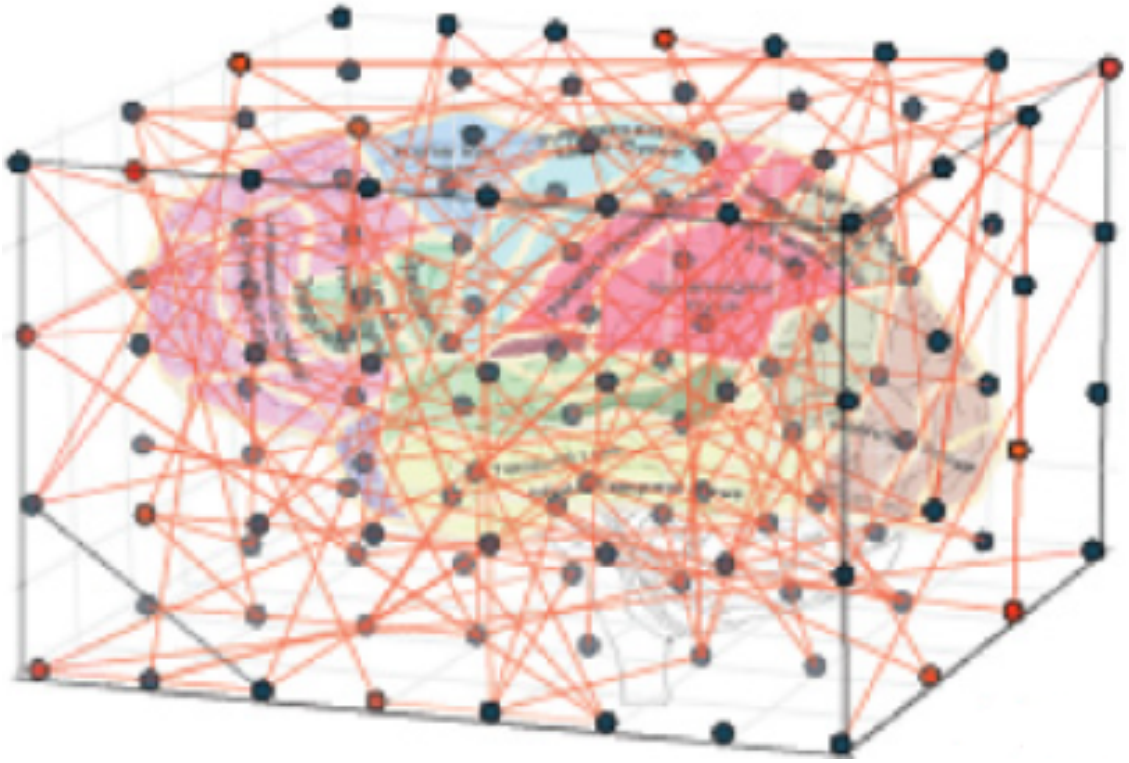
CHALMERS
UNIVERSITY OF TECHNOLOGY



Evaluation of NeuCube Spiking Neural Network Architecture for MEG/EEG Data Classification

Master's Thesis in Biomedical Engineering

YANKUN XU

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

Master's thesis 2018

Evaluation of NeuCube Spiking Neural Network Architecture for MEG/EEG Data Classification

Yankun Xu

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Signal and System
Chalmers University of Technology
Gothenburg, Sweden 2018

Evaluation of NeuCube Spiking Neural Network Architecture for MEG/EEG Data Classification
Yankun Xu

Supervisor: Artur Chodorowski, CHALMERS & MedTech West
Examiner: Artur Chodorowski, CHALMERS & MedTech West

Master's Thesis 2018
Department of Electrical Engineering
Division of Signal and System
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: 6 speed DCT, 6DCT450, GETRAG.

Typeset in LaTeX
Chalmers Reproservice/Department of Electrical Engineering
Gothenburg, Sweden 2018

Evaluation of NeuCube Spiking Neural Network Architecture for MEG/EEG Data Classification

Yankun Xu
Department of Electrical Engineering
Chalmers University of Technology

Abstract

In recent years many new methods within artificial intelligence field have been developed. Many techniques behind AI are taking a role in many practical fields, where pattern recognition in biomedical applications is one of them. Much of the research related to neuroscience use brain electrophysiological signal to investigate many neurodegenerative diseases. However, it is challenging for traditional machine learning methods due to spatio-temporal property of signal. Recently, a new framework called NeuCube has been proposed to address this problem.

In this work, the NeuCube architecture was investigated and evaluated on two pattern recognition applications, one was classification of wrist movement from public electroencephalography(EEG) data, and the other was classification of muscle sympathetic nerve activity(MSNA) from local Magnetoencephalography(MEG) data. Apart from NeuCube, two traditional machine learning methods, support vector machine(SVM) and multilayer perception(MLP), were tested on same dataset to make comparison to NeuCube. As to EEG application, NeuCube achieved 87% classification accuracy, which is much better than SVM and MLP whose accuracy were lower than 50% (5-fold cross validation, three classes). In term of MSNA classification, the performance of NeuCube and traditional machine learning methods were similar, all three methods were able to achieve around 87% accuracy (5-fold cross validation, 20 subjects, two classes), showing that NeuCube can be potentially attractive for neurological brain related applications.

Keywords: EEG, MEG, machine learning, SNN, NeuCube, MLP, SVM, pattern recognition

Acknowledgements

Foremost, I would like to express my gratitude to my thesis supervisor and examiner Artur Chodorowski of Electrical Engineering Department from *Chalmers University of Technology*, he was always available to provide help whenever I had questions about my thesis. Artur is professional in the biomedical field so that he can always give me useful suggestion and inspiration on my thesis research.

Besides my supervisor, I would like to thank Bushra Riaz Syeda who was a PhD student in MedTech West, and her supervisor Prof. Justin Schneiderman and Prof. Mikael Elam. They provided me with local clinical dataset and helped me have a good understanding on dataset.

I also would like to thank *Chalmers University of Technology*, because I have spent very happy two years here. As an international student, I was nervous about my life in the beginning of master study, but the both living and study environment in *Chalmers University of Technology* really helped me build confidence and develop my great interest in my study.

Last but not least, I must express my sincere gratitude to my parents Zhuhua Xu, and Cuilin Guo for providing me with unfailing support and encouragement during my years of master study.

Yankun Xu
Gothenburg, Sweden, 2018

Contents

List of Figures

List of Tables

List of Abbreviations

AI Artificial Intelligence. 1
ANN Artificial Neural Network. 8

BSA Ben's Spiker Algorithm. 12

CT Computed Tomography. 6
CV Cross Validation. xiii, 33

deSNN Dynamic Evolving Spiking Neural Network. 8

ECOS Evolving COnnectionist Systems. 14
EEG Electroencephalography. 1
eSNN Evolving Spiking Neural Network. 14

fMRI functional Magnetic Resonance Imaging. 3

IF Integrate and Fire. 9

LIF Leaky Integrate and Fire. 10
LTD Long-Term Depression. 13
LTP Long-Term Potentiation. 13

MEG Magnetoencephalography. 1
MLP MutiLayer Perception. 1
MSNA Muscle Sympathetic Nerve Activity. 1
MW Moving-Window Spike Encoding Algorithm. 12

PET Positron Emission Tomography. 6
PSP PostSynaptic Potential. 10

RO Rank-Order. 14

SDSP Spike Driven Synaptic Plasticity. 15
SF Step-Forward Spike Encoding Algorithm. 12
SNN Spiking Neural Network. 1
SQUID Superconducting QUantum Interference Device. 6
STDP Spike-Timing Dependent Plasticity. 8
SVM Support Vector Machine. 1

TR Thresholding Representation. 12

1
Introduction

In recent years researches into AI field has increased tremendously. Many techniques behind AI, such as machine learning, deep learning, are becoming powerful in many practical fields, for example, computer vision, natural language processing, biomedical imaging analysis, etc. Apart from these popular fields, pattern recognition with biomedical applications is another important field where AI is able to be useful. Nowadays many brain physiological signals, such as EEG, MEG are utilized as indication or biomaker to investigate many researches related to human brain, for example, mental disorder, sleeping disorder, the correlation between the neuroresponse and behaviors of human, etc.

Many previous researches on classification of EEG/MEG data in relation to specific application have been done. Sonja et al. utilized machine learning on EEG data to successfully achieve accurate classification of cholinergic intervention and Alzheimer's disease[1]. In terms of Ocular and Cardiac Artifacts, Ahmad et al. made use of MEG data to make classification automatically via deep learning approach[2]. However, many researches, such as working memory performance in schizophrenia and healthy adults, auditory perception, posed several challenges for traditional machine learning methods[3][4]. In the thesis work, two practical applications will be investigated, one is classification of human wrist movement using EEG data, the other is using local MEG data from *Sahlgrenska University Hospital* to recognize MSNA inhibition situation, which has been proved to be related to the risk of hypertension[5]. A new SNN architecture framework, so-called NeuCube[23][22], is used in the thesis, in addition another two traditional machine learning methods are used as well to against NeuCube's performance.

1.1 Purpose and goal

The purpose of this master thesis project is to implement some experiments on EEG and MEG data to evaluate the performance of NeuCube framework on spatio-temporal data. Meanwhile we hope use standard classifier to against NeuCube, in the thesis we determine to use two traditional methods, SVM and MLP, in comparison with NeuCube. NeuCube framework is a kind of built-in software/toolbox written in Matlab environment, and traditional machine learning methods are implemented in Python. Three methods will be compared in order to see whether NeuCube is advantageous or not.

And the goals of this master thesis work are:

- Study the principles of NeuCube, and other machine learning methods.
- Explore different preprocessing method for raw brain data.
- Understand the meaning of different parameters inside NeuCube and other machine learning methods, and figure out optimal ones.
- Implement spike encoding on raw data to figure out if spike train is able to simply represent original signal information.
- Implement NeuCube on public EEG data.
- Implement NeuCube on MEG data from local Sahlgrenska hospital.
- Using traditional machine learning methods, SVM and MLP in comparison with NeuCube.

1.2 Solution

Recently, NeuCube SNN architecture is proposed by Prof. Nikola Kasabov to capture both temporal and spatial characteristics of physiological brain data, it has potential to accomplish the goal of the thesis.[22][21]

1.3 Limitation

There is a big challenge for traditional machine learning methods to analyze EEG/MEG data due to its spatio-temporal property. In term of common machine learning task, such as pattern recognition on pictures or other static data, each sample contains not too much information, for example, only one matrix with pixel value can represent a picture sample. Compared to the static pictures, however, EEG/MEG is kind of dynamic data which contains hundreds or thousands data points(samples) per channel in only one second. Sometimes we have about one hundred channels and several seconds to study the behavior of human, so each sample would contain tremendous times information as static data. In addition, EEG/MEG recordings has low signal-to-noise ratio, noise can come from a variety of sources, and in real world, it is not easy to find sufficient patients as available study samples, which all make EEG/MEG classification task become much more difficult. Thus, traditional machine learning methods may not perform well on our applications in the thesis[21].

1.4 Related work

Many researches have been done using NeuCube SNN architecture. Nikola Kasabov and Elisa Capecci et al.[6][7][8] have used EEG data as a type of brain data to study Alzheimer's Disease, absence epilepsy seizure data and human cognitive process, the results are very satisfied and NeuCube has performed better than standard machine learning methods. The spatio-temporal fMRI data is another important indicator to study brain behavior, many researches have shown that NeuCube is a great feasible tool to classify and segment fMRI data.[9][10][11] Apart from brain data, Long Peng et al.[13] has proven the feasibility of NeuCube SNN architecture for electromyography pattern recognition.

It is impressive that, in addition to physiological signal data, NeuCube is able to analyze many other spatio-temporal data. Nikola Kasabov et al.[12] have used NeuCube to analyze static features of patient over 60-day period to predict stroke, and Enmei Tu et al.[14] have successfully solved traffic status classification problem using NeuCube SNN architecture combined with graph matching input mapping method.

2

Theory

2.1 What is EEG and MEG?

2.1.1 What is EEG?

EEG is a kind of electrophysiological monitoring method to record electrical activity of the brain[15]. It is a non-invasive approach to detect the fluctuations or changes of human brain, with so many electrodes placed on the scalp according to specific location. In EEG signal, one electrode means one channel, so EEG signal is also called multi-channel signal. From few channels to hundreds of channels are all used to record different brain activities. Cells in human are able to generate electric potentials via ionic current flows between cell membrane, this electric potentials usually help doctor to monitor the signs of patient. Neural activity in human brain can also generate electric potential, and EEG is used to record this electric potential. When people suffer from many brain disease or disorders, such as epilepsy, sleeping disorder, brain death, etc., some abnormalities would be appeared in the EEG records.

In Figure 2.1, we can see that many electrodes are used to record activity of human brain, and on the right side of picture, the multichannel EEG signals are shown on the screen.

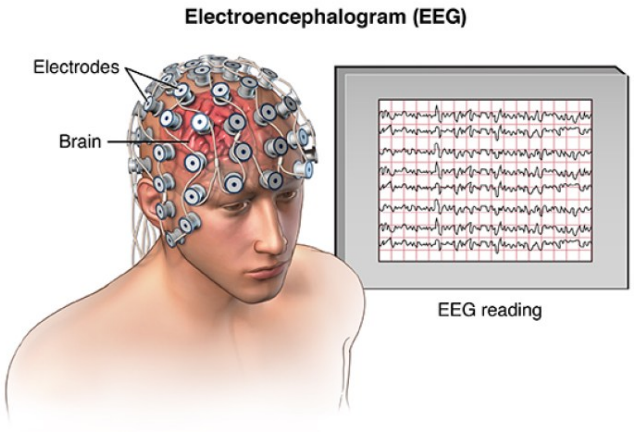


Figure 2.1: EEG schematic diagram

2.1.2 What is MEG?

Compared to EEG, MEG is a kind of neuroimaging technology to directly measure neuronal activity of human brain. The difference between EEG and MEG source is that EEG is to record scalp's electrical activity generated from the electric potential of cell from brain deeply, but MEG is to measure magnetic fields generated from neuronal sources in human brain. MEG has much better performance on spatial resolution which means source could be localized with millimeter precision that 2-3 mm for sources in the cerebral cortex. Except for spatial resolution, MEG is also very good at temporal resolution which could be better than 1ms. Fundamentally, sensitive magnetic fields, which are around 10fT - 1pT, are also generated from electric potential of neuronal cell inside brain, when brain is processing information, small currents flow in the neural system and would produce very weak magnetic fields which can be measured by multichannel SQUID, that is introduced in the late 1960s by James Zimmerman[16]. The magnetic field B at a location r of a dipolar point source with moment Q at location r_0 in a homogeneous volume is given by Sarvas[17]:

$$B(r) = \frac{\mu_0}{4\pi} Q \times \frac{r - r_0}{|r - r_0|^3} \tag{2.1}$$

In terms of MEG sensors, usually SQUID consists of two sensors - gradiometer and magnetometer. Magnetometer is used to measure a magnetic field, and provide data on its strength and direction. As to gradiometer, there are two different ways to construct the detector coils. One is to construct the detector channels from first- or second-order gradiometers, where the two or three gradiometer coils are located concentrically with a baseline on the order of centimeters. The other one utilizes planar gradiometers that two adjoining detector coils are located on the same plane[18]. In this master thesis project, the MEG data we will analyze has 102 locations and each location contains three sensors, one magnetometer and two gradiometers with planar gradiometers approach.

2.1.3 Summary about EEG and MEG

In term of EEG, multichannel electrodes are placed on skull, meanwhile SQUID is placed outside the skull, thus EEG and MEG are both completely non-invasive methods to detect brain activity. They both have similar sample frequencies which usually lies in the range 200-1000 Hz, depending on the purpose of study.

Compared to other neuroimaging technologies, such as CT, PET, fMRI, EEG and MEG have a very high temporal resolution. In addition, they also do not require injection of isotopes and not need people exposure to X-rays or magnetic fields, which means children and infants are able to be studied.

2.2 NeuCube

2.2.1 Overview

NeuCube is a SNN architecture framework for mapping, learning and understanding spatio-temporal brain data.[22][23] NeuCube consists of three main modules[21]:

- Input encoding module
- 3D-SNN Cube module
- Output/Classification module

In the first part, the encoding module utilizes properties of SNN to convert the continuous input information data into discrete spike trains which is binary. After converting, raw continuous signal is transformed into the form of spikes. There are several encoding methods we are able to use, which would be discussed in the following part.

As to the SNN cube, it is scalable which means we can change the size of it. SNN cube consists of many mimic neurons, the number of neurons is determined by three parameters: length(n_x), width(n_y), height(n_z) of reservoir, and the total number is $N = n_x \times n_y \times n_z$. For each neuron, probabilistic leaky integrate and fire model is used in this framework[24].

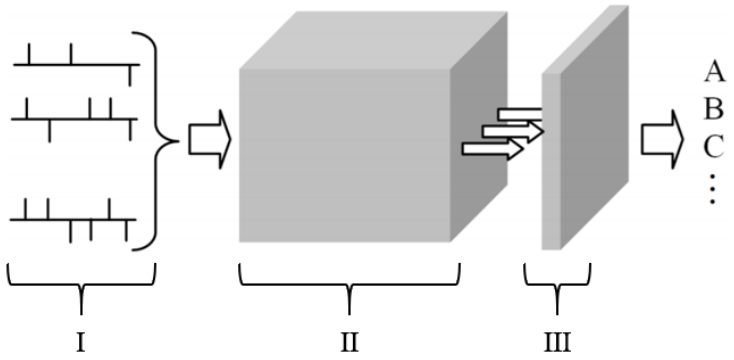


Figure 2.2: Simplified representation of NeuCube architecture which consists of I: Input encoding module; II: 3D-SNN Cube module; III: Output/Classification module.[21]

There are two learning stages inside NeuCube framework, one is unsupervised learning and the other is supervised learning. Unsupervised learning is implemented firstly, before training in cube, connection weights inside cube are initialized at first, then training samples are input into cube, for single training sample, there is a spike train for each channel, which means the number of channels determines the number of input neurons, for example, in this thesis work, each sample in EEG dataset has 14 channels, so there are 14 input neurons inside the cube, which has N neurons in

total. The next, unsupervised learning is to modify or adjust all neuronal connection weights to encode the spatio-temporal relationships between input data. Two fundamental learning rules are utilized during the training phase: Hebbian learning rule[25], and STDP learning rule[26].

The second stage occurred in output module is to make use of deSNN[39] to train an output classifier. This classifier is not similar with classifier used in traditional machine learning method. After cube training finished and before supervised learning beginning, there are none neurons existed in classifier layer. For supervised learning, each training sample is input into cube again one by one, established connection weights help every neuron inside cube have a spike train, then all neurons in cube are connected to a newly created output neuron, thus the number of training samples determines the number of output neurons of classifier. The connection weights between cube and classifier is generated and adjusted according to the learning rules inside deSNN. Due to the class information for training samples, each output neuron has their own label consistent with the label of every training sample, this is the reason why we call this learning phase as supervised learning. When all output neurons are created, every test sample can be input into the NeuCube, similar to the supervised learning step, each test sample would create a output neuron, but for recall phase, test output neuron will be used to make comparison that which training output neurons is most similar one, then the label of this test sample is consistent with the label of the closest training output neuron.

In the following parts, many terminologies and learning rules mentioned before will be illustrated elaborately.

2.2.2 Brain-inspired SNN

SNN, which is considered as the third generation of ANN, is presented in the form of spike trains for information processing. Inside SNN, there are many artificial spiking neurons containing binary spikes only in order to network is able to process huge amount of data[38]. Sometimes, the mechanism inside human brain can be seen as the ultimate inspiration to develop new generation of machine learning or artificial intelligence approach to handle spatio-temporal brain data, because human brain has amazing capacity to learn and recall patterns with different time scales, ranging from milliseconds to years. Due to functional similarity between neurons inside SNN and biological neurons inside human brain, spiking model has powerful potential to process spatio-temporal brain data by encoding both spatial and temporal information in SNN as location of synapses and neurons and time of spike trains[22]. Thus we usually call SNN as brain-inspired neural network. In term of traditional ANN, each neuron inside the whole network contains only single value, but in SNN each neuron represents a spike train.

2.2.3 Spiking neuronal model

From biophysiologic point of view, two neurons communicate by transmitting electrical pulse which is so-called action potential or spike via synapse. Figure 2.3 shows the structure of single neuron which consists of three parts, called dendrites, soma, and axon, and action potential generted from axon or pre-synaptic neuron. The action potential is a kind of electrical pulse has the range of 1-2ms duration and around 100mV amplitude[19].

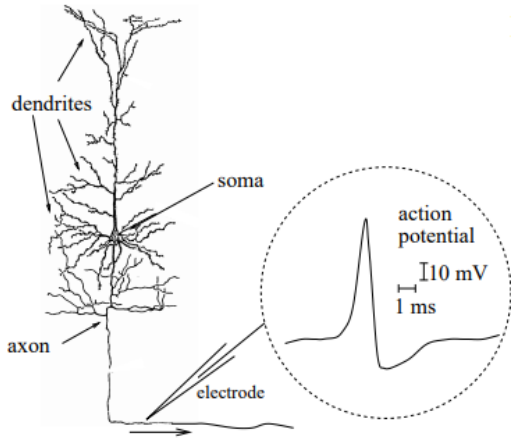


Figure 2.3: Single neuron and action potential[19]

In the past few decades, many different spiking neuronal models have been proposed. From Gerstner and Kistler[19], for most of spiking neuronal models, transmitted neural information in the synapse is represented mainly by timing of spikes rather than shape of spikes. The mathematical description for a sequence of spike trains is shown as equation 2.2:

$$S(t) = \sum_f \delta(t - t_i^{(f)}) \tag{2.2}$$

where $f = 1, 2, 3, ...$ is the label of spikes and δ is the Dirac function.

A popular choice of spiking neuronal model for SNN is IF model. Figure 2.4 shows the circuit of IF model happened inside soma cell. A current $I(t)$ coming from external current or input from presynaptic neurons is used to charge the RC circuit, where R is membrane resistance and C stands for membrane capacitance. The voltage $u(t)$ across C is compared to a potential threshold ϑ. And output spike $\delta(t - t_i^{(f)})$ would be generated when $u(t)$ reach at this threshold at time $t_i^{(f)}$[19].

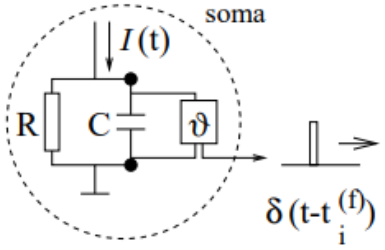


Figure 2.4: Schematic diagram of the IF model[19]

In Figure 2.4, we can see that capacitor C is parallel with resistor R and the driving current $I(t)$ goes through this two components, $I(t) = I_R + I_C$. The current passing through resistor can be calculated by Ohm's law as $I_R = u(t)/R$. For capacitive current, we need use the definition of capacity and current as $C = q/u$ and $I = dq/dt$ to get $I_C = Cdu/dt$. Thus:

$$I(t) = \frac{u(t)}{R} + C\frac{du}{dt} \tag{2.3}$$

Formula 2.3 represents the best-known spiking neuronal model LIF model, if we multiply this formula by R, the standard form is shown as:

$$\tau_m \frac{du}{dt} = -u(t) + RI(t) \tag{2.4}$$

where $\tau_m = RC$ is the term of "leakage". Equation 2.4 can be used to describe IF model as well if R tends to infinite.

LIF model can be stimulated by either external current or presynaptic spike trains, here we only discuss the condition of biological spike trains stimulus. At first, we introduce PSP $\epsilon_{ij}(t)$ whose definition is:

$$u_i(t) - u_{rest} =: \epsilon_{ij}(t) \tag{2.5}$$

where $u_i(t)$ and u_{rest} are the membrane potential and resting membrane potential of neuron i respectively. If several presynaptic neurons j transmitting spike trains to the postsynaptic neuron i, we will have time course of membrane potential $u_i(t)$ of neuron i as[19]:

$$u_i(t) = \sum_j \sum_f \epsilon_{ij}(t - t_j^{(f)}) + u_{rest} \tag{2.6}$$

where $f = 1, 2, 3, ...$ and $t_j^{(f)}$ is presynaptic spike arrival times.

Except for formula 2.6 to describe the LIF model, time course of the membrane potential $u_i(t)$ of a LIF model stimulated by presynaptic spike trains and mechanism of output firing spikes are clearly shown as following:

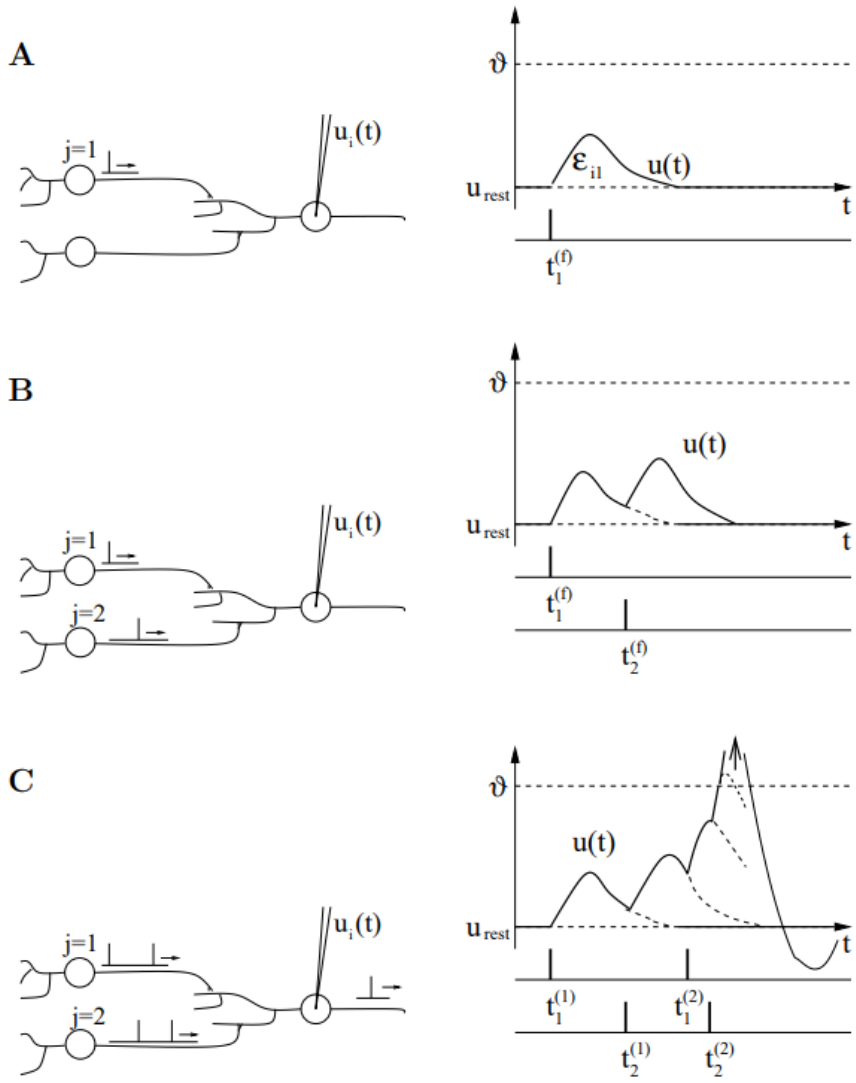


Figure 2.5: Time course of membrane potential $u_i(t)$ and output firing spikes on postsynaptic neuron i[19].

In Figure 2.5, an example of two presynaptic neurons $j = 1, 2$ sending spike trains to postsynaptic neuron i is described. In part A, only neuron $j = 1$ sends one spike to the neuron i, which leads to the time course of PSP becomes $\epsilon_{ij}(t - t_1^{(f)})$, whereas another spike driven by neuron $j = 2$ sends to neuron i as well with a short time delay in part B, causing a second PSP that adds to the first one linearly. Followed by principles shown before, several spikes driven by two presynaptic neurons are sent to neuron i, but in part C $u_i(t)$ reaches the threshold ϑ which leads to an output spike generated(shown in left-hand side of part C). After that, $u_i(t)$ will returns to a value quickly below resting potential firstly then back to u_{rest}, sometimes we call it as refractory period[20].

2.2.4 Input encoding

There are several input encoding schemes provided to convert the raw continuous data into spike trains. In the NeuCube framework package, there are four available built-in schemes: BSA[29], TR[30], SF, and MW.

Different encoding methods have distinct properties. BSA convert data into positive(excitatory) spikes only, it is suitable for signals with high frequencies[22], and it has been implemented in the EEG data transformation task[31]. Except for BSA, other three methods will generate both positive and negative(inhibitory) spikes. AER is able to capture significant changes in data via a given threshold, the positive and negative spikes generated by the sign of changes. AER was implemented initially in the artificial silicon retina[32]. Furthermore, SF and MW methods are variants of AER method[33], among these four methods or other proposed methods, which one should be chosen to carry the information of given signals depends on the characteristic of them.

2.2.5 Hebbian learning rule

Hebbian learning rule, which is proposed by neuropsychologist Donald Hebb in 1949 is claiming how biological neurons learn: the connections between two cells/neurons will be strengthened if the interaction between two neurons is persistent.[25] Sometimes, this learning rule is summarized as "Cells that fire together wire together."[34], however, it is noticed that cell j needs to just fire before, not at the same time as cell i in order to obey the learning rule.

In ANN field, hebbian learning rule can be regarded as a way to change or update the weights between two artificial neurons. The weights would be increased if two neurons activate simultaneously or synchronously and decreased if they activate asynchronously. Figure 2.6 shows synapse weight w_{ij} from presynaptic neuron j to postsynaptic neuron i.

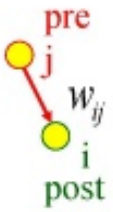


Figure 2.6: Artificial neuron model

Equation 2.7 is the formulaic description of hebbian learning rule, where w_{ij} is the weight of connection from neuron j to neuron i.

$$w_{ij} = x_i x_j \tag{2.7}$$

and equation 2.8 is used to update the weight.

$$\Delta w_{ij} = \eta x_i x_j \tag{2.8}$$

2.2.6 STDP learning rule

STDP is a temporally asymmetric form of Hebbian learning induced by tight temporal correlations between the spikes of pre- and postsynaptic neurons. Sometimes, STDP can be seen as a spike-based formulation of a Hebbian learning rule.[35]. With STDP, LTP and LTDcan be introduced. A presynaptic spike arrival a few milliseconds before a postsynaptic spike will increase synaptic strength, which leads to LTP. In contrast, a presynaptic spike arrival a few milliseconds before a postsynaptic spike will weaken synaptic strength, which leads to LTD. The change in synaptic strength depends on the timing difference between the presynaptic and postsynaptic spikes.

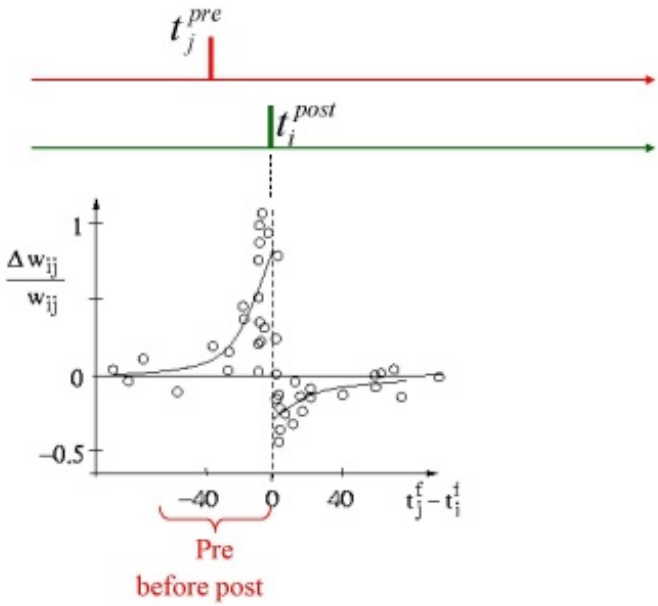


Figure 2.7: STDP schematic diagram[36]

Figure 2.7 shows the change of synaptic strength as a function of the timing difference between pre- and post-synaptic spikes, this diagram shows 60 spike pairings. The x-axis is the timing difference between two spikes, and unit is millisecond. It is noticed that here we use time of spiking on pre-synapse minus time of spiking on post-synapse, so we can see that negative part of x-axis represent presynaptic spike activated before postsynaptic spike, which leads to synaptic strength can be increased. The efficacy or degree of change in synaptic strength is shown on y-axis.

We are still able to describe STDP model as in following formula:

$$\Delta w_{ij} = \sum_{f=1}^{N} \sum_{n=1}^{N} F(t_i^n - t_j^{(f)}) \tag{2.9}$$

where Δw_{ij} denotes the weight change of a synapse from a presynaptic neuron j to postsynaptic neuron i, t_j^f means the presynaptic spike arrival times and t_i^n means postsynaptic neuron's firing times, where $f, n = 1, 2, 3, ...$

In the equation 2.9, there is another parameter F, it is a function of Δt which is time difference between pre- and post-synaptic spike arrival times as shown in the x-axis of Figure2.7. $F(\Delta t)$ denotes the STDP function illustrated before, and this function can be presented as equation 2.10:

$$\begin{aligned} F(\Delta t) &= A_+ \; exp(-x/\tau_+), \quad if \; \Delta t > 0 \\ F(\Delta t) &= A_- \; exp(-x/\tau_-), \quad if \; \Delta t < 0 \end{aligned} \tag{2.10}$$

where A_+ and A_- determine the maximum values of synaptic change, and both of them are positive, so we will see change achieve maximum modification if Δt tends to zero. τ_+ and τ_- determine the range of pre-to-postsynaptic interspike intervals when the occurrence of synaptic strength is increased or weakened[37].

2.2.7 deSNN

The eSNN will be introduced firstly before we illustrate deSNN. The eSNN makes use of principles of ECOS proposed by Kasabov[39], but as an extend of ECOS, eSNN also uses IF model for spiking neuron and RO learning[40]. Before RO learning rule was published, Thorpe and Gautrais also discovered that earlier arriving spikes between synapses contain most important information, that is an fundamental assumption for the RO learning rule. The RO learning used in SNN has many advantages, such as fast, one-pass learning(reads input only once) and asynchronous data processing. RO learning consists of two phases-learning phase and recall phase.

During learning phase, a new output neuron i is created after entering each input pattern, and the connection weights $w_{ji}(j = 1, 2, 3, ...$ represents input vector dimensions) between input vector and neuron i are calculated according to RO learning rule:

$$w_{ji} = \alpha \cdot mod^{order(ji)} \tag{2.11}$$

where α is learning parameter(in a partial case it is equal to 1); mod is a modulation factor which defines the importance of the order of the first spikes; $order(j, i)$ stands for the rank of the first spike at single synapse j, i among all synapses, by the way we only care about the first spike from each synapse and $order(j, i) = 0$ when neuron i receives the first spike, then $order(j, i) = 1, 2, 3, ...$ as following first spikes received[39].

After training phase of single input pattern is finished(input spikes from all synapses are sent to the network), we need set a threshold Th_i of neuron i to decide i spike or not when a new pattern sent in the network again in the recall phase. Formula 2.12 shows the threshold Th_i defined as a fraction(C) of total PSP of neuron i, ϵ_{imax} [39]:

$$Th_i = C \cdot \epsilon_{imax} \tag{2.12}$$

where

$$\epsilon_{imax} = \sum_{j} mode^{order(ji)} \tag{2.13}$$

When all input patterns sent to network, output neurons whose number equals to the number of input patterns are created, then recall phase is used to help testing the network. During the recall phase, the PSP of a neuron i at time l, $\epsilon_i(l)$, is calculated as[39]:

$$\epsilon_i(l) = \sum_{t=0,1,2,\ldots,l} \sum_{j} e_j(t) \cdot mod^{order(ji)} \tag{2.14}$$

where $e_j(t) = 1$ if neuron i receives a first spike at time t on synapse j. An output spike(not an output neuron as training phase does) will be generated if $\epsilon_i(l)$ reaches the threshold Th_i, then the class of this activated neuron i is able to recognize the new input pattern.

The deSNN is an extend of eSNN because only RO learning used in SNN may not be very satisfied for the classification task of spatio-temporal data, and the word "d", a short for "dynamic", stands for a new adaptive learning rule, so-called SDSP, is utilized together with RO learning rule in SNN to adjust the connection weight of synapses w_{ji} in order to synapses become dynamic[39]. There are many advantages when we use deSNN, for example, reducing computational cost, utilizing both first spike and following spikes information.

During the learning phase of deSNN, similar to steps implemented in the eSNN, a new output neuron i will be created for each training input sample, and its connection weights w_{ji} is calculated initially as $w_{ji}(0)$ based on the first spike from each synapse j according to the RO learning rule, then this synapse will become dynamic and adjust the weight based on the consecutive spikes at synapse j according to SDSP learning rule. w_{ji} will be increased with a small value if a new spike sent in synapse j at time t, whereas decreased if no spike appears at time t. The formulaic representation is[39]:

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t) \tag{2.15}$$

where

$$\Delta w_{ji}(t) = e_j(t) \cdot D \tag{2.16}$$

where $e_j(t) = 1$ if there is a consecutive spike at synapse j at time t, otherwise it equals to -1; D stands for a draft parameter to adjust the weight how much "go up" when $e_j(t) = 1$ or "go down" when $e_j(t) = -1$. In SDSP learning rule drift is bi-stable, which means if a weight is larger than defined High value (resulting in LTP) or smaller than Low value(resulting LTD), this connection weight is fixed to this threshold value for the rest of the training phase.[39]

As in the eSNN, deSNN also need a threshold Th_i of neuron i to determine this out neuron spike or not, Th_i is calculated as formula 2.12 shows, but in this situation total PSP of neuron i is calculated based on adjusted weights:

$$\epsilon_{imax} = \sum_{t=0,1,2,...,T} \sum_{j} f_j(t) \cdot w_{ji}(t) \tag{2.17}$$

where T represents the length of time period of input pattern; $f_j(t) = 1$ if there is a spike at synapse j at time t, otherwise it equals to 0; $w_{ji}(t)$ is the dynamic weight calculated using formula 2.16.

During the recall phase, PSP of neuron i is calculated as:

$$\epsilon_i = \sum_{t=0,1,2,...,T} \sum_{j} f_j(t) \cdot w_{ji}(t) \tag{2.18}$$

As decision rule in eSNN, ϵ_i of new input pattern is used to compared to Th_i to decide which output neuron will be activated in order to new input pattern is able to recognized.

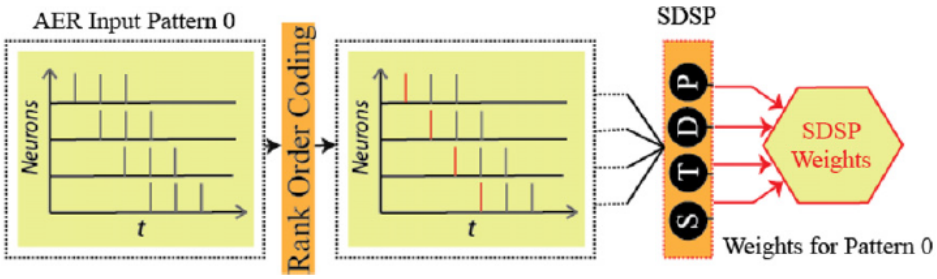


Figure 2.8: An example to illustrate principle of deSNN learning rule.[39]

There is a simple example that can help illustrate principle of deSNN learning rule. As Figure 2.8 shown, there are four neurons transmitted to single output neuron, RO learning rule is used to recognize only the first spike of a spike train from each input neuron, then output neuron receives spikes from the first neuron to fourth one in order. In this case the parameter *mod* is set to 0.8, thus the initial weights w_1, w_2, w_3, w_4 of four synapses is: $1, 0.8, 0.64, 0.512$ according to equation 2.11, then SDSP learning rule is applied to adjust these weights so that each spike train will be drifted two times due to there are three spikes in total within spike train for all neurons, in this case, SDSP's high and low value is set to 0.6 and 0 correspondingly, from equation 2.16 drift parameter D is 0.00025 and e_j is kept to 1 due to all spikes are positive in this example, thus finally the first three weights are fixed to 0.6 and the fourth one is 0.5125 because $0.512 + 0.00025 \times 2$ lies in the range $(0, 0.6)$. After deSNN learning, we can add all four weights to calculate the total information that the output neuron receives from all input neurons.

2.3 Support Vector Machine

SVM is a popular machine learning algorithm for solving problems in classification, regression.[41] Suppose in two-class classification task, inputs can be mapped into a feature space, and each data point is regarded as a vector with dimension p which is equal to the number of features, then there are many hyperplanes that are able to separate these data points into two classes, but the concept of SVM is to find best hyperplane so that the distance from the hyperplane to the nearest data point of each class is maximized.

In order to describe theory of SVM illustratively, we start from linear model on two-class classification problem in form of

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}) + b \tag{2.19}$$

where $\phi(\mathbf{x})$ and b denote fixed feature-space transformation and bias.[41] All input vectors $\mathbf{x}_n : \mathbf{x}_1, ..., \mathbf{x}_N$ has their own labels $t_n : t_1, ..., t_N$, where $t_n \in \{-1, 1\}$. For all training data points, there exists many parameter pairs $\mathbf{w}, b$ make $y(\mathbf{x}_n) > 0$ for points with $t_n = +1$ and $y(\mathbf{x}_n) < 0$ for points with $t_n = -1$ according to 2.20 if training data set is linearly separable, so that when new data points are input into this model, we can classify them according to the sign of $y(\mathbf{x})$. However, the SVM is to find the decision boundary that makes margin is maximized.

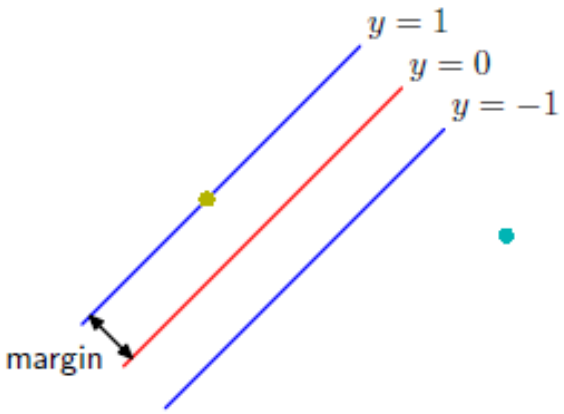


Figure 2.9: Definition of margin.

In Figure 2.9, the line $y = 0$ denotes the decision boundary, and the margin is defined as the perpendicular distance from decision boundary to the nearest data points. According to form 2.20 the margin can be described as $\frac{|y(\mathbf{x})|}{\|\mathbf{w}\|}$, we could introduce label t_n to help eliminate the absolute sign of $y(\mathbf{x})$ due to $t_n y(\mathbf{x}) > 0$ for all data points. Thus we get the margin:

$$\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|} = \frac{t_n(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n) + b)}{\|\mathbf{w}\|} \tag{2.20}$$

Then we wish to optimize the parameters $\mathbf{w}$, b in order to maximize the margin, so we need solve margin maximum problem:

$$\arg\max_{\mathbf{w},\ b}\left\{\frac{1}{\|\mathbf{w}\|}\min_{n}[t_n(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)+b)]\right\} \tag{2.21}$$

For all data points, the condition

$$t_n(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)+b)\geq 1 \tag{2.22}$$

is satisfied, because the nearest data point satisfies $t_n(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)+b)=1$, then we can transform the optimization problem 2.21 into a simple form:

$$\arg\max_{\mathbf{w},\ b}\frac{1}{2}\|\mathbf{w}\|^2 \tag{2.23}$$

subject to the constraints 2.22, the factor $1/2$ is introduced for later convenience. In order to solve this optimization problem, Lagrange function is introduced.[41]

Usually, training dataset is not linearly separable so that we need a nonlinear classifier to accomplish classification task. Kernel trick is proposed to build nonlinear classifier, kernel function which comes from dual function when we solve Lagrange function, is defined by $k(\mathbf{x},\mathbf{x}^{'})=\phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{x}^{'})$, kernel trick is to replace dot product by different nonlinear kernel functions. Sometimes, we need introduce soft-margin trick if data cannot be separated. In the thesis Gaussian kernel $k(\mathbf{x},\mathbf{x}^{'})=\exp(\gamma\left\|\mathbf{x}-\mathbf{x}^{'}\right\|^2),\gamma>0$ is used.

2.4 Multilayer Perception

MLP which is a class of feed-forward ANN, consists of three parts - one input layer, several(≥ 1) hidden layers, one output layer. Figure 2.10 shows a MLP, where each circle represents a artificial neuron or perception.

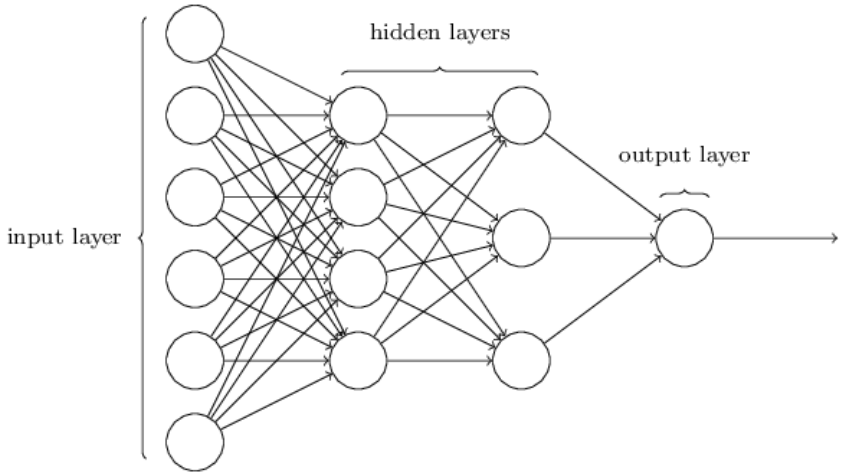


Figure 2.10: An example of MLP with one input layer, two hidden layers and one output layer.[42]

Learning algorithm is to train the parameters weight w_{ij} and bias b between every two layers. Apart from input layer, as shown in Figure 2.11, each neuron x_i is the output of all presynaptic neuron x_j $(j = 1, ..., d)$ from previous layer. The feed-forward propagation rule is:

$$y(\mathbf{x}) = f(\mathbf{w}^T \cdot \mathbf{x} + b) \tag{2.24}$$

where $\mathbf{x}$, $\mathbf{w}$ represent input and weight vector, $f(\cdot)$ denotes the nonlinear activation function, and b is bias added for setting a threshold to help activation function always work. There are many different activation functions, we use the sigmiod function $f(x) = 1/(1 + exp(x))$ in the thesis. Sometimes bias also can be regarded as a weight w_{i0} coming from a input neuron with a constant value 1.

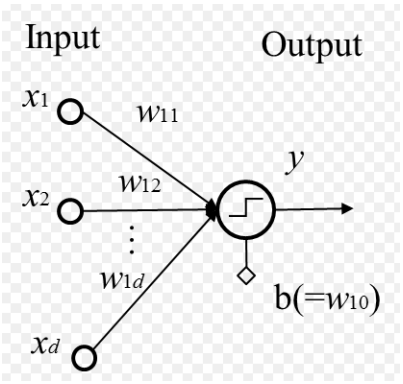


Figure 2.11: An example of output neuron x_i with $i = 1$

When we keep following the feed-forward rule in MLP, finally outputs $\mathrm{Y}'(\mathbf{x})$ would be generated in output layer, after that a loss function L is used to calculate the difference between predicted output and true label $\mathrm{Y}(\mathbf{x})$, sometimes we also call it cost function. As activation function, we still have many options for cost function, in the thesis we choose cross-entropy loss function:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} log p_{ij} \tag{2.25}$$

where N is the number of training samples, M is the number of all classes, y_{ij} is true label for the instance i, and p_{ij} is our prediction which is in form of probability for the instance i in class j. The aim of MLP is to find optimal weights and bias so that the value of loss function should be as small as possible, during the learning process we update the weights and bias via backpropagation algorithm.[42]:

$$\begin{aligned} w &\leftarrow w + \eta \Delta w \\ b &\leftarrow b + \eta \Delta b \end{aligned}$$

where η is learning rate. It is noticed that the number of neurons in output layer depends on how many classes we want to classify in the classification task.

3
Experiment

3.1 Data description

3.1.1 Human wrist movement EEG data

The HWM EEG data is distributed with the NeuCube software toolbox[23] and is free to use. This dataset is collected from human wrist movement behavior with three different patterns: moving wrist either to the left or to the right, or holding hand straight.[44] There are 60 samples, within each sample 14 channels are tested and 128 sampling points from time domain are recorded in each channel.

3.1.2 Muscle Sympathetic Nerve Activity MEG data

In this thesis work, MEG data coming from Sahlgrenska University Hospital is utilized to investigate the correlation between MEG recordings and MSNA response in brain. Many previous researcher proved that MSNA inhibitor response happened when starting/arousing stimuli, such as visual flash, auditory beep or electrical stimulation to the finger, is correlated to cardiovascular responses during stress. Usually MSNA is divided into two classes, inhibitor and non-inhibitor, and higher and sustained blood pressure response in non-inhibitor is more likely to cause cardiovascular disease[43]. In this experiment protocol, which is shown in Figure 3.1, we apply three electrical pluses on the finger with every other heartbeat and the pulse train with three pulses is repeated after 30 to 60 seconds for 72 times. Because Pulse 1 is triggered after 30 to 60 seconds during every trail, pulse 1 might be surprising, whereas Pulse 3 comes after four heartbeats of Pulse 1 therefore Pulse 3 is expected. A 1.5 seconds response after each pulse is measured by MEG whose sampling frequency is 1000Hz, thus we record 1500 sampling values during 1.5s interval.

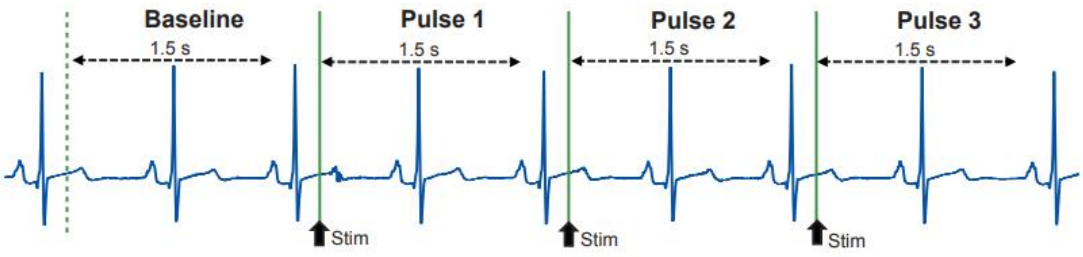


Figure 3.1: MEG study design protocol.

There are 20 subjects measured in the MEG experiment, each subject has their own MSNA values, which is shown by ascent order in Figure 3.2. Usually 30 is a threshold to distinct non-inhibitor and inhibitor, thus the first 10 subjects are non-inhibitors and the last 10 subjects are inhibitors. In practical work, I get the data with only No.2 and No.20 subjects in the beginning, and get full 20 subjects dataset later. Therefore, classification task with 2-subject situation is tested initially, after that 20-subject situation is investigated.

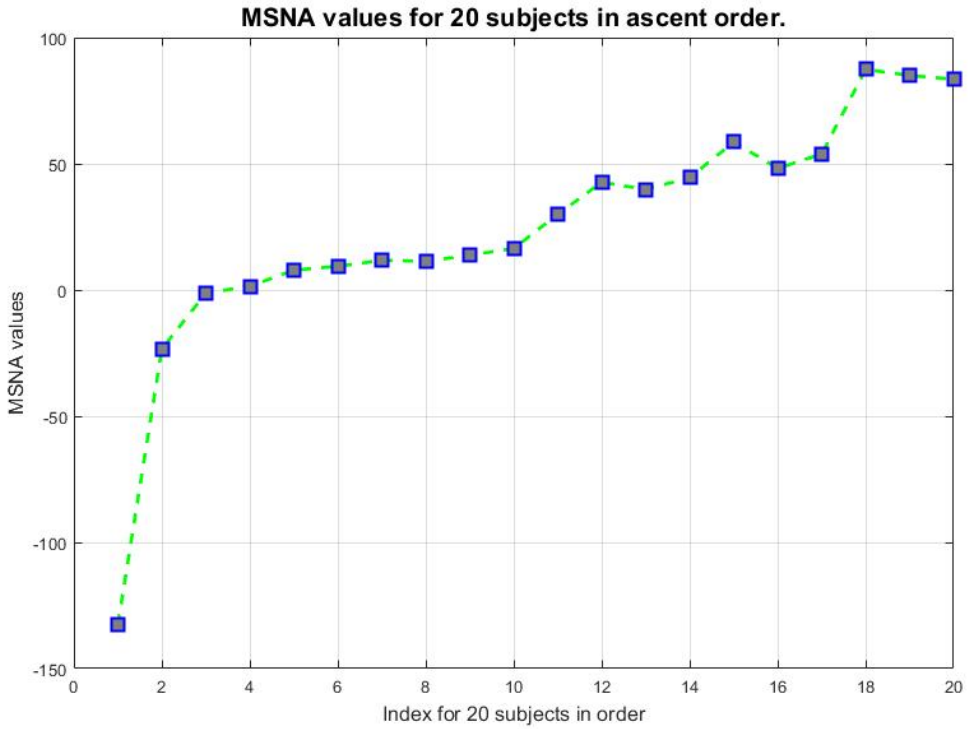


Figure 3.2: MSNA value for 20 subjects in provided order.

In term of dataset we used in the thesis work, MEG signals starting from three different pluses are separated, and within each pulse signal, we have 306 data points, 102 channels and three sources per channel - one is from magnetometer sensor and another two comes from gradiometer.

3.1.3 Data extraction and cleaning

When we look into the channel data from MEG dataset, there are many different channel information missing in different subjects, for example, some subjects have full 306 channel information, but some of them miss one or several data from magnetometer sensor. In order to keep the length of data consistent, we need find the channels all 20 subjects contain, thus, there are only 84 channels left for experiments. Furthermore, only information from magnetometer is used in the experiments, so data from only 87 magnetometer sensors is extracted for pattern recognition task.

3.2 Data preprocessing

3.2.1 Input encoding

In NeuCube SNN architecture, the raw data is encoded into spike trains initially. Encoding approach is a built-in module embedded in NeuCube toolbox, but the performance of SVM and MLP on the data with input encoding approach is evaluated in later experiments, so we have to write input encoding approach function manually for traditional machine learning methods. There are four input encoding approaches embedded in NeuCube framework, and TR method has the best performance according to Figure 4.1, thus only AER method is chosen to test the performance of input encoding on traditional machine learning methods. The AER operation can be represented as mathematical formula[30]:

$$\frac{d}{dt}logI = \frac{dI/dt}{I} \tag{3.1}$$

where I is the intensity of signal. Then adding a threshold setting, AER input encoding approach is available for traditional machine learning methods.

3.2.2 Normalization

The AER input encoding approach is "self-normalized"[30]. For traditional machine learning methods, however, it is hardly to find a useful classifier on the data with quite large or small amplitude, thus centering or normalization is an essential operation before training. In this thesis, formula:

$$x_{new} = \frac{x_{old} - mean(\mathbf{x})}{std(\mathbf{x})} \tag{3.2}$$

where $\mathbf{x}$ is all data from each channel, is used to normalize the data per channel in both two applications.

3.2.3 Averaging

Compared to time consuming of EEG task, NeuCube has to spend tremendous time training the SNN cube. In EEG task, about 20 seconds is cost to train the network for each run. As to MEG task, however, it has more information in the training set due to longer time sequence, more channels and training samples, in only 2-subject situation experiments approximately MEG task with original data has $324(\frac{1500}{128} \times \frac{101}{14} \times \frac{144 \times 0.8}{60 \times 0.5})$ times information compared to EEG task, which makes MEG task costs at least 30 minutes during training for each run. Thus, time sequence averaging and sample averaging are proposed to help NeuCube be more efficient in MEG task. In addition, averaging is able to reduce the influence of noise, and help SVM and MLP to spend less time training classifiers significantly as well.

Time sequence averaging

In both EEG and MEG data, each channel records brain information in form of a time sequence. Sometimes a time sequence has thousands sampling data points, a mean sliding window is used to help reducing the time sequence. We need set the averaging number of sliding window initially, then move sliding window forward half averaging number for each time until the sliding window reaches the end of time sequence. For example, averaging number 10 is chosen in Figure 3.3, the first data point in new sequence is the mean of first 10 data points from original time sequence, and the second data point is the mean of No.6 to No.15 data points original old time sequence, and so on. We assume original time sequence contains N sampling data points, and averaging number is n, then we will achieve a new time sequence with $\{2 \times around(\frac{N}{n}) - 1\}$ data points. The time sequence averaging should be implemented before input encoding. In the thesis work, $\{n = 20, 50, 100\}$ are evaluated on MSNA application with 2-subject situation.

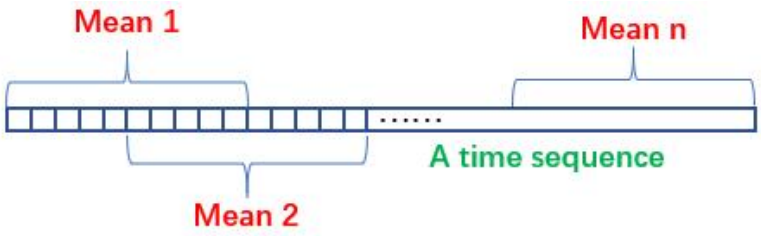


Figure 3.3: Time sequence averaging protocol

Sample averaging

As described in Section 3.1.2, each subject from MSNA application contains 72 trails/samples. Because these 72 samples come from same experiments repeatedly, we can take mean of some samples as single sample in order to reduce the the size of training set. We assume the averaging number is m, then only $\frac{72}{m}$ samples left in each subject. We only use sample averaging approach in 20-subject situation, and the averaging number 12 is chosen, thus in new dataset we have 6 samples in each subject and $120(6 \times 20)$ samples in total.

3.2.4 Training set ratio

Training set ratio determines that how many samples from whole dataset would be trained for classification model. However, in the thesis, we need manually set different ratios for different tasks. In term of EEG wrist movement task, 0.5 ratio is set because we only have 60 samples in total, and the number of test samples shall not less than 30 samples. Whereas 0.8 ratio is set for MEG classification task because we have abundant samples.

3.2.5 Alignment

The data we are using in the thesis is spatio-temporal, NeuCube SNN architecture is able to capture channels' location information, but it is a big challenge for traditional machine learning methods. Thus we need implement extra preprocessing before using SVM and MLP classifier, here we determine to align or flat the data matrix into a vector for every single sample, the order of channel follows the default setting of dataset.

3.3 Implementation of NeuCube

In this part, wrist movement EEG dataset will be used to illustrate the processing routine of NeuCube. First of all, Figure 3.4 shows the user interface of NeuCube, here classification on EEG task is chosen. After loading dataset, information panel shows the information of the loaded data, shown as Figure 3.5.

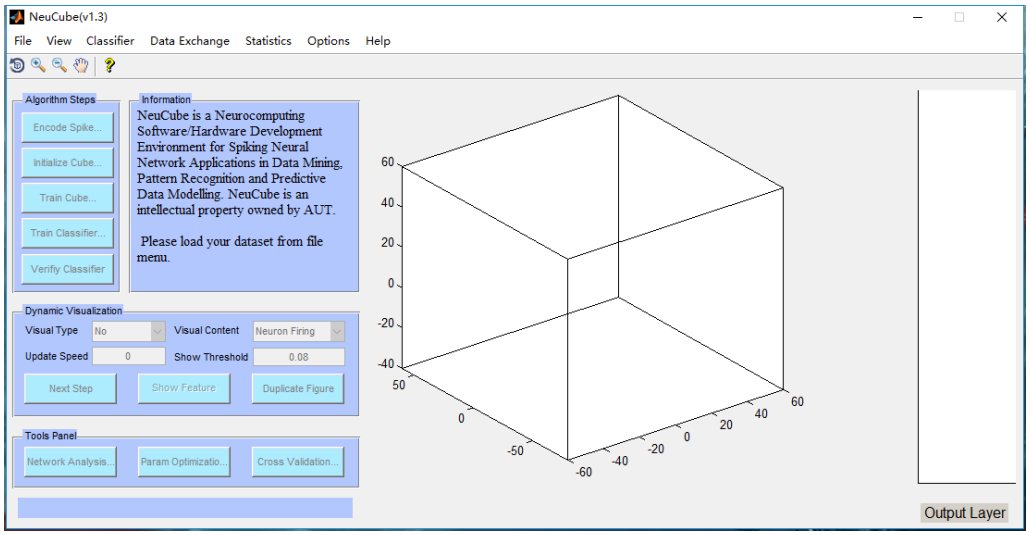


Figure 3.4: NeuCube user interface

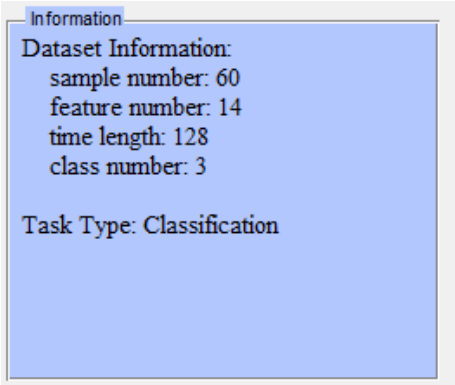


Figure 3.5: Information for loading data

Figure 3.6 is encoding panel, there are four available encoding methods as discussed before here, for expected TR method, we set the parameter "Spike Threshold" as 0.5. In terms of other settings, "Training Set Ratio" is 0.5, and full training and validation time length are chosen. After finishing the setting of encoding approach, original and encoded signal are displayed as Figure 3.7 shown where signal from only one feature/channel of one sample can be displayed once.

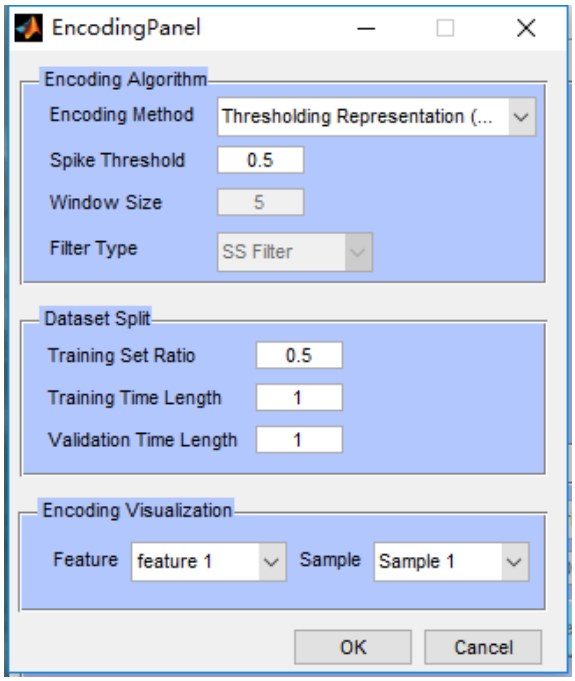


Figure 3.6: Encoding panel

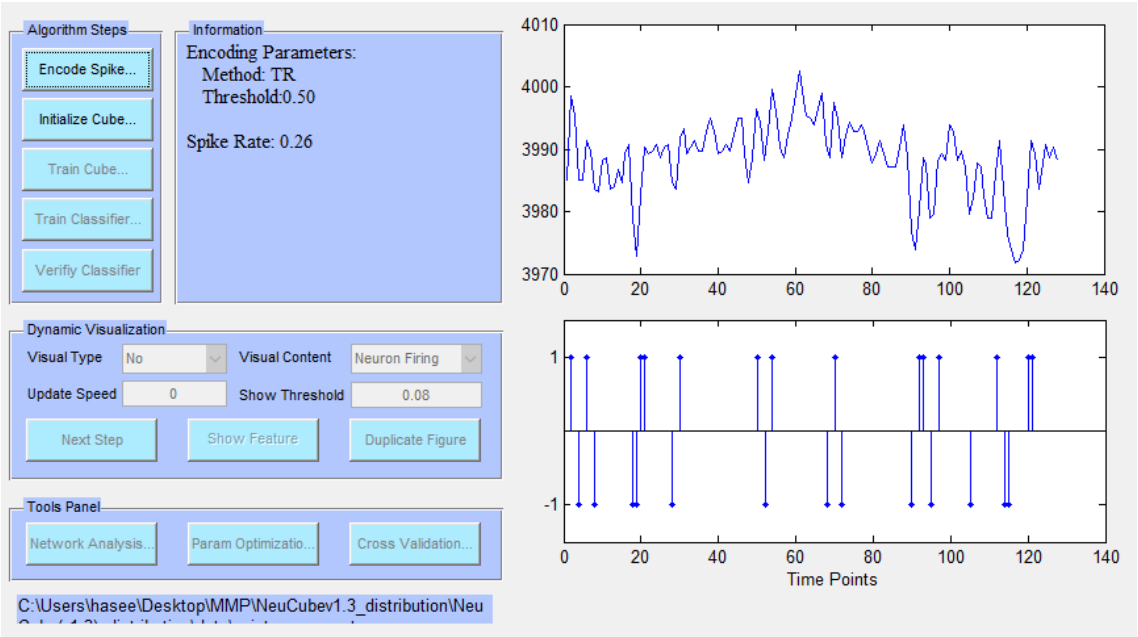


Figure 3.7: Spike encoding display

Before training SNN cube, we need initialize the Cube firstly. In the **Initialization Panel** shown as Figure 3.8, we need set the neuron coordinates and mapping location of features/channels if they are provided in the dataset. In experiments, We are able to choose brain coordinates and mapping location of features/channels based on Talairach template coordinates in the EEG task, but for MEG task, automatic neuron coordinate and graph matching[14] mapping location method are chosen. In addition only LIF neuron model is available, and in this case we choose 2.5 "Small World Radius" which is an important parameter influencing the performance. When initialization is finished, the cube including all neurons with initialized weights are generated, Figure 3.9 shows the neurons and mapped features in the cube.

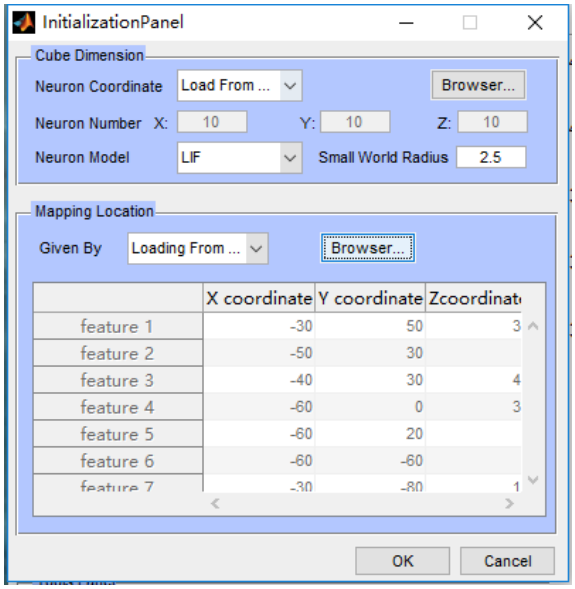


Figure 3.8: SNN cube initialization and mapping panel

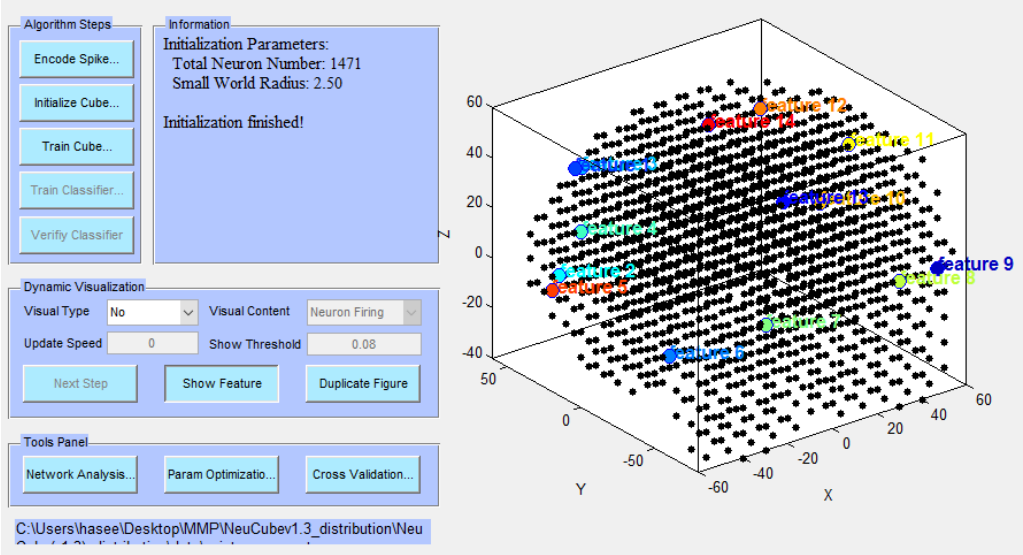


Figure 3.9: SNN cube after initialization

Then unsupervised and supervised learning will be applied. Figure 3.10 and 3.11 show the SNN cube training panel and deSNN classifier training panel respectively. There are many crucial parameters determining the performance of classifier, we need use grid search approach to find optimal ones. The optimal setting is described in the **Result** chapter.

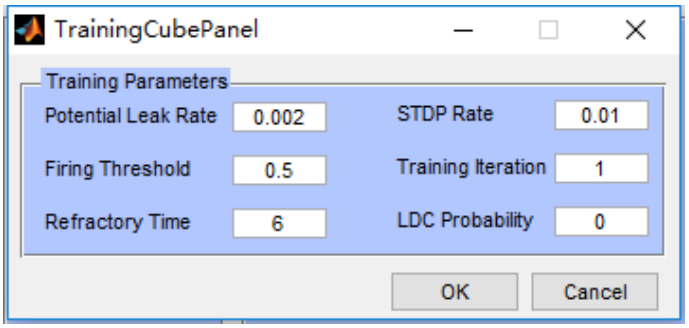


Figure 3.10: Training SNN cube panel

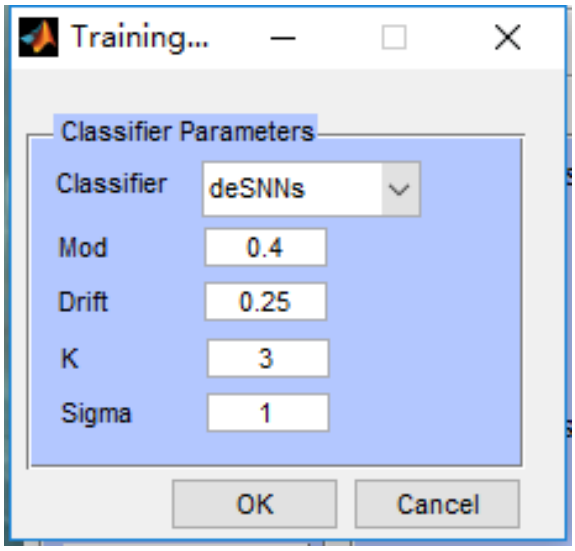


Figure 3.11: Classifier training panel

In NeuCube, we are also able to figure the status of weights before and after both SNN cube and classifier training phase. We take the EEG example to figure out weights how to update, it is stated that there are 1471 neurons existed in the cube, we only plot the weights connected each other from the first five neurons. In term of SNN cube, matrix 3.3 shows the initialized weights, and matrix 3.4 shows weights after training in the cube according to STDP learning rule. As to weights between cube and classifier layer, in this case, we set training set ratio as 0.5 which means we have 30 output neurons both in training and prediction phase, matrix 3.5 and matrix 3.6 shows weights between cube and deSNN classifier generated in training and prediction phase respectively, the statement is that the whole matrix has 30 lines and 1471 columns which means 30 output neurons in classifier layer and 1471 neurons in SNN cube.

$$\begin{pmatrix} 0 & 0.030661 & 0.033644 & 0 & 0.1046 & ... \\ 0 & 0 & 0.052697 & 0 & 0.021 & ... \\ 0 & 0.024634 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 0 & ... \\ ... & ... & ... & ... & ... & ... \end{pmatrix} \tag{3.3}$$

$$\begin{pmatrix} 0 & 0.067032 & 0.062942 & 0 & 0.19335 & ... \\ 0 & 0 & 0.081641 & 0 & 0.09338 & ... \\ 0 & 0 & 0 & 0.000585 & 0 & ... \\ 0 & 0.024634 & 0 & 0 & 0 & ... \\ 0 & 0 & 0 & 0 & 0 & ... \\ ... & ... & ... & ... & ... & ... \end{pmatrix} \tag{3.4}$$

$$\begin{pmatrix} -25.186 & -26.25 & -22.5 & -32 & -27 & ... \\ -23.186 & -26.25 & -25.75 & -32 & -27.25 & ... \\ -18.75 & -21.75 & -19.5 & -32 & -22.5 & ... \\ -19.5 & -20.75 & -18.75 & -32 & -23.25 & ... \\ -21.474 & -22.5 & -22.25 & -32 & -23.25 & ... \\ ... & ... & ... & ... & ... & ... \end{pmatrix} \tag{3.5}$$

$$\begin{pmatrix} -19.75 & -16.25 & -15.75 & -32 & -18.75 & ... \\ -22 & -16.5 & -16 & -32 & -16 & ... \\ -25.25 & -21.75 & -20.75 & -32 & -26.25 & ... \\ -22.498 & -19.25 & -19 & -32 & -23.75 & ... \\ -23.59 & -23.25 & -23.5 & -32 & -27.25 & ... \\ ... & ... & ... & ... & ... & ... \end{pmatrix} \tag{3.6}$$

Finally, results are plotted as Figure 3.12 shown where both true and prediction label are displayed and visualized.

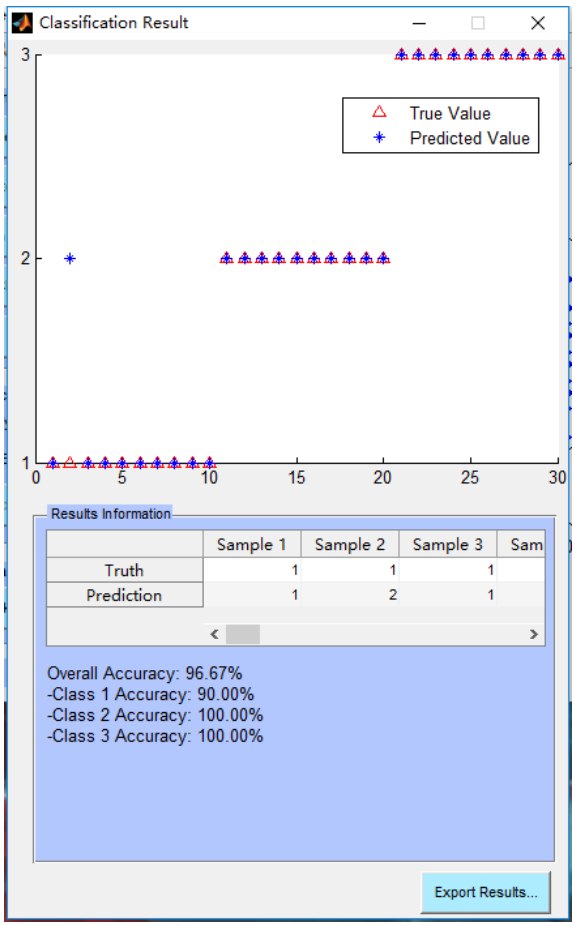


Figure 3.12: Result panel

3.4 Implementation of SVM and MLP in Python

In the experiments, the Python implementation of SVM and MLP was used via scikit-learn library[45]. In term of SVM, we use `svm.SVC()` to call the SVM classification function, the only parameter we need tune is the kernel function, here `kernel='rbf'` is set to use radial basis function kernel. As to MLP function, only one hidden layer with 50 neurons and 0.01 learning rate are chosen, other parameters are kept as default setting.

3.5 Learning process

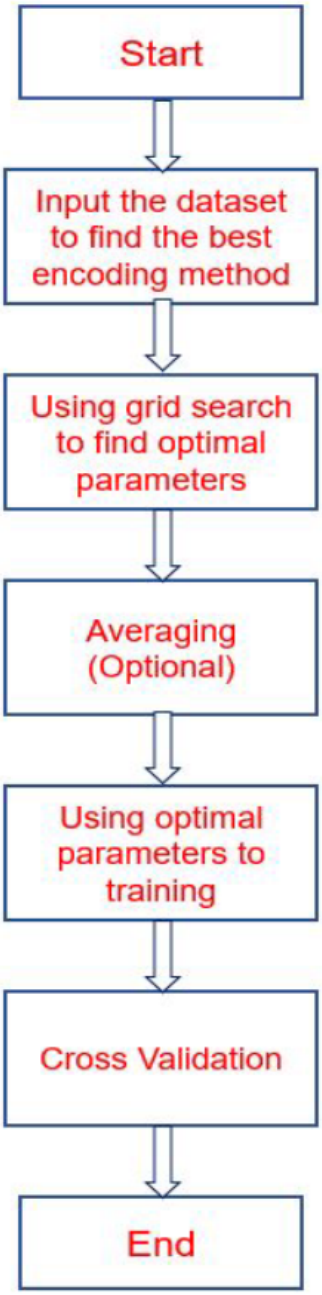


Figure 3.13: Flow chart of learning process

The learning process of the thesis is shown as the flow chart Figure 3.13. Initially we need select the optimal spike encoding method for our data to convert continuous signal into spike trains in order to later fair comparison between NeuCube and traditional machine learning methods. Then we use gird search method to figure out optimal parameters existed in each training model. Averaging procedure is optional for the task, as illustrated in section 3.2.3, the aim of averaging is to reduce computational cost, rather than an necessary step. The next step is to train our

model, in order to investigate the stability of model, 10 runs for each model is chosen during the training process, after which mean and standard deviation are recorded for showing performance of model. Finally, cross validation is used to reduce the effect of bias within dataset.

3.5.1 Cross Validation

In the thesis, we use k-fold cross validation to validate the performance of classifier. K-fold means that we separate the whole dataset as k parts, then pick up one of them once as testing set, and training classifier using the data from rest $(k-1)$ parts, after that we need loop k times, mean and standard deviation of accuracy from k times will be recorded. Usually, larger k is able to bring better performance. In the thesis, we do not have abundant samples, more folds cross validation can bring worse performance, so that 2-fold and 5-fold cross validation are used for both EEG and MEG classification applications.

3.5.2 Statement

It is noted that in the **Results** chapter, accuracy which is generated from 10 runs, is shown as the form of $\mu \pm \sigma$. By the way, for the no cross validation situation, μ and σ come from 10 accuracy, whereas for cross validation situation, each run generate one μ and one σ, where μ is the mean of 10 means, and σ is the mean of 10 standard deviations.

4

Results

In this part, results will be shown. In section 4.1, where results about EEG classification task are shown, whereas in section 4.2, MEG classification with 2 subjects will be shown firstly, then 20 subjects situation will be investigated.

4.1 Results for EEG data

Before we investigate the performance of NeuCube on EEG data, many parameters and setting of NeuCube should be set firstly. After grid search, optimal parameters and settings are found as shown in Table 4.1:

Data encoding method	Thresholding Representation(TR)
Neuron Coordinate in NeuCube	Brain Coordinates
Data mapping location	14-channels EEG locations
Spike threshold	0.5
Small World Radius	2.5
Potential Leak Rate	0.002
STDP rate	0.01
Firing Threshold	0.5
Refractory Time	6
LDC probability	0
Mod	0.4
Drift	0.25
Sigma	1
K	3

Table 4.1: Optimal parameters of NeuCube for EEG task.

Then we can test the performance of NeuCube classifier on public EEG data, the result is shown as Table 4.2

Classifier	no CV	2-fold CV	5-fold CV
NeuCube	87% ± 5%	76% ± 4.5%	58% ± 5.5%

Table 4.2: Accuracy($\mu \pm \sigma$) of NeuCube on EEG data with different numbers of fold CV.

It is satisfied that accuracy achieves around 87%, but the performance becomes worse when we implement cross validation, and there is a strange thing that per-

formance under 5-fold cross validation is worse than the performance under 2-fold cross validation. In my opinion, it is caused by insufficient samples in the dataset(60 samples in total), 5-fold means only 12 samples in the test set, thus few mis-classified samples may bias the performance significantly.

TR input encoding method is chosen in the before test, now all four available input encoding methods will be tested, meanwhile keep all other parameters and settings unchanged. According to Figure 4.1, TR achieves the highest accuracy, thus TR is chosen for the rest of tests.

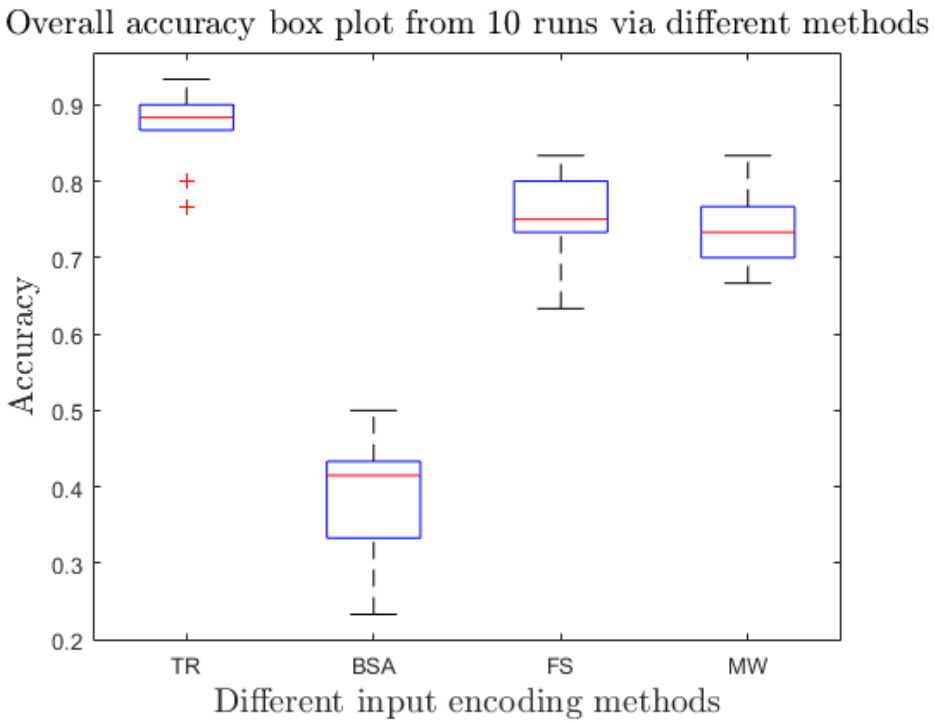


Figure 4.1: Accuracy box plot for different input encoding methods.

Then two traditional machine learning methods are used to test their performance on this three-class EEG public data. In order to make comparison fairly, EEG data is converted into spike trains firstly before using machine learning methods. Figure 4.2 shows the result that both methods perform very poorly with and without cross validation. Then we want to input the raw EEG data without input encoding into these two machine learning classifiers, the result is shown in Figure 4.3, we can see that accuracy is still lower than 50% for both classifiers with and without cross validation, which means these two classifiers fail.

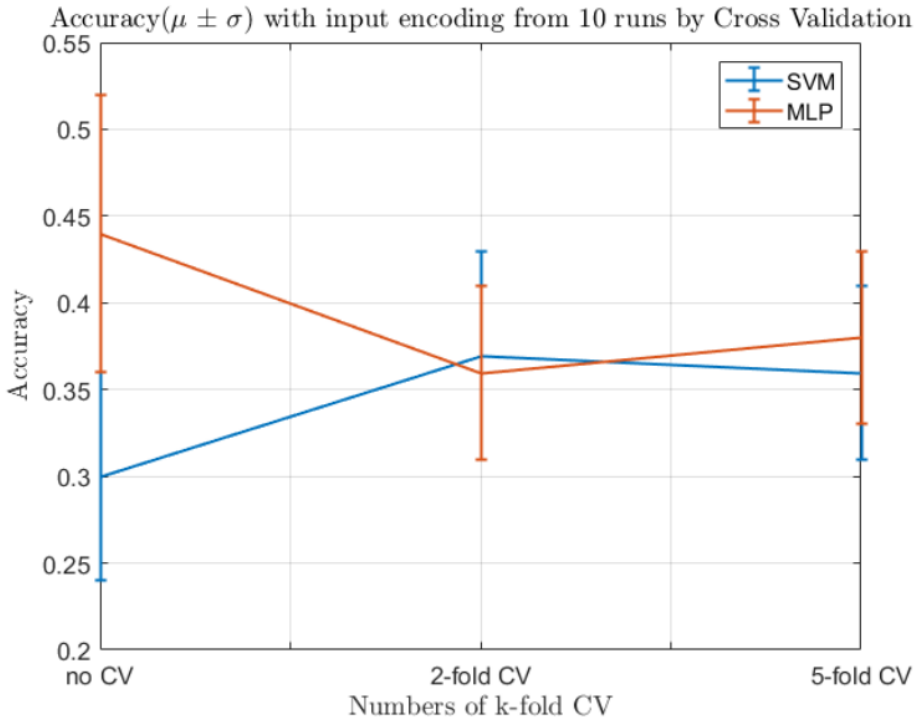


Figure 4.2: Accuracy with spike encoding via SVM and MLP methods.

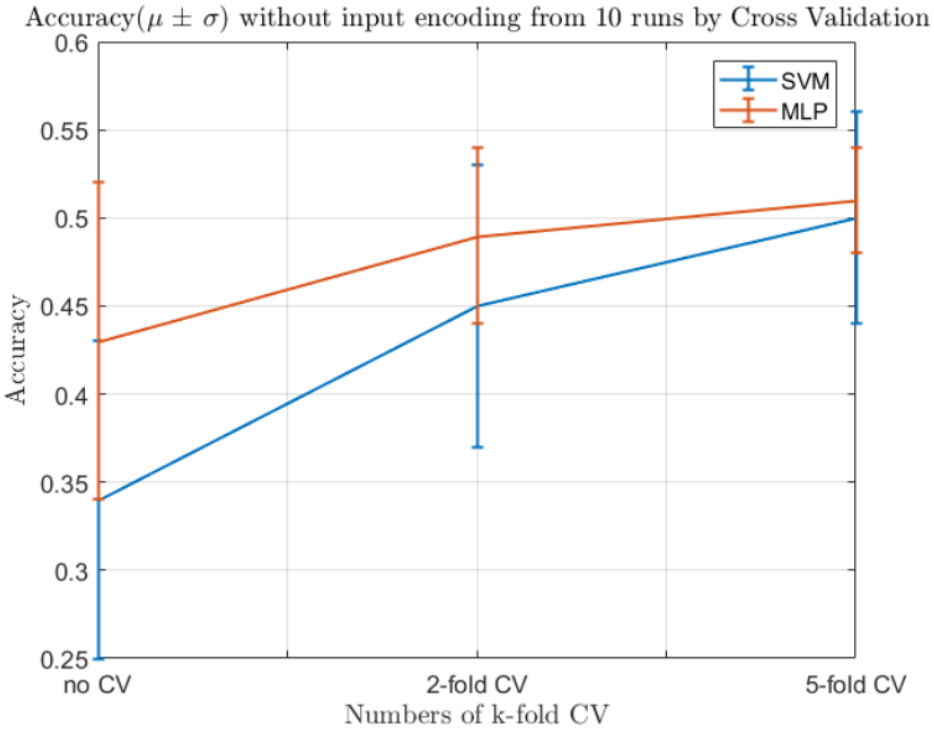


Figure 4.3: Accuracy without spike encoding via SVM and MLP methods.

In term of three-class public EEG data, three different methods are used to test and make comparison. It is obvious that NeuCube performs better than the other two

traditional machine learning methods. SVM and MLP can be seen as very efficient in machine learning field, but it does fail in this spatio-temporal EEG signals, but in my opinion, the reasons leading to this failure is that firstly, we may not have enough samples during the training procedure(only 30 samples in the training set due to training set ratio equals to 0.5), secondly this EEG data is generated by wrist movement where it is always difficult to distinct different classes/patterns in machine learning field. However, in other word, NeuCube is efficient to process and recognize spatio-temporal signals, it does not need transfer time sequence and channels matrix of each sample into a vector that is processed by classifier indeed. NeuCube not only uses the spatio-temporal signals naturally, but also intuitively mimic the processing procedure of human brain.)

4.2 Results for MEG data

4.2.1 2-subject situation

As in the EEG task, grid search is also used to figure out optimal parameters and settings in MEG task. From Table 4.3, we can see that, compared to Table 4.1, three parameters-small world radius, Mod, Drift are different, and we use automatically neuron coordinate and graph matching method for data mapping location. It is noticed that parameters used in the SNNcube training do not affect the result too much, whereas these three changed parameters determine the performance of classifier mainly.

Data encoding method	Thresholding Representation(TR)
Neuron Coordinate in NeuCube	Automatically
Data mapping location	Graph Matching
Spike threshold	0.5
Small World Radius	2
Potential Leak Rate	0.002
STDP rate	0.01
Firing Threshold	0.5
Refractory Time	6
LDC probability	0
Mod	0.35
Drift	0.1
Sigma	1
K	3

Table 4.3: Optimal parameters of NeuCube for MEG task.

As description in section 3.1.2, we have P1 and P3 both for MEG task. According to Bushra[43], there is no significant difference between P1 and P3 in the selected region of interests. As to this thesis work, P1 and P3 are tested at the same time initially by NeuCube even if P3 is the expected one. Without cross validation, training set ratio=0.8 bring quite satisfied performance to both two pulse signals,

which is shown in Table 4.4. Thus, it is "safe" for us to use P3 only for the following test, and with 2-fold and 5-fold cross validation, we get the result shown in Table 4.5.

Classifier	P1	P3
NeuCube	96% ± 2%	95% ± 1%

Table 4.4: Accuracy($\mu \pm \sigma$) of NeuCube on MEG P1 and P3 data.

Classifier	no CV	2-fold CV	5-fold CV
NeuCube	95% ± 1%	97% ± 1%	96% ± 2%

Table 4.5: Accuracy($\mu \pm \sigma$) of NeuCube on MEG P3 data with different numbers of fold CV.

According to Table 4.5, NeuCube still achieve good performance on binary MEG classification task under different numbers of fold cross validation, then performance of SVM and MLP classifier will be investigated following. As in the procedure of EEG task, the situation with input encoding is tested firstly. For both classifiers, accuracy does not reach 70% with and without cross validation, as shown in Figure 4.4. So far NeuCube performs still better than another two methods, however, accuracy of SVM and MLP classifiers is increased significantly when we only use raw data without spike encoding. From Figure 4.5, we can see that accuracy of SVM is higher than MLP situation among all three cross validation situation. Compared to accuracy achieved by NeuCube, only SVM is able to achieve the accuracy that is higher than 95%. In a short, we are able to achieve good performance using three methods under 2-subject situation, in my opinion one reason is that the MSNA values for these two subjects are quite different, the inhibitor one's MSNA value is 83.51 and the non-inhibitor one is -23.35, thus we can always find a classifier to separate two classes well. By the way, we can also see that without input encoding, the performance of SVM and MLP is much better than the situation with input encoding, I think the reason is that input encoding method reduces the information of raw data significantly, a spike train with only one and zero is not sufficient for traditional machine learning methods to recognize.

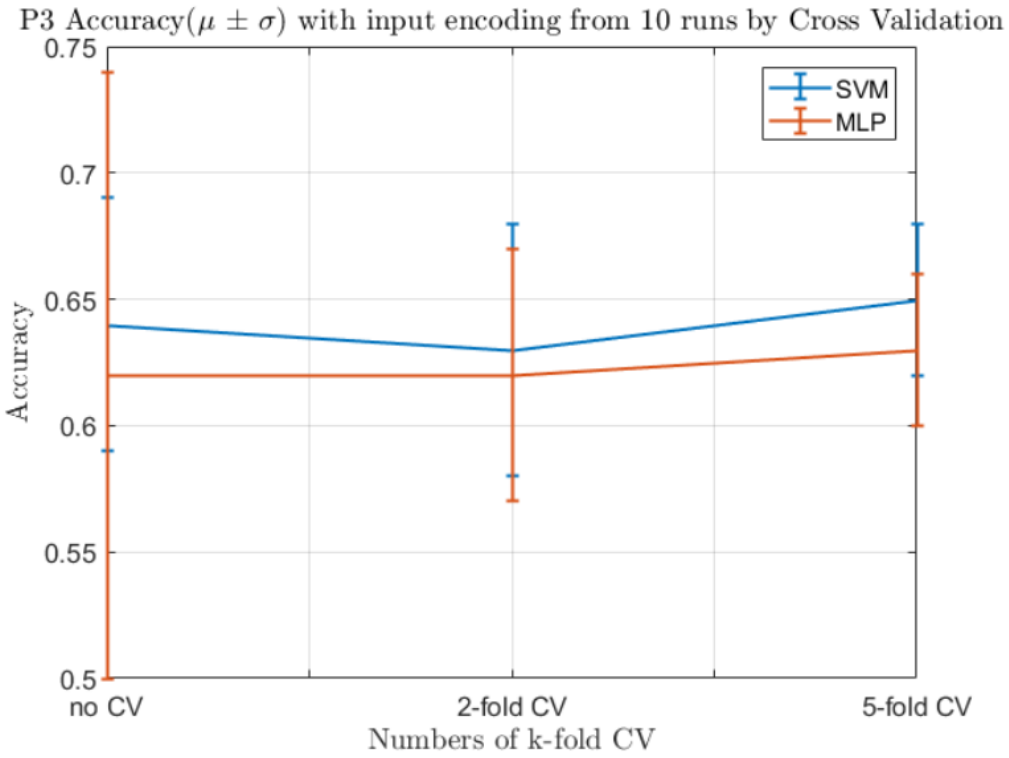


Figure 4.4: Accuracy on P3 data with spike encoding by different numbers of fold cross validation.

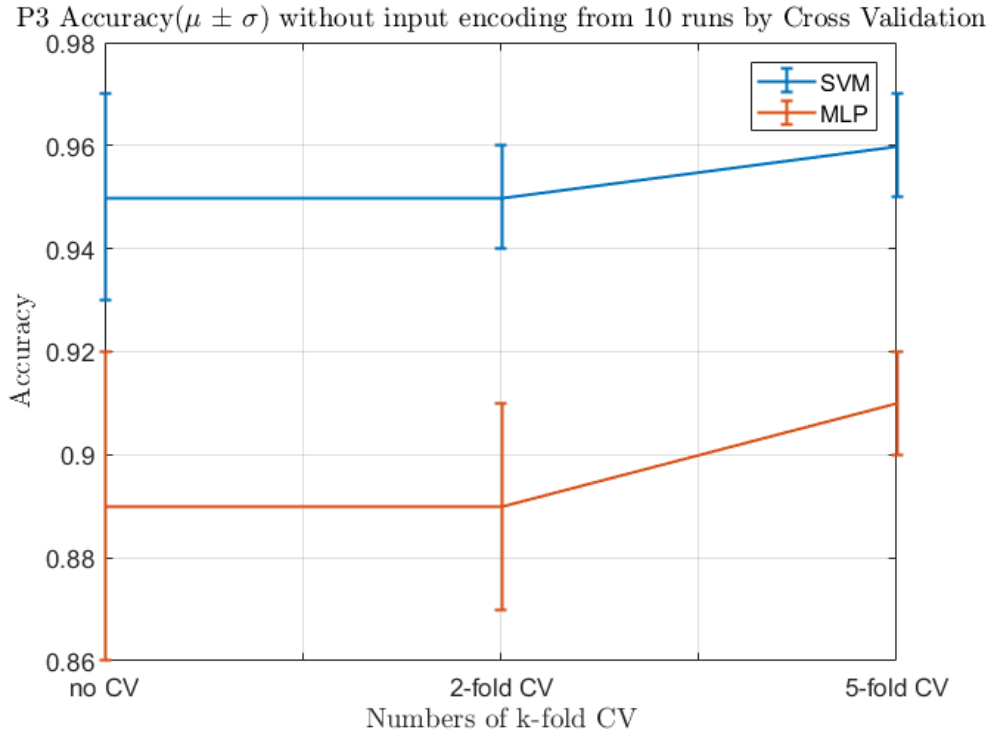


Figure 4.5: Accuracy on P3 data without spike encoding by different numbers of fold cross validation.

Next, averaging method will be investigated, but only time sequence averaging is utilized in 2-subject situation. Due to the issue of time consuming, we use SVM and MLP initially to test how many numbers of time sequence averaging used is satisfied. It is noticed that we use input encoding here because our aim is to investigate the influence of time sequence averaging on SNN. According to Figure 4.6, with 20-, 50-, 100-time sequence averaging, the accuracy is increasing for both SVM and MLP classifier, under 100-time sequence averaging, both SVM and MLP even achieve the similar accuracy compared to NeuCube's performance on raw data. Because 20-time sequence averaging cannot improve result significantly, only 50- and 100-time sequence averaging is used to test NeuCube classifier. From 4.6, we can see that time sequence averaging cannot keep the satisfied performance, only around 75% accuracy is achieved.

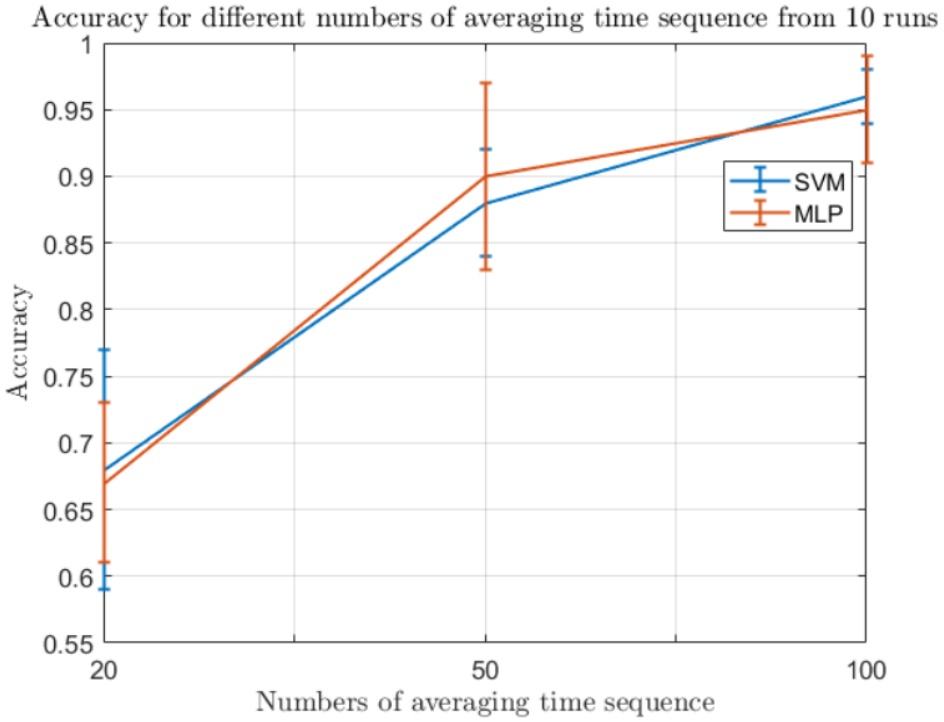


Figure 4.6: Accuracy on P3 data with spike encoding by different numbers of time sequence averaging.

Classifier	50	100
NeuCube	77.3% ± 5.5%	71.3% ± 3.8%

Table 4.6: Accuracy($\mu \pm \sigma$) of NeuCube with 50- and 100-time sequence averaging.

NeuCube still has very satisfied performance on 2-subject MEG binary classification task. For the other two traditional machine learning methods, however, with and without input encoding bring different results. SVM and MLP both perform excellent on raw MEG data with normalization only, but with input encoding, they perform worse. However, it is surprising that under input encoding situation, increasing

numbers of time sequence averaging can improve the performance significantly. In my opinion, input encoding procedure might lead to some noise or anomaly information which can be fixed by time sequence averaging approach, meanwhile NeuCube is able to fix this issue directly. If we use time sequence averaging approach in NeuCube, however, accuracy is decreased by around 20% with both 50- and 100-time sequence averaging even if NeuCube does spend much less time on training. It is strange that time sequence would improve performance for traditional machine learning methods, but impair for NeuCube, I think the reason is that NeuCube mimics human brain processing, whole time sequence contains important information, time sequence averaging might lost some information located between every spikes, but for SVM and MLP, time sequence averaging can be seen as just a preprocessing procedure that is not able to impair the performance.

4.2.2 20-subject situation

Under 20-subject situation, our task is still to train the binary classifier. Within 20 subjects, half subjects with higher than 30 MSNA value are belong to non-inhibitor class, and another half subjects are inhibitor. In this experiment, sample averaging approach is used to reduce the computational cost. Within each subject, 12 samples averaging is used, which means there are only 6(72/12=6) samples left in each subject, and 120(6×20subjects=120) samples left in whole dataset.

	NeuCube	SVM	MLP
Accuracy	86.5% ± 3.5%	72% ± 6%	74% ± 7%

Table 4.7: Accuracy($\mu \pm \sigma$) of three methods with input encoding

	SVM	MLP
Accuracy	90% ± 4%	84% ± 7%

Table 4.8: Accuracy($\mu \pm \sigma$) of SVM and MLP methods without input encoding

The results are shown in Table 4.7, we can see that NeuCube achieves around 87% accuracy which is quite satisfied because we only have around 100 samples for training and we have 20 subjects now. Due to NeuCube uses the data with input encoding, we will compare the performance of SVM and MLP with input encoding to NeuCube, and the result is that the accuracy both SVM and MLP achieve is about 10% lower than NeuCube's accuracy. If we do not use input encoding approach when training SVM and MLP classifier, the accuracy is increased. We can see from Table 4.8 that SVM achieves around 90% accuracy which is even slightly higher than NeuCube's performance, and MLP achieves around 84% accuracy. And as discussed in Section 4.2.1, for traditional machine learning methods, input encoding preprocessing would not help a lot. In addition, compared to the performance under 2-subject situation, all three methods perform worse, I think the reason is that the subjects we used in before situation have significant difference on MSNA values,

but in 20-subject situation, every subject has unique MSNA values, which means few difference on MSNA values may be existed among samples from two classes, for example, as MSNA value plot shown in Figure 3.2, the MSNA values of No.5 to No.10 subjects are closed to No.11 subject, but they are belong to two classes. Furthermore, because we always randomly select training set from all samples which mix all subjects, higher accuracy variance happened in 20-subject situation.

5
Conclusion

The thesis work investigates the performance of NeuCube on EEG/MEG data with two practical biomedical applications, and two traditional machine learning methods are used to make comparison to NeuCube. According to Chapter 4, we can see that the performance of NeuCube on both applications are quite satisfied, showing that NeuCube can be potentially attractive for neurological signals with brain related applications, and NeuCube has capacity to catch up both spatial and temporal information of signals.

In terms of two traditional machine learning methods used in the thesis, SVM and MLP success to classify the application with MEG signal, but achieve poor performance on the application with EEG signal. However, we cannot conclude that traditional machine learning method is good or worse at pattern recognition on physiological signals, because only two traditional methods are chosen in the thesis, meanwhile few preprocessing operation is utilized before we implement SVM and MLP methods.

In the future, there are many works can be continued. Firstly, as to task with MEG signal, we use graph matching method for mapping input neurons inside SNN cube, but if we know the coordinates of measured channels according to Talairach template, this prior information may help us achieve much better performance. Secondly, we could search many other preprocessing approaches on EEG/MEG data if we use traditional machine learning methods. Thirdly, deep learning methods such as convolution neural network, recurrent neural network, etc. can be utilized to make comparison to NeuCube framework. Fourthly, we can use NeuCube framework to test other biomedical applications to investigate whether or not NeuCube is suitable for other spatio-temporal signal classification task.

Bibliography

[1] Simpraga, S., Alvarez-Jimenez, R., Mansvelder, H. D., Gerven, J. M., Groeneveld, G. J., Poil, S. S., & Linkenkaer-Hansen, K. (2017). *EEG machine learning for accurate detection of cholinergic intervention and Alzheimer's disease.* Scientific Reports, 7(1), 5775.

[2] Hasasneh, A., Kampel, N., Sripad, P., Shah, N. J., & Dammers, J. (2018). *Deep Learning Approach for Automatic Classification of Ocular and Cardiac Artifacts in MEG Data.* Journal of Engineering, 2018.

[3] Johannesen, J. K., Bi, J., Jiang, R., Kenney, J. G., & Chen, C. M. A. (2016). *Machine learning identification of EEG features predicting working memory performance in schizophrenia and healthy adults.* Neuropsychiatric Electrophysiology, 2(1), 3.

[4] Stober, S., Sternin, A., Owen, A. M., & Grahn, J. A. (2015). *Deep feature learning for EEG recordings.* arXiv preprint arXiv:1511.04306.

[5] Timio M, Lippi G, Venanzi S, et al. (1997). *Blood pressure trend and cardiovascular events in nuns in a secluded order: a 30-year follow-up study.* Blood Pressure, 6(2), 81-87.

[6] Capecci, E., Morabito, F. C., Campolo, M., Mammone, N., Labate, D., & Kasabov, N. (2015). *A feasibility study of using the neucube spiking neural network architecture for modelling Alzheimer's disease eeg data.* In Advances in neural networks: Computational and Theoretical Issues (pp. 159-172). Springer, Cham.

[7] Kasabov, N., & Capecci, E. (2015). *Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes.* Information Sciences, 294, 565-575.

[8] Capecci E, Espinosa-Ramos J I, Mammone N, et al. (2015). *Modelling absence epilepsy seizure data in the neucube evolving spiking neural network architecture.* In 2015 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[9] Kasabov, N. K., Doborjeh, M. G., & Doborjeh, Z. G. (2017). *Mapping, learning, visualization, classification, and understanding of fMRI data in the NeuCube evolving spatiotemporal data machine of spiking neural networks.* IEEE Transactions on Neural Networks and Learning Systems, 28(4), 887-899.

[10] Murli, N., Kasabov, N., & Handaga, B. (2014). *Classification of fMRI data in the NeuCube evolving spiking neural network architecture.* In International Conference on Neural Information Processing (pp. 421-428). Springer, Cham.

[11] Doborjeh, M. G., Capecci, E., & Kasabov, N. (2014). *Classification and segmentation of fMRI spatio-temporal brain data with a NeuCube evolving spiking neural network model.* In 2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS) (pp. 73-80). IEEE.

[12] Kasabov N, Feigin V, Hou Z G, et al. (2014). *Evolving spiking neural networks for personalised modelling, classification and prediction of spatio-temporal patterns with a case study on stroke.* Neurocomputing, 134, 269-279.

[13] Peng, L., Hou, Z. G., Kasabov, N., Bian, G. B., Vladareanu, L., & Yu, H. (2015). *Feasibility of Neucube spiking neural network architecture for EMG pattern recognition.* In 2015 International Conference on Advanced Mechatronic Systems (ICAMechS) (pp. 365-369). IEEE.

[14] Tu, E., Kasabov, N., & Yang, J. (2017). *Mapping temporal variables into the neucube for improved pattern recognition, predictive modeling, and understanding of stream data.* IEEE Transactions on Neural Networks and Learning Systems, 28(6), 1305-1317.

[15] Electroencephalography, `https://en.wikipedia.org/wiki/Electroencephalography`

[16] Hämäläinen, M., Hari, R., Ilmoniemi, R. J., Knuutila, J., & Lounasmaa, O. V. (1993). *Magnetoencephalography—theory, instrumentation, and applications to noninvasive studies of the working human brain.* Reviews of Modern Physics, 65(2), 413.

[17] Sarvas, J. (1987). *Basic mathematical and electromagnetic concepts of the biomagnetic inverse problem.* Physics in Medicine & Biology, 32(1), 11.

[18] Malmivuo, P., Malmivuo, J., & Plonsey, R. (1995). *Bioelectromagnetism: principles and applications of bioelectric and biomagnetic fields.* Oxford University Press, USA.

[19] Gerstner, W., & Kistler, W. M. (2002). *Spiking neuron models: Single neurons, populations, plasticity.* Cambridge University Press.

[20] Schmidt, Robert F.; Thews, Gerhard. (1983). *Human physiology.* Springer-Verlag. p. 725. ISBN 978-3540116691.

[21] Kasabov, N. K., Doborjeh, M. G., & Doborjeh, Z. G. (2017). *Mapping, learning, visualization, classification, and understanding of fMRI data in the NeuCube evolving spatiotemporal data machine of spiking neural networks.* IEEE Transactions on Neural Networks and Learning Systems, 28(4), 887-899.

[22] Kasabov, N. K. (2014). *NeuCube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data.* Neural Networks, 52, 62-76.

[23] NeuCube
`https://kedri.aut.ac.nz/R-and-D-Systems/neucube`

[24] Kasabov, N. (2010). *To spike or not to spike: A probabilistic spiking neuron model.* Neural Networks, 23(1), 16-19.

[25] Hebb, D. O. (2005). *The organization of behavior: A neuropsychological theory.* Psychology Press.

[26] Song, S., Miller, K. D., & Abbott, L. F. (2000). *Competitive Hebbian learning through spike-timing-dependent synaptic plasticity.* Nature Neuroscience, 3(9), 919.

[27] Kasabov, N., Dhoble, K., Nuntalid, N., & Indiveri, G. (2013). *Dynamic evolving spiking neural networks for on-line spatio-and spectro-temporal pattern recognition.* Neural Networks, 41, 188-201.

[28] Kasabov, N. K. (2007). *Evolving connectionist systems: the knowledge engineering approach.* Springer Science & Business Media.

[29] Schrauwen, B., & Van Campenhout, J. (2003). *BSA, a fast and accurate spike train encoding scheme.* In Proceedings of the International Joint Conference on Neural Networks, 2003. (Vol. 4, pp. 2825-2830). IEEE.

[30] Lichtsteiner, P., & Delbruck, T. (2005). *A 64×64 AER logarithmic temporal derivative silicon retina.* Research in Microelectronics and Electronics, 2, 202–205.

[31] Nuntalid, N., Dhoble, K., & Kasabov, N. (2011). *EEG classification with BSA spike encoding algorithm and evolving probabilistic spiking neural network.* In LNCS: vol. 7062 (pp. 451–460). Springer.

[32] Delbruck, T., & Lichtsteiner, P. (2007). *Fast sensory motor control based on event-based hybrid neuromorphic-procedural system.* IEEE International

Symposium on Circuits and Systems (pp. 845-848). IEEE.

[33] Kasabov N, Scott N M, Tu E, et al. (2016). *Evolving spatio-temporal data machines based on the NeuCube neuromorphic framework: design methodology and selected applications.* Neural Networks, 78, 1-14.

[34] Lowel, S., & Singer, W. (1992). *Selection of intrinsic horizontal connections in the visual cortex by correlated neuronal activity.* Science, 255(5041), 209-212.

[35] Jesper S and Wulfram G. (2010) *Spike-timing dependent plasticity.* Scholarpedia, 5(2):1362.

[36] Bi, G. Q. and Poo, M. M. (1998). *Synaptic modifications in cultured Hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type.* J Neurosci, 18:10464-72.

[37] Song, S., Miller, K.D., and Abbott, L.F. (2000). *Competitive Hebbian learning through spike-timing-dependent synaptic plasticity.* Nat Neurosci 3, 919-926.

[38] Ponulak, F., & Kasinski, A. (2011). *Introduction to spiking neural networks: Information processing, learning and applications.* Acta Neurobiologiae Experimentalis, 71(4), 409-433.

[39] Kasabov, N., Dhoble, K., Nuntalid, N., & Indiveri, G. (2013). *Dynamic evolving spiking neural networks for on-line spatio-and spectro-temporal pattern recognition.* Neural Networks, 41, 188-201.

[40] Thorpe, S., & Gautrais, J. (1998). *Rank order coding.* In Computational neuroscience (pp. 113-118). Springer, Boston, MA.

[41] Bishop, C. M. (2006). *Pattern recognition and machine learning.* Springer.

[42] MA Nielsen. (2015). *Neural Networks and Deep Learning.* Determination Press.

[43] Syeda, B. R. (2018). *Bringing MEG Towards Clinical Applications.* Diss. Göteborgs Universitet, 2018.

[44] NeuCube v1.3 User Manual. (2016). Auckland University of Technology.

[45] Scikit-learn.
`https://scikit-learn.org/stable/index.html`