences increasingly demonstrate that synaptic plasticity in networks of spiking neurons is sensitive to the presence and precise timing of spikes [77]. The best-known learning paradigm for spiking neural network is Spike Timing Dependent Plasticity (STDP) induced by tight correlations between the spikes of pre- and postsynaptic neurons, which is often referred to a temporal Hebbian rule. It relies on local information driven by back-propagation of action potential through the dendrites of the postsynaptic neuron. For the computational purposes, STDP is normally modeled in SNNs using temporal window for adjusting the weight LTP and LTD which are derived from neurobiological experiments. Different shapes of STDP have been studied in [39].

A winner-take-all learning rule [78] modifies the synaptic weights using a time-variant of Hebbian learning: the weight of the synapse is increased when the start of the postsynaptic potential at a synapse slightly precedes a spike in the target neuron; earlier and later synapses are decreased in weights showing their lesser impact on the target neuron's spike time. After learning, the firing time of an output neuron reflects the distance of the evaluated pattern to its learning input pattern thus realizing a kind of RBF neuron.

2.5.2 Supervised Learning

Supervised learning contributes to the development and maintenance of lots of brain function. There is strong biological evidence that supervised learning exists in the cerebellum and the cerebellar cortex [17], [79]. However, the mechanism of supervised learning in the biological neurons remain unclear [57]. In view of this, research on supervised learning for spiking neurons and spiking neural networks is still at the early stage, and many existing learning methods have some weakness [8]. Here, we list some popular supervised learning rules base on temporal encoding, that it, during a certain running time, neurons can precisely emit spikes at appointed times through learning.

There are two types of supervised learning methods for SNNs based on temporal encoding, which are classified according to the number of spikes that need to be controller precisely. The first type is the single-spike learning, which can control only the firing time of a single spike. The most straightforward approach to implement supervised learning in spiking neural networks is as the same method used by Rumelhart et al. [80]: using the gradient descent on the error of the time difference between the desired output spike train and the actual output spike train. Different from the traditional artificial neural network, the spiking neuron’s activation function is not differentiable. Thus, backpropagation-like approach was derived for spiking networks with some additional assumptions. To overcome the discontinuous nature of spiking neurons, the threshold function is approximated, thus linearizing the model at a neuron's output spike times. The SpikeProp [81] rule has been shown to be capable of learning complex nonlinear tasks in spiking neural networks with similar accuracy as traditional sigmoidal neural networks, including the archetypal XOR classification task. SpikeProp and its variants such as QProp [82], RProp [82] have two major limitations [37], [82], [83] : (1) they do not allow multiple spikes in the output spike train, and (2) they are sensitive to spike loss, in that no error gradient is defined when the neuron does not fire for any pattern, and hence will never recover. Although single spike learning rules have good application capability, networks with only single spike output will limit the capacity and the diversity of information that they transmit. The Tempotron rule [84], another gradient descent based approach which is efficient for binary temporal classification task, encounters these two problems as well. As demonstrated in study [85], non-gradient-based methods like evolutionary strategies do not suffer from these tuning issues. However, evolution method is very time consuming which is not suitable for complex tasks.

Relative to the single spike learning, multi-spike learning methods are more consistent with the running and information transmitting model of the biological neurons. Due to the complexity of the learning targets increases significantly, most existing multi-spike learning methods focus on single spiking neurons or single-layer spiking neural networks. Temporal learning rules, such as SPAN [86], PSD [22], Chronotron [87], have been developed to train neurons to generate multiple output spikes in response to a spatiotemporal stimulus. In Chronotron, both analytically-derived (E-learning) and heuristically-defined (I-learning) rules are introduced. Both the E-learning rule and the

SPAN rule are based on error function that takes into account the difference between the actual output spike train and the desired spike train. Their application is therefore limited to tractable error evaluations, which are unavailable in biological neural networks and are computationally inefficient as well. The I-learning rule of Chronotron is based on a particular case of the Spike Response Model, which might have limitations for other spiking neuron models. In addition, it depends on weight initialization. Those synapses with zero initial value will not be updated according to the I-learning rule, which will lead to information loss from afferent neurons. For multilayer perceptron networks based on various spiking neuron models, performance comparable to SpikeProp is shown. An evolutionary strategy is, however, time consuming for large-scale networks. Carnell and Richardson [88] first applied the linear algebra into the learning of time series of spikes by using the Gram Schmidt projection process to calculate the weight change. However, this method is not a back-propagation based rule which is only applicable to a single spiking neuron or single layer of SNN. PBSNLR [89] first transform the supervised learning problem into a classification problem and solves the problem by using perception learning method. But it needs many learning epochs to achieve good learning performance when time step is precise.

ReSuMe [27], referred to as the remote supervised method, is one of the few supervised learning algorithms that is based on a learning window concept derived from STDP. In the ReSuMe learning process, the desired or supervisory signal does not directly influence the membrane potential of the corresponding learning neuron. It is described to be biologically plausible and can learn patterns in an online mode by adjusting the synaptic weights locally in time. Similar to SPAN and PSD, ReSuMe is derived from the Widrow-Hoff rule [90]. It combines two processes: one is STDP for strengthening synapses based on input spike trains and desired output spike train; the other one is anti-STDP learning window for weakening synapses based on the input spike trains and actual output spike train. Under remote supervision of instruction neuron, the output neuron can produce a desired output spike train in response to a spatiotemporal input spike pattern. The results show that ReSuMe has good learning ability and wide applications. With this method, it also can reconstruct the target input-output transformations. In [57], the ability of ReSuMe on sequence learning, classification and spike shifting are discussed. As another supervised learning rule based on STDP, SWAT (Synaptic Weight Association Training) [24] merged BCM (Bienenstock-Cooper-Munro) learning rule with STDP. Different from ReSuMe, SWAT focuses on multilayer SNN rather single neuron or single layer SNN. The SNN uses a feed-forward topology, in which the hidden layer works as a frequency filter while frequencies of input and output layers are kept as fixed values.

The common disadvantage of these multi-spike learning methods for spiking neurons is that the learning efficiency is relative low. When the desired output spike train is relatively long, the neuron needs to run a relatively long time to make the learning converge. In [57], it needed around 450 epochs to achieve high learning accuracy when a neuron learned to emit a spike train of 400ms length using ReSuMe. Thus, it lowers its learning efficiency and weakens its ability to solve real-time problems. In addition to learning efficiency, the learning accuracy of the existing learning methods will decrease when the number of desired output spikes increase to a certain degree, and this limits their ability to solve complicated tasks.

These disadvantages of the existing methods prompted us to search for supervised learning method with higher learning efficiency and accuracy.

3. CROSS-CORRELATED DELAY SHIFT SUPERVISED LEARNING METHOD

This chapter introduces a novel learning algorithm for spiking neurons, called CCDS, which is able to learn and reproduce arbitrary spike patterns in a supervised fashion allowing the processing of spatiotemporal information encoded in the precise timing of spikes. Unlike the Remote Supervised Method (ReSuMe), synapse delays and axonal delays in CCDS are variants which are modulated together with weights during learning. The CCDS rule is both biologically plausible and computationally efficient. The properties of this learning rule are investigated extensively through experimental evaluations in terms of reliability, adaptive learning performance, generality to different neuron models, learning in the presence of noise, effects of its learning parameters and classification performance. Results presented show that the CCDS learning method achieves learning accuracy and learning speed comparable with ReSuMe, but improves classification accuracy when compared to both the Spike Pattern Association Neuron (SPAN) learning rule and the Tempotron learning rule.

3.1 Introduction

Spiking neural networks [1], [2], [91], [92] (SNNs) are considered to be more biologically realistic compared to typical rate-coded networks as they can model more closely different types of neurons and their temporal dynamics [93]. SNNs exhibit interesting properties that make them particularly suitable for applications [20], [46], [94]–[96] that require fast and efficient computation and where the timing of input/output signals carries important information. Various interesting types of neuron models [1], [91], [97]–[100] have emerged for building large scale artificial SNNs. SNNs rely on three main functional parts: encoding, learning and decoding.

Recently, the learning problem in a network of spiking neurons has attracted attention from a number of researchers. One reason for this interest is that in these networks the learning process is considered as a realistic model in the biological neural networks. There is strong biological evidence that supervised learning exists in the cerebellum and the cerebellar cortex [17], [79]. It is shown that supervised signals are provided to the learning modules or neural structures in the brain. Several supervised learning algorithms have been successfully developed for nonlinear benchmark problems. Some of the existing supervised learning rules, such as SpikeProp [101], QProp [82], RProp [82] use error back propagation similar to the traditional Neural Networks (NNs). The two major limitations of these methods and their extensions [37], [82], [83] are that (1) they do not allow multiple spikes in the output spike train, and (2) they are sensitive to spike loss, in that no error gradient is defined when the neuron does not fire for any pattern, and hence will never recover. The Tempotron rule [84], another gradient descent based approach which is efficient for binary temporal classification task, encounters these two problems as well. As demonstrated in study [85], non-gradient-based methods like evolutionary strategies do not suffer from these tuning issues. For multilayer perceptron networks based on various spiking neuron models, performance comparable to SpikeProp is shown. An evolutionary strategy is, however, time consuming for large-scale networks. Other temporal learning rules, such as SPAN [86], PSD [22], Chronotron [87], have been developed to train neurons to generate multiple output spikes in response to a spatiotemporal stimulus. In Chronotron, both analytically-derived (E-learning) and heuristically-defined (I-learning) rules are introduced. Both the E-learning rule and the SPAN rule are based on error function that takes into account the difference between the actual output spike train and the desired spike train. Their application is therefore limited to tractable error evaluations, which are unavailable in biological neural networks and are computationally inefficient as well. The I-learning rule of Chronotron is based on a particular case of the Spike Response Model, which might have limitations for other spiking neuron models. In addition, it depends on weight initialization. Those synapses with zero initial value will not be updated according to the I-learning rule, which will lead to information loss from afferent neurons.

Well known biologically-inspired Spike-timing dependent plasticity (STDP) was observed through experiments on hippocampal neurons [102] which can induce either long- or short-term potentiation in synapses based on local variables such as the relative timing of spikes, voltage, and firing frequency. It shows that postsynaptic firing, which occurred within a time window of 20ms after presynaptic firing, resulted in weight potentiation; whereas postsynaptic firing within a time window of 20ms before presynaptic firing led to weight depression. STDP is widely used in unsupervised processes and pattern recognition [103]. However, simply implementing this form of learning will not always guarantee convergence for the network of neurons during learning as the rule does not formulate the competition between neurons.

ReSuMe [27] is one of the few supervised learning algorithms that is based on a learning window concept derived from STDP. It is described to be biologically plausible and can learn patterns in an online mode by adjusting the synaptic weights locally in time. Similar to SPAN and PSD, ReSuMe is derived from the Widrow-Hoff rule [90]. It combines STDP and anti-STDP learning window under remote supervision of instruction neuron to produce a desired output spike train in response to a spatiotemporal input spike pattern. With this method, it also can reconstruct the target input-output transformations.

In this study, a learning method is proposed to improve ReSuMe by integrating synaptic delay and axonal delay with the synaptic weights learning process. The importance of delays in computing with spiking neurons in defining a supervised learning rule acting on the delays of connections (instead of weights) between the reservoir and the readout neurons was proved [28], [104]. The learning rule of the readout delays is based on a temporal margin criterion inspired by Vapnik's theory [105]. Axonal conduction delays refer to the time required for an action potential to travel from its initial site near the neuronal soma to the axon terminals, where the synapse connects the soma with other neurons.

Although axonal delays do not vary continually in the brain, a wide range of delay values has been observed. Evidence shows that conduction delays in the mammalian brain can reach from a few ms up to over 50 ms [106]. The effect of delay on the processing ability of the nervous system has been extensively studied [107], [108]. There is biological evidence that the synaptic delay can be modulated instead of always being invariant [109].

The evidence supports the introduction of a novel learning algorithm for SNNs considering both axonal and synaptic delays.

Several analyses of SNNs have proved the need for programmable delays for both computational power [1] and learnability [104], [110]. Two approaches for delay learning in SNNs are delay selection [13], [37], [82], [111] and delay shift [112]. In the delay selection method, the output of a neuron is assumed to be connected to the input of another neuron by multiple connections with different fixed delays. The weights of connections related to suitable delays are enhanced while the weights related to unsuitable ones are decreased. The delay shift method adapts the actual delay values of the connections between neurons during training. Input spike patterns close to the synaptic delay vector will make the neuron emit an output spike. Such adaptation may be achieved by changing the length or thickness of dendrites and axons, the extent of myelination of axons, or the density and type of ion channels from biological aspects [15]. The weights are considered constant during the learning process [112].

In this chapter, we propose a novel learning algorithm named Cross-Correlated Delay Shift (CCDS), as a supervised learning method able to learn the association between precise patterns considering axonal delay, synapse delay with weight modulation. In the first experiment, the basic concepts of the CCDS rule are demonstrated. Then, various properties of the CCDS learning rule are investigated through experimental analysis, in which learning process, adaptive learning performance, generality, influence of noise and classification performance are discussed. Finally some discussion with retrospective evaluations is presented.

3.2 Methods

In this section, the spiking neuron model, ReSuMe and the proposed learning method are described in detail. In the proposed learning method, the synapse delays and axonal delays associated with weights are all trained. The error measurement method is also provided afterwards.

3.2.1 Spiking Neuron Model

Conductance-based neuron models such as the Hodgkin-Huxley model [98], are known to accurately reproduce the electrophysiological signals, but they remain computationally taxing. Simple phenomenon models with low computational cost are more popular for studying the learning and dynamics of spiking neural networks. If not specified, the 1-D leaky integrate-and-fire model is selected in this study. The dynamic of the *i*-th neuron can be described as in the following equation:

$$\tau_i \frac{dV_i}{dt} = E - V_i + (I_{syn} + I_{ns}) \cdot R_i\,, \tag{3.1}$$

where V_i is the membrane potential, $\tau_i = R_i C_i$ is the membrane time constant relating to the 'leakage' of charge across the neuron's membrane when it is not at rest, and R_i is the neuron's effective membrane resistance, E stands for the resting potential, I_{syn} and I_{ns} are the sum of synaptic currents entering the given neuron and background noise current, respectively. When membrane voltage V_i reaches the threshold level V_{th}, the neuron emits a spike and V_i is reset to V_{rest} for a refractory period t_{ref}.

The synaptic current is modeled as follows:

$$I_{syn}(t) = \sum_j w_j I_{PSC}^j(t), \tag{3.2}$$

where w_j stands for synaptic efficacy of the *j*-th afferent neuron, I_{PSC}^j represents the postsynaptic current from afferent spikes. The postsynaptic current with synaptic delay can be written as:

$$I_{PSC}^j(t - dt_j) = \sum_{t^f} K(t - t^m - dt_j)H(t - t^m - dt_j), \tag{3.3}$$

where t^m and dt_j are the *m*-th spike and the synaptic delay from the *j*-th afferent neuron, respectively; $H(t)$ is the Heaviside function; K refers to a normalized exponential kernel function as:

$$K(t) = V_0(\exp(-t/\tau_s) - \exp(-t/\tau_f)), \tag{3.4}$$

where V_0 is the normalized factor, τ_s and τ_f are the slow and fast decay time constant, respectively, setting $\tau_s / \tau_f = 4$.

Another two phenomenon models are also included in section 3.3.4 for the comparison purpose: the 2-D Izhikevich [99] neuron model and Spike Response Model (SRM) [32].

The dynamics of the Izhikevich model is described as:

$$\begin{cases} dV_i / dt = 0.04V_i^2 + 5V_i + 140 - U + I_{syn} + I_{ns} \\ dU / dt = a(bV_i - U) \end{cases} \tag{3.5}$$

If $V_i \geq 30mV$, then $V_i \leftarrow c, U \leftarrow U + d$.

In this model, V_i denotes the membrane potential and U represents the membrane recovery variable. The synaptic current I_{syn} has the same form of (3.2), and I_{ns} defines the background noise current. Izhikevich model can exhibit 20 of the most prominent features of biological spiking neurons with different parameters [99].

The SRM is a phenomenological model of neuron, based on the occurrence of spike firings. The membrane potential V_i of neuron N_i is a time integral over the past.

Let $\zeta_i = \{t_i^{(f)}; 1 \leq f \leq n\} = \{t \mid u_i(t) = \upsilon \wedge u_i^{'}(t) > 0\}$ denote the set of all firing times of neuron N_i, and $\Gamma_i = \{j \mid N_j \text{ is presynaptic to } N_i\}$ define its set of presynaptic neurons. The state $V_i(t)$ of neuron N_i at time t is given by

$$V_i(t) = \sum_{t_i^{(f)} \in \zeta_i} \eta_i(t - t_i^{(f)}) + \sum_{j \in \Gamma_j} \sum_{t_i^{(f)} \in \zeta_i} w_{ij} \varepsilon_{ij}(t - t_j^{(f)}), \tag{3.6}$$

where the kernel functions η_i, ε_{ij} and κ_i respectively describe the potential reset and the response to a presynaptic spike. $\varepsilon_{ij}(t - t_j^{(f)})$ is the spike response function with $\varepsilon_{ij}(t - t_j^{(f)}) = 0$ for $t \le t_j^{(f)}$. The times $t_j^{(f)}$ represent the firing times of neuron j. In our case the spike response function $\varepsilon(t)$ describes a standard post-synaptic potential:

$$\varepsilon(t) = \frac{t}{\tau} \exp\left(1 - \frac{t}{\tau}\right), \tag{3.7}$$

where $\tau > 0$ models the membrane potential time constant and determines rise and decay of the function.

3.2.2 ReSuMe

Supervised learning in temporal encoded SNNs attempts to link the input spike train with the output spike sequence. ReSuMe is such a learning method which adjusts the synaptic weights of a neuron to generate a desired spike train $S^d(t)$ in response to a spatio-temporal input spike pattern expressed as $S^{in}(t) = [s_1(t), s_2(t), \cdots, s_n(t)]$. By employing STDP and anti-STDP window, synaptic weights are modified in ReSuMe according to the following relation:

$$\frac{d}{dt}w_i(t) = [S_d(t) - S_o(t)]\left[a_d + \int_0^\infty W(s)S_i(t-s)ds\right], \quad (3.8)$$

where $S_d(t)$, $S_o(t)$ and $S_i(t)$ are the desired, actual output spike train and input spike train corresponding to the *i*-th synapse, respectively. The role of parameter a_d is to adjust the average strength of the synaptic input to impose on a neuron a desired activity. In the case of excitatory synapses, the term a_d is positive and the learning window $W(s)$ has a similar shape as STDP. In the case of inhibitory synapses, a_d is negative and $W(s)$ is defined similarly as for anti-STDP rules. When the number of spikes in the actual output spike train $S_o(t)$ is more (less) than the number of spikes in the desired spike train $S_d(t)$, a_d decreases (increases) the weights. Thus, the actual mean firing rate of $S_o(t)$ approaches the mean firing rate of signal $S_d(t)$. This will speed up the convergence of the training process. The formal proof for convergence of the ReSuMe process is illustrated in study [113]. In ReSuMe, no delay was considered.

3.2.3 CCDS

Considering synaptic and axonal delays, illustration of the neuron structure is shown in Fig. 3.1. The inputs of the spiking neuron are the times of the discrete spikes. Each spike from the afferent neuron will result in a post-synaptic current (PSC), and the weighted sum of all incoming PSCs from afferent neurons determines whether a spike fires at the current moment. The outputs of spiking neuron are the times of fired spikes. Fig. 3.1 shows a multi-connected neuron structure with axonal delays d_i, $i = 1, 2, \cdots, n$ and synapse delays dt_i, $i = 1, 2, \cdots, n$. The corresponding weight values are from w_1 to w_n, respectively.

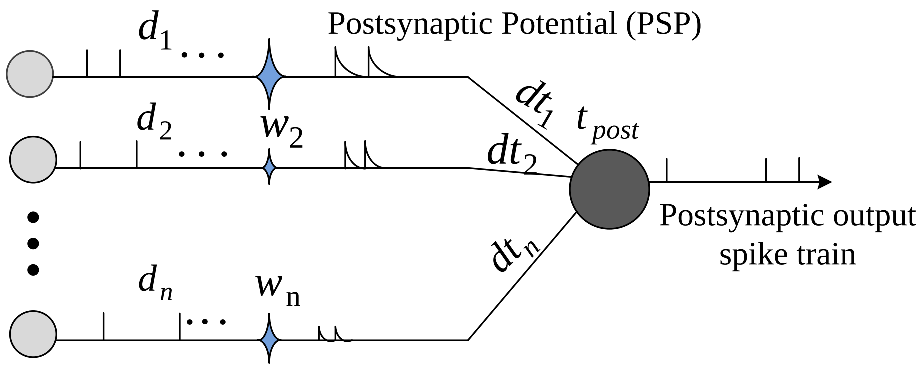


Figure 3.1 Neuron structure with multi-path connectivity

The time difference between input and output spike can be expressed as

$$\delta_{t_i} = t_{post} - (t_{pre} + d_i + dt_i), i = 1, \cdots, n. \tag{3.9}$$

Then the positive half of the learning window of STDP results in Long-Term Potentiation (LTP) of the synaptic weights that can be written as

$$\delta w_i = A_1 \exp(-\delta_{t_i} / \tau_1) \qquad \text{for } \delta_{t_i} > 0, \tag{3.10}$$

where A_1 is the maximum value of the weight potentiation, τ_1 is the width of the window for LTP and δ_{t_i} is the time differential as defined by (3.9).

Similarly, the negative part of the learning window where Long-Term Depression (LTD) occurs is given by

$$\delta w_i = -A_2 \exp(-\delta_{t_i} / \tau_2) \qquad \text{for } \delta_{t_i} < 0 \cdot \tag{3.11}$$

Again, A_2 is the maximum magnitude of weight depression and τ_2 defines the width of the window for LTD.

The weight modulation can then be written as

$$w_{i(new)} = w_{i(old)} + \delta w_i \,. \tag{3.12}$$

In order to speed up the learning process, we propose that the input spatiotemporal pattern be split into groups of input neurons, and the learning rule operates on a weight such that its relative magnitude reflects the association of a particular spike time to each group. Thus, both pre- and post-firing activity at a synaptic site and pre- and post-firing activity at other synaptic sites where presynaptic neurons have similar firing times are taken into account. This would reflect the relative occurrence of an input spike within each group during training.

Let us first consider a simple example in order to formulate the relative occurrence rule. Assume both data groups g_1 and g_2 have a total of *k* spikes occurring at various times within a temporal window T. Consider a particular spike time t_s occurring at *n* different channels (neurons). Assuming time t_s occurs *p* times within group g_1 and q times within group g_2, occurrence of t_s in g_1 relative to g_2 is

$$O(g_1, t_s) = \frac{p}{p+q} \ . \tag{3.13}$$

Similarly, the relative occurrence of t_s in g_2 is

$$O(g_2, t_s) = \frac{q}{p+q} \, . \tag{3.14}$$

The weight associating t_s to g_1 can be modified to

$$w_{ij(new)} = w_{ij(old)} + \frac{p}{p+q} \delta w_{ij}(t_s) \, , \tag{3.15}$$

where $w_{ij(old)}$ is the pre-trained value associated with connection w_{ij} between neurons *i* and *j*, and $\delta w_{ij}(t_s)$ has the same form as in (3.8). A similar rule that reflects the association of t_s with g_2 is given by

$$w_{ij(new)} = w_{ij(old)} + \frac{q}{p+q} \delta w_{ij}(t_s) \, . \tag{3.16}$$

Dividing all the input spike trains into M groups, g_r, $r = 1,...,M$, with m spike trains in each group, t_s occurs q_r times within group $g_r, i = 1,...,M$, $q_r \leq m$. F is the maximum number of spikes across all input spike trains. The updated weight can thus be written as

$$w_{ij(new)} = w_{ij(old)} + CC_i^r \cdot \delta w_{ij}(t_s), \tag{3.17}$$

where the cross correlated term of the *i*-th neuron in group *r* is given by the relation

$$CC_i^r = \frac{\sum_{f=0}^{F} O(g_r, t_i^f)}{\sum_{f=0}^{F} \sum_{j=1}^{M} O(g_j, t_i^f)}, \tag{3.18}$$

and where t_i^f is the firing time of the *f*-th (*f*=1,2,…) spike in the *i*-th input spike train, $O(g_r, t_i^f)$ defines the occurrence of spike time t_i^f in g_r in relation to the other groups.

The proposed CCDS algorithm is a heuristic method which helps the neuron generate the desired output and also remove the instance of an undesired output. In CCDS, the delay is applied to the connection that has the nearest spike before the desired time, which leads to an increase in the Post-Synaptic Potential (PSP) at the desired time. This increment brings the positive PSP produced by excitatory synapses close to the desired spike time in order to increase the level of the total PSP of the output neuron and consequently cause the neuron to reach the firing threshold and emit a spike at the desired time. In addition, the reduction of the PSP at an undesired output spike is achieved by delayed PSP. The reduction may eventually cancel the undesired spike.

The nearest previous input spike is calculated via local variable, $x_i(t)$, described in (3.19). Variable $x_i(t)$, a low-pass filtered version of spike trains, jumps to a saturated value A_o whenever a presynaptic spike arrives.

$$x_i(t) = \begin{cases} A_o \exp(-(t - t_i^f)/\tau), & \mathit{for}\ t_i^f < t < t_i^{f+1} \\ A_o & \mathit{for}\ t = \cdots t_i^f,\ t_i^{f+1}, \cdots \end{cases} \tag{3.19}$$

Amplitude A_o and time decay τ are constants.

Assuming the *n*-th synapse (*n*=1,2,…,*N*) has the nearest spike before the current time *t*, the delays d_n and dt_n shift the effect of its spike to the time *t* by using the inverse of (3.19).

$$d_n + dt_n = t - t_o^f = -\tau \ln\left(\frac{x_o(t)}{A_o}\right). \tag{3.20}$$

At desired spiking time t^{f_d} without any actual output spikes t^{f_a} , $x_o(t)$ is chosen from excitatory synapses. The chosen connection is delayed by $d_i + dt_i$. Then the spike is shifted toward the desired time, which will lead to an increment in the PSP. In contrast, at the undesired spiking time with output spikes, $x_o(t)$ is chosen from inhibitory synapses. This delay adjustment can be calculated using equation (3.21).

$$d_i + dt_i = \begin{cases} (-1)^l (d_n + dt_n) x_i(t) / x_n(t), & t = t^{f_d} \\ (-1)^{l+1} (d_n + dt_n) x_i(t) / x_n(t), & t = t^{f_a} \\ 0, & otherwise \end{cases} \tag{3.21}$$

where l takes an even value for excitatory synapses and an odd value for inhibitory synapses.

Considering both cross-correlated term and delay shift effect, the weights and delays as governed by the CCDS learning rule are updated on the basis of (3.22), where delays are updated according to (3.20) and (3.21). The CCDS rule is proposed for processing spatiotemporal patterns, where the exact time of each spike is used to convey information. In CCDS, weights are updated as follows:

$$\frac{dw_i^r(t)}{dt} = CC_i^r \cdot [s_d(t) - s_o(t)] \cdot [a + \int_0^{+\infty} W(s - d_i - dt_i) S_i(t - d_i - dt_i - s) ds] \tag{3.22}$$

where $s_d(t), s_o(t), S_i(t)$ are the desired output spike train, the actual output spike train and the input spike train, respectively. Constant a is the non-Hebbian term used to speed up the learning process. Term CC_i^r has the same form as in (3.18), and with A being the amplitude of the long-term potentiation, the learning window is

$$W(s) = \begin{cases} A\exp(-s/\tau), & s \geq 0 \\ 0, & s < 0 \end{cases}. \tag{3.23}$$

At the beginning of the training procedure, 20% of the weights are considered inhibitory and 80% of the weights are considered excitatory. In each epoch, synaptic and axonal delays are adjusted according to (3.20) and (3.21). Both delays and the connection weight can be changed many times during the learning process.

3.2.4 Error Measurement

The correlated-based metric (*C*) [114] is used to evaluate the similarity of the desired spike pattern with the actual output spike train. It takes a value between zero and one. The metric *C* equals one for identical spikes and drops down to zero for loosely correlated trains. The metric *C* is calculated after every learning epoch as

$$C = \frac{\vec{S}_d \cdot \vec{S}_o}{|\vec{S}_d| \cdot |\vec{S}_o|} \qquad (3.24)$$

where $\vec{s}_d$ and $\vec{s}_o$ are low pass filtered vectors in response to the desired spike train $s_d(t)$ and actual output spike train $s_o(t)$, respectively.

3.3 Results

The first experiment is devised to demonstrate the ability of the proposed CCDS rule for learning a spatiotemporal spike pattern. The neuron is trained to fire at desired spike times.

3.3.1 Learning Process

The trained neuron is connected with *N* afferent neurons, and each fires a spike train in the time interval (*0, T*). Input spike trains and desired spike trains are randomly generated with a homogeneous Poisson distribution with mean frequencies F_{in} and F_d, respectively. In this experiment, we set *N*=600, *T* = 200ms, $F_{in} = 10\text{Hz}$ and $F_d = 40\text{Hz}$. Since 20% of the cortex neurons are inhibitory neurons while 80% of cortex neurons are excitatory ones [1], the ratio of inhibitory and excitatory synapses is set to 1/4. The initial synaptic weights are drawn randomly from a uniform distribution with a mean value of -0.5 and a standard deviation of 0.2 for an inhibitory synapse, and with mean value of 0.75 and a standard deviation of 0.2 for an excitatory synapse. For the learning parameters, we set the membrane time constant $\tau_i = 10ms$ and the refractory period $t_{ref} = 5ms$; while the initial voltage, threshold voltage and reset voltage are selected as $V_{init} = -60mV$, $V_{th} = -55mV$ and $V_{reset} = -65mV$, respectively. By trial and error, the weights are capped within the range of [-15, 15] in order to get the optimal learning performance.

As both axonal and synapse delays are limited in the biological neurons, all axonal delays and synaptic delays in this study evolve within the interval [0, 40] ms and [0, 2] ms, respectively.

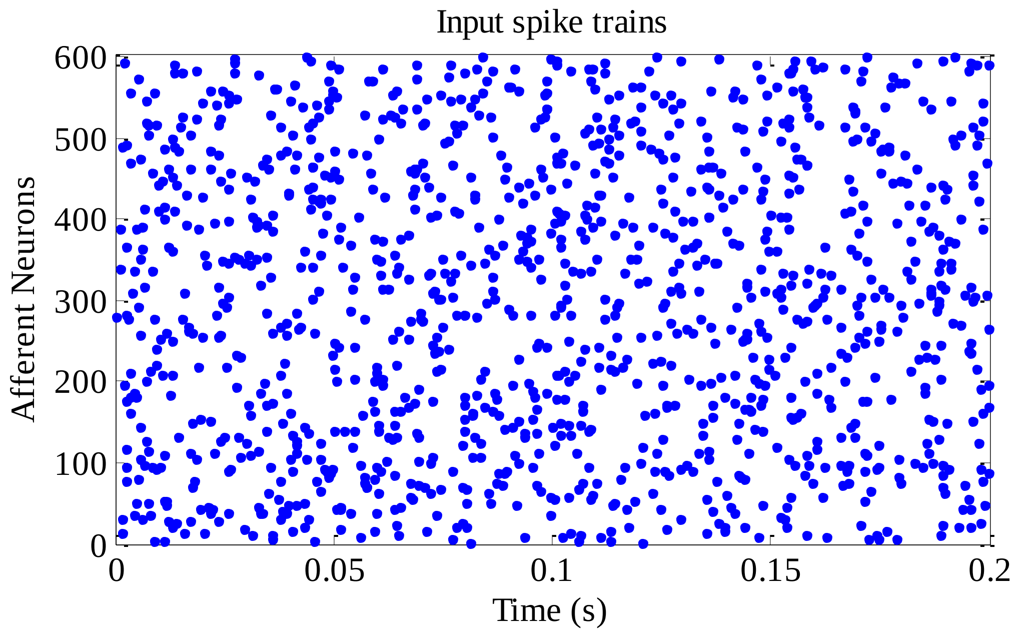
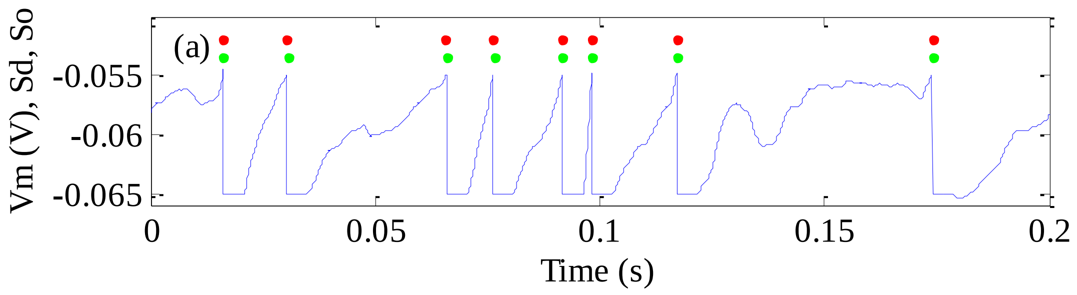


Figure 3.2 Spike raster of input spike trains when *N*=600.



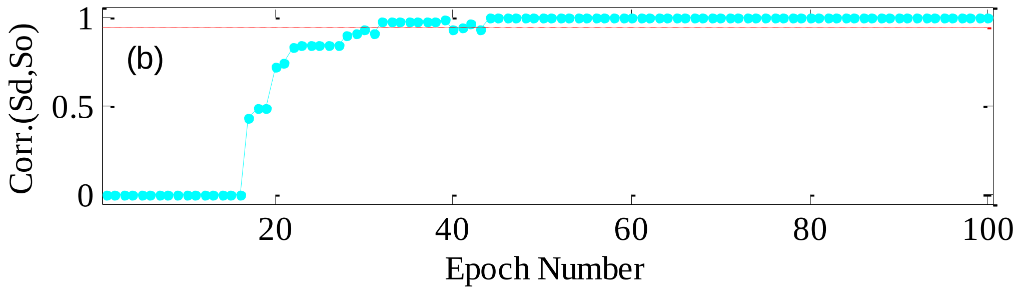
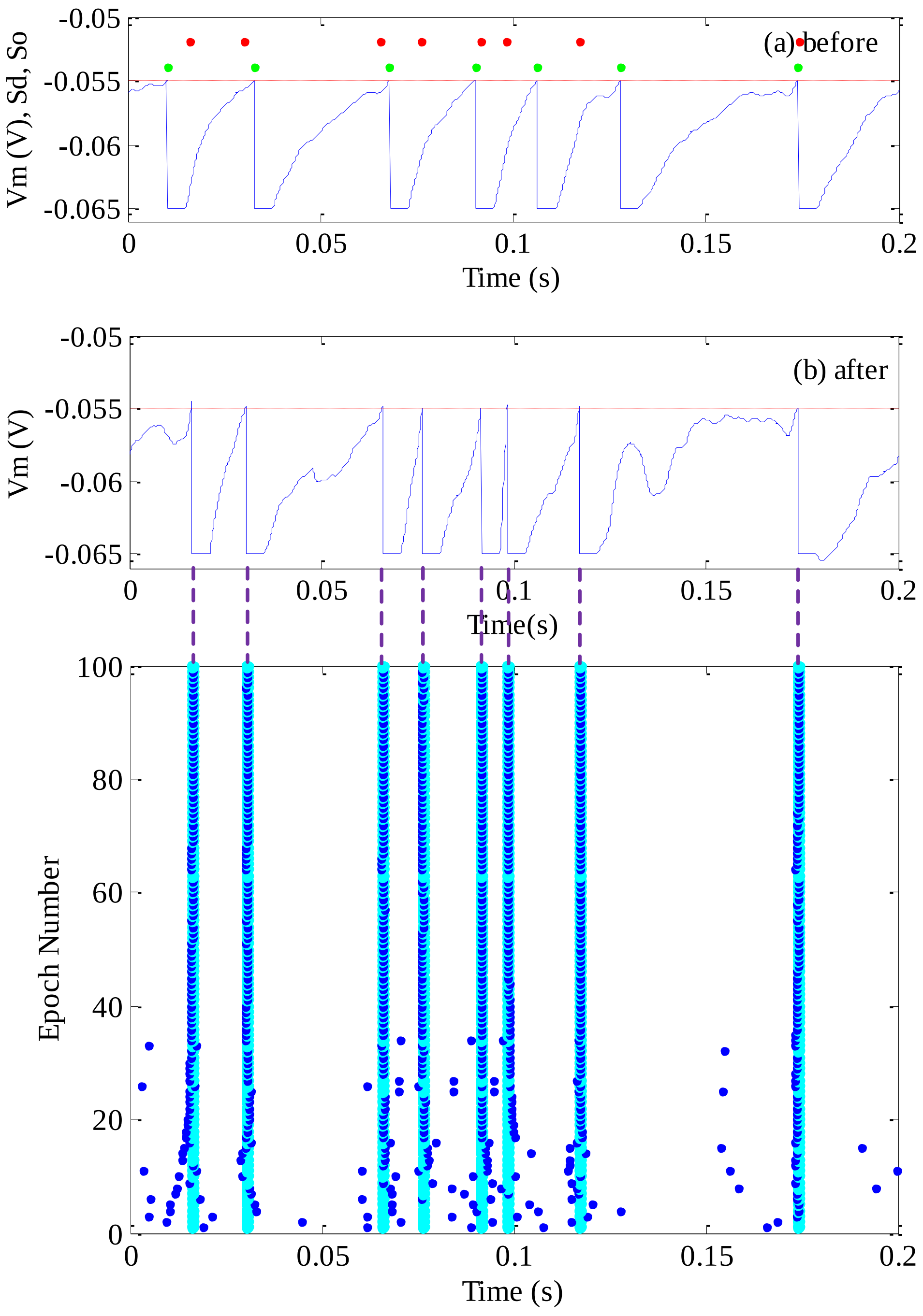


Figure 3.3 Training results without noise. (a) V_m: membrane potential after learning; red dots: target spike train; green dots: actual output spike train; (b) correlated-based metric *C* of target and output spike trains.



(c)
Figure 3.4 Temporal sequence learning of a typical run without noise (a) membrane potential before learning; (b) membrane potential after learning; (c) learning process.

Input spike trains are generated by a homogenous Poisson spike train with frequency $F_i = 10Hz$ with N=600 afferent neurons, as shown in Fig. 3.2. Frequency $F_d = 40Hz$ is chosen to produce the output spike train. Delayed version LIF is utilized for the training as can be seen from Fig. 3.3(a), with the red dots being the target spikes and the green dots being the actual spikes. Fig. 3.3(b) shows the similarity of the actual output spikes and the desired spikes. During the learning process, the neuron gradually learns to produce spikes at the target time, and it is reflected by the increasing correlation C. At around 30 epochs, the correlation C reaches a satisfactory level C>0.95. Each learning epoch takes 29.8ms in computation time, which was recorded using Matlab simulations running on a quad core PC with Intel core™ i7-2600 3.4GHz CPU and 16GB of RAM. After a small period of oscillation, the correlation *C* converges. The evolution of firing patterns generated by the neuron in consecutive learning epochs can be seen in Fig. 3.4(c), where the cyan line is the desired spike and the blue dots are the actual output spike patterns according to the learning epochs.

The dynamics of the neuron's membrane potential is also given in Fig. 3.4. The results show that the neuron can successfully learn to emit the desired spike train from the initial random output spike train after just 37 learning epochs. The weight randomly generated spike patterns converge perfectly after training. The synapse weights variance of each neuron is given in Fig. 3.5, in which both excitatory weights and inhibitory weights stop evolving around the 37-th epoch.

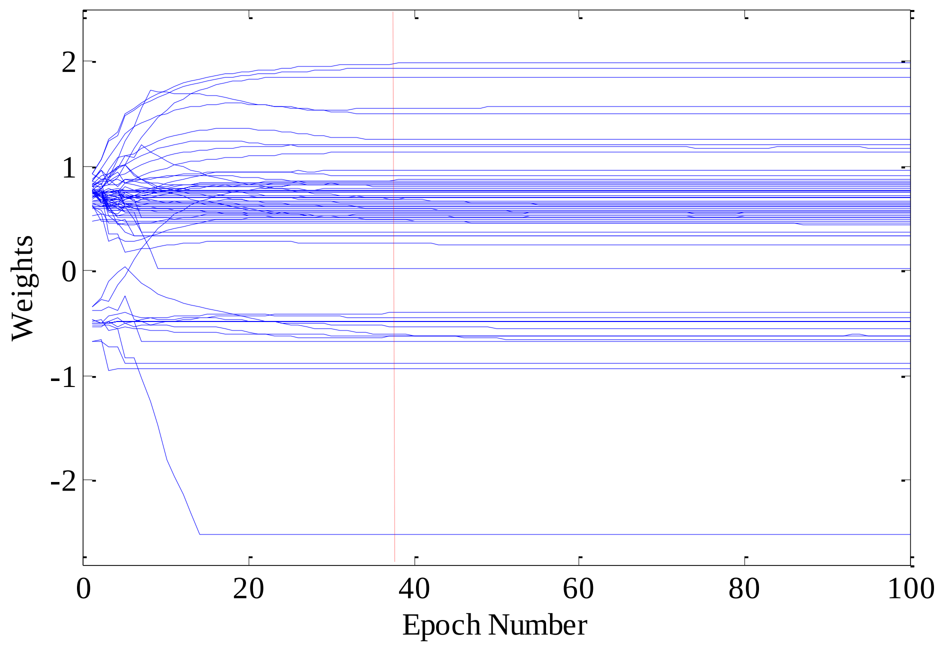
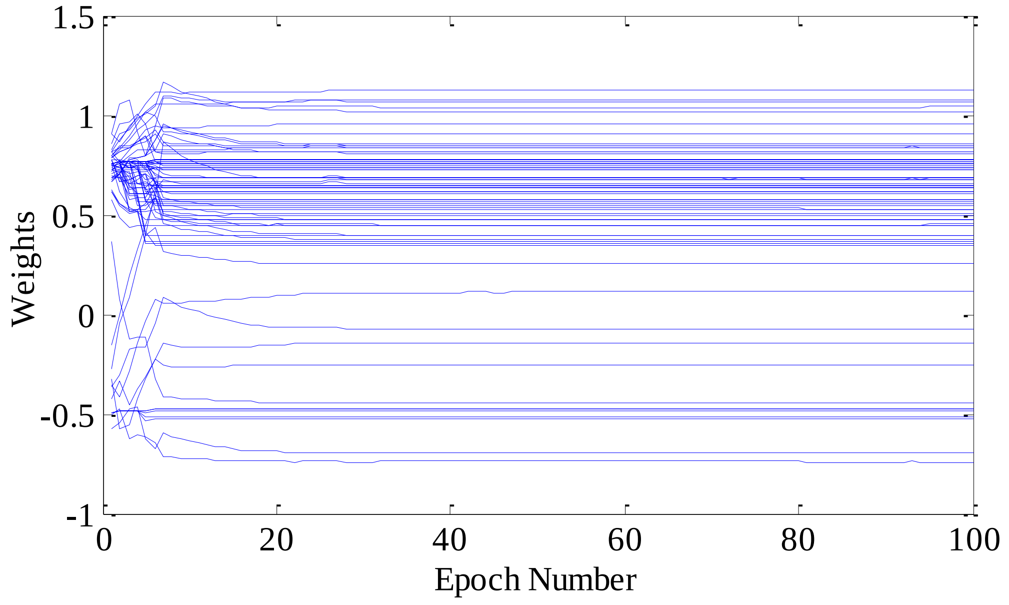
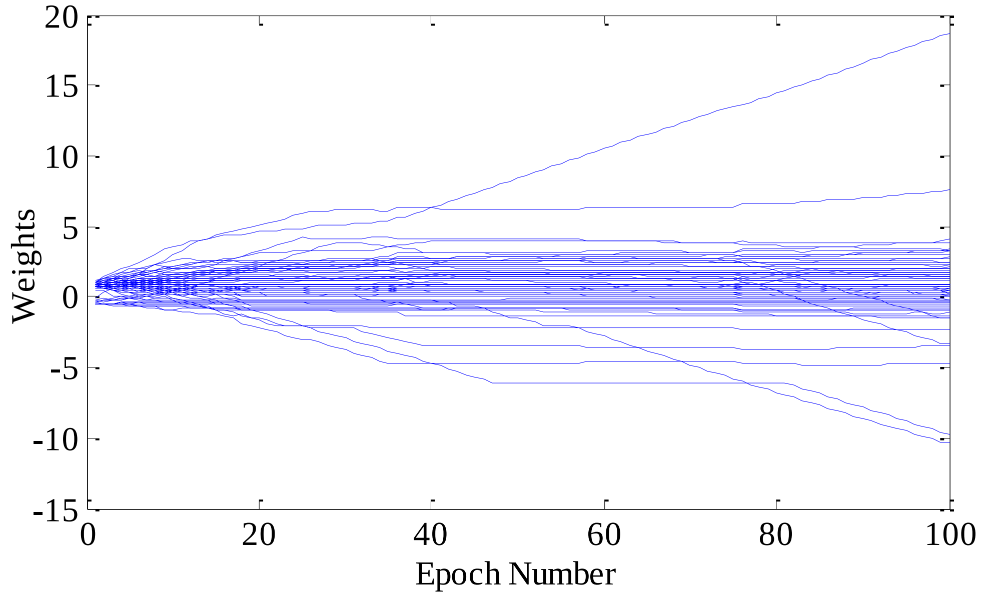


Figure 3.5 Synaptic weights during CCDS supervised learning

The learning performances of CCDS and ReSuMe are assessed in the following experiments, in which each spike train has a total time duration $T = 400ms$. At the beginning of the CCDS simulation, none of the input spike trains have delays. The same input spike train with $F_{in} = 5Hz$ and desired spike train $F_d = 100Hz$ are selected for both CCDS and ReSuMe. Each process is averaged over 20 runs. The evolution of the weights for both methods are given in Fig. 3.6(a) and Fig. 3.6(b), respectively. The weights learned by CCDS lie in the range of [-1, 1.5] which is much narrower than that of ReSuMe which has a wider range of [-10, 18] in Fig. 3.6(b). It implies that CCDS performs the same learning task better than ReSuMe, with less weight adjustments.



(a)



(b)

Figure 3.6 Comparison between CCDS and ReSuMe (a) evolution of the weights during CCDS learning; (b) evolution of the weights during ReSuMe learning.

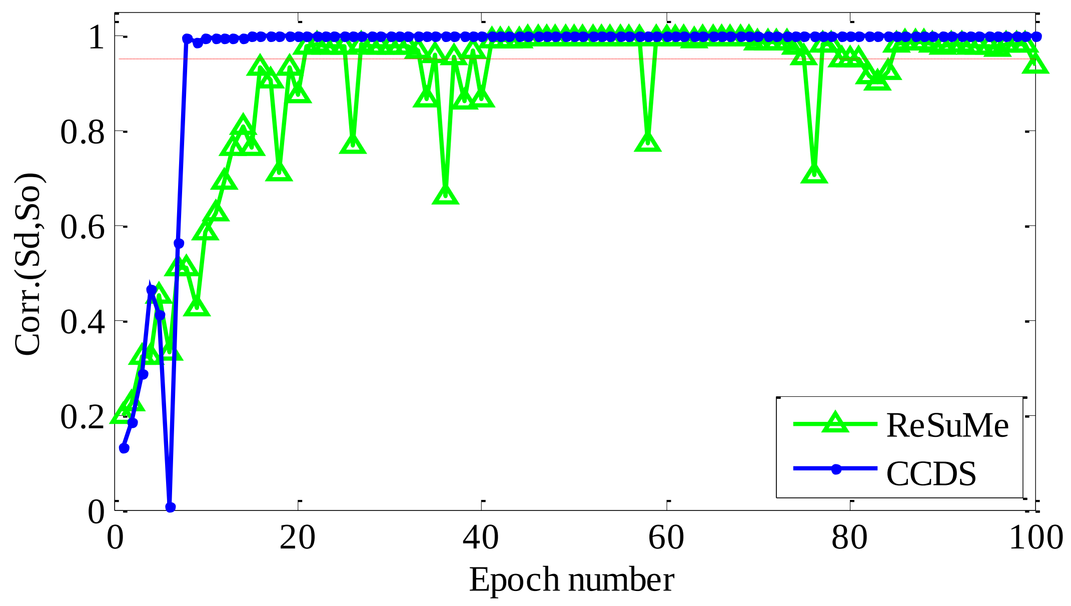
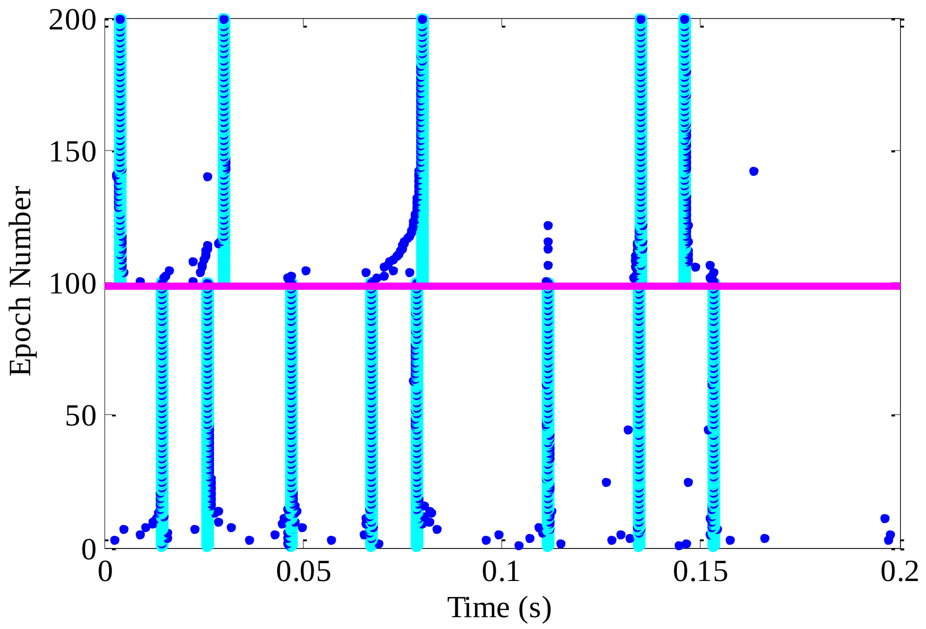


Figure 3.7 Evolution of correlated-based metric *C* for both ReSuMe and CCDS

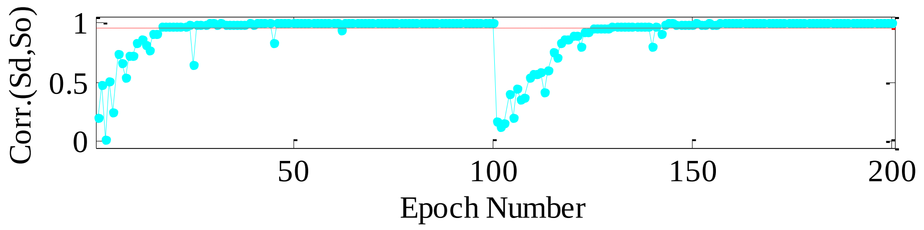
The performance of the proposed method is compared with that of ReSuMe in Fig. 3.7. These results show that the CCDS learning rule achieves high learning accuracy much faster than ReSuMe. CCDS managed to reach the satisfactory level of C>0.95 much earlier at the 8th epoch and settles on a stable set of weights thereafter. In contrast, the ReSuMe training process shows that the weights continue to adjust even after the 100th epoch.

3.3.2 Adaptive Learning Performance

In order to assess the ability of CCDS to adapt to different patterns, the adaptive learning process is evaluated. At the beginning, the neuron is trained to learn a target train as in the previous experiments. After a successful learning phase, the target spike train is changed to an arbitrarily generated train, where the precise spike time and firing rate may be different from the previous target train. We found that, we successfully train the neuron to learn the new target within several epochs using the CCDS learning rule. As shown in Fig. 3.8(a), each dot denotes a spike. At the beginning, the neuron is trained to learn one target (denoted by the cyan bar in the bottom part). After 100 epochs of learning (the dashed blue line), the target is changed to another randomly generated train (denoted by the cyan bar in the above part). Again, the neuron successfully learned the new target spike train very rapidly. Fig. 3.8(b) shows the correlated measure *C* of the new desired spike train and output spike train along the learning process.



(a)



(b)

Figure 3.8 Adaptive learning of different target trains (a) sequence learning with the changed target train; (b) Correlated-based metric *C* of target and output spike trains.

3.3.3 Generalization to Different Neuron Models

This experiment is carried out to demonstrate that the CCDS rule is independent of the neuron model. Three different neuron models are selected, the LIF neuron model, the 2-D phenomenon spiking neuron model, Izhikevich and Spike Response Model (SRM).

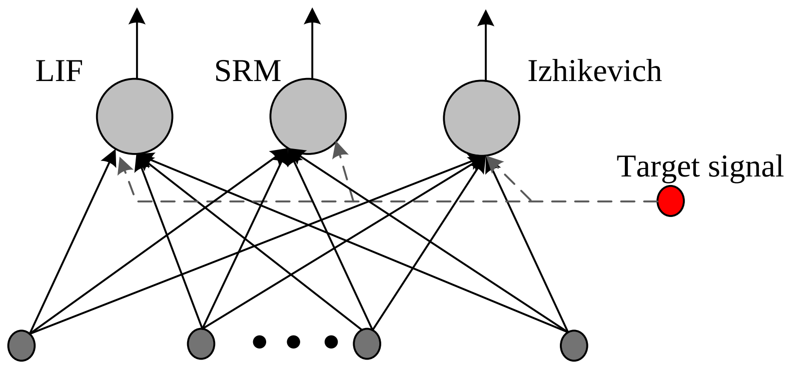


Figure 3.9 LIF neuron, SRM neuron and Izhikevich neuron are driven by the common set of presynaptic stimuli, and trained on the common target signal.

Izhikevich is known to exhibit 20 of the most prominent features of biological spiking neurons with different parameters, such as regular spiking, intrinsically bursting, fast spiking, chattering, low-threshold spiking, among other things. The parameters are chosen as $a = 0.02, b = 0.2, c = -65,$ and $d = 8$ to exhibit a Regular Spiking (RS) behavior, which is the most typical behavior in the cortex. When $a = 0.02, b = 0.2, c = -55,$ and $d = 4,$ the neurons fire a stereotypical burst of spikes followed by repetitive single spikes, which is called Intrinsically Bursting (IB).

For a fair comparison, as shown in Fig. 3.10, LIF, SRM and Izhikevich neurons are driven by the same set of presynaptic stimuli and trained on the same target signal. Except for the neuron dynamics described in (3.1) and (3.5) respectively, all other parameters are the same for these three neurons. In order to illustrate that the CCDS is not limited to homogeneous neurons, both neurons with single spiking pattern RS (Fig. 3.10b) and neurons with different kinds of spiking patterns like in the actual cortex (Fig. 3.10c) are trained. For simplification, two spiking patterns-RS and IB, are chosen for the demonstration. The input neurons are randomly selected as RS and IB, and the ratio of RS and IB is set to 1:1.

Training results are given in Fig. 3.10. After 66 epochs of training, the correlated-based measure *C* reached the criterion value, set here to 0.95. The generated sequences of spikes become highly correlated with the target pattern for these three neuron models. Although the neuron models are different, all can be trained to reproduce the target spike train successfully with the proposed CCDS learning method. A slightly faster convergence is achieved for the LIF neuron, which is caused by the simpler dynamics of LIF neuron model compared to the SRM neuron model and the Izhikevich neuron model. It is observed that the Izhikevich neuron with two spike patterns required a little more time to converge. However, comparable with the CCDS, the Izhikevich neuron with single spike pattern generated sequences of spikes which were highly correlated with the target pattern after 66 epochs. Therefore, the CCDS is proven to work well not only for homogeneous neurons in the network, but also for the network which consists of neurons having several kinds of spiking patterns like in the actual cortex.

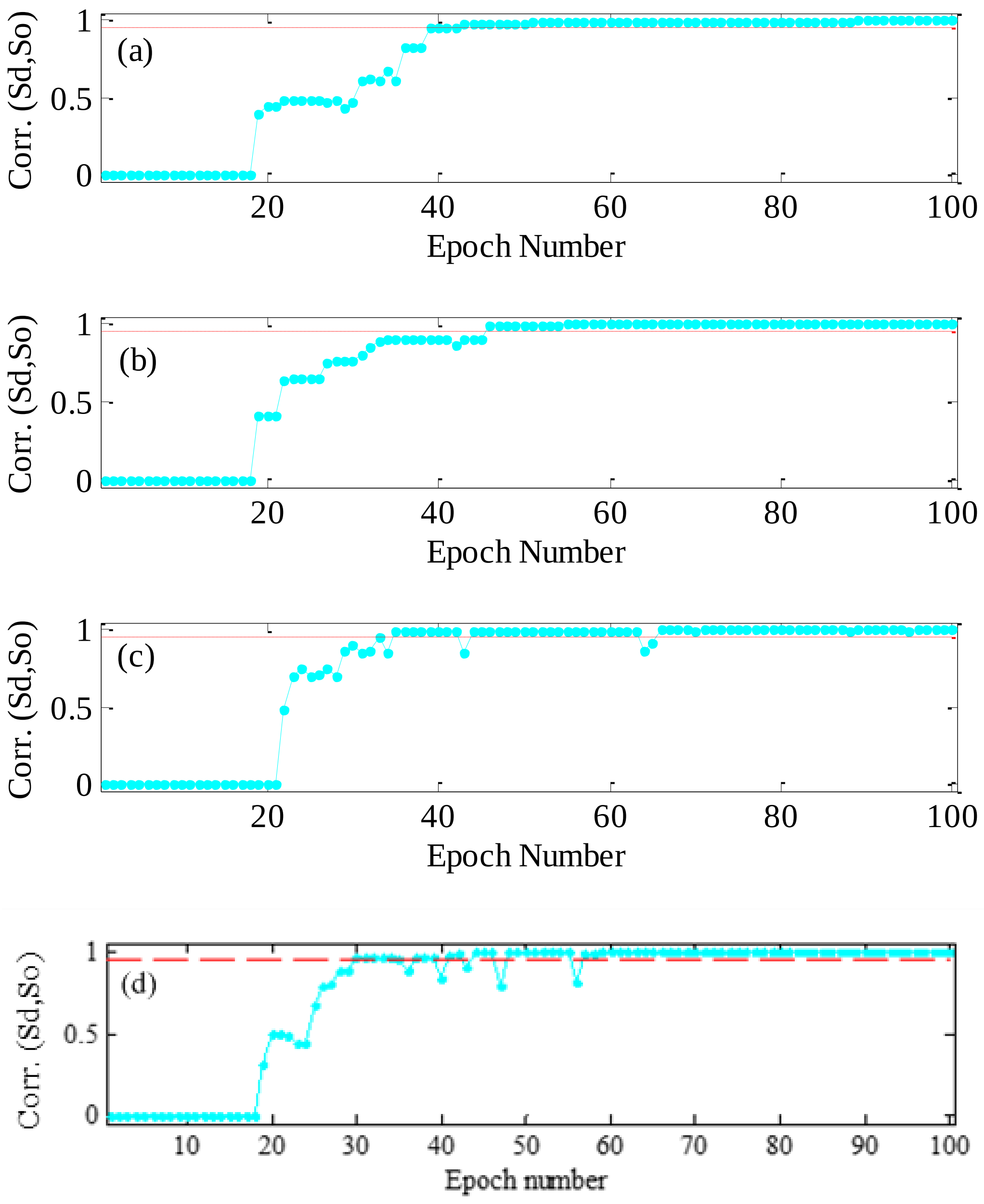


Figure 3.10 Correlated-based metric *C* of target and output spike trains with (a) LIF neuron; (b) Izhikevich neuron with single spike pattern RS; (c) Izhikevich neuron with two spike patterns RS and IB; (d) SRM neuron.

3.3.4 Learning in the Presence of Noise

In the previous experiments, the neuron is trained to learn a single pattern without the presence of noise. However, the reliability could be significantly affected by noise. In this experiment, a LIF neuron with *n=600* afferent neurons under the presence of background current noise is tested. Gaussian noise is added to the LIF neuron where $I_{ns} = 0.2$ nA. Randomly generated Poisson spike trains are used for both input and desired spike trains. As shown in Fig. 3.11, eight spike patterns still converge very rapidly. Compared with eight patterns in Fig. 3.4 without noise, the results have not been worsened by the inclusion of noise, as shown in Fig. 3.12 and Fig. 3.13.

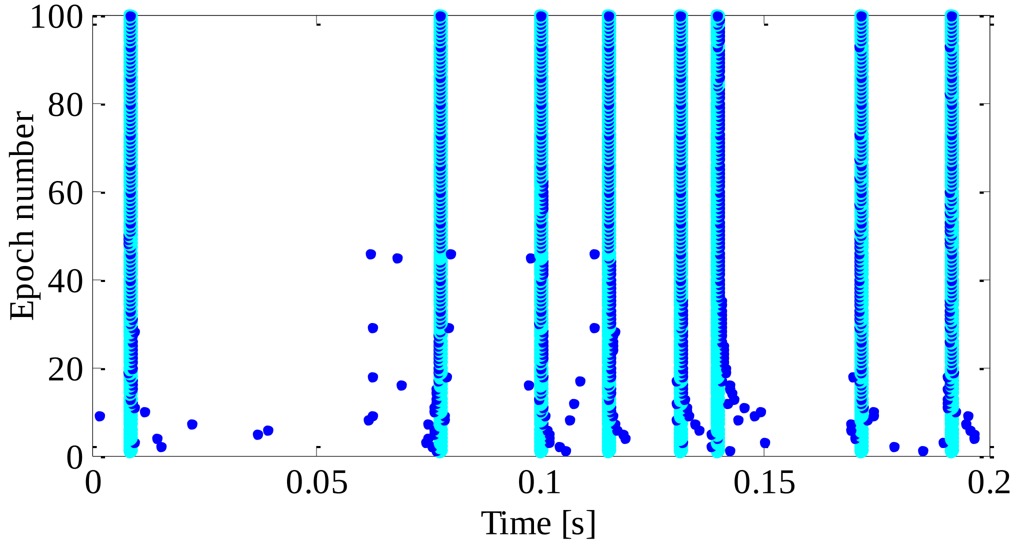


Figure 3.11 Temporal sequence learning of a typical run with noise

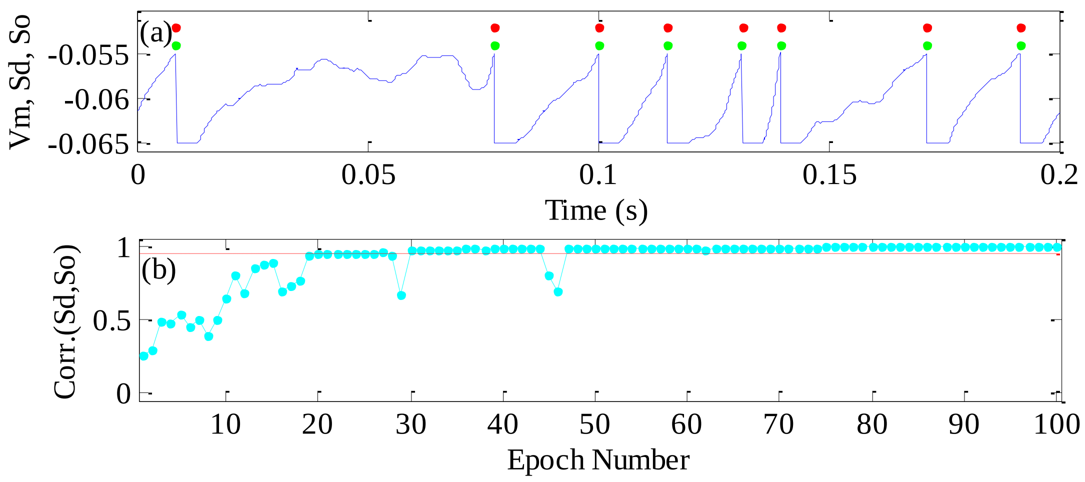


Figure 3.12 Training results with noise I_{ns}=0.2nA. (a) V_m: membrane potential after learning; red dots: target spike train S_d; green dots: actual output spike train S_o; (b) correlated-based metric C of target and output spike trains.

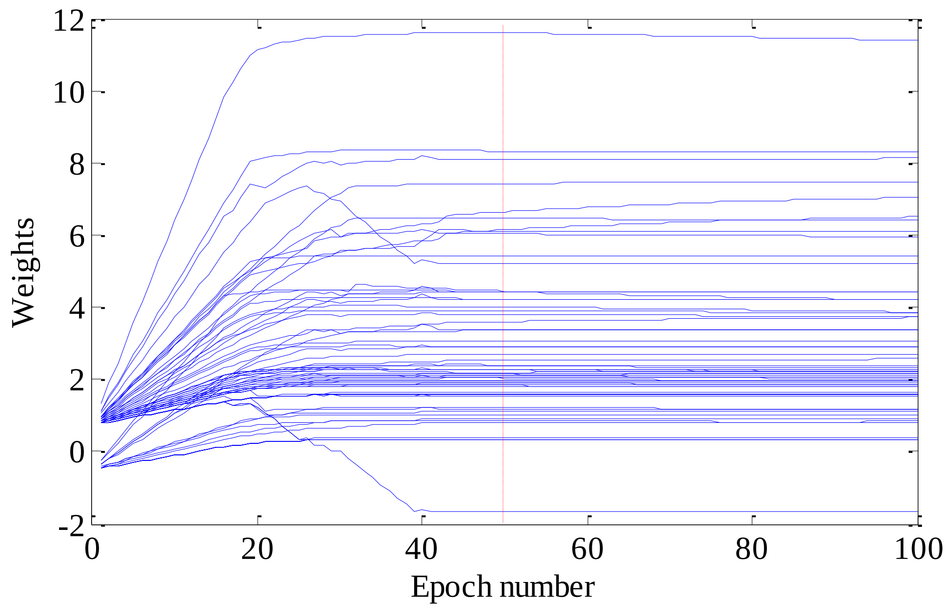


Figure 3.13 Synaptic weights during CCDS supervised learning with noise I_{ns}=0.2nA

The comparison of average learning performance (C) of CCDS and ReSuMe in the presence of different noise levels over 100 runs is given in Table 3.1. When the noise level is low, i.e. $I_{ns} = 0.1$nA, C of both methods were not influenced by the noise. Though CCDS performs slightly better than ReSuMe in the presence of noise, the correlation *C* of both methods drops almost linearly with incremental noise. Resilience to noise can be improved through training [27].

Table 3.1 Comparison of average learning performance (*C*) of CCDS and ReSuMe with different noise levels over 100 runs

I_{ns}	100th epoch		125th epoch		150th epoch	
(nA)	CCDS	ReSuMe	CCDS	ReSuMe	CCDS	ReSuMe
0.1	0.98	0.98	0.98	0.98	0.98	0.98
0.5	0.97	0.96	0.97	0.96	0.96	0.96
1.0	0.96	0.95	0.93	0.92	0.91	0.90
1.5	0.91	0.90	0.86	0.84	0.80	0.80
2.0	0.87	0.83	0.80	0.78	0.73	0.72

3.3.5 Classification of Spatiotemporal Patterns

In this experiment, the ability of the CCDS rule for spatio-temporal classification is investigated. The objective is to learn to classify three classes of input spike patterns. The pattern for each class is given as a random input spike pattern with $F_{in} = 10Hz$ homogeneous Poisson spike train, and these are fixed as templates. A Gaussian jitter with a standard deviation of 3ms is used to generate jittered patterns. Thirty copies for each of the three patterns are produced as data set, constituting $30 \times 3 = 90$ samples. Figure 3.14 illustrates some examples of spike patterns used in this experiment. The top row shows the fixed templates of three different classes. The bottom row shows one of the patterns generated for training and testing. They are generated by adding a Gaussian time jitter of 3ms standard deviation to the templates. Each dot in the figure stands for a spike. The output neuron is trained to emit a single spike at a specific time for each class. Time instances of 50, 100 and 150 ms are selected as the desired output spike time for each class, respectively, defining the three classes shown in Table 2. We allow 100 epochs for the learning method.

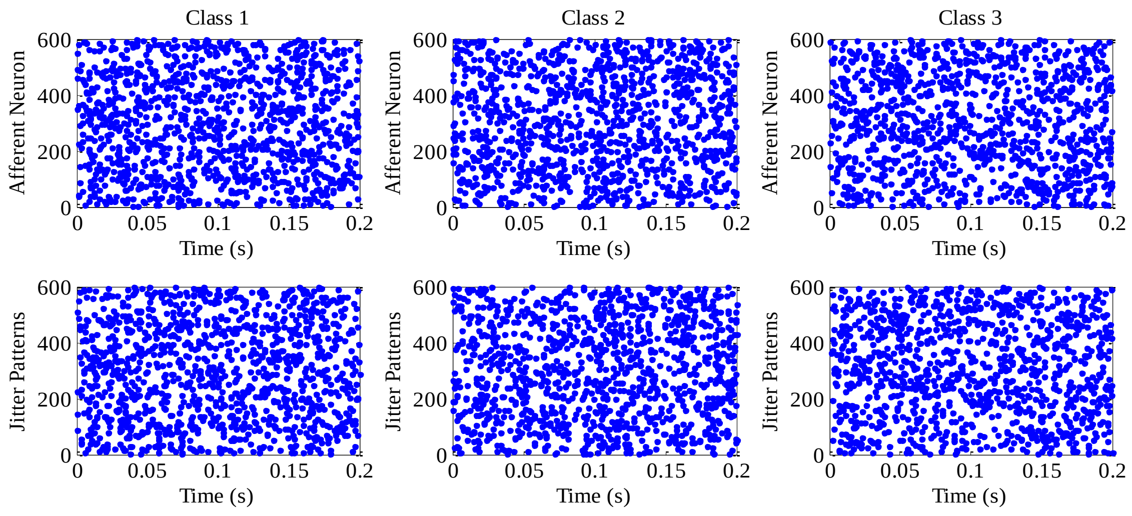


Figure 3.14 Spike pattern examples generated for training/testing

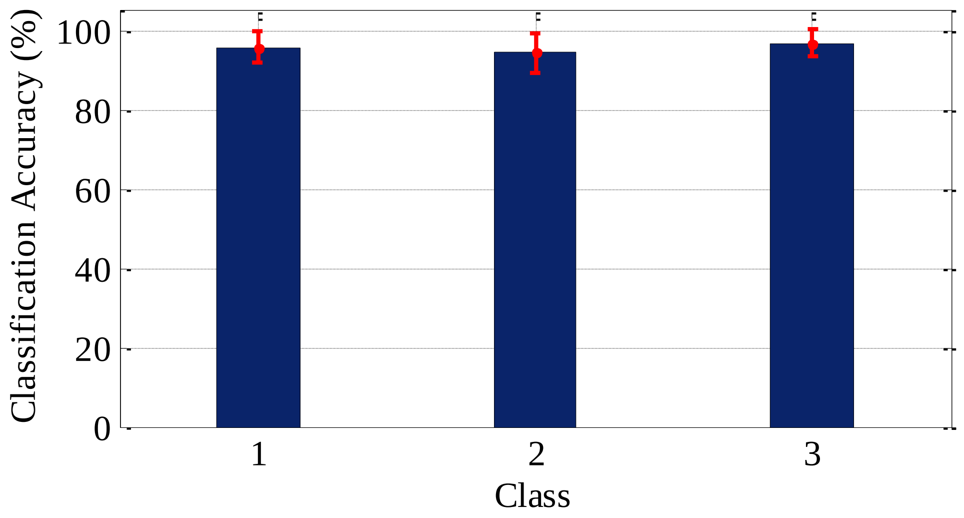


Figure 3.15 Average accuracies for the classification of spatiotemporal patterns using CCDS.

Table 3.2 Comparison of classification performance.

Classification Accuracy (%)

	Class 1		Class 2		Class 3	
Rules	Mean	S.D.	Mean	S.D.	Mean	S.D.
CCDS	96.2	3.8	94.5	5.2	97.1	3.5
SPAN	95.3	4.3	92.6	4.7	93.1	2.7
Tempotron	95.7	2.1	94.8	2.8	96.8	1.7
ReSuMe	91.4	7.2	90.2	5.8	92.1	3.8

In order to calculate the classification accuracy of the trained neuron, error metric [86] is used. If the neuron fires a single spike within $[t_d - 3ms, t_d + 3ms]$ at the desired spike time t_d, it is assumed as correctly classified. Otherwise it is considered misclassified. To avoid over fitting in the training phase and be able to estimate accurately the performance of the neural network in classifying new (unseen) data, the ten-fold cross validation process is used to measure classification accuracy. Fig. 3.15 shows the average accuracy and the standard deviation (S.D) of the 10 repetition runs for the classification of spatiotemporal patterns using the CCDS rule. Results show a mean for the classification accuracy greater or equal to 94.5%.

For comparative purposes, we include the SPAN rule [86], the Tempotron rule [84] and the ReSuMe rule to perform the same classification task. SPAN is a supervised learning method which transforms spike trains during the learning phase into analog signals so that common mathematical operations can be performed on them. These arithmetic calculations can easily reveal how networks with spiking neurons can be trained, but SPAN is not a good choice for designing networks with biological plausibility. The Tempotron rule is known as an efficient rule for spatiotemporal classification tasks. In Tempotron, neurons are trained to discriminate between two classes of spatiotemporal patterns. It is based on a gradient descent approach. The neurons could be trained to successfully distinguish two classes by firing a spike or by remaining quiescent. Since Tempotron mainly aims at decision-making tasks, it cannot memorize and generate multiple desired output spikes. In addition, the Tempotron rule is limited to a specific neuron model. Table 3.2 shows a comparative classification performance of these three different learning rules in contrast to the proposed CCDS rule. From these results, the proposed new CCDS rule is not only more biologically plausible than SPAN and Tempotron, but it also has higher classification accuracy than SPAN and ReSuMe, and is comparable with Tempotron on spatiotemporal patterns.

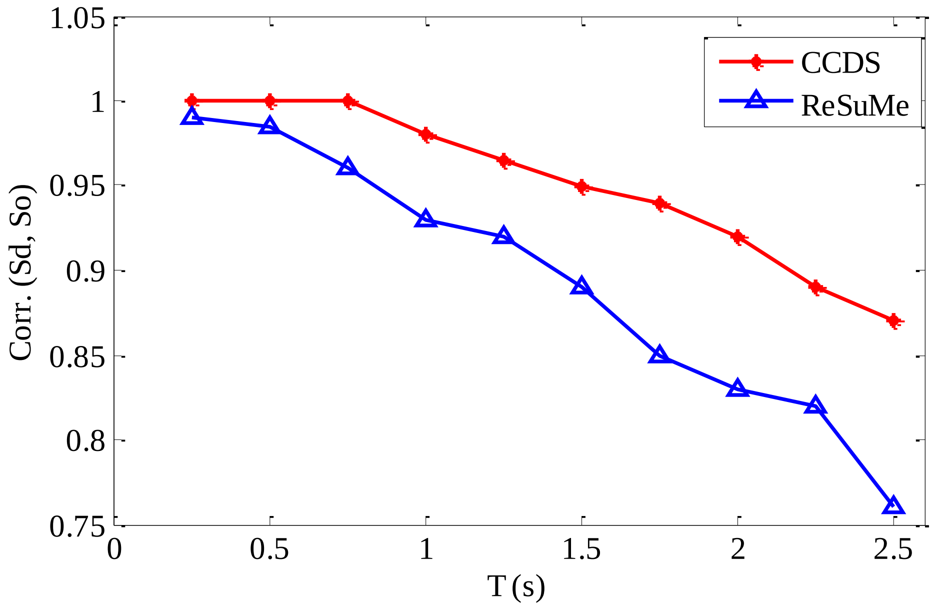
Another important property of the neuron's capacity is the maximum load (L) it can learn, which is defined as the ratio of the number of random patterns (p) that a neuron can correctly classify over the number of its synapses (N) [22]. Hence, a similar experiment as that in Ref. [22] was conducted. A number of p patterns are randomly generated as the previous experiment, where each pattern contains N spike trains. A single LIF neuron is trained to memorize all patterns using a maximum number of 200 training epochs. The learning process is considered a failure if the number of training epochs reaches 200. We run the simulation for 400, 600, 800 afferent neurons, respectively. All the data are averaged over 100 runs with different initial weights. The maximum load factors using CCDS for 400, 600, 800 synapses are 0.167, 0.151, and 0.121, respectively. The maximum load factors using ReSuMe have been examined in the similar way, and the reported L

values are 0.162, 0.141 and 0.109 for 400, 600, 800 synapses, respectively, which are a little lower than the *L* values obtained through CCDS.

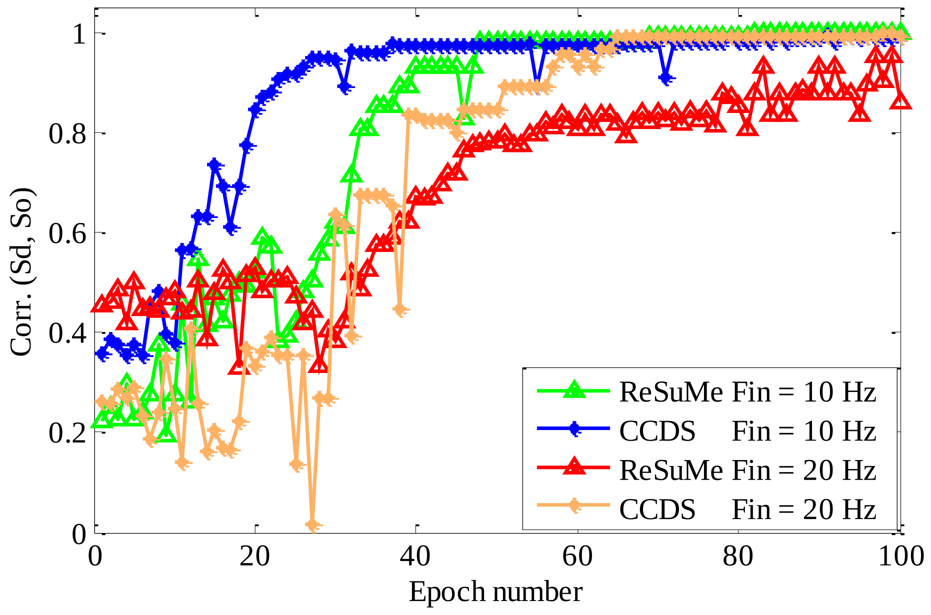
3.3.6 Effects of Learning Parameters

Two important parameters of the CCDS rule are investigated here for their effect on the learning process: the whole simulation time T and the frequency of the input spike train F_{in} . The effect of different values for the simulation times T (0.25s-2.5s) on the performance of CCDS and ReSuMe is compared in Fig. 3.16 when F_{in} = 10Hz. For each *T*, the learning is averaged over 100 runs. The mean of the correlated-based metric *C* is shown in Fig. 3.16(a). One can observe that the performance of ReSuMe drops faster when the simulation becomes longer, while CCDS performs better than ReSuMe for higher value of the simulation time. When T > 2.0s, correlated-based metric *C* of both ReSuMe and CCDS are lower than 0.9. The comparison of CCDS and ReSuMe with different input frequency is shown in Fig. 3.16(b). In this simulation, the total time duration of the spike train is set to 400ms and the output spike train is kept at 40Hz. It is observed that when the input frequency changes from 10Hz to 20Hz, the maximum value of *C* of ReSuMe decreases. It is also observed that the CCDS performance is increased when the input frequency is decreased. Meanwhile, CCDS gets a higher value of *C* faster than would ReSuMe when input spike frequencies are the same.

When the simulation time T>12s, neither ReSuMe nor CCDS do converge. This is caused by the increased number of input spikes and desired output spikes.



(a)



(b)

Figure 3.16 Comparison between CCDS and ReSuMe for (a) different simulation times between 0.25s and 2.5s when F_{in} = 10 Hz, averaged over 100 runs; (b) different input frequencies: F_{in} =10 Hz and F_{in} =20 Hz with F_{out} =40 Hz and simulation time T=400ms.

This creates inconsistencies when determining the delay learning expressed in (3.22). This results in $d_i + dt_i$ measurement that is not consistent across the desired spikes. Furthermore, when the frequency of the input spike train is larger than 80 Hz, the same situation happens. Thus, it is important to split the real-world stimuli (spikes trains) into appropriate time series segments to get the best learning performance of CCDS, since the performance drops for longer simulation times *T* and higher frequencies of the input spike train.

3.4 Discussion and Summary

In this chapter, the spatio-temporal associations of key events or patterns were investigated using the proposed CCDS training algorithm. By making use of the biological concepts of spike-timing dependent plasticity (STDP), axonal delays, and synapse delays, CCDS is able to learn the association between precise test patterns. Like ReSuMe, CCDS uses STDP to increase the PSP of the neuron at the desired spike time by increasing the weights of excitatory/inhibitory synapses that spike shortly before the desired time; CCDS also uses anti-STDP to reduce the weights of input synapses at the undesired spike time. The CCDS rule is also suitable for learning rate coded patterns. Other spatiotemporal learning algorithms, such as ReSuMe, SPAN, PSD and Chronotron cannot guarantee successful learning of arbitrary spatiotemporal spike patterns. The temporal range of desired spikes are required to be covered by the input spikes. CCDS overcomes this silent window problem: it uses delay adjustment to shift nearby input spikes to the proper time in the silent window even when no spikes exist in the input spatiotemporal pattern around the desired spike. The CCDS rule is not only suitable for a single spike code, but is amenable to multi-spike trains. The appropriate number of afferent input neurons makes the output neuron produce the desired spike train only in several epochs. When the number of spikes increases, the learning process becomes slower and is unable to converge. The results obtained confirm that the proposed method improves both the accuracy and the speed of convergence as compared to the well-known ReSuMe algorithm.

The successful learning process achieved under different spiking neuron models shows the potential for generalizations of the proposed algorithm. Its reliability is verified by reproducing the target spike trains under stochastic noisy environments. After successful training, weights and delays are kept fixed and the spiking neuron is able to reproduce the learned spatiotemporal spike pattern even in the absence of the external input, because spikes of the presynaptic neuron will stimulate the postsynaptic neuron and hence help to ‘recall’ the sequence of firing. Though CCDS outperform the ReSuMe, its performance also drops fast when the simulation time is longer. Attention in this case should be given to the selected time window to make the learning convergence more efficient. The CCDS ruls has only one layer of neurons. Extension of the structure to multiple layers could improve the performance and lead to a more biologically plausible method.

4. TEMPORAL LEARNING IN MULTILAYER SPIKING NERUAL NETWORKS

This chapter introduces a novel supervised temporal learning method for multilayer spiking neural networks. Synaptic efficacies, axonal delays and synapse delays are finely tuned for generating a desired post-synaptic firing status. The CCDS rule is extended to multiple layers, leading to a new rule termed multilayer CCDS (MutCCDS). The algorithm is benchmarked by the Iris problem, Wisconsin Breast Cancer (WBC) problem with multilayer Tempotron (MutTmptr) and multilayer ReSuMe (MutReSuMe).

4.1 Introduction

In many sensory pathways, information about the external stimuli is encoded in precise patterns of spikes. The importance of precise spike timing in neural system and cognitive information processing has been addressed in a variety of studies [102], [115]–[117]. Due to the temporal features, the spiking neural networks are more biologically plausible and computationally powerful than sigmoidal multilayer perceptron networks. Meanwhile, SNNs are widely used for the applicability for VLSI implementation with significant speed advantages [118].

Learning is the central to the exploration of intelligence to emerge, empowering living entities to perform their natural daily activities. The learning in a network of spiking neurons has attracted lots of attention from several researchers since it is considered as a realistic model in the biological neural networks. Supervised learning, contributed to the development and maintenance of a variety of brain functions, especially in sensorimotor networks and sensory systems [28], have been successfully studied for nonlinear benchmark problems. One of the most famous supervised learning rules of SNNs termed Tempotron [84], a gradient descent based approach which is efficient for binary temporal classification task, encounters two problems: (1) it does not allow multiple spikes in the output spike train, and (2) it sensitive to spike loss, in that no error gradient is defined when the neuron does not fire for any pattern, and hence will never recover. Other temporal learning rules, such as SPAN [86], PSD [22], Chronotron [87], have been developed to train neurons to generate multiple output spikes in response to a spatiotemporal stimulus. In Chronotron, both analytically-derived (E-learning) and heuristically-defined (I-learning) rules are introduced. Both the E-learning rule and the SPAN rule are based on error function that takes into account the difference between the actual output spike train and the desired spike train. Their application is therefore limited to tractable error evaluations, which are unavailable in biological neural networks and are computationally inefficient as well. The I-learning rule of Chronotron is based on a particular case of the Spike Response Model, which might have limitations for other spiking neuron models. In addition, it depends on weight initialization. Those synapses with zero initial value will not be updated according to the I-learning rule, which will lead to information loss from afferent neurons. PBSNLR [89] transforms the supervised learning into a classification problem, then solves the problem by using the perception learning rule. However, it needs lots of learning epochs to achieve good learning performance when time step is small.

Spike-Timing Dependent Plasticity (STDP) is a well-known biologically inspired plasticity process that adjusts the weights between neurons based on the relative time difference. STDP is widely used in unsupervised processes and pattern recognition [103]. However, simply implementing this form of learning will not always guarantee convergence for the network of neurons during learning as the rule does not formulate the competition between neurons. ReSuMe [27] is one of the few supervised learning algorithms that is based on a learning window concept derived from STDP. Similar to SPAN and PSD, ReSuMe is derived from the Widrow-Hoff rule [90]. It combines STDP and anti-STDP learning window under remote supervision of instruction neuron to produce a desired output spike train in response to a spatiotemporal input spike pattern. With this method, it also can reconstruct the target input-output transformations. However, the postsynaptic response to a presynaptic spike is not instantaneous. Instead, both axonal and synaptic delays contribute to the onset latency of the synaptic current with respect to the time of the presynaptic spike. Inspired by ReSuMe, the CCDS learning rule [11] is recently proposed by integrating the axonal delays and synaptic delays. Different from delay selection learning rules, the delays in CCDS are modulated together with weights instead of keeping invariant. By introducing the cross correlated term, CCDS achieves learning accuracy and learning speed improvements comparable to ReSuMe.

However, the majority of existing learning rules for the spatiotemporal spike patterns focus on training single spiking neurons or single-layer SNNs rather than multilayer networks [22], [57], [84], [86]. These learning rules are biologically plausible to some extent. Since the connections of neurons in real nervous system are extremely complex, the learning rules based on single neuron or single-layer networks are insufficient for understanding the cognitive functions of the brain. Learning rules for multilayer spiking networks are proved to be a challenge to formulate due to the discontinuous nature of spike timing of neurons. As the first supervised learning algorithm for feed-forward spiking neural networks, SpikeProp [81] only considered the first spike of each neuron. Its extension introduced in [13], [119] allows multiple spikes in the input and hidden layer, however, not in the output layer. Although single spike learning has good application capability, networks of neurons with only single spike output will have limitations in the capacity and diversity of information transmitted [120]. Multi-Spiking Neural Network (MuSpiNN) model [13] trained by Multi-SpikeProp improved the efficiency of the original single-spike SNN model by two orders of magnitude. Its applications on classification of complicated problems such as the epilepsy and seizure detection, a significantly higher classification accuracy was achieved using Multi-SpikeProp than the classification accuracy obtained using the single spiking SNN with SpikeProp. These gradient descent learning rules are effective based on the assumption that the value of the internal state of the neuron increases linearly in the infinitesimal time range around the neuronal firing time. However, this assumption limits the learning rate to a small value [121]. Also these learning rules are based on the dynamics of SRM model, which limit their generality to other neuron models. Evolutionary Strategies [122] have also been used to adjust weights and delays of a three-layer feed-forward SNN. It outperformed the SpikeProp rule in the nonlinear tasks such as XOR and Iris benchmark problems. However, evolution method is very time consuming which is not suitable for complex tasks. Sporea and Gruning [28] extended ReSuMe to multilayer by the gradient decent of the error function, proposed the Multi-layer Remote

Supervised Learning method (MutReSuMe). It combines the quality of SpikeProp with the flexibility of ReSuMe, i.e., it can be used with multiple spikes and different neuron models in multiple layers. Simulation results show that networks with hidden layers can perform nonlinear logical operations, while networks without hidden layers cannot. In [115], how pattern encodings might take place in the nervous system is represented from a systematic view. The learning rule for spiking neural networks containing hidden layers is introduced by optimizing the likelihood of generating desired output spiking patterns. It has a capability of learning a large amount of input-output spike pattern mapping, which outperforms other learning rules for SNN in terms of the number of mapping and the complexity of spiking train encodings. All of these mentioned algorithms can achieve learning, though their efficiency is much lower than that of the biological system, and do not consider the effect of learning of delays.

Here, we derive a new supervised learning rule for multilayer spiking neural networks terms Multi-CCDS. Our rule extends the single-layer CCDS learning rule to multiple layers by combining the method of gradient descent, matrix mapping with error back-propagation. The efficacy of the proposed learning rule are tested on several spike pattern transformation and classification tasks.

4.2 Multilayer Learning Rules

In this section, learning schemes for multilayer spiking neural networks are described. The spiking neuron model used in the study is introduced first. Then, the multilayer Tempotron and multilayer ReSuMe are presented sequentially. The description of multilayer CCDS is given afterwards.

4.2.1 Spiking Neuron Model

There are many spiking neuron models like Hodgkin-Huxley model, Leaky integrate-and-fire model, MAT[4] model and SRM which aim to explain the mechanism of a biological neuron. Simple phenomenon models with low computational cost are more popular for studying the learning and dynamics of spiking neural networks. The SRM treats the input spike train to produce a spike response using a simple threshold concept, i.e. when each spike arrives, a postsynaptic potential will be inducted in the neuron. After summing the effects of several incoming spikes, an output spike is triggered when the membrane potential reaches the threshold. The SRM is selected in this study as it regards as the generalization of the leaky integrate-and-fire model [32]. The membrane potential V_i of neuron N_i is a time integral over the past.

Let $F_i = \{t_i^{(f)}; 1 \le f \le n\} = \{t \mid u_i(t) = \upsilon \wedge u_i^{'}(t) > 0\}$ denote the set of all firing times of neuron N_i, and Γ_i = { $j \mid N_j$ is presynaptic to N_i } define its set of presynaptic neurons. The state $V_i(t)$ of neuron N_i at time t is given by

$$V_i(t) = \sum_{t_i^{(f)} \in F_i} \eta_i(t - t_i^{(f)}) + \sum_{j \in \Gamma_j} \sum_{t_i^{(f)} \in F_i} w_{ij} \varepsilon_{ij}(t - t_j^{(f)}), \tag{4.1}$$

where the kernel functions η_i, ε_{ij} and κ_i respectively describe the potential reset and the response to a presynaptic spike. $\varepsilon_{ij}(t-t_j^{(f)})$ is the spike response function with $\varepsilon_{ij}(t-t_j^{(f)})=0$ for $t \le t_j^{(f)}$. The times $t_j^{(f)}$ represent the firing times of neuron j. In our case, the spike response function $\varepsilon(t)$ describes a standard post-synaptic potential:

$$\varepsilon(t)=\frac{t}{\tau}\exp\left(1-\frac{t}{\tau}\right), \tag{4.2}$$

where $\tau > 0$ models the membrane potential time constant and determines rise and decay of the function.

4.2.2 Multilayer Tempotron

The Tempotron [84] is an efficient supervised learning rule allows a spiking neuron to discriminate between different categories of spike trains. The neurons could be trained to successfully distinguish two classes by firing a spike or by remaining quiescent. The Tempotron learning rule is obtained through applying the gradient descent in the space of synaptic efficacies for minimizing the following cost function:

$$C=\begin{cases} V_{thr}-V(t_{\max}), \text{if } V(t_{\max})<V_{thr}, \text{ the neuron should fire for this pattern} \\ V(t_{\max})-V_{thr}, if\ V(t_{\max})\ge V_{thr}, \text{ the neuron fired when it should be silent} \\ 0 \qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{4.3}$$

where $t_{\max}$ is the time at which the postsynaptic potential $V(t)$ reaches its maximum value. V_{thr} denotes the firing threshold.

After minimizing the above cost function, synaptic weights modified in Tempotron according to the following relation:

$$\Delta w_i = \begin{cases} \lambda \sum_{t_i^f < t_{\max}} \varepsilon(t_{\max} - t_i^f), & \text{if } \mathrm{P}^{+}\text{error} \\ -\lambda \sum_{t_i^f < t_{\max}} \varepsilon(t_{\max} - t_i^f), & \text{if } \mathrm{P}^{-}\text{error} \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

where $\lambda > 0$ is the learning rate which specifies the maximum size of the synaptic update per input spike, P^{+}error stands for the neuron should fire but it did not while P^{-}error is that the neuron should not fire but it did, respectively.

Only the direction of synaptic modification is used in the single-layer Tempotron rule while the amount of modification depends on the current Post-Synaptic Current (PSC). The multilayer Tempotron (MutTmptr) [29] is developed in the similar way as (4.5).

$$\Delta w_i = \begin{cases} \lambda I_{psc}^{i}(t_{\max} - t_i^f), & \text{if } \mathrm{P}^{+}\text{error} \\ -\lambda I_{psc}^{i}(t_{\max} - t_i^f), & \text{if } \mathrm{P}^{-}\text{error} \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

where I_{psc}^{i} is the PSC of the corresponding synapse.

The instructor signal, only contains information of modification directions, is back-propagated to the prior layer in the multilayer network. The amount of weight change depends on the corresponding PSC received by each synapse. When the output neuron fires negative patterns in MutTmptr, the LTD process will occur; when the output neuron keeps silent to a positive pattern, the LTP process will occur; no synaptic modification occurs when the output neuron fires correctly as desired.

4.2.3 Multilayer ReSuMe

The ReSuMe learning rule [57] was introduced to train a single neuron to associate input output spatiotemporal spike patterns. Although ReSuMe allows multiple spikes, it can only be applied to a single layer network or to train the readout layer of neurons in liquid state machines. Linear Poisson neuron model [28], [60] was used to analyze the relation between the input and output spike trains for the extension of ReSuMe in the multilayer SNN. The smooth firing rate $R(t)$ was used for the derivation of the learning rule, then it was replaced by the corresponding discontinuous spike train $S(t)$. The instantaneous rate function is defined as:

$$R(t) = < S(t) > = \frac{1}{M}\sum_{j=1}^{M} S_j(t) \tag{4.6}$$

where M is the number of trails and $S_j(t)$ is the spike train for each trial.

The network error is defined as the difference between the actual firing rate $R_o^a(t)$ and the desired firing rate $R_o^d(t)$ for output neurons:

$$E(t) = \frac{1}{2}\sum_{o\in O}(R_o^a(t) - R_o^d(t))^2 \tag{4.7}$$

To minimize the network error, weight modification for the output neurons can be obtained using the gradient descent and the chain rule as follows:

$$\begin{aligned}\Delta w_{oh}(t) = {} & \frac{1}{n_h} S_h(t)\left[\int_0^\infty a^{pre}[S_o^d(t) - S_o^a(t)]ds\right] \\ & + \frac{1}{n_h}[S_o^d(t) - S_o^a(t)]\left[a + \int_0^\infty a^{post}(s)S_h(t-s)]ds\right]\end{aligned} \tag{4.8}$$

where n_h is the number of hidden neurons. The integration variable s represents the time difference between the actual firing time of the output neuron and the firing time of the hidden neuron, and the desired firing time and the firing time of the hidden neuron, respectively. The kernel a^{pre} and a^{post} defined the learning window function $W(s)$ [28].

Weight modifications for the hidden neurons can be obtained in the similar way as follows:

$$\begin{aligned}\Delta w_{hi}(t) = {} & \frac{1}{n_i n_h} S_i(t)\sum_{o\in O}\left[\int_0^\infty a^{pre}[S_o^d(t) - S_o^a(t)]ds\right]w_{oh} \\ & + \frac{1}{n_i n_h}\sum_{o\in O}[S_o^d(t) - S_o^a(t)]\left[a + \int_0^\infty a^{post}(s)S_i(t-s)]ds\right]w_{oh}\end{aligned} \tag{4.9}$$

where n_i is the number of input neurons.

(4.8) and (4.9) consist the multilayer ReSuMe (MutReSuMe). By adding the hidden layer, the network is able to learn nonlinear problems and complex classification tasks, the computational power increases as well [28]. In Multilayer ReSuMe, no delay was adjusted during learning.

4.2.4 Multilayer CCDS

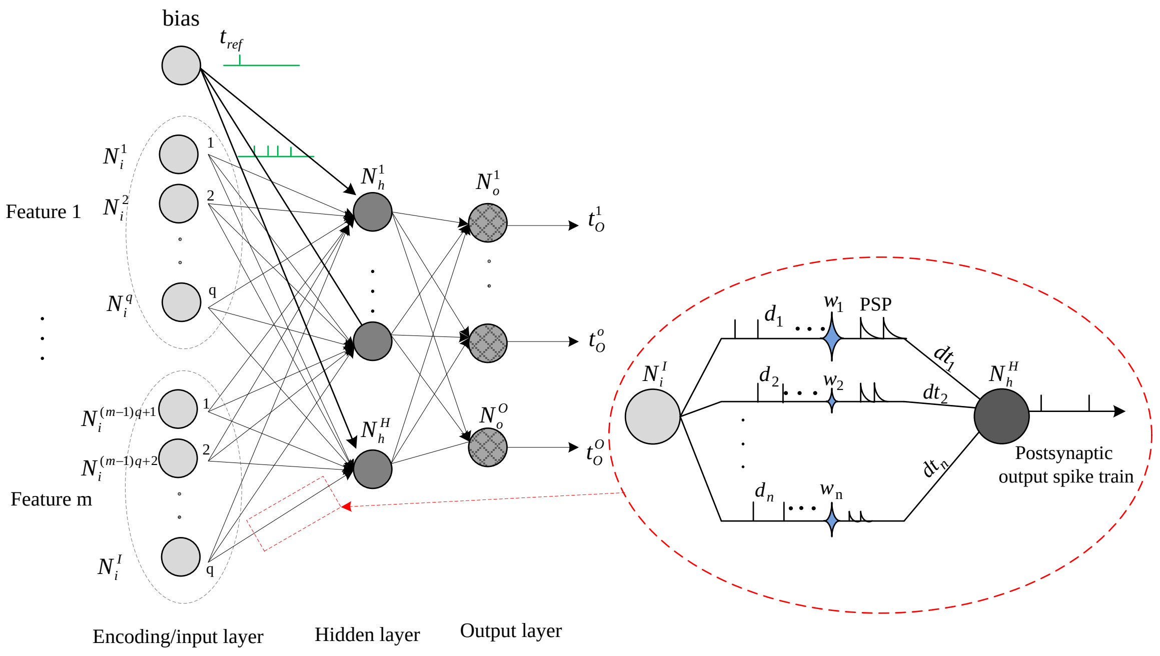


Figure 4.1 Multilayer structure of feed-forward spiking neural networks with *n* delayed axonal and synaptic connections, *I* input layer neurons, *H* hidden layer neurons and *O* output layer neurons.

Figure 4.1 shows the multilayer structure of feed-forward spiking neural networks with axonal delay $d_i, i=1,\cdots,N$ and synapse delay $dt_i, i=1,\cdots,N$. It consists of a layer of encoding neurons, a layer of hidden neurons and a layer of output neurons. The neurons in the encoding layer convert the input feature into a set of spiking times using population encoding (Gaussian receptive field encoder/square cosine encoder). Different from traditional feed-forward ANNs where two neurons are connected by one synapse only, the connection between two neurons in SNN is modeled by multiple synapses with axonal delays and synapse delays. The network is assumed to be fully connected containing a single hidden layer for simplicity in this study. The obtained algorithm can be extended to networks with multiple hidden layers similarly.

The instantaneous error is defined as the difference between actual instantaneous firing rate $R_o^a(t)$ and the desired instantaneous firing rate $R_o^d(t)$ for output neurons:

$$E(t) = E(R_o^a(t)) = \frac{1}{2}\sum_{o\in O}(R_o^a(t) - R_o^d(t))^2 \qquad (4.10)$$

According to the generalized delta learning rule, the weight adjustment is computed as

$$\Delta w_{ij}(t) = -\eta \frac{\partial E(R_o^a(t))}{\partial w_{ij}} \qquad (4.11)$$

where η is the learning rate and $\Delta w_{ij}(t)$ is the weight between presynaptic neuron i and postsynaptic neuron j.

Weights and delays can be presented as $\mathbf{w} = [w_1^1, \cdots, w_N^M]^T$ and $\mathbf{d} = [d_1 + dt_1, \cdots, d_N + dt_N]^T$, respectively. Both are the $N \times 1$ vectors. In order to minimize the error function considering the effect of delay adaptation, we map these two vectors into a higher dimension vector as:

$$\begin{aligned}
\mathbf{Z} &= [z_1, \cdots, z_p, \cdots, z_{2N}]^T = [\mathbf{w}^T, \mathbf{d}^T]^T \\
&= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \cdot \begin{bmatrix} w_1^1 \\ \vdots \\ w_p^r \\ \vdots \\ w_N^M \end{bmatrix} + \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} d_1 + dt_1 \\ \vdots \\ d_p + dt_p \\ \vdots \\ d_N + dt_N \end{bmatrix} \\
&= A\mathbf{w} + B\mathbf{d}
\end{aligned} \tag{4.12}$$

where *A* and *B* are mapping matrices.

$$z_p = \begin{cases} w_p & , p \le n \\ d_{p-n} + dt_{p-n}, & p > n \end{cases} \tag{4.13}$$

The weights and delays modification for the output neurons can be obtained through the derivation of the error function using the chain rule as follows:

$$\begin{aligned}\frac{\partial E(R_o^a(t))}{\partial z_{oh}} &= \frac{\partial E(R_o^a(t))}{\partial R_o^a(t)} \cdot \frac{\partial R_o^a(t)}{\partial w_{oh}} \cdot \frac{\partial w_{oh}}{\partial t} \cdot \frac{\partial t}{\partial z_{oh}} \\ &= \frac{1}{n_h}[S_o^d(t) - S_o^a(t)]S_h(t) \cdot \frac{\partial w_{oh}}{\partial t} \cdot \frac{1}{\partial z_{oh} / \partial t}\end{aligned} \tag{4.14}$$

Based on (3.18), (3.20)-(3.22), the weight modification can be written as (4.15).

$$\begin{aligned}\Delta w_{oh} = \frac{CC_h^r}{n_h} \Bigg\{ & S_h(t)\left[\int_0^\infty a^{pre}[S_d(t) - S_o(t)]ds\right] \\ & + [S_d(t) - S_o(t)]\left[a + \int_0^\infty a^{post}(s - d_h - dt_h) \cdot S_h(t - d_h - dt_h - s)ds\right]\Bigg\}\end{aligned} \tag{4.15}$$

where n_h is the number of hidden neurons; $S_d(t), S_o(t)$ and $S_h(t)$ stand for desired spike train, actual spike train in the output layer of neurons, and spike train in the hidden layer of neurons, respectively. The weight change is based on the CCDS rule as state in [11]. The kernel $a^{pre}(s)$ and $a^{post}(s)$ define the learning window $W(s)$ as in [28]:

$$W(s) = \begin{cases} a^{pre}(-s) = -A^- \exp(s/\tau^-), \text{if } s \le 0 \\ a^{post}(s) = A^+ \exp(-s/\tau^+), \text{if } s > 0 \end{cases} \tag{4.16}$$

where A^+, A^- are the amplitudes and τ^+, τ^- are the positive time constants of the learning window.

The axonal delay and synapse delay adaptation can be express as:

$$d_h + dt_h = \begin{cases} \dfrac{(-1)^l [S_d(t) - S_o(t)](d_k + dt_k) x_h(t)}{n_h x_k(t)}, t = t^{f_d} \\ \dfrac{(-1)^{l+1} [S_d(t) - S_o(t)](d_k + dt_k) x_h(t)}{n_h x_k(t)}, t = t^{f_a} \\ 0 \qquad\qquad\qquad\qquad , \text{otherwise} \end{cases} \tag{4.17}$$

where $d_k + dt_k = t - t_o^f = -\tau \ln(x_o(t)/A_o)$.

The weight modification for the hidden neurons can be derived as the similar way as:

$$\Delta w_{hi} = \frac{CC_i^r}{n_i n_h} \left\{ S_i(t) \sum_{o \in O} \left[\int_0^\infty a^{pre} [S_d(t) - S_o(t)] ds \right] w_{oh} \right. \\ \left. + \sum_{o \in O} [S_d(t) - S_o(t)] \left[a + \int_0^\infty a^{post}(s - d_i - dt_i) \cdot S_i(t - d_i - dt_i - s) ds \right] w_{oh} \right\} \tag{4.18}$$

The axonal delay and synapse delay adaptation for the connections between input and hidden neurons is:

$$d_i + dt_i = \begin{cases} \dfrac{(-1)^l [S_d(t) - S_o(t)](d_h + dt_h) x_h(t)}{n_i n_h x_i(t)}, t = t^{f_d} \\ \dfrac{(-1)^{l+1} [S_d(t) - S_o(t)](d_h + dt_h) x_i(t)}{n_i n_h x_i(t)}, t = t^{f_a} \\ 0 \qquad\qquad\qquad\qquad , \text{otherwise} \end{cases} \tag{4.19}$$

where n_i stands for the number of neurons in the input layer.

(4.15)-(4.19) consist the multilayer CCDS (MutCCDS) rule. It can be applied to any neuron model in principle, as the weights and delays adaptation depend only on the input, desired and output spike trains instead of the specific dynamics of the neuron model.

4.3 Results and Discussion

In this section, we used a simple three layer feed forward network to analyze the learning process of MutTmptr, MutReSuMe and our MutCCDS rule on classifying patterns. Fisher's Iris dataset and Wisconsin Breast Cancer (WBC) benchmark dataset from the UCI Machine Learning Repository [139] are selected. Our learning rule is benchmarked against several existing methods. Two measures of performance are investigated: number of convergence epochs and classification accuracy.

4.3.1 Fisher's Iris Benchmark

As one of the best know multivariate pattern recognition databases, the Fisher Iris species classification problem consists of four flower features (petal width, petal length, sepal width and sepal length) and three classes, including Iris Versicolor (class 1), Iris Virginica (class 2) and Iris Setosa (class 3) [37]. The first two classes are not linearly separable. The full dataset contains a total of 150 data samples, 50 for each species. The range for attribute 1 is [4.3, 7.9], attribute 2 is [2.0, 4.4], attribute 3 is [1.0, 7.0] and attribute 4 is [0.1, 2.5]. The features of Fisher's Iris data are real values.

Temporal receptive fields is one of the most famous population encoding schemes could also be utilized for phase encoding to improve the encoding resolution [72], [73]. As shown in Fig. 4.2, input data a is encoded into temporal spike-time patterns for the input-neurons encoding this input-variable, using multiple local Gaussian Receptive Fields (GRF). For a data range $[I^n_{\min},\cdots,I^n_{\max}]$ of a variable n,m neurons were used with Gaussian receptive files. For a neuron i its center was set to $I^n_{\min}+(2i-3)/2\cdot\{I^n_{\max}-I^n_{\min}\}/(m-2)$ and width $\sigma=1/\beta\{I^n_{\max}-I^n_{\min}\}/(m-2)$. Each feature is encoded as m spike times between 0 and 9 ms. The learning parameter β is chosen by trial and error. Here β takes value 2.

Each encoding neuron fires only once during the time encoding interval. As an example in Fig. 4.2, a real-valued feature of 5.4 is converted into spiking times through eight Gaussian Receptive Fields neurons. The encoded spiking time can be obtained from intersects points between real-valued 5.4 and eight GRFs where marked dash lines, the resulting values are 0, 0.0263, 0.8012, 0.4276, 0, 0, 0 and 0, respectively. These valued will be linearly mapped to the nearest integer in the range of [0, 9] through the following equation as [123] does:

$$t=-9*y+9 \tag{4.10}$$

where y represents the encoded value. Thus, the highest encoded value 1 response to early firing spiking time $t=0\text{ms}$ while the lowest encoded value 0 associated with late firing time $t=9\text{ms}$.

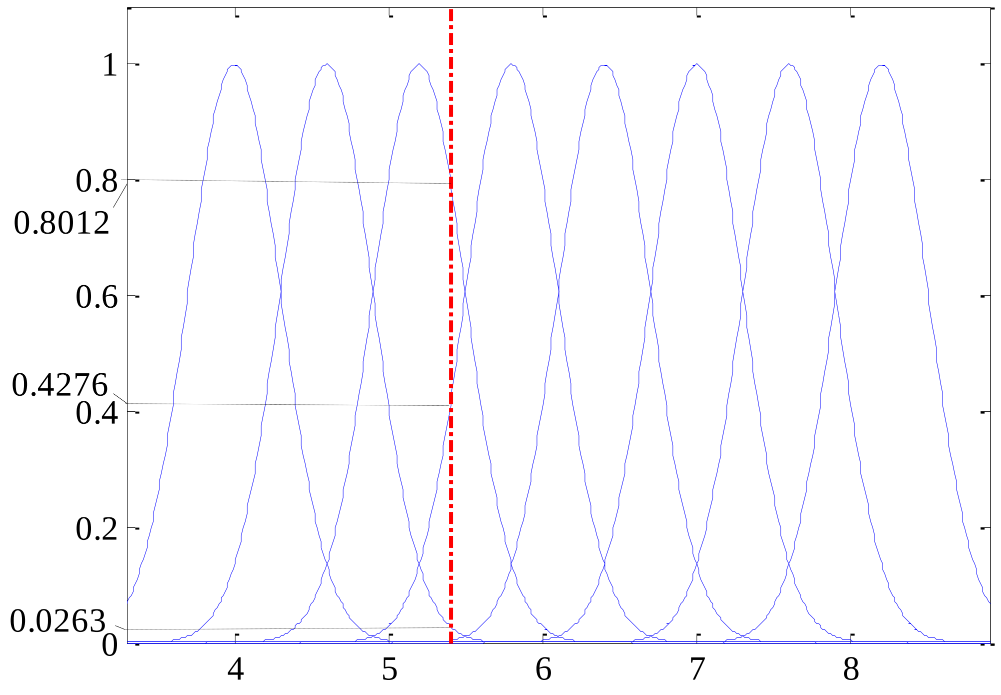


Figure 4.2 Encoding of a real-valued feature of 5.4 using Gaussian Receptive Fields neurons

As a result of population encoding, M input neurons are required per input attribute plus a bias neuron, resulting total $qM+1$ input neurons, here $q=8$. The number of hidden neurons are obtained through trail and test, 8 can get the best learning performance in the MutCCDS rule.

Table 4.1 shows the comparison results of the classification performance of different training algorithm for Iris dataset, in which I, H, O stands for number of input neurons, number of hidden neurons, and number of output neurons, receptively. It indicates MutTemptr, MutReSuMe and MutCCDS need less number of neurons for learning than BP, SpikeProp and SWAT. Especially the training methods for spiking neural networks

(SpikeProp, SWAT, MutTemptr, MutReSuMe, MutCCDS) require much less iterations to converge than the traditional BP. The learning accuracies for these methods are all acceptable, in which the learning accuracies of SpikeProp get the highest value in the testing set. As an extension of the CCDS rule, MutCCDS is not only biologically plausible, but also achieve high classification performance in terms of few number of neurons and fast convergence. Meanwhile, the MutCCDS rule and the MutReSuMe can memory and generate multiple desired output spikes while other learning rules cannot.

Table 4.1 Comparison of the classification performance of different training algorithm: result for Iris dataset

Rules	I	H	O	Iterations	Training (%)	Testing (%)
BP	50	10	3	2.6E+6	98.2 ± 0.9	95.5 ± 2.0
SpikeProp	50	10	3	1000	97.4 ± 0.1	96.1 ± 0.1
SWAT	16	208	3	500	95.5 ± 0.6	95.3 ± 3.6
MutTemptr	33	8	3	100	97.6 ± 0.3	95.8 ± 2.1
MutReSuMe	33	8	3	200	96.2 ± 0.7	94.9 ± 0.1
MutCCDS	33	8	3	120	97.1 ± 0.6	95.6 ± 0.3

4.3.2 Wisconsin Breast Cancer Benchmark

The Wisconsin breast cancer dataset are from the University of Wisconsin hospitals and consists of 699 instances divided into benign and malignant cases. Each case has 9 attributes, and each attributes is assigned an integer between 1 and 10. 16 samples out of 699 instances that have missing data have been removed in this experiment for simplicity.

Thus, the remaining 683 samples are divided into two sets, i.e., the first 455 samples as the training set while the rest 228 samples as the testing set. The same network topology and training process as in the last experiment are used. Each attribute value is encoded using spike times as the SNN inputs using 10 Gaussian receptive fields, which results 90+1=91 neurons in the encoding layer. Results gained from this experiment as in Table 4.2 shows that MutTempr, MutReSuMe and MutCCDS achieve higher than 95% classification accuracy at 100 epochs, 200 epochs and 120 epochs, respectively. They are much faster than using SWAT, SpikeProp and BP to do the same task. For this dataset, the test data accuracy is comparable to that of the other approaches.

Table 4.2 Comparison of the classification performance of different training algorithm: result for Wisconsin Breast Cancer dataset

Rules	I	H	O	Iterations	Training (%)	Testing (%)
BP	64	15	2	9.2E+6	98.1 ± 0.4	96.3 ± 0.6
SpikeProp	64	15	2	1500	97.6 ± 0.2	97.0 ± 0.6
SWAT	9	117	2	500	96.2 ± 0.4	96.7 ± 2.3
MutTemptr	91	15	2	100	97.7 ± 0.2	96.8 ± 0.3
MutReSuMe	91	15	2	200	96.8 ± 0.4	95.7 ± 0.3
MutCCDS	91	15	2	120	97.2 ± 0.3	96.5 ± 0.4

4.4 Discussion and Summary

In this chapter, the performance of the network and training algorithms were investigated using two different classification problems. As the extension of Tempotron, MutTmptr mainly aims at decision-making tasks, it is not capable of carrying out specific tasks that contains additional temporal data information. In addition, the MutTmptr rule is limited to a specific neuron model. MutReSuMe derived from biological plausible remote supervised learning rule, however, it only tunes one parameter – synaptic weight. MutCCDS is similar to standard discrete-time back-propagation, but it is derived as a functional derivative in continuous time. By considering effect of delay adaptation in learning, MutCCDS shows a little higher classification accuracy compared to MutReSuMe and achieve compatible learning performance with MutTemptr. It is also found that MutCCDS learns the Fisher Iris problem and Wisconsin Breast Cancer benchmark in one-tenth of epochs compared with SpikeProp and requires only three-fifth the number of neurons. The MutTmptr rule converges faster, while the MutCCDS rule and MutReSuMe rule give better generalization ability. The proposed multilayer learning rule shows an efficient and biologically plausible scheme, representing how synapses, axonal delays and synapse delays in the multilayer networks are adjusted to facilitate learning. Simulations in this study were conducted using Matlab and Neural Simulation Tool [124] (NEST) with custom made neuron models.

5. APPLICATION TO INTERICTAL SPIKE DETECTION

The stimuli from the real world typically have a complex statistical structure. It is quite different from the ideal case of random generated patterns. This chapter presents the application of the CCDS rule on interictal spike detection of electroencephalogram signals from epilepsy patients.

5.1 Introduction

Motivated by recent findings in biological systems, a more complex system is constructed to process real-world stimuli from a view point of temporal signal processing. This experiment evaluates the ability of CCDS to detect Interictal Spikes (IS) in scalp recorded electroencephalogram (EEG) data. EEG signals are measurements of brain neurophysiology activities, and thus serve as a fundamental way to diagnose many neurological disorders [125], among which diagnosis and prediction of seizures for patients with epilepsy is one important application of EEG signal analysis [126]. After analyzing EEG signals from epilepsy patients, researchers found that there is a special kind of transient EEG discharges, dubbed as interictal spikes, which are spikes that occur in between ictal events in patients with epilepsy. These are brief, morphologically defined events observed in the EEG of patients predisposed to seizures with focal onsets. Although the relationship between IS and epileptic seizures are not fully understood so far, neurologists believe that IS could be initiation precursor to seizures [127], or the causation of seizures in the way that they could be sufficient to induce long-term potentiation of synapse between neurons and cause excessive network synchronization [128]. The detection of interictal spikes is an essential task for 3D source localization as well as in developing algorithms for essential in seizure prediction and guided therapy. Generally, the detection of epilepsy can be achieved by visual scanning of EEG recordings for interictal and ictal activities by an expert neurophysiologist. However, visual review of vast amount of EEG data is tedious, time consuming and inefficient. In addition, disagreement among experts on a same recording are due to the subjective nature of the analysis and to the variety of interictal spikes morphology, leading to confusion in ascertaining the 3D source of seizures. Therefore, methods for effectively detecting these neural transients are desirable.

The recognition of IS from EEG recordings depends on the characters of IS, including the temporal shape, frequency features, and the synchronization and causation among multiple recording channels. Early attempts for automatic detection of IS were based on extracting peaks with certain amplitude, duration, and sharpness [129], yet such methods are not robust to learn the difference of IS shapes among different patients, nor able to suppress variance introduced by different measurement devices or environment noise. Adjouadi et al. [26] applied the discrete Walsh transform instead of commonly used continuous wavelet in the EEG signal decomposition, and designed spike duration filter mechanism together with an adaptive threshold to further increase the detection accuracy. ANN is another promising tool to detect IS. Since the neural network evolved to the third generation, the processing power of ASNN in manipulating temporal signals inspired many experiments which used ASNN to analysis raw EEG data [37], [63], [130]. Two multivariate techniques based on simulated leaky integrate-and-fire neurons were investigated for detecting and predicting seizures in non-invasive and intracranial long-term EEG recordings [131]. However, algorithms using SNN to detect special transients, such as IS, from EEG records are still rare.

5.2 Wavelet Encoding Device

One of the major challenges of applying spiking neurons/ networks to practical problems is that the proper encoding method is required for representing the external stimuli into spike trains [132], [133]. The EEG data obtained is a sequence of real-valued temporal vectors. In seeking compatibility with the spiking neurons, each real-valued input time series from one electrode is transformed into a spike train using a spike encoding method. However, how the brain encodes the information is still unclear. Many encoding mechanisms have been proposed for converting an EEG signal into spikes, such as the Bens Spiker Algorithm (BSA) [6], Hough Spiker Algorithm (HSA) [6], Threshold-Based Representation method (TBR) [134], and Wavelet Encoding Device (WED) [7]. BSA and HSA are widely used as a rate encoding method for artificial spiking neural network applications. The major problem for this type of rate encodings is that an averaging time window is required for each sampling of the input signal, which as a consequence limits the temporal resolution of the encoded signal. WED is selected due to the following advantages: (1) it is compatible with the artificial spiking neural network platform; (2) it could encode input signals online, while previous wavelet decomposition preprocessing methods are mostly off-line; and (3) it is more efficient in parallel computing implementations. As can be viewed as a subset of phase encoding scheme, WED in this study mapping the receptive field to the amplitude dimension instead of the temporal dimension, which yielded good performance for static input data.

A two-stage spike triggered modulate-and-integrate module is designed for processing the input signal, as shown in Fig. 6.1. The processing unit could decompose the input signal into a wavelet spectrum, and further encode the spectrum amplitude into the delay amounts between output spikes and the clock signals. By using the preprocessing module together with a Leaky Integrate-and-Fire (LIF) neuron, the input EEG signal could be decomposed into wavelet spectrum.

The WED model introduced in [135] was adopted here as the encoding neuron, with Mexican-hat wavelet been chosen as the encoding kernel function. As shown in [7], a single regular spike LIF neuron N_{clk} is implemented as the clock neuron, which is recursively stimulated by its own output. An initializing stimulation I_{init} is designed as a short pulse sufficient to initial the first output spike in N_{clk}, the output spikes from N_{clk} feed back to the clock neuron itself with a time delay T_{clk}, and a synapse weight sufficient to induce another output spike from N_{clk}.

This configuration ensures that the clock neuron could generate series of output spikes with intervals approximate to T_{clk}. These output clock spikes are sent to the two synaptic channels of all WEDs, with two different delays indicated by the green color and blue color in Fig. 5.1.

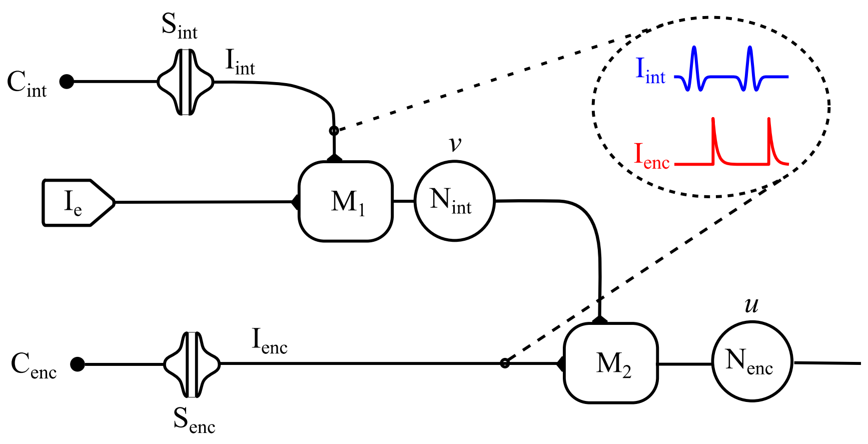


Figure 5.1 Wavelet Decomposition and Spike Phase Encoding with Two-Stage Modulate-and-Integrate Module

In reference to Figure 5.1 [7], C_{int} and C_{enc} are delay synchronized clock spikes satisfying:

$$t_l^{enc} - t_l^{\text{int}} = T_e \,, \tag{5.1}$$

where T_e is the delay phase, t_l^{int} and t_l^{enc} are time of spikes in C_{int} and C_{enc}, respectively, with $l = 1,2,\cdots,n$ being the index of each spike. The interval of spikes is T_{clk}. Terms C_{int} and C_{enc} are converted into post-synaptic current I_{enc} and I_{int}, respectively. Input signal I_e is multiplied by I_{int}, and integrated by neuron N_{int} into its state variable v. N_{enc} is a normal LIF neuron, stimulated by the absolute amplitude modulated with I_{enc}.

The overall dynamics of this encoding unit could be expressed as:

$$\tau \frac{du(t)}{dt} = -u(t) + \frac{\tau}{C_m} |\, v(t) \,|\, I_{enc}(t) \,, \tag{5.2}$$

$$a\frac{dv(t)}{dt} = I_e(t)I_{\text{int}}(t)\,, \tag{5.3}$$

where u is the state variable of N_{enc}, I_{enc} and I_{int} are summations of the post-synaptic currents of spikes in C_{enc} and C_{int} respectively, and are defined as follows:

$$I_{enc}(t) = \sum_i \exp\left(-\frac{t - t_l^{enc}}{\tau}\right) H(t - t_l^{enc}) \tag{5.4}$$

$$I_{\text{int}}(t) = \sum_l \sqrt{a}\Psi(t - t_l^{\text{int}} - d, \sigma) H(t - t_l^{\text{int}}) \tag{5.5}$$

where Ψ is a mother wavelet used as the PSC for S_{int}, a is the scale of the wavelet, σ represents the time scale of the wavelet related to the sampling frequency f_s, d is an offset parameter, and H is a Heaviside step function.

A shifted Mexican-hat mother wavelet for Ψ is selected here. Assuming the length of integration period $T_l < T_{clk}$, with $d = T_l / 2$ being assumed to make the wavelet function centered within each integration window.

Assuming $T_l < T_e < T_{clk}$, each spike in C_{enc} could reset the state variable from u to u_c for neuron N_{enc}, and (5.2) could be solved for the defined range $t_l^{enc} \le t < t_{l+1}^{enc}$ as:

$$u(\Delta t) = u_c \exp(-\Delta t / \tau) + V(\Delta t) \tag{5.6}$$

$$V(\Delta t) = \frac{\tau \Delta t}{C_m} \exp(-\Delta t / \tau) \, | X_w(t_l^{\text{int}} + T_l / 2, \sigma) | \tag{5.7}$$

where Δt is the elapsed time since the last input spike from C_{enc} arrived at the neuron. Furthermore, when $u_c < u_{th} < 0$, as long as $T_{clk} - T_e > \tau \ln(u_{th} / u_c)$, the membrane potential will exceed the threshold and emits an output spike during $\left[t_l^{enc}, t_{l+1}^{enc}\right)$. The fire delay T could then be solved from:

$$| X_w | = \frac{u_{th} C_m}{\tau T \exp(-T / \tau)} \tag{5.8}$$

where, X_w is the wavelet transform of input I_e at translation $t_l^{\text{int}} + T_l / 2$ and time scale σ.

Synapses and neurons are implemented in NEST with a single customized neuron model. In order to balance the accuracy and efficiency while solving ODEs, exponential integration method has been adopted to solve the state variable *u*, and Simpson's rule was applied to the integration for state variable *v*.

The WED model incorporates two types of spike receptors to distinguish whether a spike is send to S_{int} or S_{enc}, in the same manner as any other neuron model implemented in NEST which could receive spike input from more than one type of synapses. Input spikes with receptor type I are recognized as spikes sent to S_{int}, which could reset v_n to zero and set t^{int}

to the current time; while input spikes with receptor type II are recognized as spikes sent to S_{enc}, which in turn could reset u to u_{c} and s to zero.

A normal LIF neuron N_{clk} with an exponential decay synapse is implemented in this network as the clock generator. This LIF neuron is connected to itself with axon delay T_{clk} and synaptic efficacy large enough to generate a new output spike from itself. A short strong pulse injected to N_{clk} could initialize the first firing of N_{clk}, and generate oscillatory clock spikes at constant interval approximate to T_{clk}. These clock spikes are sent to type I receptors of WED neurons with a short delay D_0, and type II receptors with a longer delay.

Then the wavelet spectrum of input signal I_e is encoded into delay T which is the time difference between each output fire and the most recent input spike in C_{enc}.

5.3 Interictal Spike Detection

The features of interests in the EEG recordings are the interictal spikes as described in [26], which in epilepsy are a key feature used for 3D source localization of seizure onsets. The detection of interictal spikes will also help delineate EEG records that could lead to seizures [136]. Interictal spikes could be found synchrony in multiple channels between ictal events, characterized as fast EEG transients (faster than 50 ms) with steep rising and falling slopes, and habitually followed by a slow potential. This experiment evaluates the ability of CCDS to detect IS in scalp recorded EEG data. These are spikes that occur in between ictal events in patients with epilepsy. These are brief, morphologically defined events observed in the EEG of patients predisposed to seizures with focal onsets.

The experimental work of this study was approved by the Office of Research Integrity, Florida International University, Miami and the Institutional Review Board (Protocol number: IRB-052708-03). The consent forms were provided to the patients or legal representatives. Recordings were performed at Miami Children's Hospital, Miami, Florida, USA, using XLTEK Neuroworks Ver. 3.0.5 (equipment manufactured by Excel Tech Ltd. Ontario, Canada). Sampling rate of 512Hz with 0.1-70 Hz bandpass filter setting and 22 bits A/D conversion were used to obtain the digital EEG recordings. Multichannel scalp EEG signals from twenty patients with focal epilepsy were recorded using referential montage following the standard 10-20 electrode placement system. Nineteen electrodes of scalp EEG recordings were considered in this study. The raw digital EEG data used one reference electrode located in the midline of the scalp. The electrode recordings were then referenced to the average of all referential recordings. The lengths of independent sets of the EEG data recorded on the twenty pediatric patients varied from 20 min to 24h. The file segments from patients with epilepsy for training and testing were only interictal (i.e. without seizure activity), twenty minutes long, one to two hours ahead of the seizure. The number of interictal spikes in the file segments for each subject varied from 5 to 18 occurrences. EEG recordings from each patient were used as the input signal. In order to improve the signal to noise ratio (SNR), EEG data were preprocessed using standard steps before encoding.

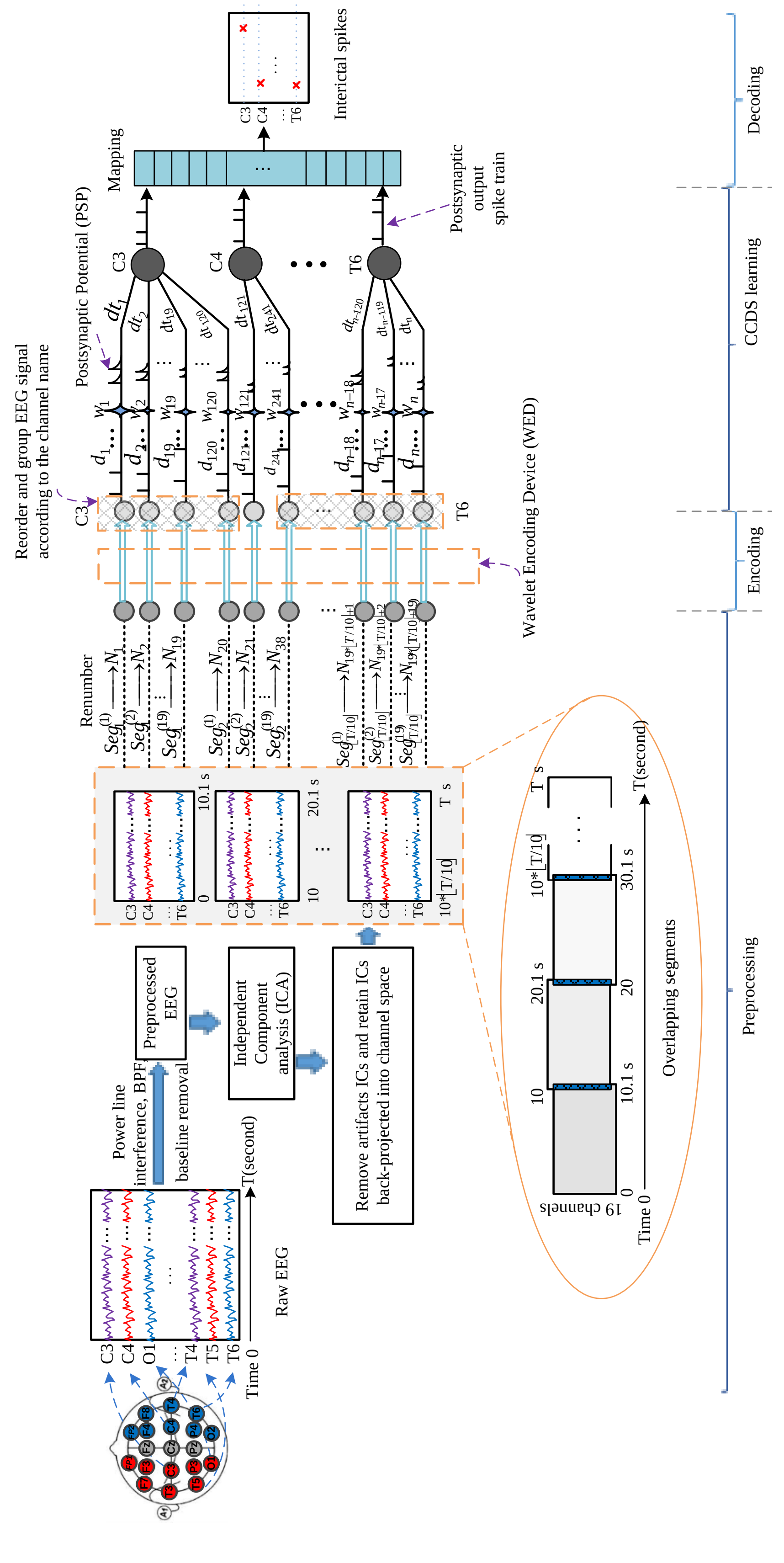
Since the shape of interictal spike is similar to the Mexican-hat wavelet mother function, WED with time scale matched to the duration of these spikes will generate much faster output spikes. We implemented 19 groups of WED arrays, each connected to one individual channel of the EEG signal. Time constants $T_{\mathrm{clk}} = 200$ ms, $D_0 = 1.0$ ms, $T_{\mathrm{i}} = 85$ ms, $T_{\mathrm{e}} = 100$ ms and $\tau = 100$ ms were used for all WEDs in this network, with σ varies between 5 ms and 70 ms.

The interictal spike detection procedure is as shown in Fig. 5.2, and consists of the following steps:

Step 1. Preprocess scalp EEG recordings using a notch filter to eliminate power line noise (60Hz and its harmonic), and an 8th order Butterworth band-pass filter with a range of [0.5, 70] Hz to avoid the influence of stochastic drifts. Baseline is also removed.

Step 2. Apply the FastICA algorithm[35] to the filtered EEG to remove the physiological artifacts, such as eye movement and eye blinks. Data are visually inspected to remove channels and data segments heavily contaminated by non-physiological artifacts, such as momentary spikes in electrode impedance, or high amplitude voltage fluctuations across all channels.

Step 3. Divide the artifacts-free EEG recordings into several 10.1s windows, overlapping by 0.1 seconds to avoid to miss interictal spikes around the cutoff point. Put all segments of each patient in order.



Raw EEG
Time 0
T(second)
C3
C4
O1
T4
T5
T6
Power line interference, BPF
baseline removal
Preprocessed EEG
Independent Component analysis (ICA)
Remove artifacts ICs and retain ICs back-projected into channel space
Overlapping segments
19 channels
0
10
10.1 s
20
20.1 s
30.1 s
10*⌊T/10⌋
T s
Renumber
Reorder and group EEG signal according to the channel name
Wavelet Encoding Device (WED)
Postsynaptic Potential (PSP)
Postsynaptic output spike train
Mapping
Interictal spikes
Preprocessing
Encoding
CCDS learning
Decoding

Figure 5.2 A schematic representation of procedures for interictal spike detection problem

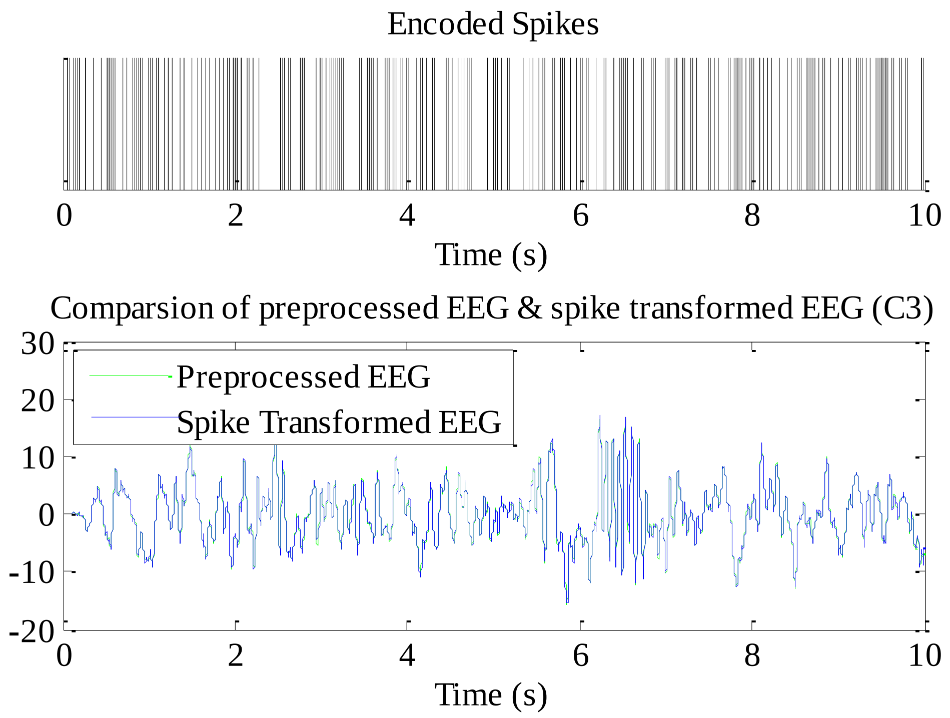


Figure 5.3 Illustration of wavelet encoding for EEG on patient 3

Step 4. In order to obtain an input compatible with the spiking neurons, each real-valued input time series, i.e. the preprocessed EEG segments in one electrode is transformed into a spike train by using the WED method.

Step 5. Put 8 patients' segments and 12 patients' segments as training set and testing set, respectively.

Step 6. Train the EEG data in the training set one by one using CCDS. The obtained weights, axonal delay and synaptic delay in the most recent training process are stored as initial values for the next training.

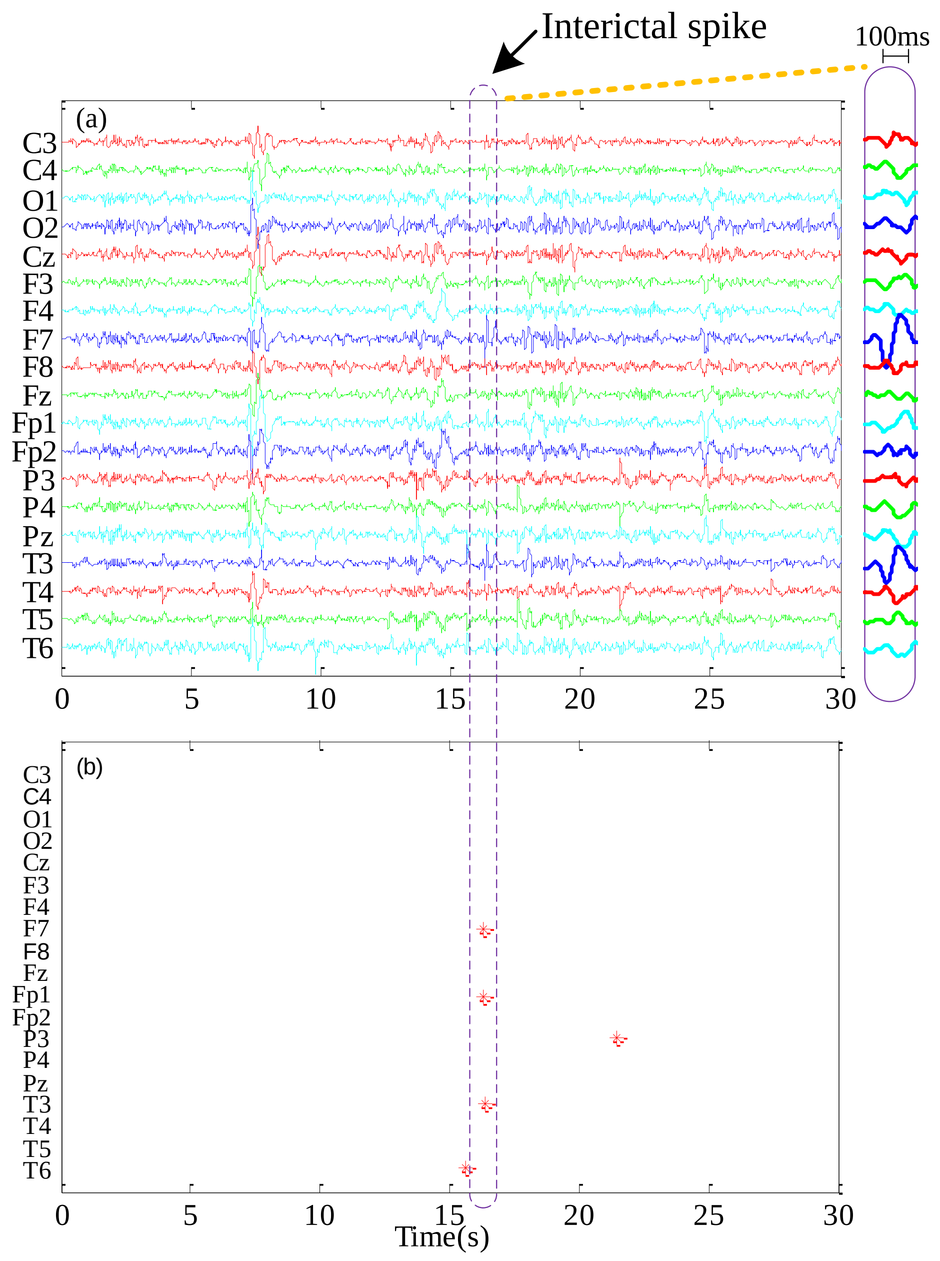


Figure 5.4 Interictal spike detection on patient 3 in the first 3 segments. (a): raster plot of processed EEG signal with spikes; and (b) detected spikes

Step 7. Stop training when correlated-based metric $C > 0.88$.

Step 8. Test EEG segments in the testing phase.

Step 9. Map the output spike train to actual spike times.

Step 10. Compare both training and testing readout results with interictal spikes marked by EEG experts according to 10 criteria characterizing interictal spikes[26] to assess the performance of the system.

The top of Fig. 5.3 is the spike representation of one channel (C3) EEG signal obtained using wavelet encoding for the duration of 10s. The bottom of Fig. 5.3 shows the actual channel of C3 of EEG signal has been superimposed with the reconstructed EEG signal from the wavelet encoded spikes. The similarity between these two signals illustrates the applicability of the wavelet encoding method.

To demonstrate the performance of CCDS on the interictal spike detection task, the first three segments of patient 3 are selected for illustrative purposes as shown in Fig. 5.4. The zoomed view of the IS at 100ms scale is illustrated on the right side of Fig. 5.4(a). Detected spikes are marked by red stars in Fig. 5.4(b). Our proposed system identified 69 spikes out of the 82 in the testing set of EEG recordings for an accuracy of 84%. It also identified 17 "spikes" that have not been annotated by expert clinicians. The Leave-One-Out-Cross-Validation (LOOCV) was performed across all subjects. The accuracy in this case improved significantly to 92.67%.

The most two common standard spike detection methods are: 1) template matching algorithm [25]; and 2) feature extraction detection algorithm [26]. For comparative purposes, the template matching algorithm in study [25], the feature extraction detection rule using Walsh transform [26] and ReSuMe have been applied to the same IS detection task. After performing the LOOCV on these three IS detection methods, the learning performance on the same encoded EEG segments is shown in Table 5.1. Observe that CCDS outperforms ReSuMe and the feature extraction method using Walsh transform for the given spike detection task. Template matching resulted in higher averaged accuracy, but required more computational time due to the exhaustive template matching process.

Table 5.1 Comparison of interictal spike detection methods on accuracy, sensitivity and specificity (%)

Rules	Accuracy	Sensitivity	Specificity
CCDS	92.67 ± 2.84	93.72 ± 2.88	92.38 ± 2.78
ReSuMe	88.42 ± 3.70	88.67 ± 3.67	86.79 ± 3.25
Template matching[25]	96.25 ± 3.24	96.70 ± 2.73	95.26 ± 2.80
Walsh transform[26]	87.62 ± 6.78	89.12 ± 6.80	86.16 ± 5.69

5.4 Discussion and Summary

Encoding of analog signals into spike trains is one of the most important steps for information processing in biological nervous systems. Our newly proposed WED encoding method incorporates the concepts of synaptic current modulation with phase encoding representation. Combining a preprocessing unit with a LIF neuron, it could perform the wavelet decomposition of the input signal, and convert the wavelet spectrum amplitude at certain translation and time scales into the output fire delay of the designed neuron.

The efficacy of the CCDS learning rule is validated through a real-world example involving the detection of interictal spikes in EEG data on patients with epilepsy. The system operates in a temporal framework, where precise timing of spikes is considered for information processing and cognitive computing. By integrating with encoding and learning function parts, it got a reasonably high detection accuracy. Simulation results show that the applied CCDS rule outperforms ReSuMe and rule-based detection method using Walsh transform for identifying these spikes.

6. CONCLUSIONS AND FUTURE WORK

In this chapter, the main results of this dissertation are summarized. Then the possible directions for future work are also discussed.

6.1 Summary

In this dissertation, we considered a problem of the supervised learning in the context of spiking neural networks. Our research was also performed on the applications of SNN to real-life tasks, such as automated detection of interictal spikes in EEG records of patients with epilepsy.

A novel temporal supervised learning rule dubbed Cross-Correlated-Delay-Shift (CCDS) learning rule was developed for learning hetero-association of spatiotemporal spike patterns in chapter 3. Various properties of the proposed learning rule were also investigated through an extensive experimental analysis. It was found that the CCDS rule could not only successfully train neurons to associate a sparse spatiotemporal pattern with a desired spike train, but also can perform classification of spatiotemporal spike patterns. The CCDS rule is both biologically plausible and computationally efficient. It was demonstrated that CCDS possesses the following properties desirable from the point of view of the consider application: 1) its ability to efficiently learn and process the sequences of spikes - it was compared with ReSuMe in Chapter 3.3.1 using the same input spike train, the result shows that CCDS outperform ReSuMe in both accuracy and converge speed; 2) its online processing ability – similar to the ReSuMe rule, synaptic weights and delays are updated incrementally each time the particular spikes in the desired spike or the actual output spike appear rather than after the whole pattern is presented. The CCDS rule converges in a few dozens of learning epochs. The online processing ability make it is possible implemented in hardware like FPGA or VLSI for real-time applications; 3) it is a local-type rule – the learning process at the particular synapse is independent on the concurrent processes at the other synapses. It make CCDS easily adapt to different network structure, especially, recurrent network and reservoir computing.

In chapter 4, a new temporal learning rule, named multilayer CCDS (MutCCDS), was proposed for multilayer spiking neural networks. It is an extension of the single layer CCDS. Correlated neurons are connected through fine-tuned weights and delays. Comparing with the multilayer Remote Supervised Method (MutReSuMe) and multilayer Tempotron rule (MutTmptr), MutCCDS shows better generalization ability and faster convergence. The proposed multilayer rules provide an efficient and biologically plausible mechanism, describing how delays and synapses in the multilayer networks are adjusted to facilitate the learning.

In chapter 5, a spiking neural network system for interictal spike detection in epilepsy patients was developed. The CCDS rule was applied and further investigated for practical applications in this study. It was found that different function parts such as encoding, learning, and decoding can be cooperate consistently within a temporal framework for a specific learning task. The results show that with proper encoding, the CCDS rule achieves good recognition performance. It outperforms ReSuMe for identifying interictal spikes.

6.2 Future Work

It still a great challenge to understand how brain processing information due to the limitation of current technology. Based on realistic mimic neuron models, SNNs has great potentials for solving complicated time-dependent pattern recognition problems defined by time series due to its inherent temporal features. However, its wide acceptance and application is limited by the taxing training time.

The whole system for processing information in spiking neural networks consists of three parts: encoding, learning and decoding. The encoding focuses on converting real-world stimuli into representations in the form of spikes. A good encoding method should retain adequate information of the original signals and facilitate the later learning process. In this study, only temporal based framework are considered since precise timing of individual spikes plays an important role in cognition computation and it has significant computational advantages over the rate based ones. Since rate coding also exists in some function of the brain, it would be valuable to explore computation considering both temporal and rate coding. As another research direction, aiming at improving the use of biologically plausible spiking neural networks for pattern recognition, it is important to investigate a proper encoding scheme. We will also search for output decoding methods for multiple spikes that yield the best classification performance. Additionally, research into optimal parameters will be another logical step for future investigations.

We can also extend our learning algorithm to other network architecture than the feed-forward structure in the future. The algorithm can in principle train all kinds of networks, including recurrent architectures and reservoir. In the traditional recurrent neural networks, it is impossible to perform the back-propagation of the error since it is unclear the state of a specific neuron is the cause or the effect of another neurons' state. This problem doesn't exist in spiking neural networks as the state of a spiking neuron can only influence the state of another neuron in the future according to its temporal feature. The network could learn to remember certain information as long as it needs by using a recurrent architecture, which is helpful to find temporal correlations on a large and variable time-scale. As an implementation of reservoir computing, liquid state machine acts as a set of filter project the low-dimensional temporal input into a high-dimensional state. Another possible future direction is to combine our learning rules with LSM. By replacing the conventional classifiers as the output layer, our approach combined with LSM would make a more biologically plausible learning rule, and hopefully result better performance.

Meanwhile, this ongoing work will further be evaluated on dynamic temporal datasets which can be obtained in dynamically environments as the ultimate goal of the proposed approach is not limited to static datasets. Future work will also examine how this algorithm scales to larger networks. Exploration of an appropriate feature selection strategy which may further enhance the performance of the proposed online supervised learning rule will also be studied.

Neuromorphic applications may also be the possible future research directions. Because of the nature of spiking neuron communication, SNNs are also suited for Very Large Scale Integration (VLSI) implementation with significant speed advantages. Meanwhile, the online processing and efficient learning abilities increase the prospects of the CCDS rule for its implementation in larger scale and real-time SNNs. We are interested in developing analog circuits to build temporal encoded Spiking neural networks and implement our learning rule, so that VLSI methods could be used to build a highly parallel neuromorphic system. As we seek to reach this implementation goal, future research work will focus on building biologically plausible SNN using the CCDS learning rule on parallel computation platforms such as the General Purpose Graphic Process Unit (GPGPU) and Field-Programmable Gate Array (FPGA) devices, and applying such ASNN to resolve a multitude of real-world problems associated with pattern recognition and pattern classification, among other things.

LIST OF REFERENCES

[1] S. Ghosh-Dastidar and H. Adeli, “Spiking Neural Networks,” *Int. J. Neural Syst.*, vol. 19, no. 4, pp. 295–308, 2009.

[2] N. Kasabov and E. Capecci, “Spiking Neural Network Methodology for Modelling, Classification and Understanding of EEG Spatio-Temporal Data Measuring Cognitive Processes,” *Inf. Sci. (Ny).*, vol. 294, pp. 565–575, 2015.

[3] E. M. Izhikevich, “Simple model of spiking neurons,” *Neural Networks, IEEE Trans.*, vol. 14, no. 6, pp. 1569–1572, 2003.

[4] R. Kkobayashi, Y. Tsubo, and S. Shinomoto, “Made-To-Order Spiking Neuron Model Equipped with a Multi-Timescale Adaptive Threshold,” *Front. Comput. Neurosci.*, vol. 3, no. 9, 2009.

[5] Z. Nadasdy, “Information Encoding and Reconstruction from the Phase of Action Potentials,” *Front. Syst. Neurosci.*, vol. 3, p. 6, 2009.

[6] B. Schrauwen and J. Van Campenhout, “BSA, A Fast and Accurate Spike Train Encoding Scheme,” in *2003 International Joint Conference on Neural Networks*, 2003, vol. 4.

[7] Z. Wang, L. Guo, and M. Adjouadi, “Wavelet Decomposition and Phase Encoding of Temporal Signals using Spiking Neurons,” *Neurocomputing*, vol. 173, pp. 1203–1210, Jan. 2016.

[8] A. Kasinski and F. Ponulak, “Comparison of Supervised Learning Methods for Spike Time Coding in Spiking Neural Networks,” *Int. J. Appl. Math. Comput. Sci.*, vol. 16, no. 1, pp. 101–113, 2006.

[9] H. Fang, J. Luo, and F. Wang, “Fast Learning in Spiking Neural Networks by Learning Rate Adaptation,” *Chinese J. Chem. Eng.*, vol. 20, no. 6, pp. 1219–1224, 2012.

[10] R. V Florian, “A Reinforcement Learning Algorithm for Spiking Neural Networks,” 2005.

[11] L. Guo, Z. Wang, M. Cabrerizo, and M. Adjouadi, "A Cross-Correlated Delay Shift Supervised Learning Method for Spiking Neurons with Application to Interictal Spike Detection in Epilepsy," *Int. J. Neural Syst.*, 2016.

[12] Y. Meng, Y. Jin, and J. Yin, "Modeling Activity-Dependent Plasticity in BCM Spiking Neural Networks with Application to Human Behavior Recognition," *IEEE Trans. Neural Networks*, vol. 22, no. 12, pp. 1952–1966, 2011.

[13] S. Ghosh-Dastidar and H. Adeli, "A New Supervised Learning Algorithm for Multiple Spiking Neural Networks with Application in Epilepsy and Seizure Detection," *Neural Networks*, vol. 22, no. 10, pp. 1419–1431, 2009.

[14] S. Y. Bonabi, H. Asgharian, R. Bakhtiari, S. Safari, and M. N. Ahmadabadi, "FPGA Implementation of a Cortical Network based on the Hodgkin-Huxley Neuron Model," *Front. Neurosci.*, vol. 8, no. November, pp. 1–12, 2014.

[15] R. Wang, G. Cohen, K. M. Stiefel, T. J. Hamilton, J. Tapson, and A. van Schaik, "An FPGA Implementation of a Polychronous Spiking Neural Network with Delay Adaptation," *Front. Neurosci.*, vol. 7, no. February, pp. 1–14, 2013.

[16] R. M. Wang, T. J. Hamilton, J. C. Tapson, and A. van Schaik, "A Neuromorphic Implementation of Multiple Spike-Timing Synaptic Plasticity Rules for Large-Scale Neural Networks," *Front. Neurosci.*, vol. 9, pp. 1–17, 2015.

[17] H. Jörntell and C. Hansel, "Synaptic Memories Upside Down: Bidirectional Plasticity at Cerebellar Parallel Fiber-Purkinje Cell Synapses," *Neuron*, vol. 52, no. 2, pp. 227–238, 2006.

[18] K. Dhoble, N. Nuntalid, G. Indiveri, and N. Kasabov, "Online Spatio-Temporal Pattern Recognition with Evolving Spiking Neural Networks Utilising Address Event Representation, Rank Order, and Temporal Spike Learning," in *The 2012 International Joint Conference on Neural Networks (IJCNN)*, 2012, pp. 1–7.

[19] Y. Zhang, P. Li, S. Member, Y. Jin, Y. Choe, and S. Member, "A Digital Liquid State Machine With Biologically Inspired Learning and Its Application to Speech Recognition," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 11, pp. 2635–2649, 2015.

[20] J. L. Rossello, V. Canals, A. Oliver, and A. Morro, “Studying the Role of Synchronized and Chaotic Spiking Neural Ensembles in Neural Information Processing,” *Int. J. Neural Syst.*, vol. 24, no. 5, p. 1430003(11 pages), 2014.

[21] S. Davies, “Learning in Spiking Neural Networks,” *Doctoral Dissertation*, 2012. [Online]. Available: http://apt.cs.manchester.ac.uk/people/daviess/thesis.pdf. [Accessed: 25-Aug-2014].

[22] Q. Yu, H. Tang, K. C. Tan, and H. Li, “Precise-Spike-Driven Synaptic Plasticity: Learning Hetero-Association of Spatiotemporal Spike Patterns,” *PLoS One*, vol. 8, no. 11, p. e78318, 2013.

[23] B. Rekabdar, M. Nicolescu, and R. Kelley, “A Biologically Inspired Approach to Learning Spatio-Temporal Patterns,” in *5th International Conference on Developement and Learning and on Epigenetic Robotics*, 2015, pp. 291–297.

[24] J. J. Wade, L. J. McDaid, J. A. Santos, and H. M. Sayers, “SWAT: A Spiking Neural Network Training Algorithm for Classification Problems,” *IEEE Trans. Neural Networks*, vol. 21, no. 11, pp. 1817–1830, 2010.

[25] P. J. Karoly, D. R. Freestone, R. Boston, D. B. Grayden, D. Himes, K. Leyde, U. Seneviratne, S. Berkovic, T. O’Brien, and M. J. Cook, “Interictal Spikes and Epileptic Seizures: Their Relationship and Underlying Rhythmicity,” *Brain*, vol. 139, no. 4, pp. 1066–1078, 2016.

[26] M. Adjouadi, D. Sanchez, M. Cabrerizo, M. Ayala, P. Jayakar, I. Yaylali, and A. Barreto, “Interictal Spike Detection Using the Walsh Transform,” *IEEE Trans. Biomed. Eng.*, vol. 51, no. 5, pp. 868–872, 2004.

[27] F. Ponulak and A. Kasinski, “Supervised Learning in Spiking Neural Networks with ReSuMe: Sequence Learning, Classification, and Spike Shifting,” *Neural Comput.*, vol. 22, pp. 467–510, 2010.

[28] I. Sporea and A. Gruning, “Supervised Learning in Multilayer Spiking Neural Networks,” *Neural Evol. Comput.*, vol. 25, no. 2, pp. 473–509, 2013.

[29] Q. Yu, “Temporal Coding and Learning in Spiking Neural Networks,” *Doctoral Dissertation*, 2014. .

[30] A. L. Hodgkin, A. F. Huxley, and B. Katz, “Measurement of current-voltage relations in the membrane of the giant axon of Loligo,” *Physiol. J.*, vol. 116, no. 4, pp. 424–448, 1952.

[31] A. N. Burkitt, “A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input,” *Biol. Cybern.*, vol. 95, no. 1, pp. 1–19, 2006.

[32] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge: Cambridge Univ. Pr., 2002.

[33] R. Brette and W. Gerstner, “Adaptive Exponential Integrate-and-Fire Model as an Effective Description of Neuronal Activity,” *Neurophysiol. J.*, vol. 94, no. 5, pp. 3637–3642, 2005.

[34] Y.-H. Liu and X.-J. Wang, “Spike-frequency adaptation of a generalized leaky integrate-and-fire model neuron,” *Comput. Neurosci. J.*, vol. 10, no. 1, pp. 25–45, 2001.

[35] G. B. Ermentrout and N. Kopell, “Parabolic bursting in an excitable system coupled with a slow oscillation,” *SIAM J. Appl. Math.*, vol. 46, no. 2, pp. 233–253, 1986.

[36] R. Naud, N. Marcille, C. Clopath, and W. Gerstner, “Firing patterns in the adaptive exponential integrate-and-fire model,” *Biol. Cybern.*, vol. 99, no. 4–5, pp. 335–347, 2008.

[37] S. Ghosh-Dastidar and H. Adeli, “Improved Spiking Neural Networks for EEG Classification and Epilepsy and Seizure Detection,” *Integr. Comput. Aided. Eng.*, vol. 14, no. 3, pp. 187–212, 2007.

[38] W. Gerstner and R. Naud, “How Good Are Neuron Models?,” *Science (80-.).*, vol. 326, no. 16, pp. 379–380, 2009.

[39] E. M. Izhikevich, “Which model to use for cortical spiking neurons?,” *IEEE Trans. Neural Networks*, vol. 15, no. 5, pp. 1063–1070, 2004.

[40] D. Yudanov and L. Reznik, “Scalable multi-precision simulation of spiking neural networks on GPU with OpenCL,” in *The 2012 International Joint Conference on Neural Networks (IJCNN)*, 2012, pp. 1–8.

[41] P. Arena, L. Fortuna, M. Frasca, and L. Patane, “Learning anticipation via spiking networks: application to navigation control,” *Neural Networks, IEEE Trans.*, vol. 20, no. 2, pp. 202–216, 2009.

[42] M. Migliore, C. Cannia, W. W. Lytton, H. Markram, and M. Hines, “Parallel network simulations with NEURON,” *Comput. Neurosci. J.*, vol. 21, pp. 119–129, 2006.

[43] T. Yamanishi, J.-Q. Liu, and H. Nishimura, “Modeling Fluctuations in Default-mode Brain Network using a Spiking Neural Network,” *Int. J. Neural Syst.*, vol. 22, no. 4, 2012.

[44] Y. Asai and A. E. P. Villa, “Integration and Transmission of Distributed Deterministic Neural Activity in Feed-Foward Networks,” *Brain Res.*, vol. 1434, no. 24, pp. 17–33, Jan. 2012.

[45] N. Fourcaud-Trocme, D. Hansel, C. Van Vreeswijk, and N. Brunel, “How Spike Generation Mechanisms Determine the Neuronal Response to Fluctuating Inputs,” *Neurosci. J.*, vol. 23, no. 37, pp. 11628–11640, 2003.

[46] E. Nichols, L. J. McDaid, and N. H. Siddique, “Case Study on a Self-Organizing Spiking Neural Network for Robot Navigation,” *Int. J. Neural Syst.*, vol. 20, no. 6, pp. 501–508, 2010.

[47] J. Iglesias and A. E. P. Villa, “Emergence of Preferred Firing Sequences in Large Spiking Neural Networks during Simulated Neuronal Development,” *Int. J. Neural Syst.*, vol. 18, no. 4, pp. 267–277, 2008.

[48] A. Mohemmed, S. Schliebs, S. Matsuda, and N. K. Kasabov, “SPAN: Spike Pattern Association Neuron for Learning Spatio-Temporal Spike Patterns,” *Int. J. Neural Syst.*, vol. 22, no. 4, p. 1250012, Aug. 2012.

[49] N. R. Luque, J. A. Garrido, R. R. Carrillo, S. Tolu, and E. Ros, “Adaptive Cerebellar Spiking Model Embedded in the Control Loop: Context Switching and Robustness Against Noise,” *Int. J. Neural Syst.*, vol. 21, no. 5, pp. 385–401, 2011.

[50] N. R. Luque, J. A. Garrido, J. Ralli, J. J. Laredo, and E. Ros, “From Sensors to Spikes: Evolving Receptive Fields to Enhance Sensorimotor Information in a Robot Arm,” *Int. J. Neural Syst.*, vol. 22, no. 4, p. 1, 2012.

[51] J.-H. Shin, D. Smith, W. Swiercz, K. Staley, J. T. Rickard, J. Montero, L. A. Kurgan, and K. J. Cios, "Recognition of partially occluded and rotated images with a network of spiking neurons," *Neural Networks, IEEE Trans.*, vol. 21, no. 11, pp. 1697–1709, 2010.

[52] Q. Yu, H. Tang, K. C. Tan, and H. Li, "Rapid Feedforward Computation by Temporal Encoding and Learning with Spiking Neurons," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 24, no. 10, pp. 1539–1552, 2013.

[53] J. Mishra, J. M. Fellous, and T. J. Sejnowski, "Selective Attention through Phase Relationship of Excitatory and InhibitoryInput Synchrony in a Model Cortical Neuron," *Neural Networks*, vol. 19, no. 9, pp. 1329–1346, 2006.

[54] M. Gilson, A. Burkitt, and J. L. van Van Hemmen, "STDP in Recurrent Neuronal Networks," *Front. Comput. Neurosci.*, vol. 4, 2010.

[55] H. Jaeger, "The 'Echo State' Approach to Analysing and Training Recurrent Neural Networks," 2001.

[56] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt, "An Experimental Unification of Reservoir Computing Methods," *Neural Networks*, vol. 20, no. 3, pp. 391–403, 2007.

[57] F. Ponulak and A. Kasinski, "Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting," *Neural Comput.*, vol. 22, no. 2, pp. 467–510, Feb. 2010.

[58] S. Panzeri, R. S. Petersen, S. R. Schultz, M. Lebedev, and M. E. Diamond, "The role of spike timing in the coding of stimulus location in rat somatosensory cortex," *Neuron*, vol. 29, no. 3, pp. 769–777, Mar. 2001.

[59] R. S. Johansson and I. Birznieks, "First spikes in ensembles of human tactile afferents code complex spatial fingertip events.," *Nat. Neurosci.*, vol. 7, no. 2, pp. 170–177, Feb. 2004.

[60] H. Sprekeler, C. Michaelis, and L. Wiskott, "Slowness: an Objective for Spike-Timing-Dependent Plasticity?," *PLoS Comput. Biol.*, vol. 3, no. 6, pp. 1136–1148, Jun. 2007.

[61] R. R. Kerr, A. N. Burkitt, D. A. Thomas, M. Gilson, and D. B. Grayden, "Delay Selection by Spike-Timing-Dependent Plasticity in Recurrent Networks of Spiking Neurons Receiving Oscillatory Inputs," *PLoS Comput. Biol.*, vol. 9, no. 2, 2013.

[62] H. De Garis, M. Korkin, F. Gers, E. Nawa, and M. Hough, "Building an Artificial Brain using an FPGA based CAM-Brain Machine," *Appl. Math. Comput.*, vol. 111, pp. 163–192, 2000.

[63] N. Nuntalid, K. Dhoble, and N. Kasabov, "EEG Classification with BSA Spike Encoding Algorithm and Evolving Probabilistic Spiking Neural Network," *Lect. Notes Comput. Sci.*, vol. 7062 LNCS, pp. 451–460, 2011.

[64] Y. Chen, J. Hu, N. K. Kasabov, Z. Hou, and L. Cheng, "NeuCubeRehab: A Pilot Study for EEG Classification in Rehabilitation Practice Based on Spiking Neural Networks," in *International Conference on Neural Information Processing*, 2013, pp. 70–77.

[65] N. Kasabov, K. Dhoble, N. Nuntalid, and G. Indiveri, "Dynamic Evolving Spiking Neural Networks for On-Line Spatio- and Spectro-Temporal Pattern Recognition," *Neural Networks*, vol. 41, no. 0, pp. 188–201, 2013.

[66] J. Lazzaro and J. Wawrzynek, "A multi-sender asynchronous extension to the AER protocol," in *Conference on Advanced Research in VLSI*, 1995, pp. 158–169.

[67] R. Serrano-Gotarredona, M. Oster, P. Lichtsteiner, A. Linares-barranco, R. Paz-vicente, F. Gómez-rodríguez, L. Camuñas-mesa, R. Berner, M. Rivas-pérez, T. Delbrück, S. Liu, R. Douglas, P. Häfliger, G. Jiménez-moreno, A. C. Ballcels, T. Serrano-gotarredona, A. J. Acosta-jiménez, and B. Linares-barranco, "CAVIAR : A 45k Neuron , 5M Synapse , 12G Connects/s AER Hardware Sensory–Processing–Learning–Actuating System for High-Speed Visual Object Recognition and Tracking," *IEEE Trans. Neural Networks*, vol. 20, no. 9, pp. 1417–1438, Sep. 2009.

[68] K. Dhoble, N. Nuntalid, G. Indiveri, and N. K. Kasabov, "Online spatio-temporal pattern recognition with evolving spiking neural networks utilising address event representation, rank order, and temporal spike learning," in *International Joint Conference on Neural Networks*, 2012, pp. 1–7.

[69] N. Kasabov, V. Feigin, Z.-G. Hou, Y. Chen, L. Liang, R. Krishnamurthi, M. Othman, and P. Parmar, "Evolving spiking neural networks for personalised modelling,

classification and prediction of spatio-temporal patterns with a case study on stroke," *Neurocomputing*, vol. 134, pp. 269–279, Jun. 2014.

[70] R. C. DeCharms, "Information Coding in the Cortex by Independent or Coordinated Populations," in *the National Academy of Sciences*, 1998, pp. 15166–15168.

[71] H. Fang, Y. Wang, and J. He, "Spiking Neural Networks for Cortical Neuronal Spike Train Decoding," *Neural Comput.*, vol. 22, no. 4, pp. 1060–1085, 2010.

[72] F. Ponulak, "Supervised Learning in Spiking Neural Networks with ReSuMe Method," *Doctoral Dissertation*, 2006. .

[73] S. G. Wysoski, L. Benuskova, and N. K. Kasabov, "Evolving spiking neural networks for audiovisual information processing," *Neural Networks*, vol. 23, no. 7, pp. 819–835, Sep. 2010.

[74] J. A. Wall, L. J. McDaid, L. P. Maguire, and T. M. McGinnity, "Spiking neural network model of sound localization using the interaural intensity difference," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 23, no. 4, pp. 574–586, Apr. 2012.

[75] T. Rumbell, S. L. Denham, and T. Wennekers, "A Spiking Self-Organizing Map Combining STDP, Oscillations, and Continuous Learning," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 25, no. 5, pp. 894–907, 2014.

[76] B. Cessac, H. Paugam-Moisy, and T. Viéville, "Overview of Facts and Issues about Neural Coding by Spikes," *J. Physiol. Paris*, vol. 104, no. 1–2, pp. 5–18, 2010.

[77] S. M. Bohte, J. N. Kok, and H. La Poutre, "Error-Backpropagation in Temporally Encoded Networks of Spiking Neurons," *Neurocomputing*, vol. 48, pp. 17–37, 2002.

[78] L. F. Abbott and S. B. Nelson, "Synaptic Plasticity: Taming the Beast," *Nat. Neurosci.*, vol. 3, pp. 1178–1183, 2000.

[79] H. Markram, J. Lübke, M. Frotscher, and B. Sakmann, "Regulation of Synaptic Efficacy by Coincidence of Postsynaptic APs and EPSPs," *Science (80-.).*, vol. 275, no. 5297, pp. 213–215, 1998.

[80] T. Natschläger and B. Ruf, “Spatial and Temporal Pattern Analysis via Spiking Neurons,” *Network*, vol. 9, no. 3, pp. 319–327, 1998.

[81] F. Ponulak and A. Kasinski, “Introduction to Spiking Neural Networks: Information Processing, Learning and Applications,” *Acta Neurobiol Exp*, vol. 71, pp. 409–433, 2011.

[82] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning Representations by Back-Propagation Errors,” *Nature*, vol. 323, pp. 533–536, 1986.

[83] Sander M. Bohte, Joost N. Kok, and J. A. La Poutre, “SpikeProp Backpropagation for Networks of Spiking Neurons,” in *ESANN 2000, 8th European Symposium on Artificial Neural Networks*, 2000.

[84] S. McKennoch, D. Liu, and L. G. Bushnell, “Fast Modifications of the SpikeProp Algorithm,” *2016 Int. Jt. Conf. Neural Networks*, pp. 3970–3977, 2006.

[85] S. B. Shrestha and Q. Song, “Adaptive Learning Rate of SpikeProp based on Weight Convergence Analysis,” *Neural Networks*, vol. 63, pp. 185–198, Dec. 2015.

[86] R. Gutig and H. Sompolinsky, “The Tempotron: a Neuron that Learns Spike Timing-Based Decisions,” *Nat. Neurosicence*, vol. 9, pp. 420–428, 2006.

[87] A. Beltreche, L. P. Maguire, M. Mcginnity, and Q. Wu, “Evolutionary Design of Spiking Neural Networks,” *New Math. Nat. Comput.*, vol. 2, no. 3, pp. 237–253, Nov. 2006.

[88] A. Mohemmed, S. Schliebs, S. Matsuda, and N. Kasabov, “SPAN: Spike Pattern Association Neuron for Learning Spatio-Temporal Spike Patterns,” *Int. J. Neural Syst.*, vol. 22, no. 4, pp. 1–17, 2012.

[89] R. V Florian, “The Chronotron: A Neuron That Learns to Fire Temporally Precise Spike Patterns,” *PLoS One*, vol. 7, no. 8, pp. 1–27, 2012.

[90] A. Carnell and D. Richardson, “Linear algebra for time series of spikes,” *Proc. Eur. Symp. Artif. Neural Networks*, no. April, p. 7, 2005.

[91] Y. Xu, X. Zeng, and S. Zhong, “A New Supervised Learning Algorithm for Spiking Neurons,” *Neural Comput.*, vol. 25, no. 6, pp. 1472–1511, 2013.

[92] B. Widrow and M. Hoff, “Adaptive switching circuits,” *1960 IRE WESCON Conv. Rec.*, no. 4, pp. 96–104, 1960.

[93] S. Deneve, “Bayesian Spiking Neurons II: Learning,” *Neural Comput.*, vol. 20, no. 1, pp. 118–145, 2007.

[94] H. Adeli and S. Ghosh-Dastidar, *Automated EEG-based Diagnosis of Neurological Disorders - Inventing the Future of Neurology*. Boca Raton, Florida: CRC Press, 2010.

[95] B. STRACK, K. M. JACOBS, and K. J. CIOS, “Simulating Vertical and Horizontal Inhibition With Short-Term Dynamics in a Multi-Column Multi-Layer Model of Neocortex,” *Int. J. Neural Syst.*, vol. 24, no. 5, p. 1440002(19 pages), 2014.

[96] Q. Yu, R. Yan, H. Tang, K. C. Tan, and H. Li, “A Spiking Neural Network System for Robust Sequence Recognition,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 27, no. 3, pp. 621–635, 2016.

[97] G. Zhang, H. Rong, F. Neri, and M. J. Perez-Jimenez, “An Optimization Spiking Neural P System for Approximately Solving Combinatorial Optimization Problems,” *Int. J. Neural Syst.*, vol. 24, no. 5, p. 1440006(16 pages), 2014.

[98] J. L. Rossello, M. L. Alomar, A. Morro, A. Oliver, and V. Canals, “High-Density Liquid-State Machine Circuitry for Time-Series Forecasting,” *Int. J. Neural Syst.*, vol. 26, no. 5, p. 1550036, 2016.

[99] Z. Wang, L. Guo, and M. Adjouadi, “A Generalized Leaky Integrate-And-Fire Neuron Model with Fast Implementation Method,” *Int. J. Neural Syst.*, vol. 24, no. 5, p. 144004, 2014.

[100] A. L. Hodgkin and A. F. Huxley, “A Quantitative Description of Membrane Current and its Applications to Conduction and Excitation in Nerve,” *J. Physiol.*, vol. 117, no. 1–2, pp. 500–544, 1952.

[101] E. M. Izhikevich, “Simple Model of Spiking Neurons,” *IEEE Trans. Neural Networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[102] S. Shapero, M. Zhu, J. Hasler, and C. Rozell, “Optimal Sparse Approximation With Integrate and Fire Neurons,” *Int. J. Neural Syst.*, vol. 24, no. 5, p. 1440001(1-16), 2014.

[103] R. P. N. Rao and T. J. Sejnowski, “Spike-Timing-Dependent Hebbian Plasticity as Temporal Difference Learning,” *Neural Comput.*, vol. 13, no. 10, pp. 2221–2237, 2001.

[104] J. Garrido, N. R. Luque, S. Tolu, and E. D’Angelo, “Oscillation-Driven Spike-Timing Dependent Plasticity Allows Multiple Overlapping Pattern Recognition in Inhibitory Interneuron Networks,” *Int. J. Neural Syst.*, vol. 26, no. 5, p. 1650020, 2016.

[105] H. Paugam-Moisy, R. Martinez, and S. Bengio, “Delay Learning and Polychronization for Reservoir Computing,” *Neurocomputing*, vol. 71, no. 7–9, pp. 1143–1158, 2008.

[106] V. N. Vapnik, “Statistical Learning Theory,” *Wiley-Interscience*, pp. 0-471-3003–1, 1989.

[107] H. A. Swadlow, “Physiological Properties of Individual Cerebral Axons Studied in Vivo for As Long As One Year,” *J. Neurophysiol.*, pp. 1346–1362, 1985.

[108] B. Glackin, J. A. Wall, T. M. McGinnity, L. P. Maguire, and L. J. McDaid, “A Spiking Neural Network Model of the Nedial Superior Olive using Spike Timing Dependent Plasticity for Sound Localization,” *Front. Comput. Neurosci.*, vol. 4, 2010.

[109] M. Gilson, M. Bürck, A. N. Burkitt, and J. L. van Hemmen, “Frequency Selectivity Emerging from Spike-Timing-Dependent Plasticity,” *Neural Comput.*, vol. 24, no. 9, pp. 2251–2279, 2012.

[110] J. W. Lin and D. S. Faber, “Modulation of Synaptic Delay during Synaptic Plasticity,” *Trends Neurosci.*, vol. 25, no. 9, pp. 449–455, 2002.

[111] P. W. Wright and J. Wiles, “Learning Transmission Delay in Spiking Neural Networks: a Novel Approach to Sequence Learning based on Spike Delay Variance,” in *WCCI 2012 IEEE World Congress on Computational Intelligence*, 2012, vol. June 10-15.

[112] T. J. Strain, L. J. McDaid, T. M. McGinnity, L. P. Maguire, and H. M. Sayers, “An STDP Training Algorithm for a Spiking Neural Network with Dynamic Threshold Neurons,” *Int. J. Neural Syst.*, vol. 20, no. 6, pp. 463–480, 2010.

[113] P. Adibi, M. R. Meybodi, and R. Safabakhsh, “Unsupervised Learning of Synaptic Delays based on Learning Automata in an RBF-like Network of Spiking Neurons for Data Clustering,” *Neurocomputing*, vol. 64, pp. 335–357, 2005.

[114] F. Ponulak, “ReSuMe-Proof of convergence (Tech. Rep.),” 2006.

[115] S. Schreiber, J. M. Fellous, D. Whitmer, P. Tiesinga, and T. J. Sejnowski, “A New Correlation-Based Measure of Spike Timing Reliability,” *Neurocomputing*, vol. 52–54, pp. 925–931, 2003.

[116] B. Gardner and I. Sporea, “Encoding Spike Patterns in Multilayer Spiking Neural Networks,” *Neural Evol. Comput.*, pp. 1–31, 2015.

[117] N. Caporale and Y. Dan, “Spike Timing–Dependent Plasticity: A Hebbian Learning Rule,” *Annu. Rev. Neurosci.*, vol. 31, no. 1, pp. 25–46, 2008.

[118] J.-M. P. Franosch, S. Urban, and J. L. van Hemmen, “Supervised Spike-Timing-Dependent Plasticity: A Spatiotemporal Neuronal Learning Rule for Function Approximation and Decisions,” *Neural Computation*, 2013. .

[119] F. Ponulak, “Supervised Learning in Spiking Neural Networks,” *Neuromorphic Eng.*, 2012.

[120] O. Booij and N. T. Hieu, “A Gradient Descent Rule for Spiking Neurons Emitting Multiple Spikes,” *Inf. Process. Lett.*, vol. 95, no. 6, pp. 552–558, 2005.

[121] M. Zhang, H. Qu, J. Li, and X. Xie, “A New Supervised Learning Algorithm for Spiking Neurons,” in *the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems*, 2015, vol. 1.

[122] B. Gardner, I. Sporea, and A. Gruning, "Learning Spatiotemporally Encoded Pattern Transformations in Structured Spiking Neural Networks," *Neural Comput.*, vol. 27, pp. 2548–2586, 2015.

[123] A. Belatreche, L. P. Maguire, M. Mcginnity, and Q. X. Wu, "An Evolutionary Strategy for Supervised Training of Biologically Plausible Neural Networks," in *Proceedings of the Sixth International Conference on Computational Intelligence and Natural Computing*, 2003, pp. 1524–1527.

[124] Q. X. Wu, T. M. McGinnity, L. P. Maguire, B. Glackin, and A. Belatreche, "Learning under Weight Constraints in Networks of Temporal Encoding Spiking Neurons," *Neurocomputing*, vol. 69, no. 16–18, pp. 1912–1922, Oct. 2006.

[125] C. Koch and I. Segev, "The role of single neurons in information processing.," *Nat. Neurosci.*, vol. 3, pp. 1171–1177, Nov. 2000.

[126] M.-O. Gewaltig and M. Diesmann, "NEST (NEural Simulation Tool)," *Scholarpedia*, vol. 2, no. 4, p. 1430, Apr. 2007.

[127] S. Sanei and J. A. Chambers, *EEG Signal Processing*. England, UK: John Wiley & Sons, Ltd, 2007.

[128] S. Noachtar and J. Rémi, "The role of EEG in epilepsy: A critical review," *Epilepsy Behav.*, vol. 15, no. 1, pp. 22–33, 2009.

[129] A. Cukiert, J. A. Buratini, E. Machado, A. Sousa, J. O. Vieira, M. Argentoni, C. Forster, and C. Baldauf, "Results of surgery in patients with refractory extratemporal epilepsy with normal or nonlocalizing magnetic resonance findings investigated with subdural grids," *Epilepsia*, vol. 42, no. 7, pp. 889–894, 2001.

[130] J. S. Bains, J. M. Longacher, and K. J. Staley, "Reciprocal interactions between CA3 network activity and strength of recurrent collateral synapses.," *Nat. Neurosci.*, vol. 2, no. 8, pp. 720–726, 1999.

[131] J. Gotman and P. Gloor, "Automatic recognition and quantification of interictal epileptic activity in the human scalp EEG," *Electroencephalogr. Clin. Neurophysiol.*, vol. 41, no. 5, pp. 513–529, Nov. 1976.

[132] S. Schliebs, H. N. A. Hamed, and N. K. Kasabov, “Reservoir-Based Evolving Spiking Neural Network for Spatio-temporal Pattern Recognition,” in *ICONIP*, 2011, pp. 160–168.

[133] A. Schad, K. Schindler, B. Schelter, T. Maiwald, A. Brandt, J. Timmer, and A. Schulze-Bonhage, “Application of a Multivariate Seizure Detection and Prediction Method to Non-Invasive and Intracranial Long-Term EEG Recordings,” *Clin. Neurophysiol.*, vol. 119, no. 1, pp. 197–211, 2008.

[134] S. Soltic and N. Kasabov, “Knowledge Extraction from Evolving Spiking Neural Networks with Rank Order Population Coding,” *Int. J. Neural Syst.*, vol. 20, no. 6, pp. 437–445, 2010.

[135] J. Friedrich, R. Urbanczik, and W. Senn, “Code-Specific Learning Rules Improve Action Selection By Populations of Spiking Neurons,” *Int. J. Neural Syst.*, vol. 24, no. 1, p. 1450002(1-16), 2014.

[136] M. G. Doborjeh, G. Wang, N. Kasabov, R. Kydd, and B. R. Russell, “A Spiking Neural Network Methodology and System for Learning and Comparative Analysis of EEG Data from Healthy versus Addiction Treated versus Addiction Not Treated Subjects,” *IEEE Trans. Biomed. Eng.*, vol. 63, no. 9, pp. 1830–1841, 2016.

[137] Z. Wang, “System Design and Implementation of A Fast and Accurate Bio-Inspired Spiking Neural Network,” Florida International University, 2015.

[138] M. Cabrerizo, M. Ayala, M. Goryawala, P. Jayakar, and M. Adjouadi, “A new parametric feature descriptor for the classification of epileptic and control EEG records in pediatric population.,” *Int. J. Neural Syst.*, vol. 22, no. 2, p. 1250001, Apr. 2012.

[139] UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/

APPENDICES

Appendix A. Wavelet Encode Device (WED)

Since LIF model is a reasonable simplification of biological neuron with balanced accuracy and efficiency, LIF is selected for the explanation of Wavelet Encode Device (WED). LIF spiking neuron is described by following one-dimensional ordinary differential equations:

$$\tau \frac{du(t)}{dt} = -u(t) + \frac{\tau}{C_{\mathrm{m}}} I_{\mathrm{all}}(t) \tag{A.1}$$

$$\text{if } u = u_{\mathrm{th}} \text{ and } \frac{du(t)}{dt} > 0,\ u \leftarrow u_{\mathrm{c}} \tag{A.2}$$

where u is the membrane potential, τ and C_{m} are the time constant and capacitance of the neuron, respectively, with I_{all} defining the overall afferent current. When u reaches the firing threshold u_{th} and the derivative of u takes positive value, post-fire potential will set to u_{c}. Such derivative condition ensures that the neuron only fires when its membrane potential in an upward trend crosses the threshold, thus it avoid accidental fires if the resting potential of the neuron is higher than its firing threshold.

The stimulation to LIF neuron can be written as:

$$I_{\mathrm{all}}(t) = I_{\mathrm{e}}(t) + \sum_{j} w_j I_{\mathrm{s}}(t - s_j)\,, \tag{A.3}$$

where $I_e(t)$ is the external current, $I_s(t)$ is the shape function of the post-synaptic current (PSC), s_j is the time that the j-th spike arrives at the synapse, and w_j is the connection efficacy corresponding to the j-th input spike.

Although linear summation of synaptic currents and external current as performed in [1] has been widely accepted as a simplified relationship among the afferent stimulations in large-scale artificial spiking neural networks, the interaction between post-synaptic currents was found to be more complicated in biological nervous system. Two-stage modulate-and-integrate module (Fig. 6.1), where the multiplication is performed instead of summation between the input signal and synaptic currents. The first stage of the module incorporates the integration of the multiplication of external current and a wavelet shape synaptic current, while the second stage modulate the output from first stage with an exponential decay synaptic current. By using the preprocessing module together with a Leaky Integrate-and-Fire (LIF) neuron, input EEG signal could be decomposed into wavelet spectrum, and such spectrum amplitude could be encoded into synchronized spike trains.

Overall dynamics of the encoding unit could be expressed as

$$\tau \frac{du(t)}{dt} = -u(t) + \frac{\tau}{C_m} |v(t)| I_{enc}(t) \tag{A.4}$$

$$a \frac{dv(t)}{dt} = I_e(t) I_{\text{int}}(t), \tag{A.5}$$

where u is the state variable of N_{enc}, I_{enc} and I_{int} are summations of the post-synaptic currents of spikes in C_{enc}, and C_{int} respectively, and are defined as follows:

$$I_{enc}(t) = \sum_i \exp\left(-\frac{t - t_l^{enc}}{\tau}\right) H(t - t_l^{enc}) \tag{A.6}$$

$$I_{\text{int}}(t) = \sum_l \sqrt{a}\,\Psi(t - t_l^{\text{int}} - d, \sigma) H(t - t_l^{\text{int}}) \tag{A.7}$$

where Ψ is a mother wavelet used as the PSC for S_{int}, a is the scale of the wavelet, σ represents the time scale of the wavelet related to the sampling frequency f_s, d is an offset parameter, and H is a Heaviside step function.

A shifted Mexican-hat wavelet mother function for Ψ is selected as follows:

$$\Psi(t,\sigma) = \frac{2}{\sqrt{3}\pi^{1/4}}\left(1 - \frac{t^2}{\sigma^2}\right)\exp\left[-\frac{t^2}{2\sigma^2}\right] \tag{A.8}$$

Assuming that the length of integration period *T*l satisfies $T_l < T_{clk}$, we could define *d* = *T*l / 2 in (A.7) so that the wavelet function is centered within each integration window. Note that both *I*enc and *I*int are constructed in a unitless manner for the model simplification.

Suppose each spike in C_{int} could reset the state variable v of neuron N_{int} to zero, assuming $\sigma << T_{clk}$ and $\sigma << T_i$, and $\Psi(t,\sigma) \to 0$ when $t < 0$ or $t > T_i$, then (A.5) could be resolved as follows:

$$\begin{aligned} v(t) &= \frac{1}{\sqrt{a}}\int_{-\infty}^{\infty} I_e(\zeta)\Psi(\zeta - t_l^{\text{int}} - T_l/2, \sigma)d\zeta \\ &= X_w(t_l^{\text{int}} + T_l/2, \sigma) \end{aligned} \tag{A.9}$$

where X_w is the wavelet transform of input I_e at translation $t_l^{\text{int}} + T_l / 2$ and time scale σ.

Assuming $T_l < T_e < T_{clk}$, each spike in C_{enc} could reset the state variable from u to u_c for neuron N_{enc}, and (A.4)-(A.5) could be solved for the defined range $t_l^{enc} \le t < t_{l+1}^{enc}$ as:

$$u(\Delta t) = u_c \exp(-\Delta t / \tau) + V(\Delta t) \tag{A.10}$$

$$V(\Delta t) = \frac{\tau \Delta t}{C_m} \exp(-\Delta t / \tau) \,|\, X_w(t_l^{\text{int}} + T_l / 2, \sigma) \,| \tag{A.11}$$

where Δt is the elapsed time since last input spike from C_{enc} arrived at the neuron. Note that the absolute value operation applied to v makes $V(\Delta t)$ a function of the absolute spectrum of the wavelet transform X_w. The absolute spectrum is preferable to power spectrum of the wavelet transform, in the sense that it ensures that the units in equation (A.10) are balanced without need for extra constants.

Furthermore, when $u_c < u_{th} < 0$, as long as $T_{clk} - T_e > \tau \ln(u_{th} / u_c)$, the membrane potential will exceed the threshold and emits an output spike during $[t_l^{enc}, t_{l+1}^{enc})$. The fire delay T could then be solved from:

$$| X_w | = \frac{C_m}{\tau T} [u_{th} \exp(T / \tau) - u_c] \tag{A.12}$$

When $u_c = 0$ and $u_{th} > 0$, considering that $V(\Delta t)$ is a bell function which reaches its maximum when $\Delta t = \tau$, the membrane potential will exceed the threshold only if $| X_w | \ge X_{th} = \dfrac{u_{th} C_m}{e \tau^2}$, and fire delay T could be solved from:

$$|X_w| = \frac{u_{th} C_m}{\tau T \exp(-T/\tau)} \tag{A.13}$$

Note that T is always less than τ in (A.13), which ensures that T is a monotonic decreasing function of $|X_{\mathrm{w}}|$ when the amplitude spectrum $|X_{\mathrm{w}}|$ is larger than the threshold X_{th}. If the wavelet spectrum amplitude is smaller than X_{th}, the LIF neuron N_{enc} will not fire during $\left[t_i^{\mathrm{enc}}, t_{i+1}^{\mathrm{enc}}\right)$.

Then the wavelet spectrum of input signal I_e is encoded into delay T which is the time difference between each output fire and the most recent input spike in C_{enc}. Larger wavelet spectrum amplitude corresponds to faster firing after each clock spike.

VITA

LILIN GUO

2006	B.S., Information and Computational Science Wuhan University of Technology Wuhan, China
2009	M.S., Control Science and Control Engineering Huazhong University of Science and Technology Wuhan, China
2016	Ph.D. candidate, Electrical Engineering Florida International University Miami, Florida

PUBLICATION AND PRESENTATIONS

1. Lilin Guo, Z. Wang, M. Cabrerizo, M. Adjouadi, "A Cross-Correlated Delay Shift Supervised Learning Method for Spiking Neurons with Application to Interictal Spike Detection in Epilepsy," *International Journal of Neural Systems*. DOI: 10.1142/s0129065717500022, 2016.

2. Lilin Guo, Z. Wang, M. Cabrerizo, M. Adjouadi, "Application of Cross-Correlated Delay Shift Rule in Spiking Neural Networks for Interictal Spike Detecion," *Proceedings of 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society 2016 (IEEE EMBC'16)*, Orlando, FL, Aug. 17-20, 2016

3. Lilin Guo, Z. Wang, M. Adjouadi, "A Supervised Learning Rule for Classification of Sptiotemporal Spike Patterns," *Proceedings of 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society 2016 (IEEE EMBC'16)*, Orlando, FL, Aug. 17-20, 2016

4. Z. Wang, Lilin Guo, M. Adjouadi, "Wavelet Decomposing and Phase Encoding of Temporal Signals using Spiking Neurons," *Neurocomputing*, vol. 173, no. 3, pp. 1203-1210, Jan. 2016.

5. Lilin Guo, Z. Wang, M. Adjouadi, "A Novel Biologically Plausible Supervised Learning Method for Spiking Neurons," *Proceedings of 17th International Conference on Artificial Intelligence (WorldComp ICAI'15)*, Las Vegas, NV, Jul. 27-29, 2015, pp.578-584

6. Z. Wang, Lilin Guo, M. Adjouadi, “Spiking Neuron Model for Wavelet Encoding of Temporal Signals,” *Proceedings of 17th International Conference on Artificial Intelligence (WorldComp ICAI’15)*, Las Vegas, NV, Jul. 27-29, 2015, pp.693-699

7. Lilin Guo, H. Deng, B. Himed, T. Ma, Z. Geng, “Optimization for Transmit Beamforming with MIMO Radar Antenna Arrays,” *IEEE Transactions on Antennas and Propagation*, vol. 63, no. 2, pp.543-552, Feb. 2015

8. Z. Wang, Lilin Guo, M. Adjouadi, "A Generalized Leaky Integrate-and-Fire Neuron Model with Fast Implementation Method," *International Journal of Neural Systems*, vol. 24, no. 05, DOI: 10.1142/S0129065714400048, Article No: 1440004, 2014.

9. Z. Wang, Lilin Guo, M. Adjouadi, “A Biological Plausible Generalized Leaky Integrate-and-Fire Neuron Model,” *Proceedings of 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE EMBC’14)*, Chicago, IL, August 26–30, 2014, pp: 6810–6813

10. Lilin Guo, L. Galarza, J. Fan, C. Choi, “High accuracy three-dimensional radar sensor design based on fuzzy logic control approach,” *Proceedings of 2013 IEEE 45th Southeastern Symposium on System Theory (IEEE SSST’13)*, Mar. 11, 2013, pp:22-26.

11. Lilin Guo, T. Ma, H. Deng, “Optimization of Antenna Excitation Phases for Transmit Beam Nulling with MIMO radar,” *Proceedings of 2012 IEEE Antennas and Propagation Society International Synposium (IEEE APSURSI’12)*, Jul. 8, 2012.

12. Lilin Guo, J. Yi, Y. Huang, J. Xiao, “Adaptive Synchronization of nonlinearly stotchastically coupled netwoks with two types of time-varying delays,” *Proceedings of 2011 IEEE 30th Chinese Control Conference (IEEE CCC’11)*, Jul. 22, 2011, pp: 854-858.

13. Lilin Guo, N. Zhao, Y. Huang, J. Xiao, Y. Wang, “Delay-Dependent Stability of Complex Dynamical Networks with Mode-Dependent Parameters and Time Delays,” *Proceedings of 2010 IEEE 29th Chinese Control Conference (IEEE CCC’10)*, Jul. 29, 2010, PP: 4671-4676.

14. Lilin Guo, Y. Wang, H. Wang, Y. Huang, “Delay-Dependent Stability of Discrete-Time Complex Networks with Mode-Dependent Uncertain Parameters and Time Delays,” *Proceedings of 2009 IEEE Control Applications and Intelligent Control (IEEE CCA&ISIC’09)*, Jul. 8-10, 2009, Saint Petersburg, Russia, pp: 177-182.

15. Y. Wang, Lilin Guo, D. Xiao, J. Xiao, “Adaptive Synchronization of GLHS with unknown parameters,” *International Journal of Circuit Theory and Applications*, vol. 37, no. 8, pp:920-927, 2009.