

**SUBJECT-INDEPENDENT
ERP-BASED BRAIN-COMPUTER
INTERFACE USING ADAPTIVE
AND ENSEMBLE LEARNING**

by

Anh Kha Vo

Thesis submitted in fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

School of Electrical and Data Engineering
Faculty of Engineering and Information Technology
University of Technology Sydney

2018

Abstract

The breakthrough of deep learning in recent years opens up a wide range of applications, mostly from computer vision to brain-computer interface (BCI), which is the research topic of this thesis. In our study, we propose a unified framework of the subject-independent event-related potential (ERP) based BCI. In other words, we attempt to overcome a major challenge of discrepancies in ERP patterns across different subjects by employing a deep learning technique, accompanied with different strategies of bagging-stacking ensemble, dynamic stopping, and adaptive incremental learning. The main contributions of this thesis are summarised as follows:

(1) Employ the subject-adversarial neural network (SANN) to learn the optimal representation for the original preprocessed ERP features. This network serves as a feature encoder which attempts to find a better feature space that is subject-independent for the testing procedure.

(2) Employ three machine learning algorithm as base learners: support vector machine (SVM), Fisher’s discriminant analysis (FDA), and fully-connected feedforward neural network (NN). The soft scores of these base learners serves as inputs for the ensemble strategy which is comprised of two widely-used techniques: bagging and stacking. The goal of the whole ensemble strategy is to perform the binary classification problem of ERP trials with lower generalisation error and higher accuracy.

(3) Employ two post-processing tasks for the P300-Speller (P3S) to enhance the information transfer rate, namely dynamic stopping (DS), and adaptive (or incremental) learning (AL). DS is performed to let the system produce the subject’s output whenever the algorithm is sufficiently confident about its decision, while AL is used to reinforce the existing classifier by analytically integrating newly-classified samples into the classifier’s decision function in real-time.

The research of the thesis is motivated by the strong representative characteristics of ERP features in EEG signal. The distinctive properties of ERP, especially P300 component in this thesis, and its variability across multiple subjects, can be well-exploited by the high complexity and employment of deep adversarial neural network framework. The robust representation features output of this network are served as learning features for the subsequent step of ensemble learning by multiple machine learning algorithms. Finally, some post-processing and enhancement methods specifically proprietary to this thesis such as dynamic stopping and incremental learning are applied to yield an additional boost in our performance.

Certificate of Original Authorship

I, Anh Kha Vo, declare that this thesis, is submitted in fulfillment of the requirements for the award of the degree of Doctor of Philosophy, in the School of Electrical and Data Engineering, Faculty of Engineering and Information Technology, at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise reference of acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature:

Production Note:
Signature removed prior to publication.

Anh Kha Vo

Date: 20th, October 2018.

Acknowledgements

First and foremost, I would like to express my sincerest gratitude to my principle supervisor, Prof. Eryk Dutkiewicz (University of Technology Sydney, Australia), who has helped me through some difficulties during my PhD study. He has been always encouraged me to strive for excellence in my career. This work would not have been completed without his invaluable guidance and generous support.

I would also like to acknowledge my co-supervisors Dr Rifai Chai, and Dr Diep Nguyen for their support, advice and encouragement throughout this work.

Last but not least, I am so much indebted to my dear parents, my sisters, and my dear partner for their infinite and unconditional love during my abroad study.

List of Publications

The publications of the author during his PhD study are listed as follow.

Journal Papers:

[1] **Kha Vo**, Thuy Pham, Diep Nguyen, Kha Ha & Eryk Dutkiewicz, “Subject-Independent ERP-based Brain-Computer Interfaces”, IEEE Transactions on Neural System and Rehabilitation Engineering, Vol.26, No.4, pp. 1-10, **2018**. (related to Chapter 3, 5, 6)

[2] **Kha Vo**, Rifai Chai, Kha Ha & Eryk Dutkiewicz, “Subject-Adversarial Neural Network for ERP Classification”, IEEE Transactions on Biomedical Engineering, **2018** (Under Review/Submission) (related to Chapter 4).

[3] **Kha Vo**, Kha Ha & Michael Blumenstein, “Extraction of Dynamic Trajectory on Multi-Stroke Static Handwriting Images Using Loop Analysis and Skeletal Graph Model”, IEEE REV Journal on Electronics and Communications, Vol. 6, No. 1-2, **2016**.

Conference Papers:

[4] **Kha Vo** & Eryk Dutkiewicz, “Optimal Length-Constrained Segmentation and Subject-Adaptive Learning for Real-time Arrhythmia Detection”, ACM/IEEE Conference on Connected Health: Applications, Systems, and Engineering Technologies (CHASE), Big Data for Health Workshop (BIG-DATA4HEALTH) , Washington DC, USA, **2018**. (related to Chapter 6)

-
- [5] **Kha Vo**, Diep Nguyen, Kha Ha & Eryk Dutkiewicz, “Subject-Independent P300 BCI using Ensemble Classifier, Dynamic Stopping and Adaptive Learning”, IEEE Global Communications Conference (GLOBECOM), Singapore, **2017**. (related to Chapter 3, 6)
- [6] **Kha Vo**, Diep Nguyen, Kha Ha & Eryk Dutkiewicz, “Real-Time Analysis on Ensemble SVM Scores to Reduce P300-Speller Intensification Time”, 39th International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Jeju Island, Korea, **2017**. (related to Chapter 5)
- [7] **Kha Vo**, Diep Nguyen, Kha Ha & Eryk Dutkiewicz, “Dynamic Stopping Using eSVM Scores Analysis for Event-Related Potential Brain-Computer Interfaces”, 11th International Symposium on Medical Information and Communication Technology (ISMICT), Lisbon, Portugal, **2017**. (related to Chapter 6)
- [8] **Kha Vo** & Kha Ha, “Extraction of dynamic features from static handwritten image and recognition using Hidden Markov model”, International Symposium on Electrical and Electronics Engineering (ISEE), Saigon, Vietnam, **2015**.
- [9] Kha Ha, **Kha Vo** & H. Dinh, “Brainwave-Controlled Applications with the Emotiv EPOC Using Support Vector Machine”, 4th International Conference on Information Technology, Computer, And Electrical Engineering (ICI-TACEE), Semarang, Indonesia, **2016**. (related to Chapter 3)
- [10] Kha Ha & **Kha Vo**, “Real-Time Brainwave-Controlled Interface Using P300 Component in EEG Signal Processing”, 12th IEEE-RIVF International Conference on Computing and Communication Technologies, Hanoi, Vietnam, **2016**. (related to Chapter 3)

Abbreviations

Ag Silver

AgCl Silver-chloride

AL Adaptive Learning

ALI Adversarially Learned Inference

ALS Amyotrophic Lateral Schlerosis

ANN Artificial Neural Network

Au Gold

Bagging Bootstrap Aggregating

BCI Brain-Computer Interface

BLDA Bayesian Linear Discriminant Analysis

BMI Brain-Machine Interface

CNN Convolutional Neural Network

CNS Central Nervous System

CNV Contingent Negative Variation

CT Computed Tomography

DANN Domain Adversarial Neural Network

DCGANN Deep Convolutional Generative Adversarial Neural Network

deoxy-Hb Deoxy-hemoglobin

DNI Direct Neural Interface

DNN Deep Neural Network

DS Dynamic Stopping

ECoG Electrocorticography

EEG Electroencephalography

ERD Event-Related Desynchronisation

ERP Event-Related Potential

ERS Event-Related Synchronisation

FDA Fisher's Discriminant Analysis

FES Functional Electrical Stimulator

fMRI Functional Magnetic Resonance Imaging

FN False Negative

fNIRS Functional Near Infrared Spectroscopy

FP False Positive

GANN Generative Adversarial Neural Network

GPU Graphics Processing Units

ISVM Incremental Support Vector Machine

KKT Karush-Kuhn Tucker

KL Kullback-Leibler

LDA Linear Discriminant Analysis

LFP Local Field Potential

LgR Logistic Regression

LR Linear Regression

MLP Multilayer Perceptrons

MMI Mind-Machine Interface

MRI Magnetic Resonance Imaging

N1 Negative 100ms

N100 Negative 100ms

N2 Negative 200ms

N200 Negative 200ms

NN Neural Network

OOF Out-of-fold

oxy-Hb Oxy-hemoglobin

P1 Positive 100ms

P100 Positive 100ms

P2 Positive 200ms

P200 Positive 200ms

P3 Positive 300ms

P300 Positive 300ms

P3S P300-Speller

PCA Principle Component Analysis

PET Positron Emission Tomography

PPT Peak-to-Peak Thresholding

Pt Platinum

ReLU Rectified Linear Unit

SANN Subject Adversarial Neural Network

SCI Spinal Cord Injury

SCP Slow Cortical Potential

SGD Stochastic Gradient Descent

SMR Sensorimotor Rhythm

Sn Tin

SNR Signal-to-Noise Ratio

SPECT Single Photon Emission Computed Tomography

SQUID Superconducting Quantum Interference Device

SSVEP Steady-State Evoked Potential

SVM Support Vector Machine

TN True Negative

TP True Positive

t-SNE t-distributed Stochastic Neighbor Embedding

TTD Thought Translation Device

UCLA University of California, Los Angeles

VEP Visually Evoked Potential

WPPT Windowed Peak-to-Peak Thresholding

Contents

Chapter

1	Introduction	1
1.1	General Background	1
1.2	Motivations	3
1.2.1	Subject-Independence	3
1.2.2	A Significantly Faster but Accuracy-Preserving BCI . . .	4
1.2.3	An Adaptive Learning Algorithm in Real-time	5
1.3	Contribution of the Thesis	5
1.3.1	Subject-Adversarial Neural Network as a Feature Encoder	6
1.3.2	An Ensemble Learning Strategy of 3 Base Learners . . .	6
1.3.3	Dynamic Stopping and Adaptive Learning	7
1.4	Organisation of the thesis	7
2	LITERATURE REVIEW	11
2.1	Brain Neurophysiology	11
2.2	Brief Introduction on Brain-Computer Interface (BCI)	13
2.3	BCI Data Acquisition Technologies	14
2.3.1	Intra-cortical Recording	14
2.3.2	Electrocorticography (ECoG)	15
2.3.3	Magneto-encephalography (MEG)	16
2.3.4	Functional Near Infrared Spectroscopy (fNIRS)	16
2.3.5	Functional Magnetic Response Imaging (fMRI)	17
2.3.6	Positron Emission Tomography (PET)	18
2.3.7	Single Photon Emission Computed Tomography (SPECT)	18
2.3.8	Electroencephalography (EEG)	19
2.4	EEG-based BCI	20

2.4.1	Event-Related Potential (ERP)	21
2.4.2	Steady-State Evoked Potential (SSVEP)	22
2.4.3	Slow Cortical Potential (SCP)	22
2.4.4	Sensorimotor Rhythms (SMR)	23
2.4.5	Mental Tasks	24
2.5	ERP-Based BCI	25
2.6	Chapter Summary	31
3	EVENT-RELATED POTENTIALS, DATA ACQUISITION, AND FEATURE ENGINEERING	33
3.1	Event-Related Potentials (ERP)	33
3.1.1	A Brief History of ERP	33
3.1.2	The ERP types in this Thesis	35
3.2	Data Acquisition Framework	40
3.2.1	P300-Speller Interface	40
3.2.2	Flashing Paradigm	41
3.3	Evaluation Metrics	45
3.3.1	Accuracy	45
3.3.2	Iteration	45
3.3.3	Bit rate	46
3.3.4	Precision	46
3.3.5	Sensitivity (or Recall)	47
3.3.6	False Counts	47
3.3.7	F-score	47
3.4	Signal Preprocessing	48
3.4.1	Averaging	48
3.4.2	Artifact Removal	53
3.4.3	EEG Filtering	56
3.4.4	Decimation	57
3.4.5	Benefits of Preprocessing	61
3.4.6	Feature Construction and Analysis	62
3.5	Dataset Specifications	63
3.6	Chapter Summary	66

4	SUBJECT-ADVERSARIAL NEURAL NETWORK	67
4.1	Motivation	67
4.2	Adversarial Neural Network Review	69
4.2.1	Generative Adversarial Neural Network (GANN)	69
4.2.2	Applications of GANN	71
4.2.3	Domain-Adversarial Neural Network (DANN)	71
4.3	Subject-Adversarial Neural Network (SANN)	73
4.3.1	Context Motivation	73
4.3.2	Theory of SANN	74
4.3.3	Theory Proofs	77
4.3.4	SANN Algorithm	80
4.4	An SANN Case Study	81
4.5	Experiments	87
4.5.1	t-Distributed Stochastic Neighbor Embedding (t-SNE)	87
4.5.2	Classification Improvement of SANN	87
4.6	Chapter Summary	90
5	ENSEMBLE LEARNING	93
5.1	Bagging	94
5.1.1	Brief Review of Bagging	94
5.1.2	Bagging reduces generalisation errors	95
5.2	Stacking	97
5.2.1	Overview and Algorithm of Stacking	97
5.2.2	Stacked learner performs best if base learners are com- parable	97
5.3	Combining Bagging and Stacking: Avoid Overfitting	99
5.4	Base Learners	100
5.4.1	Support Vector Machine	101
5.4.2	Neural Network	104
5.4.3	Fisher's Discriminant Analysis	107
5.4.4	Hyperparameters Selection for Base Learners	108
5.5	Meta Combiner: Linear Regression	109
5.6	Performance Evaluation	110
5.7	Chapter Summary	114

6	DYNAMIC STOPPING AND ADAPTIVE LEARNING	117
6.1	Dynamic Stopping	117
6.1.1	Dynamic Stopping Framework	118
6.1.2	Dataset Partitioning	119
6.1.3	Proposed Criteria for Dynamic Stopping	122
6.1.4	Estimation $\tilde{\theta}_1$ and $\tilde{\theta}_2$ through dynamic training	125
6.2	Adaptive Learning using Incremental SVM	126
6.2.1	Adaptive Learning as a Black Box	126
6.2.2	Brief Review on SVM	128
6.2.3	ISVM Theory	129
6.3	Experiments	134
6.3.1	Influence of Partitioning on Adaptive Learning	134
6.3.2	Classification Schemes	136
6.3.3	Grid Search Result	137
6.3.4	Validation Stage Performance	139
6.3.5	Influence of Expected Accuracy Parameter a of the Grid Search Algorithm	140
6.3.6	Influence of DS and AL on the Performance of Each Subject	141
6.3.7	Comparisons with Related Studies	142
6.4	Chapter Summary	143
7	CONCLUSIONS AND FUTURE WORK	145
7.1	Innovations in the Thesis	145
7.1.1	Application of Adversarial Neural Network Training on an ERP-based BCI	145
7.1.2	An Ensemble Strategy of Bagging and Stacking	146
7.1.3	Dynamic Stopping and Adaptive Learning in Real-time	146
7.2	Future Work	147
7.2.1	Perform SANN of Deep Architectures	147
7.2.2	Optimal Segmentation of EEG Data in Real-time	147
7.2.3	Language Model	148

Appendix

A	Derivations of Some Expressions in Chapter 4	149
----------	---	------------

Bibliography	153
---------------------	------------

Tables

Table

2.1	Comparison of Brain Data Acquisition Technologies.	19
3.1	The Confusion Matrix Performance of WPPT and Windsorizing	56
3.2	Dataset Specifications	64
3.3	Subject Information of Datasets in This Thesis	65
4.1	The SANN Architectures of Different Classification Cases in the Case Study	85
4.2	The Classification Performance with and without SANN on Dataset A.	91
5.1	Hyperparameters Grid Search for Base Learners.	114
5.2	The Classification Performance of Bagging and Stacking.	114
6.1	Classification Schemes' Settings.	137
6.2	The grid search results of 4 classification schemes.	138
6.3	Comparison of Our Study and Related Studies, Conducted on The Subject-Independent Basis.	142
6.4	Classification Result Averaged on All Validation Subjects.	144

Figures

Figure

1.1	Relations of the main chapters in this thesis.	8
2.1	The human brain architecture.	12
2.2	The diagram of a typical BCI.	13
2.3	The most widely used methods for data acquisition in BCI. Electroencephalography (EEG) is the data acquisition method used in this thesis.	15
2.4	Typical types of EEG-based BCI, including the event-related potential (ERP) that is the main theme of this thesis.	21
3.1	The typical ERP components.	40
3.2	The averaged ERP components found in this thesis.	40
3.3	The timeline of signal stream and flashing paradigm of a one-letter spelling session.	42
3.4	The steps of the row-column flashing paradigm of the P3S . . .	43
3.5	The classification process of the row-column flashing paradigm after each iteration.	44
3.6	The averaged ERP waveforms of subject A	50
3.7	The averaged ERP waveforms of subject B	51
3.8	The averaged ERP waveforms of subject C	52
3.9	The illustration of highpass filtering with cut-off frequency of 20 Hz on an EEG signal segment of 1.5 seconds.	58
3.10	The illustration of lowpass filtering with cut-off frequency of 0.1 Hz on an EEG signal segment of 15 seconds.	59

3.11	The illustration of bandpass filtering with cut-off frequencies of [0.1, 20] Hz on an EEG signal segment of 15 seconds.	60
3.12	The filter response of the Butterworth bandpass filter with cut-off frequencies of [0.1, 20] Hz.	61
3.13	The trial-wise accuracy performance with and without preprocessing techniques	62
3.14	The symbol-wise accuracy performance with and without preprocessing techniques	62
4.1	The general workflow of GANN.	70
4.2	The general workflow of DANN.	73
4.3	The general workflow of SANN.	74
4.4	The architecture of subject-adversarial neural network (SANN) used in this thesis.	81
4.5	The initial distributions of data samples of two subjects mentioned in the case study.	83
4.6	The sample distribution of case A in the SANN case study. . . .	84
4.7	The sample distribution of case B in the SANN case study. . . .	84
4.8	The sample distribution of case C in the SANN case study. . . .	85
4.9	The sample distribution of case D in the SANN case study. . . .	86
4.10	The classification performance of 4 cases in the SANN case study. Train domain and test domain refer to the classification accuracy of samples from subject 1 and subject 2, respectively. Train class and test class refer to the classification accuracy of samples from class 1 and class 2, respectively.	86
4.11	The data distribution before SANN learning. The class labels are depicted by marker number (0 for non-target, 1 for target), and the subject labels are depicted by colours.	88
4.12	The data distribution after SANN learning. The class labels are depicted by marker number (0 for non-target, 1 for target), and the subject labels are depicted by colours.	89

4.13	The evaluation steps of SANN. This figure is for evaluating the performance with SANN, as shown in Table 4.2. The performance without SANN is achieved by simply directly plugging in $\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}}$ into the training algorithms of SVM, FDA, and NN, equivalent to bypassing the SANN learning and feature encoding steps.	90
5.1	A topology of a conventional fully-connected feedforward neural network with 3 hidden layers.	105
5.2	The Hyperparameter Tuning Performance on two datasets . . .	112
5.3	The number of False Counts of Various Ensemble Combinations across 3 Base Learners.	113
6.1	The EEG signal timeline and row-column flashing paradigm. . .	119
6.2	The main diagram of our proposed dynamic stopping framework for a one-letter P3S session.	120
6.3	Time vs. new learning samples, with no partitioning. 4 lines indicate 4 different numbers of existing samples in the available classifier solution.	136
6.4	Time vs. new learning samples, with partitioning. 4 lines indicate 4 different numbers of existing samples in the available classifier solution.	136
6.5	Time vs. existing samples in the classifier solution and classifier partitions.	137
6.6	The accuracies of 4 schemes on the learning set with different values of DS parameters $\hat{\theta}_1$ and $\hat{\theta}_2$	138
6.7	The number of iterations of 4 schemes on the learning set with different values of DS parameters $\hat{\theta}_1$ and $\hat{\theta}_2$	139
6.8	The subject-wise results for all test subjects using different input values of dynamic stopping parameters $\hat{\theta}_1$ and $\hat{\theta}_2$	141

Chapter 1

Introduction

In this chapter, the historic development of brain-computer interface is presented in Section 1.1. Then section 1.2 addresses the motivations of this thesis, followed by the summary of the main contributions in section 1.3. Finally, the organisation of the thesis is described in section 1.4

1.1 General Background

A brain-computer interface (BCI) provides a means of direct communication between a brain, of human or animals, and the computer, or an external device (i.e., spelling robot assistant, wheelchair, thought indicator, robotic arm, etc) driven by the computer. BCIs are also known as mind-machine interface (MMI), brain-machine interface (BMI), or direct neural interface (DNI). Since the first BCI research conducted by the University of California, Los Angeles (UCLA) in 1970 [1], there have been a large number of research in this field, especially for the tasks of assisting and rehabilitation of human cognitive or sensory-motor functions.

The methods to collect cognitive information are mainly based on three major technologies, namely electroencephalography (EEG), magnetoencephalography (MEG), and functional magnetic resonance imaging (fMRI). EEG is a method of monitoring the electrical activity of the brain, and is collected typically by non-invasive electrodes placed along the scalp, although there are also a few invasive methods such as electrocorticography (EoCG). Informally, EEG can be regarded as the process of recording the multi-channel spontaneous

electrical activity of the brain over a period of time [2]. On the other hand, MEG uses extremely sensitive magnetometers to record the magnetic fields provoked by the brain's natural electrical activity. As compared to EEG, the complexity, financial and installation requirements of MEG are much higher, hence it is not as popular as EEG for research. There are also major technical differences between EEG and MEG [3, 4], since the former can detect signals from both the sulci and the top of the cortical gyri, while the latter only at the sulci. However MEG signal is more accurate in its visible area. The third major method, fMRI, works totally differently from EEG or MEG, as it detects changes in the brain's blood flow, assuming that cerebral blood flow and neuronal activation are mutually coupled. This is an indirect approach as compared to EEG, hence fMRI is usually used simultaneously with EEG to provide data with high quality in both temporal and spatial resolution.

Although there also exists a few other BCI techniques such as positron emission tomography (PET), or computed tomography (CT) which can offer high spatial resolution, EEG is still considered a valuable means for research, analysis and diagnosis thanks to its popularity, mobility, economical installment, availability of public data sets, and most importantly, high temporal resolution (millisecond), which cannot be achieved with fMRI, PET or CT. The detailed review of each brain signal acquisition technology will be presented in chapter 2.

This thesis focuses primarily on EEG BCI systems, especially ones based on event-related potentials (ERP), which is considered as one of the most dominant aspect of EEG. An ERP is a spontaneous responses produced by the brain after it receiving a specific outside sensory, cognitive, or motor event, which may be of various types, from visual, tactile, auditory, olfactory, to gustatory. The advent of the first ERP in the 1930s [5] had triggered a huge number of BCI applications. Currently, ERP is still one of the most widely used methods in various disciplines, ranging from neuroscience, cognitive psychology, cognitive science, to psycho-physiological.

1.2 Motivations

BCI, as its name suggests, is most, and to some extent, only, useful for people suffered from severe disabilities of communications. Some locked-in syndromes, such as amyotrophic lateral sclerosis (ALS), cervical spinal cord injury (SCI), or brain stem stroke cause the patients progressively becomes severely physically impaired, such as limited head and neck movement, or complete paralysis of arm, body and legs. More seriously, those impairments can massively affect swallowing, speech, and even breathing. These physical disabilities, surprisingly, do not affect their brain functionalities, since they can still think, make requests, and willingly make statements or response to outside questions normally. [6, 7].

Designing and developing an ERP-based BCI for such patients are practically much more challenging as compared to merely conducting experiments on healthy subjects in a research laboratory. The major difficulties, or also can be regarded as the major motivations for this thesis, are listed below chronologically, in terms of the order when they occurred during the course of this study:

- How can we overcome the issue of discrepancies of ERP patterns across different subjects?
- How can we estimate the confidence of the classification algorithm about its decision in real-time in order to stop the paradigm to boost up the system's speed?
- How can we adapt the existing model with the new subject-specific samples?

We will go through the related aspects of each question in the subsequent sections.

1.2.1 Subject-Independence

The first question, as well as the utmost motivation of our study, relates to the subject-independent requirement of the BCI. More specifically, we are

driven by the idea that new users of the pre-trained system are not required to perform the exhaustive training stage. This would save a huge amount of time and offer easy usage for new subjects, hence make ERP-based BCIs more practical in real-life everyday use, especially for disabled people. Indeed there had been a few notable approaches to fulfill this idea [8–11], which used no training samples to form the classifier. However, it is completely unnecessary to use no prior data to construct the classifier, since the availability of EEG data sets are abundantly provided by research institutes over the years [12–16].

For the new subjects to straightforwardly use the BCI in their first experiment, we must overcome the major issue of ERP discrepancies across subjects. More specifically, if we formulate the BCI task into a classification problem, then the input feature vectors will have significantly differing distributions across different subjects. Therefore if we train a model using an existing data from some subjects, the resultant classifier may not generalise well on new future subjects.

1.2.2 A Significantly Faster but Accuracy-Preserving BCI

The second question pays concern to the information transfer rate, i.e., the communication speed of the BCI. There must certainly be compromises between the system speed and accuracy. However, past research suggested that there were an abundant amount of unnecessary stimuli conducted for a fixed-length experimental session of ERP-based BCI [17–19]. In other words, any experiment can be terminated much sooner than expected while still preserves an expected level of accuracy. In chapter 2, we will deeply discover the past studies related to *dynamic stopping* (DS), which is the major approach to enhance the communication speed. Regrettably, to our knowledge there had only been a humble number of past works which employed DS in a proper way. Specifically, although various method had been proposed with practical effectiveness, the theoretical foundation was still not sound, or was ignored by some reasons.

In this thesis, we attempt to attain the objective of boosting the communication speed by integrating a DS algorithm into the P300-Speller (P3S)

paradigm based on two statistical coefficients. The theoretical proofs that back our method will also be proposed. More specifically, we analytically search for those coefficients of an algorithm that decides when the P3S should be stopped when it is sufficiently confident about its decision.

1.2.3 An Adaptive Learning Algorithm in Real-time

The third question imposes another tough challenge on the subject independence of BCI. The question can be restated as how to integrate newly-classified patterns of the ongoing experimental subject into the pre-trained classifier? This integration, or adaptation, must be carried out with careful considerations. First, the adaptive strategy must process the new patterns in a real-time fashion, hence computational complexity is one major concern. Second, selecting the appropriate new patterns to learn is fundamental, since incorrect-labeled samples will detrimentally affect the generalisation performance of the subsequent experiments. Finally, the proposed algorithm needs to be developed to be applicable to a general binary classification problem. Disregarding the feature selection step and some refinements specifically only to the BCI context, we are also interesting in applying the proposed algorithm in other classification contexts, such as arrhythmia detection [20]. The desirable results are those competitive to state-of-the-art methods [10, 21–24], in terms of classification accuracy, learning speed, and computational complexity.

1.3 Contribution of the Thesis

For each question described above, we propose a method to resolve the mentioned issues. All of these methods are independent and can be combined into a unified BCI framework, from the first step of data acquisition to the last step of postprocessing feedback reinforcement learning. The objectives of this thesis are therefore to tackle the challenges as mentioned in section 1.2.1, 1.2.2, and 1.2.3. The contributions of this thesis are presented as follows.

1.3.1 Subject-Adversarial Neural Network as a Feature Encoder

We propose a state-of-the-art deep learning technique applied on ERP feature vectors, namely the subject-adversarial neural network (SANN). SANN possesses a complete architecture of a typical neural network, but has three parties stacked in a proper order. The goal of SANN is to find a better feature space as compared to the original feature space induced by the preprocessed ERP trials. SANN can be viewed as an extra feature encoder that maps each input vector into another vector space that provides better generalisation capability before the classification task. In SANN, we take into account these information: the features (i.e., data points) of each ERP trial, the class label, and the subject identity. Three parties in SANN (feature encoder, class predictor, and subject separator) are trained simultaneously and adversarially to attain the final objective. The optimal feature encoder is desirable for the class predictor, but detrimental for the subject separator. Therefore, the subject-independence property of our BCI can be reached. Our developed SANN architecture is heavily based on the domain adversarial neural network (DANN) [25], which is a modified version of the original generative adversarial neural network (GANN) [26]. These adversarial architecture was found to practically perform well on various computer vision applications, i.e., gray-scale image colorisation [27], low-resolution image upsampling [28], image synthesis from textual descriptions [29, 30], or image editing tools [31–33].

1.3.2 An Ensemble Learning Strategy of 3 Base Learners

For the classification task, we propose a unified ensemble learning strategy which is the combination of the most two widely-used techniques in practice: bagging [34] and stacking [35]. Bagging and stacking were proved to practically perform well, and won various machine learning competitions [36]. Although various machine learning algorithms had been proposed in the context of ERP-based BCIs (i.e., convolutional neural network [37, 38], SVM [39, 40], LDA [21, 23]), there is still, to our knowledge, no complete study which simultaneously employ multiple classifiers of different approaches into a unique ensemble strategy. We attempt to do so by applying three base learners, namely support vector machine (SVM), fully-connected feedforward neural

network (NN), and Fisher’s discriminant analysis (FDA) on the ERP classification task. The soft scores of these learners are fed as inputs for the ensemble strategy.

1.3.3 Dynamic Stopping and Adaptive Learning

For the final major contribution, we propose an efficient adaptive (or incremental) learning algorithm, named incremental support vector machine (ISVM), which can integrate newly-classified samples into the existing solution without re-training (i.e., re-solving the quadratic program of SVM) from scratch. Specifically, we need to develop a new classification algorithm, based on SVM, that can analytically tune the existing classifier using a few new feature samples of the current experimental subject. An analytical research on how to integrate new labeled samples into the existing model needs to be exhaustively conducted. Firm theoretical proof and convergence analysis are of utmost necessity.

Practical experiments showed that the classification accuracy can be gradually improved with respect to experimental time. In a more general context, our proposed adaptive incremental technique can be applied to other classification problems. We have reported our results in ERP-based BCI [41–44], as well as in arrhythmia detection [20].

1.4 Organisation of the thesis

The layout and relations of the chapters in this thesis are depicted in Figure 1.1. Specifically, the organisation of this thesis is presented as follows.

Chapter 1 is Introduction. Section 1.1 briefly reviews the most general background of this thesis. Section 1.2 states three major challenges we aim to tackle, which are the main motivations of this research. Section 1.3 then sequentially describes the contributions in this thesis, each of them corresponds to one challenge mentioned previously. Section 1.4 depicts and describes the organisation of all chapters of this thesis.

Chapter 2 is Literature Review. Section 2.1 presents the neurophysiology aspects of the human brain. Section 2.2 describes the three major components of a typical BCI. Section 2.3 reviews state-of-the-art brain signal acquisition technologies. Section 2.4 then focuses on various types of EEG BCIs. Section 2.5 presents all aspects and issues related to ERP-based BCIs, followed by the chapter summary in section 2.6.

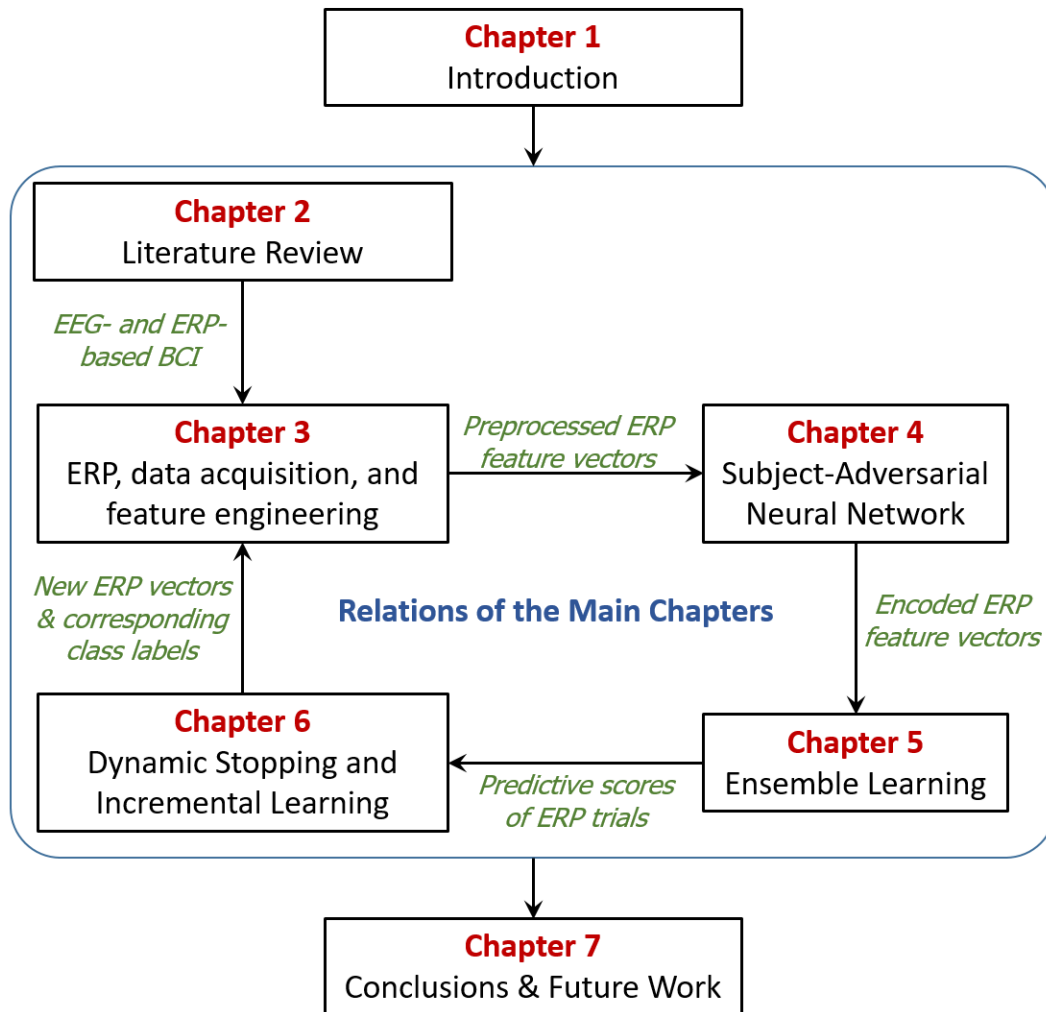


Figure 1.1: Relations of the main chapters in this thesis.

Chapter 3 is Event-Related Potentials, Data Acquisition, and Feature Engineering. Section 3.1 reviews the history of ERPs and explains their relevance to this thesis's context. Section 3.2 elaborates on each flashing paradigm iteration of the P3S, which is the interface to collect EEG signal of our framework.

Section 3.3 describes the evaluation metrics that will be used thoroughly in our research. Section 3.4 presents the preprocessing techniques on raw EEG signal, in order to suppress noise and detect artifacts, followed by the chapter summary in section 3.6. The output deliverables from this chapter are the preprocessed (filtered, artifact-removed) feature vectors which can directly be fed as inputs to the adversarial encoding method presented in Chapter 4.

Chapter 4 is Subject-Adversarial Neural Network. Section 4.1 presents the motivations of our BCI framework induced by the ERP discrepancies across different subjects. Section 4.2 reviews the most widely-used adversarial training algorithm: GANN and DANN. Section 4.3 describes our proposed subject-adversarial neural network (SANN), including the theoretical foundation and related training issues. Section 4.4 provides a simple case study of SANN for better understanding. Section 4.5 presents the experiments of SANN on BCI data, followed by the chapter summary in section 4.6. The output deliverables of this chapter are the optimally encoded feature vectors, which can be directly be fed as inputs to the ensemble classification strategy presented in Chapter 5.

Chapter 5 is Ensemble Learning. Section 5.1 describes and explains why bagging ensemble can reduce generalisation errors. Section 5.2 describes and explains why stacking ensemble can boost up the classification or regression accuracy if the base learners are comparable. Section 5.3 discuss the benefits of combining bagging and stacking into a unified ensemble strategy. Section 5.4 briefly reviews three different base learners that we employ in this thesis: SVM, NN, and FDA. Section 5.5 describes the simple linear regression method as the meta combiner. Section 5.6 presents the experimental results of our proposed ensemble strategy on BCI data, followed by the chapter summary in section 5.7. The output deliverables from this chapter are the ensemble scores of each ERP feature vector, which serves as inputs for the dynamic stopping and adaptive incremental learning presented in Chapter 6.

Chapter 6 is Dynamic Stopping and Incremental Learning. Section 6.1 presents the dynamic stopping (DS) method applied on the P300-Speller (P3S)

with two stopping criteria backed by theoretical proofs. Section 6.2 describes the incremental support vector machine (ISVM), which is an adaptive learning method to analytically integrate newly-classified samples into the existing SVM hyperplane solution. Section 6.3 presents the experimental results of DS and ISVM in the context of BCI, followed by the chapter summary in section 6.4. The output deliverables from this chapter serve as final decision of the BCI system of the current experimental session. They can be used as reinforcement samples for future learning processes, beginning by re-collecting and preprocessing EEG signal as presented in Chapter 3.

Finally, Chapter 7 is Conclusions and Future Work, which summarises the results of this thesis and recommends related future work.

Chapter 2

LITERATURE REVIEW

In this chapter, we will present the literature review on typical brain-computer interface systems, from the broadest aspects to the most specific area that this thesis is focused on. First, section 2.1 describes the neurophysiology properties of the human brain, including the major brain areas and the functionalities of each area. In section 2.2, a brief introduction on BCI is presented with three main steps: data acquisition, processing algorithms, and output/feedback. To narrow down the specific BCI area that this thesis is focused on, we begin exploring brain data acquisition technologies in section 2.3, including electroencephalography (EEG), which is the selected technology in this thesis, as well as other technologies (fMRI, MEG, etc). Next, section 2.4 points out all typical EEG-based BCIs systems, including the event-related potential (ERP), which is the selected EEG-based BCI system in this thesis, as well as other state-of-the-art EEG-based BCIs (SMR, SCP, etc). Subsequently, all aspects of EEG-ERP-based BCIs will be discussed in section 2.5. Finally, the chapter is summarized and concluded in section 2.6.

2.1 Brain Neurophysiology

The human brain, which is the essential part of the central nervous system (CNS), possesses a complex structure including all functions of physiology and cognition, i.e., decision, emotions, senses, muscle movements, memory, and the most general, thoughts. It is also responsible for unconscious functions such as breathing, digestion, pulse, or conscious functions such as planning or reasoning.

The brain structure can be divided into three major parts: the large convoluted cerebrum, the rippled cerebellum, and the brain stem. The large convoluted cerebrum consists of two hemispheres. The left hemisphere controls the right side, and the right hemisphere controls the left side of the body. Above all, the cerebral cortex is the layer that connects all components of the brain together. This cerebral cortex is comprised of four major lobe areas: frontal, occipital, parietal, and temporal [45, 46], as depicted in Figure 2.1. It is responsible for most high-order functions of the brain, including vision, auditory, movement, sense, and planning.

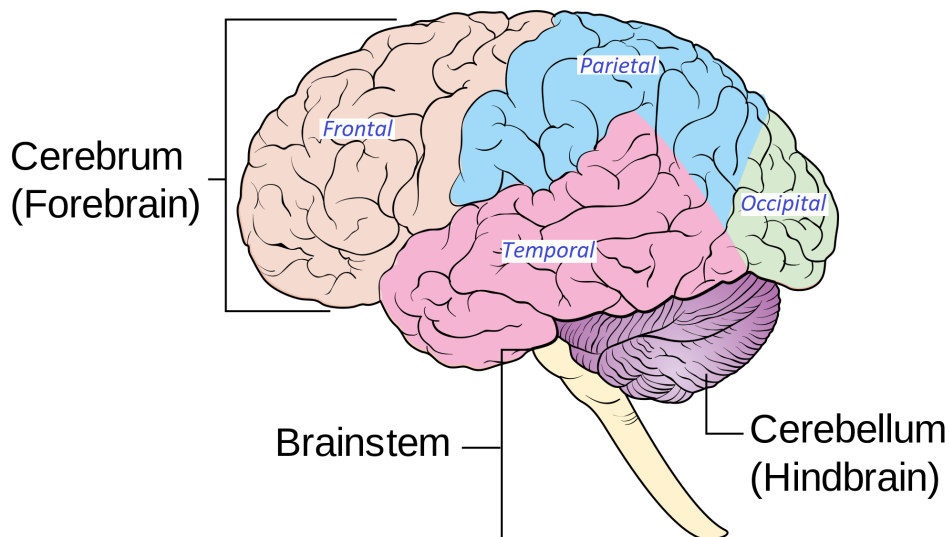


Figure 2.1: The human brain architecture.

The frontal area controls high-order execution functions such as complex cognitive behaviours, personality, and decision making. This area includes the prefrontal area. The frontal lobe also contains the primary motor cortex which is related to movements of particular body areas, i.e., legs, feet, trunk, arm, hand, and face. The occipital lobe's major function is vision, and is located at the posterior pole of the brain, as shown in Figure 2.1. The parietal lobe is in charge of tactile discrimination, motivation, reading, writing, and arithmetic calculation. It is a region of multimodal association cortex which receives input from the surrounding somatosensory, visual, and auditory sensory areas. The temporal lobes controls higher-level visual and auditory signal processing, as

well as responsible for memorizing [47]. In brain-computer interface research, the most widely used area of the brain is the cerebral cortex thanks to its high involvement in cognitive functions and its accessibility for both invasive and non-invasive electrodes. The cerebral cortex is the outer layer of neural tissue of the cerebrum of the brain. The electrical activities retrieved from this area can be used directly for mental-task classification in BCI applications.

2.2 Brief Introduction on Brain-Computer Interface (BCI)

Brain-computer interface (BCI) is a means to help people send commands to the computer by their brain's electrical activities. There are three principle steps in any BCI design: the signal (or data) acquisition, the processing algorithms, and the output/feedback to the subject [48–50], as depicted in Figure 2.2.

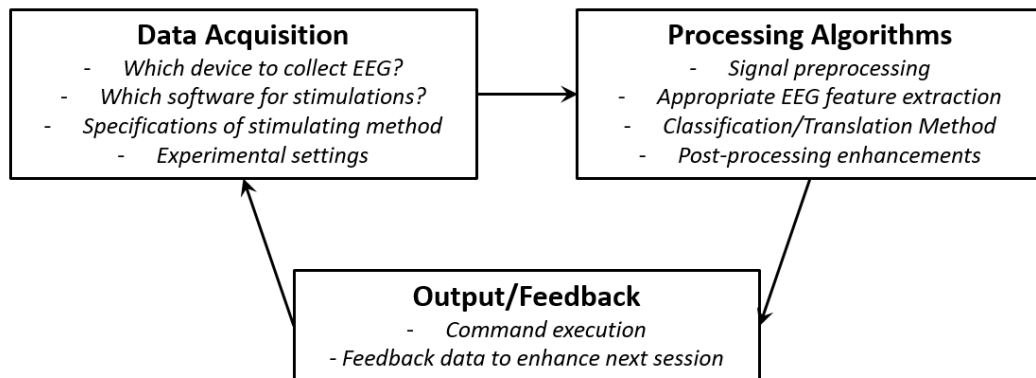


Figure 2.2: The diagram of a typical BCI.

The main processes of a typical BCI are described as follows. First, the brain signal is collected by the BCI instrumentation, i.e., an EEG headset, or an fMRI device. The accompanying aspects such as acquisition length, software/method to collect signal and its detailed specifications also belong to the first major step of a BCI: the data acquisition step. This is followed by the second major step of processing algorithms. This step is considered the most important and is comprised of several essential sub-steps. Initially, the acquired signal after the data acquisition step is filtered and digitized to

decrease storage capacity and remove unnecessary noise. Then, the digitized and filtered signal is passed through a feature extraction method to keep only appropriate brain patterns that are specifically useful to a particular application. Then, the extracted features are further processed/constructed to become the input of a classification/translation algorithm. The output of the classification/translation algorithm is a logical command, which is passed through the final major step: the output/feedback. This step is used to execute the command, i.e., to veer the wheelchair, or to pop up a target word letter on the screen. The output can also be fed back to the system to enhance the subsequent experimental sessions.

2.3 BCI Data Acquisition Technologies

From the perspective of brain data acquisition method, there are two major types: the invasive method and non-invasive method. The invasive method requires a surgery to implant the electrodes/sensors into the subject's inner brain. This method hence therefore, extremely expensive in terms of data acquisition instruments, people involvement, and preparation time. Two typical examples for invasive BCI are intra-cortical recording, and electrocorticography (ECoG). On the other hand, non-invasive method consists of more ways to collect brain signal without surgery, such as electroencephalography (EEG), magnetoencephalography (MEG), functional near infrared spectroscopy (fNIRS), functional magnetic resonance imaging (fMRI), positron emission tomography (PET), or single photon emission computed tomography (SPECT) [48, 51–53]. The diagram presenting all of the most widely used data acquisition methods is shown in Figure 2.3. In this section, we will briefly review and discuss the advantages and drawbacks of each BCI data acquisition method, as well as explaining the motivation on our employment of EEG as the main means of data acquisition in this thesis.

2.3.1 Intra-cortical Recording

In intra-cortical recording, sensors are implanted into the human brain via surgery to directly collect electrical activities from single neurons, or from the areas that produce combined activities from surrounding neurons and synapses

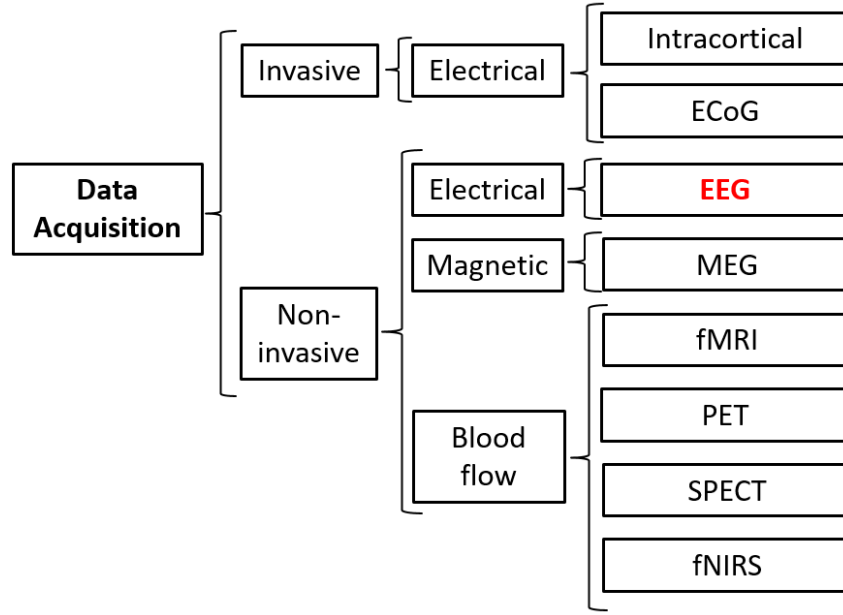


Figure 2.3: The most widely used methods for data acquisition in BCI. Electroencephalography (EEG) is the data acquisition method used in this thesis.

called the local field potentials (LFP). One clear advantage of intra-cortical recording is that it offers the highest signal resolution in both temporal and spatial domains [54]. However, a huge drawback of intra-cortical recording cancels out its high resolution, as there exists great risks and expensive costs to implant sensors into the inner brain tissues. The risks are related to problems of chronic tissue damages and long-term sensor stability [55]. As a result, the major part of BCI research using intra-cortical recording only involves experiments in animals, such as monkeys [56]. On human, there is great potential to use intra-cortical recording implanted in the motor cortex areas to rehabilitate the lost motor functions of the subjects with tetraplegia [57].

2.3.2 Electrocorticography (ECoG)

Another invasive BCI method is electrocorticography (ECoG), which also captures electrical activities of the brain. The difference between ECoG and intra-cortical recording is that, ECoG locations is underneath the skull but not within the brain itself. ECoG also requires surgical operations, but sensors are placed by screws on the epidural, i.e., the surface of the dura matter. ECoG therefore, is less invasive than intra-cortical recording and hence less signal

quality, but it offers a much easier approach, and is still high in signal resolution and frequency, as well as invulnerable to noisy artifacts [58]. As compared to electroencephalography (EEG) which will be described next in this section, ECoG provides better spatial resolution (1.25mm for subdural [59], or 1.4mm for epidural [60], as compared to over 10mm of EEG), higher amplitude range that is invariant to muscular artifacts [61], or wider frequency bandwidth (up to 500Hz as compared to 40Hz of EEG) [62]. ECoG was used in various BCI applications, including auditory imagery [63], speech imagery [64], motor imagery [65], or mental calculation [66].

2.3.3 Magneto-encephalography (MEG)

Magnetic field can also be generated by electrical currents from the human brain, hence it can be collected as another means of BCI. Magneto-encephalography (MEG) is a non-invasive method which exploits the magnetic field of the human brain. MEG works by placing sensors, which are the induction coils lying stationary around the helmet, on the human scalp. A superconducting quantum interference device (SQUID) is used to collect the magnetic field produced by the brain [3]. As compared to EEG, MEG has roughly the same temporal resolution, but lower spatial resolution. However, MEG has a huge limitation that it is affected by external geomagnetic field that is 8 to 9 orders higher in magnitude than the magnitude field generated by the brain. As a result, it requires a room with a magnetic shield as a tool to prevent artifacts from electrical devices or prevent geomagnetic field, to achieve high-quality signal [4]. Also, MEG experimental subjects have to take off their personal magnetic items. MEG is also affected by subjective artifacts such as eye blinks or muscle movements, and it has huge installation cost, as well as complex instruments.

2.3.4 Functional Near Infrared Spectroscopy (fNIRS)

Besides electrical and magnetic, blood flow is another means to measure the activities of the brain, Functional near infrared spectroscopy (fNIRS) is a

non-invasive method that can track the changes in hemoglobin, which reflect the hemodynamic activities of the brain. fNIRS works by measuring the level of oxy-hemoglobin (oxy-Hb) and deoxy-hemoglobin (deoxy-Hb) of the brain. The amount of oxy-Hb and deoxy-Hb changes at some particular areas of the brain when the subject performs a particular experiment. Those changes can be measured by the absorbance spectra in the near infrared wavelength zone, which differs with respect to different levels of oxy-Hb or deoxy-Hb [67, 68]. fNIRS is comprised of pairs of light source detectors placed on the forehead of the scalp. Although being a new technology in BCI measurements, fNIRS has some advantages such as non-invasive, easy to use, and inexpensive set-up cost. However, it has significantly lower temporal and spatial resolution as compared to EEG or MEG. Applications of BCI using fNIRS had been proposed, including motor imagery on healthy subjects [69, 70], or disabled subjects [71].

2.3.5 Functional Magnetic Response Imaging (fMRI)

Functional magnetic response imaging (fMRI) is similar to fNIRS since it is also a non-invasive method working by measuring the level of oxy-Hb and deoxy-Hb in the brain tissue. fMRI is the method that offers the best spatial resolution [72]. The main instrument is the magnetic resonance imaging (MRI) scanner, which is comprised of six main components: the large magnet, the radio frequency transmitter, the receiver antenna, the gradient coils, the front-end computer, and the reconstruction computer [73]. fMRI has a clear advantage that it can measure almost all areas of the brain, and been used in BCI research [66, 74]. However, fMRI is still unpopular due to its bulky instruments, high installation cost, inconvenience use of helium liquid for superconducting coil cooling, and the requirement of radio frequency (RF) shielded room.

2.3.6 Positron Emission Tomography (PET)

Another BCI measuring method that exploits the blood flow is the positron emission tomography (PET), which is a nuclear medical imaging technique which generates three-dimensional maps of the brain and metabolic processes. Prior to the experiments, subjects must be injected into their brain the radioactive materials in a cyclotron with small doses to avoid harmful biological effects. Then, the PET scanner begins acquiring the images by measuring the level of radiation using its ring detectors. A positron is emitted by a radioactive oxygen atom, then collides with an electron in the surrounding areas and generate two gamma photons. These photons travel in two opposite directions and hit a scintillator, which is a crystal block on the subject's scalp, and produce flashes of light which can be detected by an optical camera arranged outside the crystal block. These images provides insights of the density distributions of the radioactive tracers in the brain. The blood flow increases in a particular brain areas where the subject's cognition is active, leading to increased radioactive liquid flowing into the brain tissue. The PET instrument then detects these changes in radioactive levels and create the corresponding images [75]. PET has some major disadvantages, including the low resolution in both temporal and spatial domains [75]. The expensiveness also hampers PET from practical BCI applications and research. The majority of systems using PET is to diagnosing serious brain disease such as dementia, strokes, or tumours.

2.3.7 Single Photon Emission Computed Tomography (SPECT)

A photomultiplier tube is used to measure the photon generated by the gamma rays in the method of single photo emission computed tomography (SPECT). It is a nuclear imaging technique and is similar to the PET scanner that measures blood flow. The main difference of SPECT is that a radiopharmaceutical emitting gamma rays is injected into the subject's blood vessel before the experimental session. These rays serve as radioactive tracers that are circulated through the cardiovascular system. Their concentration will be high in the areas with high blood flow. As compared to PET, SPECT is less

expensive and has higher resolution, but both are still extremely bulky and inconvenient due to the injection requirements. Similar to PET, SPECT is mainly used to diagnose strokes and brain tumour [76, 77]

TABLE 2.1: COMPARISON OF BRAIN DATA ACQUISITION TECHNOLOGIES.

Data Acquisition Method	Signal Type	Advantages	Disadvantages
Intra-cortical	Electrical	Highest temporal Highest spatial	Most invasive Surgery
ECoG	Electrical	High temporal High spatial	Invasive Surgery
MEG	Magnetic	Moderate temporal High spatial Non-invasive	Expensive Bulky
fNIRS	Blood flow	Moderate spatial Non-invasive Portable Inexpensive	Low temporal Not well-researched
fMRI	Blood flow	High spatial Non-invasive	Low temporal Expensive Bulky
PET	Blood flow	Moderate spatial Non-invasive	Low temporal Expensive Bulky Injection
SPECT	Blood flow	Non-invasive	Low temporal Low spatial Injection Expensive Bulky
EEG	Electrical	Moderate temporal Non-invasive Portable Inexpensive Well-researched	Low spatial

2.3.8 Electroencephalography (EEG)

Electroencephalography (EEG) is the selected brain data acquisition method in this thesis. As its name suggested, EEG measures electrical activities of the brain non-invasively, and it does so by simply placing electrodes directly on the scalp. EEG is the most widely used in BCI research thanks to its many great advantages. First, it does not require surgery or liquid injection, hence alleviates the burden of experimental preparation as well as is applicable to

almost any subject. Second, EEG does not go with bulky and expensive devices. Most EEG devices are manufactured as portable headsets. The electrodes's positions are fixed on the headset, corresponding to provided brain sites. When wearing the EEG headset into their head, the subject is required to slightly adjust the headset position on their head until the referenced electrodes are correctly placed. This can be done solely by the subject by an integrated software that comes with the device. The electrodes are mostly made from platinum (Pt), gold (Au), silver/silver-chloride (Ag/AgCl), or tin (Sn). The gold electrodes provide the best quality of electrical brain signal as it can minimize drifts and less noisy in high frequency as compared to silver or silver/chloride [2], while tin electrodes are inexpensive but extremely noisy in low frequency (below 1Hz). It should be note that the signal quality can be massively enhanced by using a conductive gel to connect the surface of the scalp to the electrodes. This is done by wetting the electrodes by the gel first before placing the headset onto the scalp. Although several commercial EEG devices can offer dry electrodes, it was soundly shown that dry electrodes are inferior to wet electrodes, mostly due to their low temporal resolution [23].

A significant drawback of EEG is its noisy nature. It is highly sensitive to ocular, muscular, and cardiac noises, i.e., muscle movements, eye blinks, heart beats, breathing rhythms. While eye artifacts have a stereotypical waveform, muscular artifacts are broadband and found above 10Hz. By employing some preprocessing and averaging techniques, EEG artifacts can be effectively detected and removed. These techniques will be discussed in Chapter 3.

2.4 EEG-based BCI

With regards to BCI systems that employ EEG as the data acquisition method, they can be divided into two major types based on the nature of neural responses produced by the subjects. The first type is stimulated in which external stimulations are present to provoke brain responses. These stimulations can be of different kinds, such as event-related potential (ERP) or steady-state evoked potential (SSVEP). In stimulated EEG-based BCI, the brain patterns are triggered by external stimulations, i.e., visual flash or au-

ditary click. The second type of EEG-based BCI is not based on any stimulations, and the brain patterns of the subjects are completely spontaneous by the internal self-regulation. Typical systems of this type include slow cortical potential (SCP), sensorimotor rhythm (SMR), motor imagery tasks, or mental tasks (non-motor imagery mental tasks). In this section, brief review including the advantages, disadvantages, and applications of both stimulated and spontaneous EEG-based BCI type will be presented.

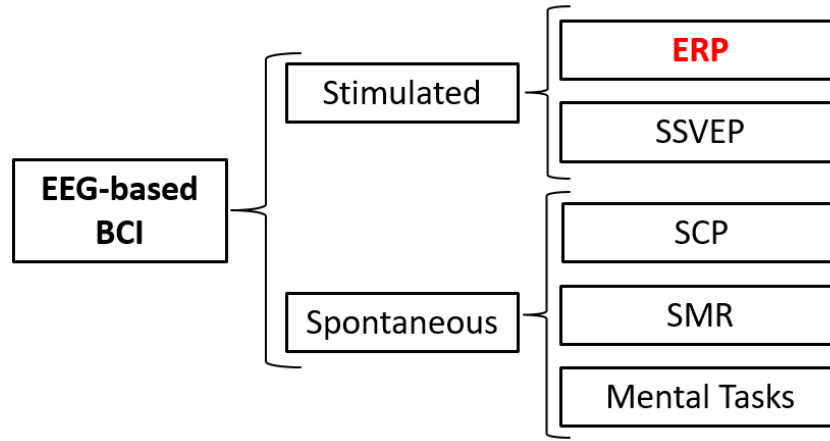


Figure 2.4: Typical types of EEG-based BCI, including the event-related potential (ERP) that is the main theme of this thesis.

2.4.1 Event-Related Potential (ERP)

Event-related potential (ERP) is one of the most widely-used method to provoke brain patterns in various BCI applications. The subjects are present with series of stimulations, mainly of two major types: the target and non-target response. The probability of target stimulations is much less than the probability of the non-target stimulations. For instance, the subjects are instructed to press a button when hearing low-tone clicks, which are rare and mixed with high-tone clicks. Prior to the physical response, the subject will generate a specific cognitive response, called the P300 (or P3) response. P3 is a component of ERP, and is exploited to design a wide range of BCI applications [78–80]. Typical examples of ERP responses can be listed, such as the visual ERPs (C1, P1, N1, P2), or the cognitive responses (N2, P3). As ERP-based BCI is the main theme of this thesis, we will present all aspects of ERP, including the thorough literature review and pilot experiments in Chapter 3.

2.4.2 Steady-State Evoked Potential (SSVEP)

In steady-state evoked potential (SSVEP) BCIs, as its name suggests, uses steady visual stimulations, i.e., flashes or changes in color, to evoke one's brain pattern. For instance, each letter on the computer screen is flashed at a fixed frequency, so the brain pattern will be specific and corresponds to a unique letter. SSVEP responses are mostly collected near the primary visual cortex of the occipital lobe [81]. The brain patterns produced by SSVEP-based BCIs are achieved by conducting signal analysis in the frequency domain. The peak at the frequency of visual stimulation is found in one's EEG signal [82]. Hence in SSVEP-based BCI, the subjects pay attention to one desired stimulation image with a specific flashing frequency, which also corresponds to a particular BCI output command, i.e., turn the wheelchair left.

One clear advantage of SSVEP as compared to ERP is that it can be straightforwardly used by the subjects. The level of usage is easy as the subjects only need to gaze at the target stimulation that they want to command. Furthermore, as the frequency peak of the subject's EEG is identical to the frequency of stimulation, no machine learning technique is required to perform classification. However, SSVEP also has its own advantages as compared to ERP. First, the discriminative capability of the frequency analysis algorithm is heavily governed by the number of selections present to the subject. Since the EEG bandwidth is narrow, i.e., 0-60Hz, too many selections, which are highly desirable for spelling applications, detrimentally affect the detection of peak frequency. As a result there were scarcely SSVEP applications which require a large number of selection outputs. However, with regards to the stable output and easy-to-use nature, SSVEP were used in various controlling BCI systems, such as the functional electrical stimulator (FES) for knee flexion initialisation [83], the aeronautical navigation system [84], artificial legs [85], or car driving [86–88].

2.4.3 Slow Cortical Potential (SCP)

Slow cortical potentials (SCP) are slow event-related and direct-current shifts in the EEG. SCP relates to the specific sensorimotor events. The shifts in

SCP are mostly negative, and can be found to occur before the imagined/actual movement, or cognitive tasks, i.e., preparation in imagined activations [89]. One major component of SCP is the Bereitschaft's potential, or also known as the readiness potential, which is a negative shift beginning at approximately 100-500ms before a spontaneous movement [90]. The onset time, amplitude, and pattern of this shift are governed by the subjective factors such as muscles involved, psychological state, or movement type [91]. Although negative shifts dominate SCP which reflect increased excitability of neurons, there also exist positive shifts which reflect the consumption of cognitive resources during testing. Both negative and positive shifts are found in healthy and disabled subjects [90].

SCP-based BCI were used in thought translation device (TTD), where training is required by conducting visual feedback and positive reinforcement [92]. More specifically, the subjects inspect their experimental session as the vertical movement of a cursor on the computer screen. Their goal is to move this cursor towards to polarity at the bottom half of the screen, with the SCP amplitude reflecting the movement. On the other side, auditory feedback can also be used for subjects with visual difficulties, i.e., subjects in their late ALS state [93,94].

SCP has an extremely huge drawback that it requires massive training data, i.e., weeks or months of training sessions, for each subject to regulate the SCP signal. As a result, SCP systems are slow and cannot be used for applications with too many output selections [95]. However, SCP was employed successfully in some practical applications, i.e., wheelchair control [96,97].

2.4.4 Sensorimotor Rhythms (SMR)

Beside ERP-based BCI, sensorimotor rhythm (SMR) based BCI is another well-explored and popular research in recent years. SMR consists of three main frequency bands: μ (8-13Hz), β (15-30Hz), and γ (30-200Hz). SMR-based BCIs are used mostly in motor imagery tasks using the oscillatory EEG on the sensorimotor cortex. There are two major components in SMR, namely the event-related desynchronization (ERD) and the event-related synchronization

(ERS), which both relate to actual or imagined movement of foot, leg, hand, or tongue [98,99]. The ERD is a decreased SMR that occurs at the movement onset, and the ERS is an increased SMR that occurs when the SMR completes.

There are two distinct ERD patterns in the μ rhythm band. The lower frequency at 8-10Hz occurs at entirely sensorimotor cortex in almost all of the motor behaviours, hence its ERD pattern is non-specific. On the other hand, the higher frequency at 10-13Hz reflects the task-specific ERD patterns of the experimental session. Similarly, the β rhythm can provide task-specific aspects of experiments. For instance, it produces ERD patterns associating somatosensory stimulation and motor behaviours, or ERS patterns associating with movements [100,101].

To detect ERD and ERS patterns in SMR, frequency analysis is fundamental. The basic signal processing techniques include band pass filtering and multi-trial averaging (which are described in details in Chapter 3). The filtered and averaged trial then is analyzed to detect the proportional power decrease in ERD, or proportional power increase in ERS, relative to a specific reference interval of several seconds before the event onset [102]. Then, a time-frequency map is generated. Spatial features can be extracted by doing some spatial filtering techniques in some particular cortical areas [103]. Since SMR-based BCIs use EEG as the data acquisition method, they are heavily affected by artifacts such as eye blinks, facial muscles, or external electromagnetic signal. The techniques to remove these artifacts are discussed in Chapter 3.

SMR-based BCIs were used in various applications, such as cursor movement control using μ rhythms in one dimension [104,105], in two dimensions [105,106], or three dimensions [107]; spelling (words or sentences) [97,108], neuroprosthesis hand grasp rehabilitation [109,110], or wheelchair control [111].

2.4.5 Mental Tasks

Mental tasks, or also known as non-motor imagery mental tasks, are desirable for subjects with severe disabilities. The BCI illiteracy phenomenon [112] suggests that selective motor imagery tasks using SMR-based BCIs malfunction.

tion in disabled patients [113]. Mental tasks, which is different from motor-imagery, is preferable in patients who have motor cortex impairments, amputations, or paralysis. [114,115].

Some typical and popular mental tasks include relaxation (relax and think nothing), multiplication (do a simply math multiplication), geometric figure rotation (imagine a rotating object around an axis), letter composing (compose letters), visual counting (count the number on the imagined blackboard), auditory imagery (imagine an audio tune), spatial navigation (imagine moving around a familiar place), face imagining (imagine a familiar face) [116–119].

2.5 ERP-Based BCI

ERP-based BCIs are well researched throughout BCI history, especially on patients with amyotrophic lateral sclerosis (ALS) [21,120]. Generally, an ERP is an electrical response produced by the human brain as a reaction to a specific evoke or stimulus. On top of that, P3, or also known as P300 [121] is one of the most robust components of ERP which had been used in various BCI studies [10,22,38,122–130]. P3 responses manifest as a positive voltage peak in EEG and presents approximately 300 ms after the visual stimulus onset. The details of P3 component as well as other dominant ERP components will be discussed more in Chapter 3.

To better exploit the predominant aspect of P3, Farwell first introduced the so-call P300-Speller (P3S) [131]. The ultimate goal of P3S BCIs is to help people indicate their choosing among the preset options on the computer screen. The set of available images on the screen can be a matrix of alphabetical letters or symbols from any language, or it can even be a group of graphical images indicating people’s every basic needs (food, light, hygiene, etc.,). We only care about the stimulus timing rather than the meaning of the images or the symbols on the screen. Therefore the set of user options can be almost anything. Each symbol in that set is flashed sequentially according to a specific flashing paradigm. In most 36-letter P3S, each one-letter spelling session is usually conducted by a fixed number of iterations. Each iteration

consists of 12 letter intensifications (or flashes), corresponding to 6 rows and 6 columns of the P3S. A signal timeline for a P3S is illustrated in Fig. 3.3.

Since then, various developments and research were carried out to boost up its performance, i.e, accuracy, speed, or bit rate. These developments can be the modifications on stimulus flashing paradigms [122–124], language model integration [10, 22, 125, 126], feature processing and frameworks [127–129], or classification algorithms [38, 130]. However, the original method remains the same that the subjects focus their eyesight at one desired option out of the offered set of letters appearing on the screen, thereby to convey their spelling intentions.

In this section, a thorough literature review is presented on most of the state-of-the-art methods on BCI research.

- Sutton [132] (1965) first discovered the Evoked-Potential correlates of stimulus uncertainty. This discovery was a major breakthrough and had laid a strong foundation for P300 research, or in a more general sense, for evoked-related-potentials (ERP).
- Farwell [131] (1988) for the first time developed a flashing paradigm to exploit the P300. This work was of great importance and currently is the must-be-cited article for any new P300 research. Although this flashing paradigm is not optimally specific to P300, it opened a vast and undiscovered field in ERP research.
- Picton [121] (1992) was the first one to conduct extensive analysis on P300 responses of human EEG in the context of human disorders: schizophrenia, autism, psychiatric disorders, dementia, children’s cognitive disorders, alcoholism, and Korsakoff’s syndrome.
- Wolpaw [133] (2002), who is widely considered as one of the most influential researcher in BCI, wrote a comprehensive review in EEG and BCI developments in communication and control.

- Benjamin and his research group [13] (2003) organized a BCI Competition which attracted the BCI community. This competition introduced different fields of BCI research as well as motivated BCI experiments all over the world. Up until this time there were scarcely projects on BCI due to the expensiveness of brainwave equipments.
- Dornhege [130] (2004) proposed a scheme for boosting bit rates in noninvasive single-trial classifications by feature combination and multiclass paradigms. They used Bayesian Linear Discriminant Analysis (BLDA) as their main classification method. The limitation of this method is the low-complexity and probabilistic nature of BLDA, as compared to several other later discovered deterministic algorithms such as support vector machine, or especially neural network.
- Hoffmann [134] (2005) proposed a gradient boosting method for P300-based BCI. Their approach is to build one strong classifier from many weak classifiers, which is also one of multiple approaches used in this report (ensemble classifier).
- Sellers [120] (2006) for the first time tested the P300 performance on amyotrophic lateral sclerosis (ALS) patients (although their P300 interface had only 4 choices, as compared to the conventional of 36 letters). The results were promising that two of the 3 ALS patients' classification rates were equal to those achieved by the non-ALS participants. This work opened the possibility for BCI applications on ALS subjects, as well as other severely damaged patients (locked-in, paralyzed).
- Birbaumer [135] made a review on non-invasive BCI and their clinical utility for direct brain communication in paralysis and motor restoration in stroke. The review was conducted on neuroelectric and neuromagnetic BCI, slow cortical potential BCI, sensorimotor rhythm BCI, P300 BCI, and magnetoencephalography BCI.

- Up to this point there were barely any ERP-based BCI which paid attention on dynamic stopping mechanism (that is, we can stop any particular P300 flashing session once a certain level of confidence is reached). Lenhardt [18] (2008) initially noticed this huge gap and proposed an adaptive on-line P300 BCI. He proposed a relatively simple dynamic stopping criteria based on accumulative scores of rows/columns in P300. As a result, he achieved the accuracies of around 80% while mean bit rate were boosted from 32 bits/min to 45 bits/min with a slight accuracy loss of 5%.

- The BCI Competition III were organized [40] (2008) with the winner of the P300-based BCI awarded to a team who implemented SVM as their main classification method. Moreover, they proposed a bagging approach (or, ensemble) for individual SVM classifiers. Instead of training the whole bunch of data into one SVM classifier, they divided the data (in chronological order) into separate partitions and combined them with a specific strategy. Their result was striking: 96.5% (with 15 flashing iterations) and 73.5% (with 5 iterations). This work interestingly suggested that in fact, discriminative classifier (i.e., SVM) greatly outperformed probabilistic classifier (i.e., LDA). This research also shed light on this thesis to focus on the family of SVM method.

- Panicker [136] (2010) proposed a method to combine two classifiers which are trained in two modes (Bayesian LDA and Fisher LDA). Although with the proposed ensemble strategy, their approach still has limitations in combining different algorithms of completely different nature, such as probabilistic methods (LDA) with deterministic methods (SVM, gradient-boosted decision tree). In the same year, Liu [19] proposed a dynamic stopping method almost similar to [18], except that Liu used a different flashing paradigm.

- Blankertz [137] (2011) conducted a P300 BCI in a totally novel flashing paradigm. Instead of flashing the rows and columns of the P300 matrix, they group all letters into 6 groups and divide the flashing task into 2 stages (stage 1: 6 groups flash sequentially, then the classifier output the focused group, then stage 2: all letters in that group flash). This approach can extend the spelling possibilities for users, from 36 letters up to 50 letters.

- Cecotti [37] (2011) proposed a totally different classification method than LDA or SVM. He constructed a multiclassifier approach combined by the convolutional neural network. The result although did not outperform ensemble SVM [40] and was just slightly better than LDA, but his work is of great value since it can combine several classifiers using a reliable neural network method, which is an idea never discovered before.
- Panicker [122] (2011) went out of the usual way in EEG spelling devices. He did not use P300 responses as the main means of detecting human intention, but using steady-state visually evoked potentials (SSVEP). The flashing paradigm of each P300 row/column is conducted all at the same time, but in different frequencies.
- Although a large number of BCI research had been done, it is until 2012 that Kindermans [8, 9] noticed a huge research gap of the subject independence issue (that is, exploitation of the brain patterns of the current experimental subject to boost up the performance). He proposed a Bayesian learning model for inter- as well as intra-subject classification. Later in 2013, he and his team improved the original proposal [10] with the extra integration of a language model.
- Schreuder [138] (2013) made a summary of dynamic stopping methods used in previous studies. He also proposed a systematic evaluation of dynamic stopping methods. The value of this work is that it showed the benefit of stopping methods, both fixed and dynamic, as well as thorough comparisons of 8 different dynamic stopping approaches.
- Ivo [139] (2014) analyzed the effects of mental workload and fatigue on the P300 BCI. This study concluded that satisfactory accuracies can be achieved by healthy participants with the P300 speller despite constant distraction and prolonged operation. However, to improve usability for the target user group of severely paralyzed patients, it was suggested to implement a BCI that adapts to the mental state of the user. This suggestion is also equivalent to the main

concern of this thesis (subject independence).

- Speier [125] (2014) modeled the P300 as a Markov process that can be indirectly observed through EEG response signals. This study shows that incorporating this natural language information significantly improves the performance of a BCI communication system. The result in this study, however, was only compared with LDA-based algorithms, hence cannot prove that it is better or worse than SVM-based methods.
- Among a humble number of research concerning ALS patients, the work of Mainsah [21] (2015) was of great value. They conducted P300 experiments on 10 disabled subjects with various levels of deceases, to evaluate the performance. Although the employed classification method was not innovative (Bayesian LDA), the merit of their work lies in the ALS dataset and the significance on ALS patients.
- Speier [140] (2015) incorporated advanced language models into the P300 speller using particle filtering. Although the general signal processing and classification framework were not novel, they proved that with their proposed error correction regime, the accuracy of P300 can be boosted by approximately 20%. We can view this research as an extra language model plug-in for any BCI system.
- Clements [23] (2016) was the first to test the P300 performance on dry EEG electrodes. They claimed that using dry electrodes is desirable for reduced set-up time, but the online performance is significantly poorer than wet electrodes for both non-disabled and disabled subjects. Another merit of this work is the demonstration that dynamic stopping mechanisms can drastically boost up dry electrode performance.
- A review on integrating language models into P300 BCI was conducted by Speier [126] (2016).

2.6 Chapter Summary

In this chapter, we have provided a thorough literature review on the following aspects:

- The architecture of the human brain, including the functional description of particular cortex areas.
- A typical framework of a BCI system, including the major three steps: data acquisition, processing algorithms, and output/feedback.
- Brief review on all brain data acquisition technologies, including invasive (intracortical, ECoG) and non-invasive methods (EEG, MEG, fMRI, PET, SPECT, fNIRS). On top of that, EEG is selected as the technology used in this thesis thanks to its popularity, inexpensiveness, and excessive availability of public datasets.
- Since EEG is selected, typical types of BCI systems employing EEG are reviewed, including stimulated EEG-based BCI (ERP, SSVEP), and spontaneous EEG-based BCI (SCP, SMR, mental tasks). On top of that, ERP is the selected main theme in this thesis thanks to its distinctive properties, which is hugely beneficial for machine learning algorithms.
- Then, typical studies on EEG-ERP-based BCI are reviewed thorough the history, covering all aspects: interface paradigms, machine learning methods, preprocessing techniques, dataset archival, postprocessing techniques (language model, reinforcement learning), or feature engineering.

We can now move on to review and analyze various aspects of ERP-based BCI, including the major components of ERPs and their properties, the stimulation paradigm, preprocessing techniques, dataset descriptions, and evaluation metrics, which will be mentioned in the next chapter.

Chapter 3

EVENT-RELATED POTENTIALS, DATA ACQUISITION, AND FEATURE ENGINEERING

In this chapter, we will present all aspects of our ERP-based BCI system. First, section 3.1 presents the history and the characteristics of ERPs, as well as their relevance to this thesis framework. Next, section 3.2 explains the formulation from the spelling application in general into the P3S interface, and describes the data acquisition method. The evaluation metrics used thoroughly in this thesis is presented in section 3.3. A couple of essential preprocessing techniques that are applied on the EEG signal collected from the P3S is described in section 3.4. Finally, section 3.5 outlines the specifications of two datasets employed in our study, followed by the chapter summary in section 3.6.

3.1 Event-Related Potentials (ERP)

3.1.1 A Brief History of ERP

The first EEG experiments conducted on human brain was reported by Hans Berger in 1929 [5]. Hans Berger demonstrated that one could monitor the electrical activity of the human brain by attaching the electrodes non-invasively and directly on the human scalp. The signal is coupled with an amplifier, and can be plotted in voltages with respect to time by a galvanometer with the support of an assistant. This study also marked the first advent of the term electroencephalography (EEG). Consequently, other respected phys-

iologists, such as Adrian and Matthews in 1934 [141], Jasper and Carmichael in 1935 [142], Gibbs, Davis, and Lennox in 1935 [143] carried out their own studies to confirm the phenomenal discovery of EEG.

Since then, EEG became a useful tool for scientists and clinical experts. EEG at the time was used for research in normal and abnormal brain rhythms, more specifically the alpha waves (or known as the Berger's waves) and beta waves [144]. These research was proved to have some impact on diagnosing epilepsy [144]. However, in its most primitive and coarse form, EEG at the time did not provide meaningful information to the field of cognitive neuroscience. In other words, due to the mix of hundreds of coarse EEG channels, one could hardly discriminate and extract useful neuro-cognitive responses of the subject. As a consequence, specific stimuli come up as a tool to stimulate those brain cognitive responses. These are called event-related potentials (ERP), as they are the potentials generated as the reaction to a specific stimulating event. ERP can be simply discriminated at the time by a simple averaging method [145].

The first research in ERP was done by Pauline and Hallowell Davis in 1939 [146], which confirmed the existence of clear ERP responses on awake humans. Then, it was not until the 1950s that ERP research was re-ignited, due to the World War II. The most notable work was reported by Walter [147] in 1964, in which he and his colleagues claimed the first cognitive ERP response, namely the contingent negative variation (CNV). More specifically, the subject is first shown a warning signal (i.e., a click), followed by a visual stimulus 500ms later. The subject is required to press a button whenever he/she notices the latter stimulus. In the EEG signal, the authors observed that besides the two sensory responses of the warning and visual signal, there also exists another large negative voltage that separates those two sensory responses. This abnormal pattern reflects subject's preparation the target visual stimuli. The remarkable discovery had sparked a light and led other experts to research on cognitive ERP patterns.

In 1965, a major discovery in ERP was published by Sutton and his col-

leagues [148]. In this study, the subject was presented with two different kinds of trials: the certain kind and uncertain kind. In each kind, there were 2 stimuli occurring consecutively: the warning stimulus and the test stimulus, which followed a random interval of 3 to 5 seconds after the warning stimulus. In the certain kind of trial, the warning stimulus let the subject know the sensory modality, or the type, of the test stimulus, i.e., a sound or a flash. On the other hand, in the uncertain kind, another different warning stimulus was presented which does not let the subject know the type of the following test stimulus. In each experiment, the subject was presented approximately 1000 trials of two kinds in a random sequence. The subject is required to state the type of the test stimulus after being presented with the warning stimulus. The experiments were conducted on 5 subjects. The results showed that, for all subjects, there exists an abnormal pattern in the averaged waveform of the uncertain kind, as compared to the averaged waveform of the certain kind. This pattern occurred roughly 300ms after the warning stimulus. Hence, the so-called name of P300, or P3, which stands for positive 300ms, was born.

The advent of P300 attracted a huge amount of subsequent research by cognitive psychologists and scientists. Up to 2006, there were approximately more than 3000 journal articles having P300 in the title or abstract [145]. A large number of these research aimed to develop methods in recording and experimenting cognitive trials, or analyze or identify ERP's components. P300, however, still plays a major role nowadays as one of the most prominent form of ERP, and is exploited regularly for the development of various cognitive applications [78–80]. Besides P300, some other major types of ERP are briefly reviewed in the next section.

3.1.2 The ERP types in this Thesis

In this section, all of the ERP that are produced by our EEG data acquisition framework are briefly summarized. Before continuing, it is convenient to retrieve a compact definition of ERP.

Definition 3.1: An event-related potential (ERP) is the electrical response

of the brain to a sensory event (related to human sense, i.e., visual, auditory), a cognitive event (related to human cognition processes, i.e., preparation state, surprising state), or a motor event (related to movements, i.e., left/right lifting) [145].

In this thesis, the P3S is used as the backbone interface framework, which will be presented in section 3.2.1. As a result, we only take care with some visual and cognitive ERP responses that can be produced through the visual flashing paradigm of the P300-Speller (or also known as the *oddball paradigm* which will be presented in Section 3.1.2.2). For the extensive exploration of all other kinds of ERP, the reader is suggested to look at the following thorough review articles [149–153].

3.1.2.1 Visual ERP

C1: C1 was the first major discovered visual sensory ERP, or visually evoked potentials (VEP) [154]. The polarity of C1 is uncertain, as it can be a negative-going or a positive-going waveform. C1 starts at approximately 40-60ms and peaks 80-100ms post-stimulus onset. C1 is highly sensitive to stimulus settings, i.e., luminance, contrast, spacial frequency. C1 is typically found at the primary visual, or striate, cortex (V1), and is invariate to different levels of attention [155, 156]. Practically, however, C1 is often overlaps with P1, which will be presented next.

P1: The P1 is the positive wave typically begins rising 60-90ms post-stimulus and peaks at 100-130ms at the lateral occipital region [145]. The early P1 part arises from the dorsal extrastriate cortex (or middle occipital gyrus), while the later part arises from the fusiform gyrus [157]. Similar to C1, the latency of P1 is also highly sensitive to stimulus parameters such as spacial attention [156], the subject's state of arousal [158], or the visual contrast [145]. Unlike C1, P1 is variate to different levels of attention [159].

N1: The N1 is the negative-going wave which occurs after the P1, at approximately 150-200ms post-stimulus. Unlike C1 and P1, N1 is found at almost brain regions, including the occipital, parietal, central, and frontal electrode

sites [160]. There are several sub-components in N1 which arise at different sites and are influenced by the spacial attention [156,161]. N1 can be generated from both the right and left hemispheres. However, the laterality (centrally, laterally, or bilaterally) of the stimulus dictates the laterality of N1. For instance, if the stimulus is presented centrally or bilaterally (one in each visual side), the N1 is bilateral. If the stimulus is presented unilaterally, the N1 is larger and earlier contralaterally (i.e., on the opposite side) [162]. This property of laterality enables N1 to be a useful ERP to develop multi-purpose VEP systems.

P2: The P2 wave follows N1 and occurs approximately 150-275ms post-stimulus. P2 is most distinct at the centro-frontal, parieto-occipital, and especially frontal areas. As suggested by Freunberger [163] and Luck [164], the P2 may be a part of cognitive matching system that compares sensory inputs with stored memory. As a result the amplitude of P2 is larger for infrequent stimuli [164]. This notion may, in some sense, coincides with the occurrences of P300 (or P3), which will be presented next. However, the P2 effect occurs only as a simple reaction to visual stimuli, while P3 effects can be modulated by more complex cognitive target parameters. In practice, however, the P2 wave is often overlapped with N1, N2, and P3 [145]. The vagueness of P2 is demonstrated in Figure 3.2. Distinguishing P2 is therefore a challenging task.

3.1.2.2 Cognitive ERP

N2: One of the most prominent ERP in cognitive processes is the N2, or also known as N200. N2 is a negative-going wave that typically peaks at 200-350ms post-stimulus and is primarily detected at the anterior scalp sites [165]. N2 can be produced through the Eriksen flanker task [166], or the go/no-go task [167]. In this thesis, the P3 wave (which will be presented next) is the fundamental ERP, hence N2 only plays a minimal role due to the irrelevance of P3S in producing N2. The N2 wave, in this case, can be considered an extra visual ERP to enhance the classification performance.

P3: P3, or also known as P300, is one of the most remarkable cognitive ERP discovery. As described in the last paragraph of section 3.1.1, P3

was originally produced by the paired-stimuli experiments [148] in which the subjects were not certain about the modality of the second stimulus of the uncertain kind. Some studies claimed that P3 reflects the human brain's processes of evaluation or categorization the stimulus, or the process of *context updating* [168]. However, these are still proposed hypotheses, and there is still no firm consensus about what is the exact neural or cognitive process the P3 reflects [164]. Since there are thousands of published P3 studies to date, we will summarize the most important key points about P3 that are relevant in this thesis's P3S framework.

- *The Oddball Paradigm:* The P3 wave is produced by the *oddball paradigm*. In this paradigm, there are stimuli (or trials) of 2 kinds: the target trials and the non-target trials. The two kinds stimuli can be auditory: soft-tone clicks (70 dB) and loud-tone clicks (90 dB), or visual: flash and non-flash. The probability of one kind is significantly less than the other (i.e., soft-tone with 0.1 probability and loud-tone with 0.9 probability, or flash with 0.2 probability and non-flash with 0.8 probability). The trials with less probability are defined as the target trials, and the other defined as the non-target trials. When presented the mixed consecutive trials of these two kinds, the subject typically produces the P3 wave in target trials.
- *Sub-components of P3:* Squires and his colleagues [169] suggested that there are 2 sub-components in P3. The P3a sub-component occurs in an unattended task. Specifically, when experimenting the soft-tone/loud-tone test, the subjects were instructed to silently read a text and ignore the auditory stimuli. This is called the unattended task. Squires found that, the less-frequent tone (or target trials) elicited a positive-going wave at typically 220-280ms post-stimulus, which was later named as P3a. On the other hand, in the attended task in which the subjects were told to pay attention to (i.e., silently count) the target tones, another positive-going potential usually peaks at 310-380ms post-stimulus, and at different scalp sites as compared to the P3a. They termed this as the P3b. Experts, however, usually refer

P3 to the P3b, as the major designed paradigm focused on attended tasks [145]. In this thesis, this is also true as the P3S flashing paradigm is an attended task.

- *Factors influencing the amplitude of P3:* There are numerous factors influencing the amplitude and latency of P3, which were extensively reviewed in [170–174]. In terms of amplitude influence, the most dominating factor is the target probability. In the work of Duncan-Johnson [175], the P3 amplitude was proved to get significantly larger when the target probability drops. Moreover, the level of influence is also affected by the local probability. For instance, two consecutive target trials would decrease the P3 amplitude, or consecutive non-target trials would enhance the P3 discriminativeness of the subsequent target trial [145]. The difficulty of the task is also a factor influencing the P3 amplitude. The P3 peak tends to be smaller when the subject is uncertain about the kind of the target (i.e., a middle tone between soft-tone and loud-tone) [176]. Beside the target's probability, and the difficulty of the task which can be viewed as objective factors, there are also subjective factors. The most prominent subjective factor is the subject's level of attention. The P3 amplitude is found to be larger when the subject puts more efforts into the task [177]. This leads to a trade-off between the difficulty of the task and the subject's level of attention. An experiment with more kinds of stimuli, or an experiment with complicated instructions, would inevitably increase the subject's concentration and enhance P3, while at the same time diminish it due to the difficulty of the task [170].
- *Factors influencing the latency of P3:* Since P3 depends on the probability of the categories of the stimuli, it is logically necessary to assume that P3 is produced after the categorization cognitive process has been done [145]. As a result, P3 latency is influenced by the time required for this categorization process. For example, a blurred flashed symbol, which requires an extra time for the human brain to categorize it into the target or non-target class, will postpone the P3 occurrence. However, P3 is proved to be insensitive to post-categorization processes.

For example, if the subjects are required to press the left-hand button when they see stimulus A, and press the right-hand button when they see stimulus B, the P3 latency is found to be constant if they are required to press the right-hand button when they see stimulus A and press the left-hand button when they see stimulus B [178, 179].

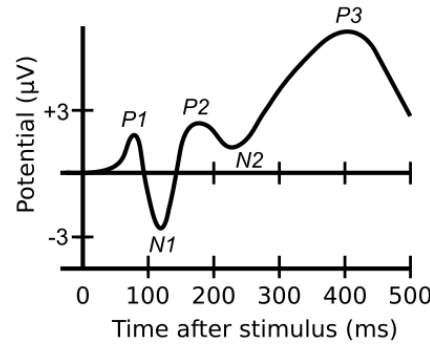


Figure 3.1: The typical ERP components.

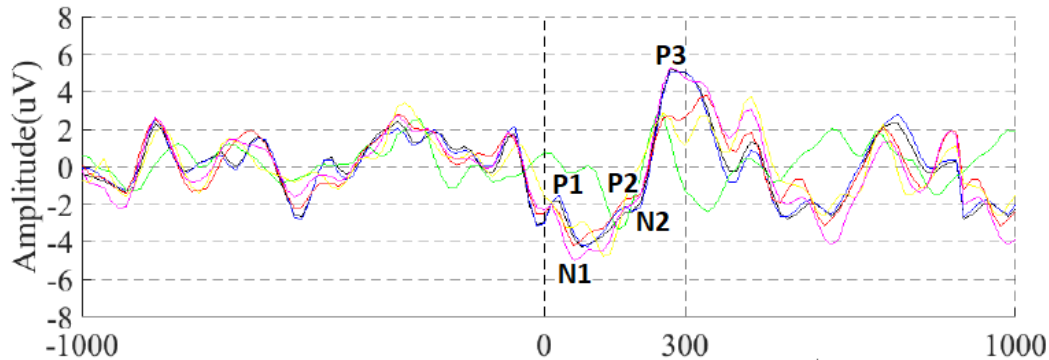


Figure 3.2: The averaged ERP components found in this thesis.

3.2 Data Acquisition Framework

3.2.1 P300-Speller Interface

An EEG headset is worn by the subject and is connected to the computer in real-time. On the screen, there is a matrix of alphabetical letters, or called as the P300-Speller (P3S). The number of entries in this matrix can be varied, as well as the design of each letter's position can be designed to match specific applications. For instance, the conventional P3S consists of 36 letters in a matrix of 6 rows and 6 columns (which is also the P3S used in this thesis).

Each row or columns in the set of 12 rows and columns is flashed, then faded sequentially on the screen. This flashing sequence is referred to as the row-column flashing paradigm.

The goal of the subject is to spell any letter in the P3S as he/she wants to. This is done by the subject focusing his/her eyesight on the letter on the screen that he/she wants to spell. There is no further action required for the subject except keeping his/her eyesight on the desirable letter, until an output letter result is popped up on the screen indicating that the spelling session for that letter is complete. This result may be correct or incorrect, and the subject can choose his/her next desirable letter to spell in the next session.

Before using the system, the subject may be required to conduct a training phase to let the system have the knowledge of his/her specific P3 brain patterns. In each spelling session of the training stage, he/she is instructed to spell the letter that the system knows beforehand.

In this thesis, as the subject-independent aspect is employed, all new subjects are exempted from this time-consuming and elaborate training stage. The system model is trained from past subjects' patterns which are sufficiently statistical for the ensemble and adaptive mechanism. Figure 3.3 describes in details the design of the P3S, its row/column numbering, and an example of a typical 15-iteration spelling session for 1 letter. In Figure 3.3, there are 15 iterations of 12 scramble flashes of 12 rows/columns, in which there are 2 target and 10 non-target flashes. Ideally, there is a P3 response occurred approximately 300ms after the onset of each target flash. Each flash lasts for a duration of inter-stimulus interval (ISI). In the classification stage, each flash is associated with a feature vector beginning at the flash's onset. As there are $12 \times 15 = 180$ flashes in one session, we have the maximum of 180 post-stimulus feature vectors in a static stopping flashing paradigm.

3.2.2 Flashing Paradigm

This section will describe in detail each step of the P3S flashing paradigm. The described steps are corresponding with the subfigures in Figure 3.4.

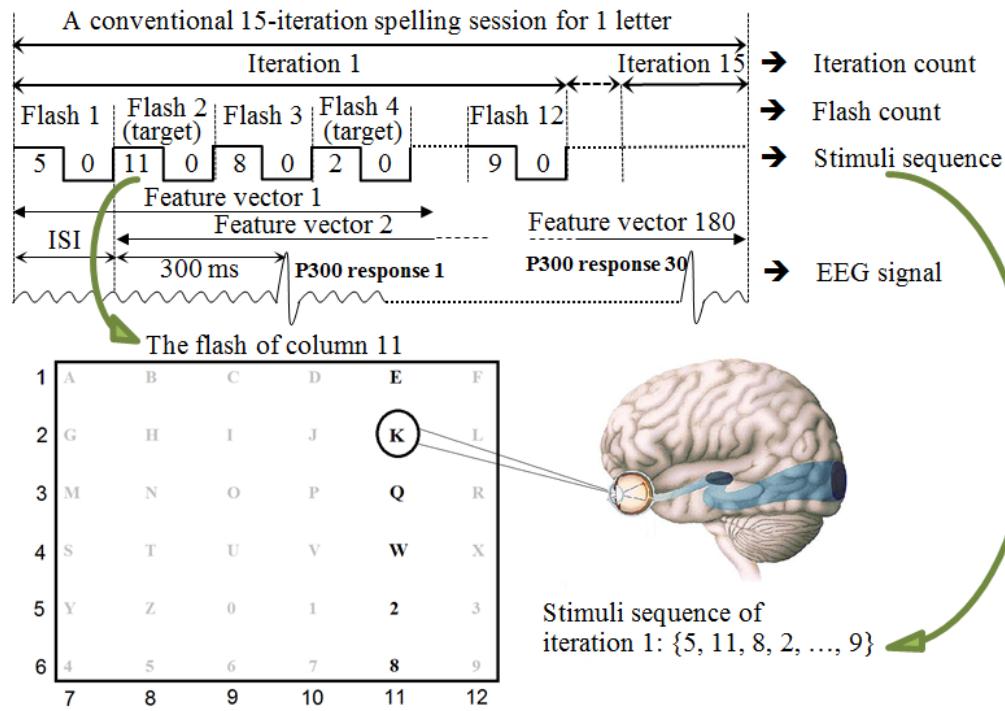


Figure 3.3: The timeline of signal stream and flashing paradigm of a one-letter spelling session.

Step 1: This step corresponds to Figure 3.4a. In this step, the subject begins his/her spelling session. He/she sits in front of a screen displaying the P3S. He/she veers his/her eyesight to the desirable letter that he/she wants to spell, which is the letter K in this figure. The flashing paradigm begin. Each of 6 rows and 6 columns of letters will flash consecutively in a random order. In this step, the first column is flashed for 125ms, then faded for the next 125ms. The EEG signal is collected for the whole 250ms, and is indicated as the light blue square in the EEG signal stream.

Step 2: This step corresponds to Figure 3.4b and is the subsequent step to step 1. The flashing of the fourth row is conducted (as a result of randomly generated order of flashing sequences of rows/columns). This fourth row of letters is flashed for 125ms, then faded for 125ms, resulting in the added signal as indicated in the light blue square. This process of 250ms is repeated for all other rows/columns.

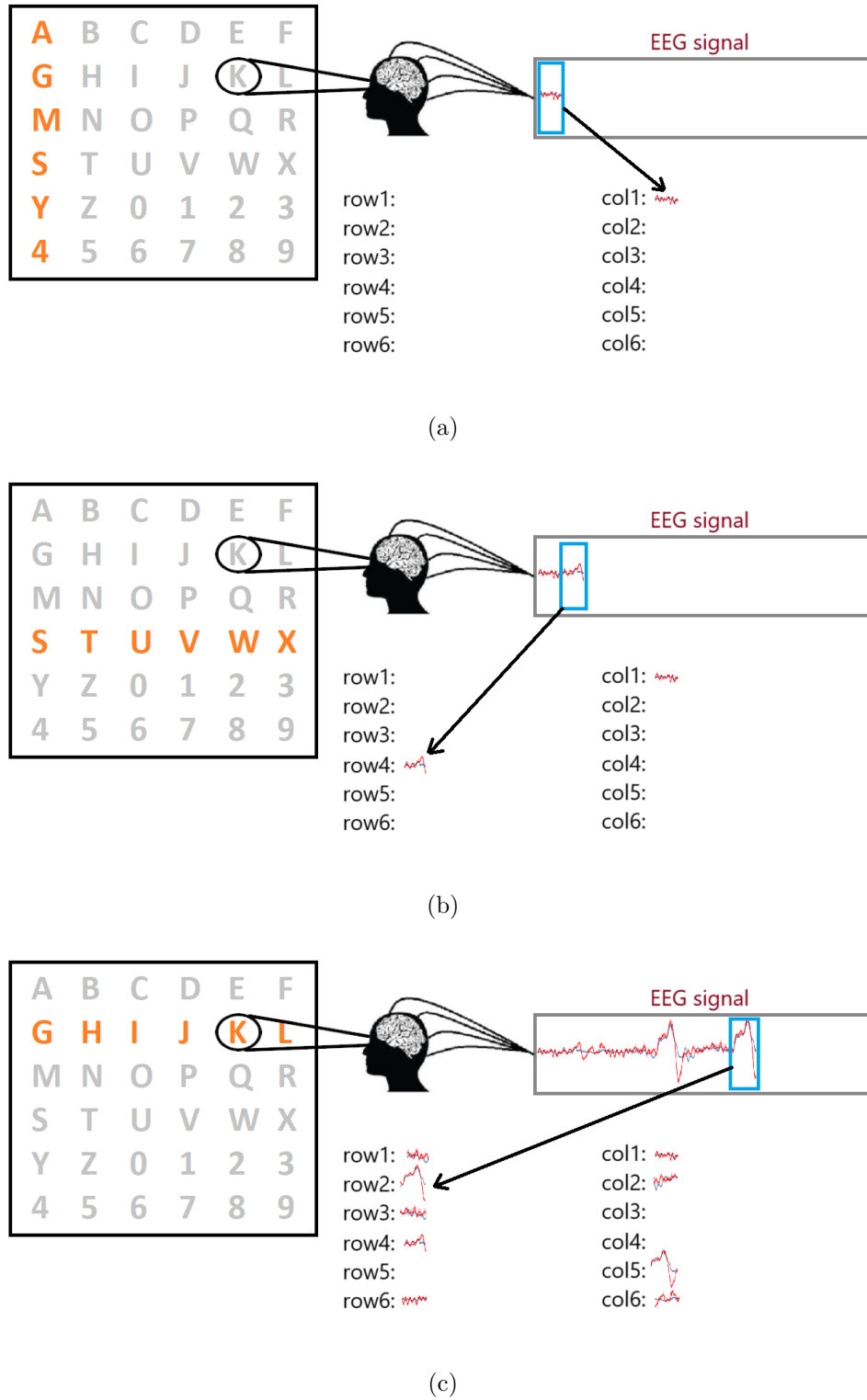


Figure 3.4: The steps of the row-column flashing paradigm of the P3S

Step 3: This step corresponds to Figure 3.4c. After some repeated flashes as described in step 2, it is now necessary for us to describe a step in which

the flashed row/column is a target one. In this step, the second row is flashed, which contains letter K, which is aimed by the subject. As a result, the signal stream will typically have some unique P3 pattern specific to this target flash (or target trial). As depicted in the light blue square, this signal segment has an unusual higher amplitude. There also presents another short segment of higher amplitude that precedes somewhere before this step. That is the signal segment corresponding to the flash of the fifth row from the left hand side.

Step 4: This step corresponds to Figure 3.5. After finishing a set of 12 flashes, the classification score for each row/column is computed. As shown, typically the second row and the fifth column will have the best scores that correspond to P3 response. This results in the letter K as the intersection of the second row and the fifth column, and is output by the computer.

In practice, only one set of 12 flashes does not provide scores that can significantly discriminate between target and non-target responses. Therefore, a couple of sets of 12 flashes will be repeated until the system is confident enough about its decision. This is called *dynamic stopping* (DS), and will be presented in Chapter 6.

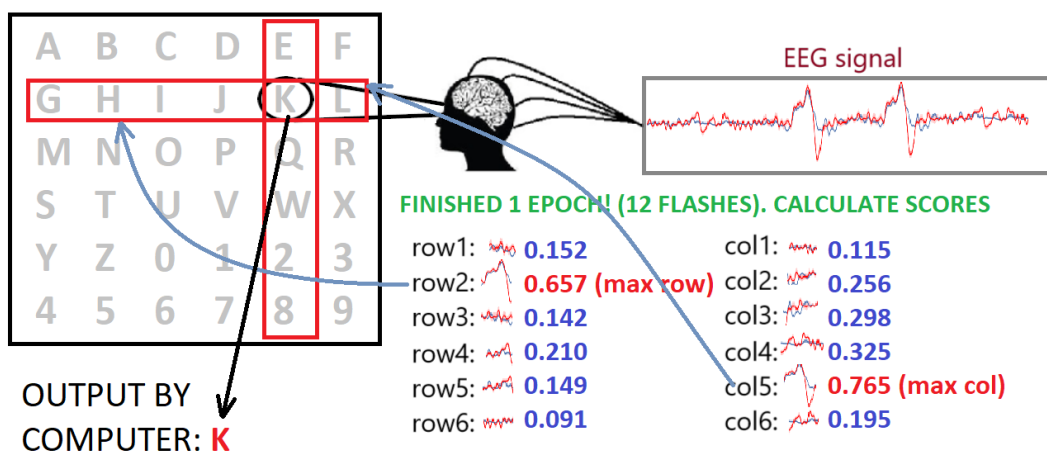


Figure 3.5: The classification process of the row-column flashing paradigm after each iteration.

3.3 Evaluation Metrics

Throughout this thesis, we employ different evaluation metrics depending on different experimental contexts. With regards to the general performance of the BCI system, especially P3S, there are three main metrics: accuracy, iterations, and bit rate. With regards to evaluating the capability of a machine learning algorithm, there are several measures, such as accuracy, precision, F-score, etc. We will describe each metric and the contexts in which it is employed.

3.3.1 Accuracy

The term *accuracy* may refer to different metrics. In the context of measuring the accuracy of the P3S, we refer accuracy to the symbol-wise accuracy, or letter-wise accuracy, i.e., the ratio of the number of correctly classified symbols (or letters) over the total number of symbols (or letters) spelled by a subject

$$A_{\text{symbol}} = \frac{\text{correctly classified letters}}{\text{total letters spelled}} = \frac{\text{correctly classified symbols}}{\text{total symbols spelled}}. \quad (3.1)$$

In some cases, there arise the need to calculate the classification accuracy trial-wise, i.e., the ratio of the number of correctly classified trials (or data samples) over the total number of trials (or data samples) extracted from a spelling session

$$A_{\text{trial}} = \frac{\text{correctly classified trials}}{\text{total trials extracted}}. \quad (3.2)$$

3.3.2 Iteration

Iteration is simply the number of repetitions of one spelling sequence that covers all symbols in the P3S

$$\text{Iteration} = \text{number of repetitions that covers all P3S symbols}. \quad (3.3)$$

In the P3S of 6 rows and 6 columns using the row/column flashing paradigm, one iteration equals 12 flashes or 12 inter-stimulus intervals. If we use the same P3S design but with the single symbol flashing paradigm (flash each symbol after another rather than the whole row/column), an iteration will consist of 36 flashes. Iteration is the a metric usually used to measure the amount of time spent to spell one symbol. In the static stopping flashing paradigm, i.e.,

there are a fixed of 15 iterations for every symbol, iteration is also fixed at 15. In this case this metric does not provide any useful information. Iteration is used, therefore, more frequently in dynamic stopping flashing paradigm, i.e., there are a dynamic number of iterations per each spelling symbol session depending on the subject's concentration. In this case, iteration is a useful metric. It directly refers to the number of averaging repetitions to provide statistically sufficient data to determine the target symbol.

3.3.3 Bit rate

Accuracy and time are two metrics with opposing trends. One can significantly decrease the amount of time spent on a session with the trade-off in accuracy. In BCI, a third metric to combine the accuracy and speed, called *bit rate*, was developed to measure the amount of correct information produced in one minute

$$\text{Bit rate} = \frac{B \times 60}{\text{ISI} \times (\text{number of total flashes})} \quad (\text{bits/min}) \quad (3.4)$$

where ISI is the inter-stimulus interval (seconds), and B is the bit rate per trial which can be computed as

$$B = \log_2 L + A_{\text{symbol}} \log_2 A_{\text{symbol}} + (1 - A_{\text{symbol}}) \log_2 \frac{1 - A_{\text{symbol}}}{L - 1} \quad (\text{bits/trial}), \quad (3.5)$$

where L is the number of symbols of the P3S.

3.3.4 Precision

The metric of accuracy has one clear disadvantage if the ratio of the classes is unbalanced. For instance in binary classification of P3S, if the number of non-target trials is far higher than the number of target trials, i.e., 9 times higher, the resulting accuracy may be always high. Let us imagine the case when the classification algorithm always assigns every input sample as non-target, the accuracy is therefore always kept at 90%, which is very high.

Precision is a metric to measure the accuracy of positive labels. In the context of P3S, precision can be understood as the accuracy of data samples classified as target. A high precision means that the algorithm is trustworthy

whenever it outputs a sample as target (positive). Conversely, a low precision means that the algorithm produces too many false alarms, or false positive samples. Precision, on the other hand, does not take into account any negative samples (i.e., the samples classified as non-target), hence we cannot conclude anything about the accuracy of negative-classified samples from precision.

$$\text{Precision} = pr = \frac{\text{correct trials classified as positive (or target)}}{\text{total trials classified as positive (or target)}}. \quad (3.6)$$

3.3.5 Sensitivity (or Recall)

Sensitivity, or sometimes known as *recall*, is a metric to measure the capability of an algorithm to correctly detect positive samples (target trials). A high sensitivity means that the algorithm has high capability to detect positive samples. A low sensitivity, on the other hand, can deem the algorithm almost useless. Let us consider the same context as mentioned previously in the precision section. If the algorithm is able to correctly detect one target trial, the accuracy will be shifted from 90% to 91% instantly. Sensitivity is computed as the ratio of correctly-classified target trials and the total number of true positive trials

$$\text{Sensitivity} = se = \frac{\text{correct trials classified as positive (or target)}}{\text{total true positive (target) trials}}. \quad (3.7)$$

3.3.6 False Counts

In some experiments when the total number of samples is very small, both precision and sensitivity are extremely sensitive to the classification result. A few samples, when swapping their classified labels, may heavily affect precision and sensitivity. Therefore, the count of falsely classified samples is necessary to provide a better understanding on the classification result. False counts is simply calculated as the total of falsely-classified trials, including the true target trials classified as non-target and the true non-target samples classified as target.

$$\text{False counts} = fc = \text{Number of falsely classified trials}. \quad (3.8)$$

3.3.7 F-score

Equivalently to bit rate in P3S metrics which is a harmonic combination of accuracy and speed, F-score is a harmonic average of precision and sensitivity.

F-score, or also known as F1-score or F-measure, approaches 1 at high precision and high sensitivity, and approaches 0 at low precision and low sensitivity

$$\text{F-score} = f_s = \frac{2 \times pr \times se}{pr + se}. \quad (3.9)$$

For notational convenience, we can define four types of classified samples, namely *true positive* (TP), *true negative* (TN), *false positive* (FP), and *false negative* (FN). All of these measures are derived from the following *confusion matrix*

		True label	
		Target	Non-target
Predicted	Target	TP	FP
	Non-target	FN	TN

As a consequence, precision, sensitivity, and false counts can be rewritten as

$$\begin{aligned} pr &= \frac{TP}{TP + FP}, \\ re &= \frac{TP}{TP + FN}, \\ fc &= FP + FN. \end{aligned} \quad (3.10)$$

3.4 Signal Preprocessing

3.4.1 Averaging

Averaging is the simple technique to enhance the classification performance by elongating the experimental session. The resulting averaged trials representing the classes are likely to converge to the typical pattern specifically to the subject. The averaging process is conducted in a point-by-point manner, and the amount of noise is decreased as the number of trials for averaging increases.

Given N trials with the amount of noise in each trial denoted by γ , the amount of noise in the averaged trial is given by [180]

$$\frac{\gamma}{\sqrt{N}}. \quad (3.11)$$

In other words, the averaged noise decreases as a function of the square root of the number of trials. As a result, doubling the signal to noise ratio (SNR) of the averaged trial requires four times as many trials. This leads to the

fact that, to achieve a significantly better SNR, one would need to massively prolong the experimental session. As a result, one would usually find ways to avoid noise rather than increasing experimental time.

Influence of Averaging on P3 Amplitude:

There are several assumptions when using the averaging technique. First, the underlying neural response in every trial is the same. This assumption, however, can be violated in some ways. As an example, the visual P1 waveform's amplitude may vary, but the averaged waveform's amplitude can reflect the actual P1 amplitude specifically to the subject. Another example is that, some different ERPs may not be present at the same specific single trial simultaneously, but the averaged waveform contain all those ERPs.

In this thesis, we conduct the ERP analysis on three different subjects. The details of the experiments and the subjects' information are presented in section 3.5. The averaging waveforms of target and non-target trials of all three subjects, namely subject A, subject B, and subject C are presented in Figure 3.6, Figure 3.7, and Figure 3.8 respectively. It should be noted that, all mentioned figures plot the averaged trials that had been artifact-removed, filtered, and decimated. The artifact removal, filtering and decimation process will be described in the next section.

The most distinct and theoretically correct target ERP waveform is found in subject A. The averaged waveform of subject A with 100 trials or higher (the sub-figures on the right column) are almost identical, reflecting the stability of individual responses. More interestingly, this subject also produces all of the typical ERPs in his/her target waveform. The average target waveform (the blue line plot) of 480 trials of subject A is extremely similar to the typical single ERP response presented previously in Figure 3.1, as the P1, N1, P2, N2, and P3 components are all present. The average non-target waveform (the orange line plot) of 480 trials of subject A, also reflects the typical non-target response by presenting three low peaks, which represent three visual ERP responses. These three small responses correspond to three flashes in the interval of 625ms (since the inter-stimulus interval of our experiment is

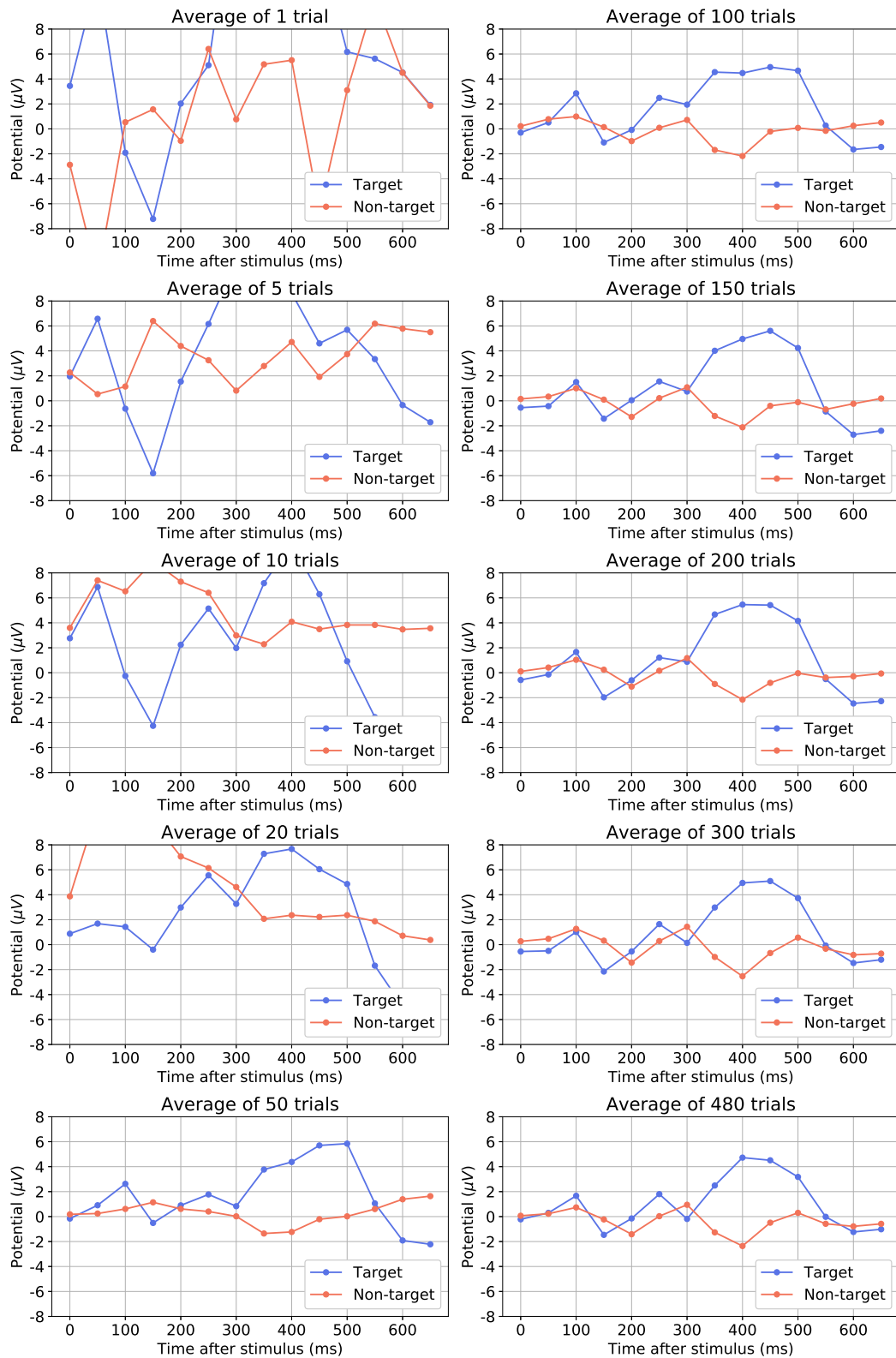


Figure 3.6: The averaged ERP waveforms of subject A

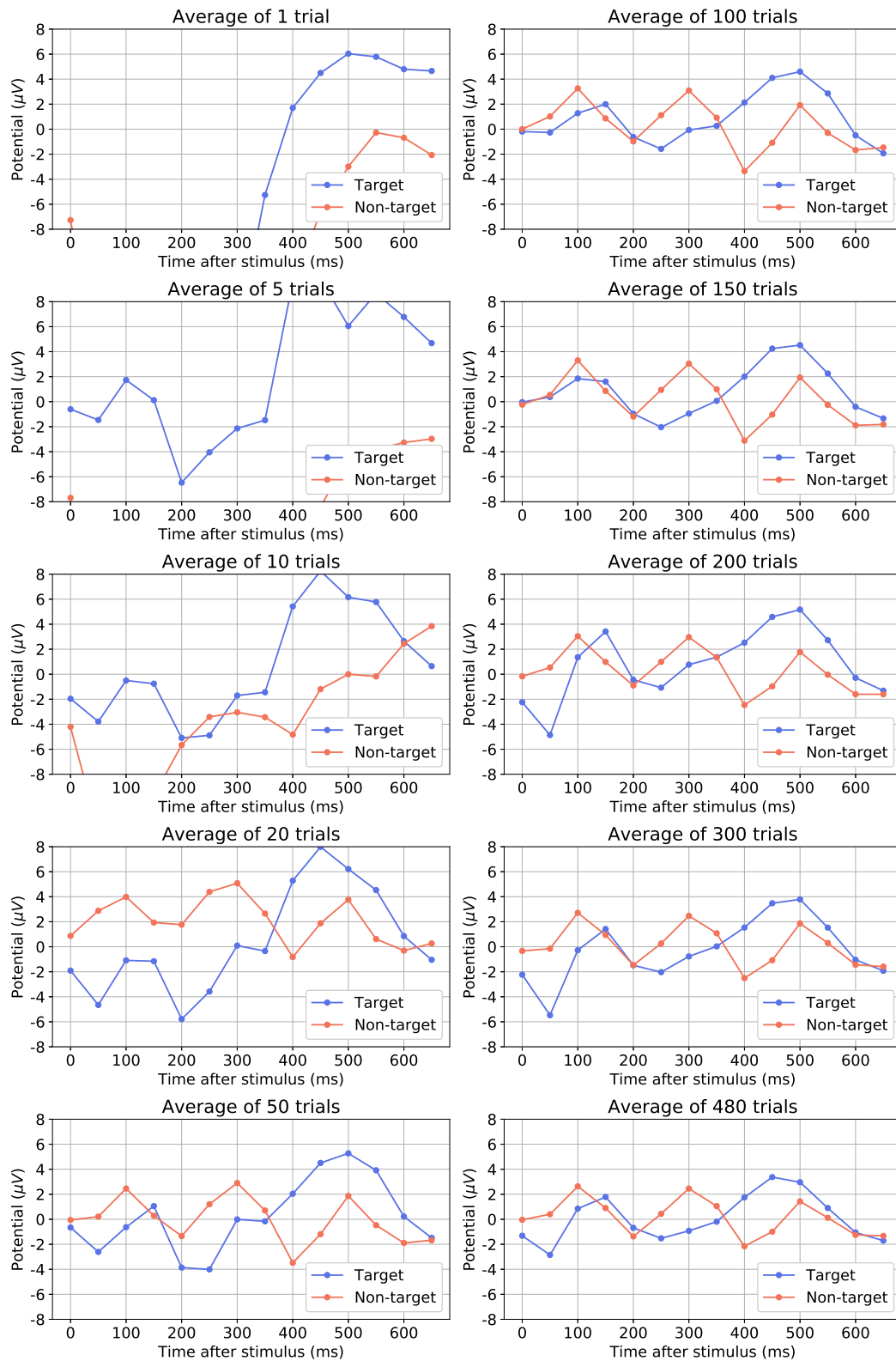


Figure 3.7: The averaged ERP waveforms of subject B

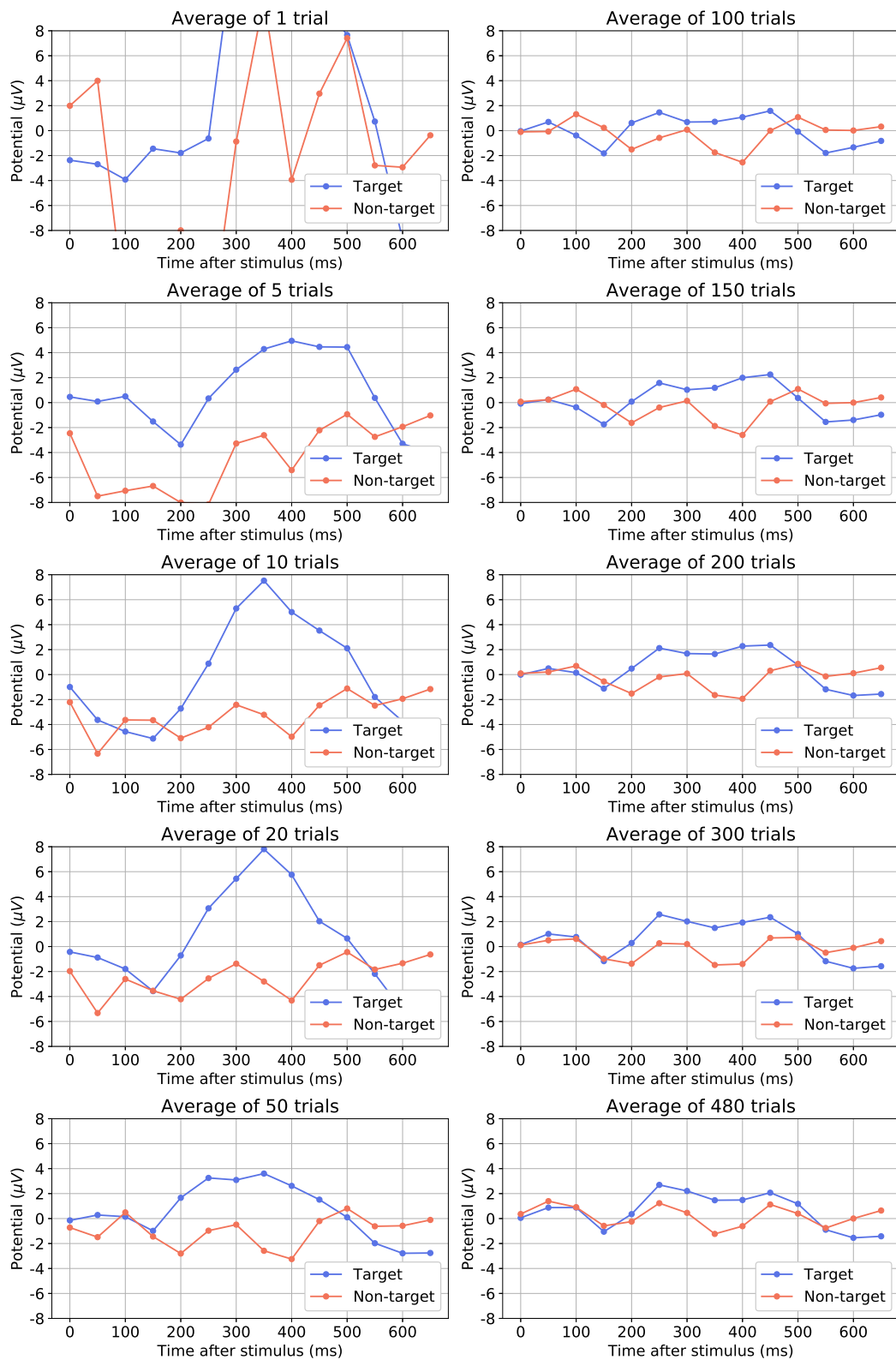


Figure 3.8: The averaged ERP waveforms of subject C

250ms). In subject B, as depicted in Figure 3.7, the average of 480 non-target trials is more discriminative, and the peaks are nearly equal to the P3 peak of the average target trial. Although in each non-target peak, we do not know the exact contribution of the visual ERP components (P1, N1, and P2), we however have enough evidence to entrust our experimental scheme that it can produce the desirable ERP responses.

Influence of Averaging on P3 latency:

Although amplitude variability across trials may not be problematic, the latency variability, however, is a significant concern in averaging technique. One example is that the averaged peak's amplitude may be broader and significantly lower than the typical single peak. In the worst case, the averaged target pattern can become indiscriminative as compared to the non-target pattern [181].

The latency variability problem is present in subject C, as depicted in Figure 3.8. The peak of the 480-trial averaged target waveform is significantly broader than those found in subject A and subject B. This is the consequence of the variability of P3 latency across target trials.

There are various method to tackle the problem of latency variability, such as area measures, response-locked averages [145], Woody filter [182], or time-locked spectral averaging [183, 184]. In this thesis, the variability of P3 amplitude and latency is internally embedded and evaluated by the classification algorithm. The specific pattern of each subject is, therefore, of more concern than the general properties of ERP responses. In other words, we care more about how to discriminate target and non-target trials specific to a subject (subject-independent) regardless of the target and non-target ERP patterns. This is done by the adaptive learning method using adversarial neural network, which will be presented in Chapter 4.

3.4.2 Artifact Removal

Unlike the effects of the averaging technique on classification performance which can be resolved by the classification algorithm itself as explained in the previous section, EEG artifacts are hugely detrimental and must be handled

separately. There are several artifacts which can massively distort the EEG signal, such as eye blinks, eye movements, or muscle movements. These artifacts are very large as compared to the desired ERPs. A single trial with the present of an artifact can hugely ruin the averaging process.

In this thesis, it is only required to deal with eye blinks as the most influential artifact in the P3S framework. Other types of artifacts are found to be negligible. For instance, the position of the letter the subject wants to elicit is fixed on the screen, hence eye movement artifacts are not usually present. Also, there is no requirement for the subject to conduct muscle movements like pressing buttons or moving hands, hence muscle-related artifacts can also be neglected.

There are two major approaches to handle eye blinks. First, one would completely discard all contaminated trials (the trials considered as possessing an eye blink) from the averaging process. Second, one would keep all trials but try to reduce the influence of eye blinks in contaminated trials. We will analyze both approaches, make comparisons, and select the better approach as the final artifact removal technique.

When blinking, a large voltage deflection occurs at the surrounding regions of the eyes on the scalp. An eye blink typically produce a deflection of absolute 50 to 100 μV with duration of 200ms to 400ms [185].

Windowed Peak-to-Peak Thresholding (WPPT):

For the discarding approach, one common and simple method is the peak-to-peak thresholding (PPT) [145]. In PPT, one would simply get the lowest and the highest point in a trial and make a subtraction. If the subtracted value is greater than a pre-defined threshold, that trial is considered as having an eye blink and should be discarded from the averaging process. However, this simple method can malfunction if there exists voltage shifts that cause one end of a trial to be significantly lower or higher than the other end. As a result, one would use a slightly modified method called *windowed peak-to-peak thresholding* (WPPT).

In WPPT, we define the width of the window, i.e., 100ms. For each point in the trial, we subtract the mean of the points in the preceding window with the mean of the points in the subsequent window. The thresholding process is carried on the windowed signal. The benefit of WPPT as compared to PPT is that WPPT can remove high-frequency oscillations caused by external sources (i.e., system measurement noise) or internal neural activities (i.e., muscle movements). Another benefit of WPPT is that it can minimize the effects of gradual voltage shifts, resulting in eye blinks considered only as sudden voltage changes.

Windsorizing:

For the artifact suppressing approach, the most widely-used method is windsorizing [41]. In this method, we define two threshold percentiles, i.e., the 10th percentile and the 90th percentile. These percentiles are computed for the data points from each electrode. The data points lying below the 10th percentile or above the 90th percentile will be replaced by the 10th percentile or the 90th percentile, respectively.

Windsorizing has two major advantages as compared to WPPT. First, the calculation can be done on the whole EEG signal rather than single trials. This property yields great benefit to the signal processing framework in the real-time fashion. Second, no trial will be discarded, resulting in shorter session. There is, certainly, a trade-off that the averaged trial may be heavily distorted if windsorizing is not properly tuned.

Comparing WPPT and Windsorizing:

To inspect and compare WPPT and windsorizing, we conducted an experiment using the EEG Eye State dataset [186]. This dataset consists of a 117-second continuous EEG data collected by the Emotiv EPOC EEG headset [187]. The state of the eye was labeled by a camera by the dataset provider. We only extracted the eye blinks, i.e., the EEG intervals in which the state transitions occur. We selected the intervals of 1 second and let the blinks occur at approximately middle of the intervals. We also extracted the intervals of 1 second in which there is no eye blinks. A total of 100 blink samples and 100 non-blink

TABLE 3.1: THE CONFUSION MATRIX PERFORMANCE OF WPPT AND WINDSORIZING

		True label	
		Blink	Non-blink
WPPT Predictions	Blink	93	36
	Non-blink	7	64

		Blink	Non-blink
		Blink	Non-blink
Windsorizing Predictions	Blink	98	7
	Non-blink	2	93

samples were extracted to learn a simple SVM classifier. This trained classifier is used to evaluate the windsorizing technique. The evaluation is done on another 100 blink samples, and another 100 non-blink samples. On the other hand, for WPPT as the unsupervised approach, the performance measurement is calculated directly. The window size of 100ms and the threshold of $40\mu V$ were chosen. The evaluation samples for WPPT is the same as those used for windsorizing.

Table 3.1 presents the classification confusion matrix results of two approaches. As shown, the performance of windsorizing is significant better than WPPT. Although we do not know why exactly windsorizing outperformed WPPT, we somehow can lean on the truth that practically supervised machine learning approaches usually outperforms unsupervised ones. Moreover, as explained previously, data acquisition errors (voltage shifts) and various unknown factors may heavily govern the performance of WPPT, while in windsorizing, the false negative samples (the blink samples with classified label as non-blink) does not cause too much detrimental effects thanks to the percentile-swapped data points. As a result, in this thesis we select windsorizing as the artifact removal method.

3.4.3 EEG Filtering

EEG data filtering is an essential and even indispensable process in almost every EEG applications. There are two major reasons to explain why filtering is important. First, it allows us to significantly reduce the amount of collected

data. The reduction can be massive when the EEG data consists of multiple, i.e., 64 or 128 channels (or electrodes). The Nyquist theorem [188] states that there is no information loss if the digitized rate is at least twice as the highest frequency of the signal. Second, and more importantly, filtering reduces noise. For instance, muscle activities usually produce high-frequency oscillations (i.e., above 100 Hz), and this would detrimentally affect typical cognitive activities related to P3S which is lower than 20 Hz. As shown in Figure 3.9, the high-pass filtered signal is smoother, and the high-frequency noise is removed and depicted in the last sub-figure.

On the other hand, suppressing low frequencies (i.e., below 0.1 Hz) is also necessary to remove slow voltage shifting due to electrode impedance shifting during data acquisition process. As depicted in Figure 3.10, a signal segment of 15 seconds is lowpass filtered with a cut-off frequency of 0.1 Hz. The slow voltage shifting is removed and illustrated as the residual signal in the last sub-figure. The full bandpass filtering process is depicted in Figure 3.11. The removed signal is comprised of both the low voltage shifting and the high oscillations.

Although there is no firm consensus on the range of cut-off frequencies, the order, and the type of filter, their variabilities however have minimal and negligible effects [189]. In this thesis, we employ the 4th order Butterworth bandpass filter with cut-off frequencies of $[0.1, 20]$ Hz. The filter responses with different orders are plotted in Figure 3.12. Generally, a filter with higher order tends to maintain the frequencies surrounding the cut-off frequency.

3.4.4 Decimation

High-resolution and high-dimension signal may hamper the rate of subsequent processes, especially the classification stage. In this thesis, the BCI framework is designed to be implemented on-line, hence the amount of data to be processed even governs the performance speed with more influence. For instance, a 625-ms 64-channel ERP trial with the sampling rate of 240 Hz, when flattened (i.e., all the data points of different channels are concatenated consecutively), can have over 9500 data points. This high-dimensional sample

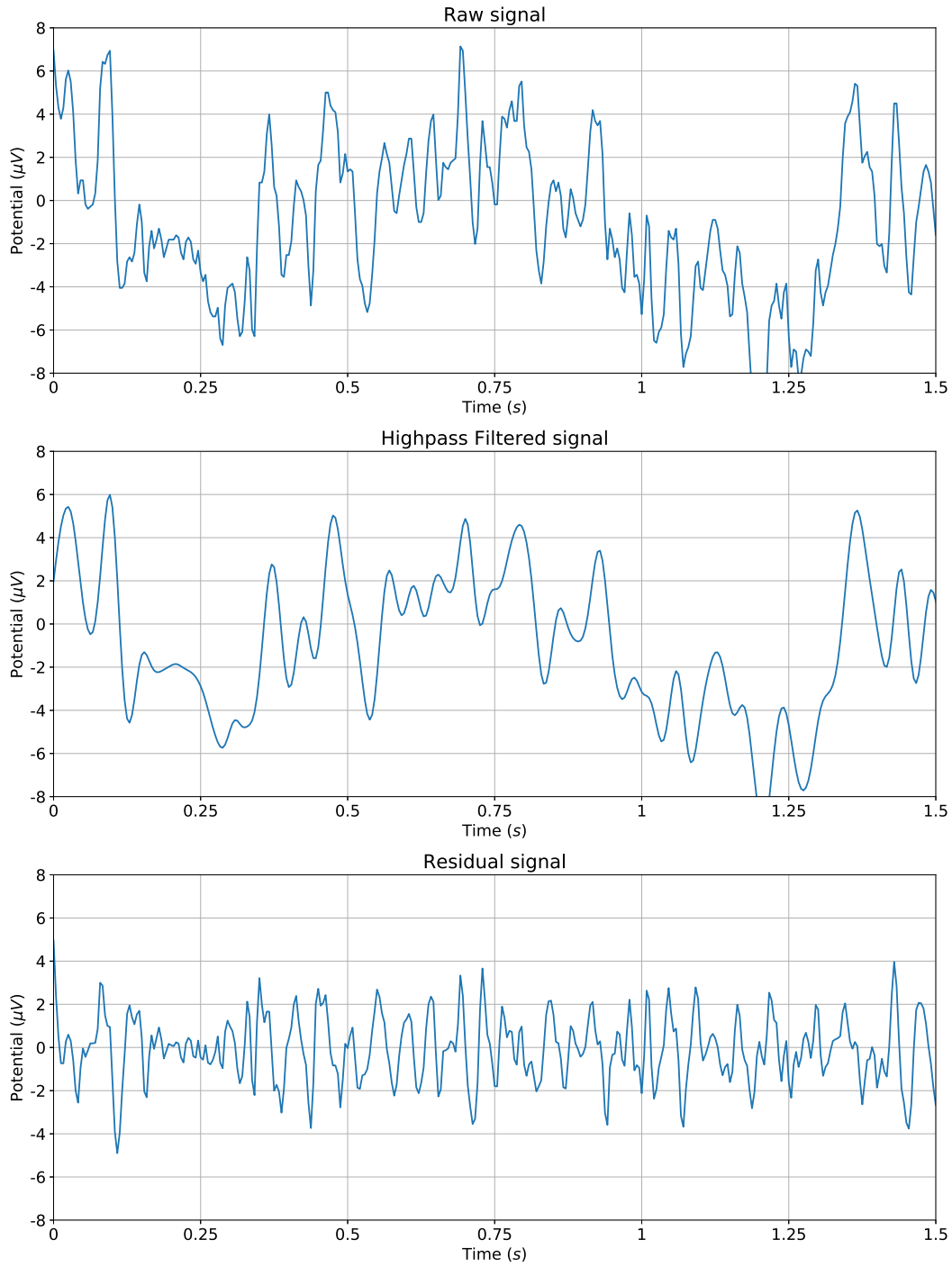


Figure 3.9: The illustration of highpass filtering with cut-off frequency of 20 Hz on an EEG signal segment of 1.5 seconds.

is definitely undesirable for any machine learning algorithms, especially our adaptive and ensemble framework in this thesis, which will be present in the next chapters.

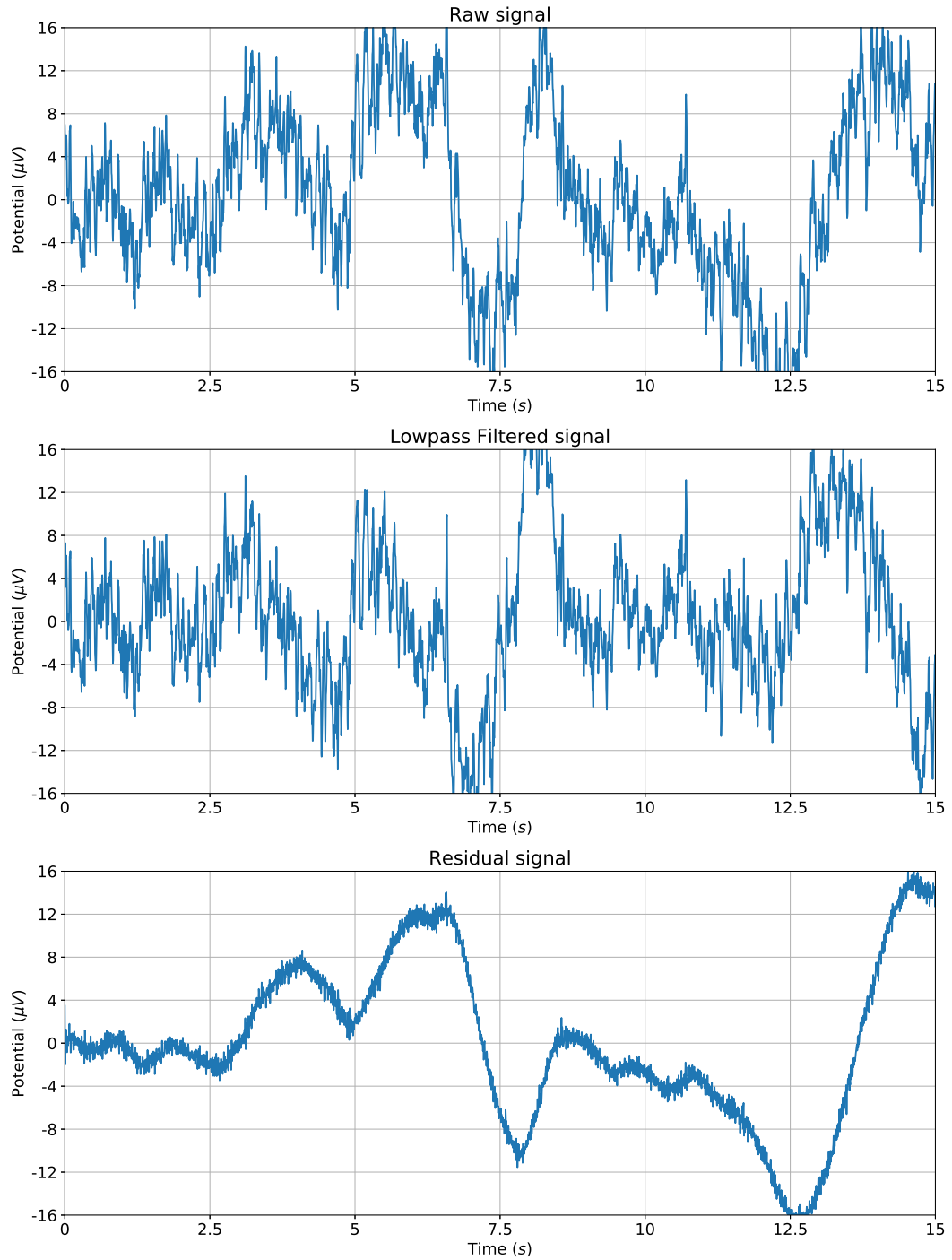


Figure 3.10: The illustration of lowpass filtering with cut-off frequency of 0.1 Hz on an EEG signal segment of 15 seconds.

We employ the simple decimation, or down sampling, method for the essential need of dimensional reduction for two reasons. First, decimation is extremely fast, as it works by only selecting the m -th data points from the original signal (with m as the decimation factor). Second, thanks to the low-

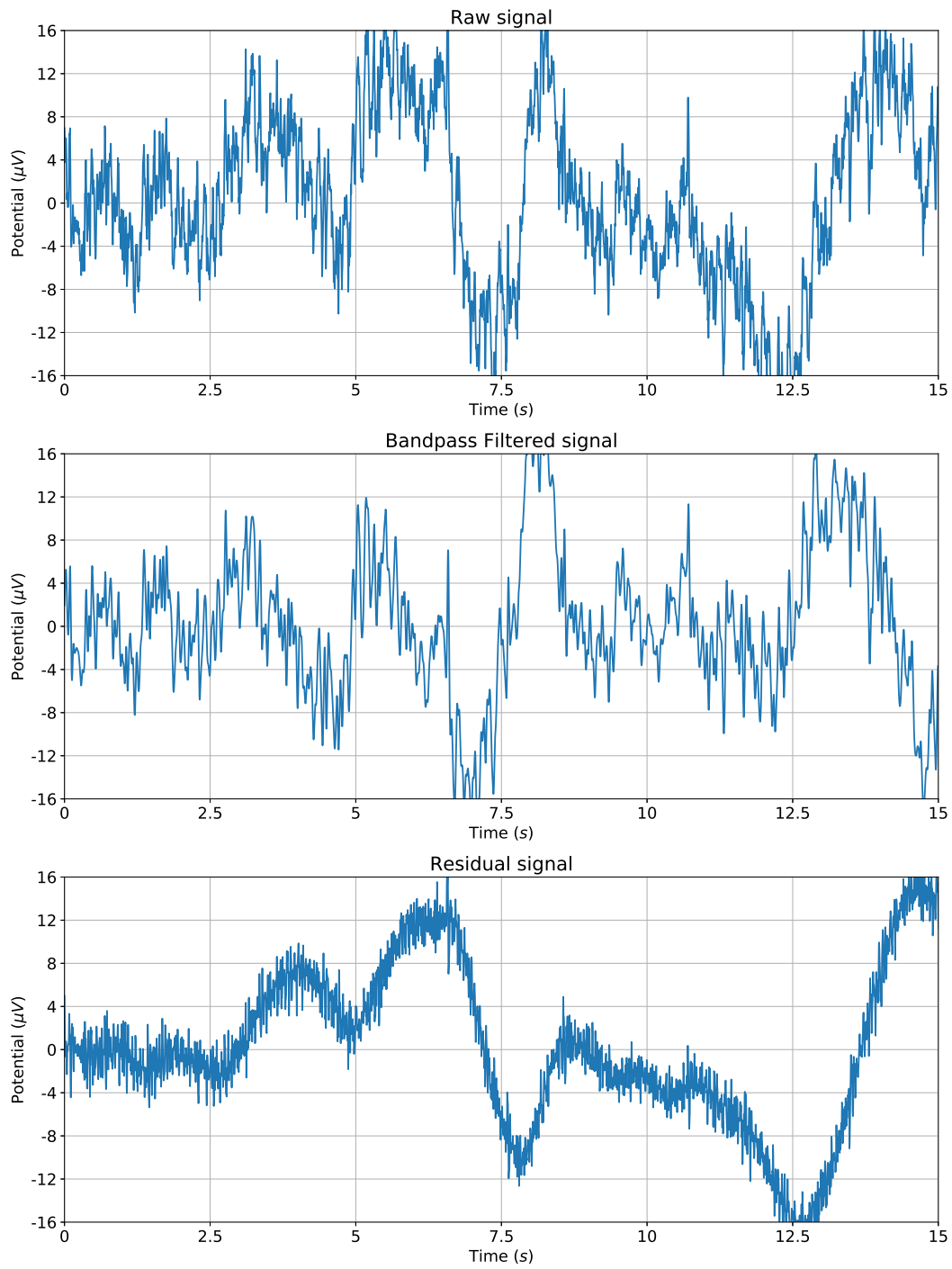


Figure 3.11: The illustration of bandpass filtering with cut-off frequencies of $[0.1, 20]$ Hz on an EEG signal segment of 15 seconds.

frequency characteristic of ERP components, decimation does not remove any useful featuring information, especially the P3.

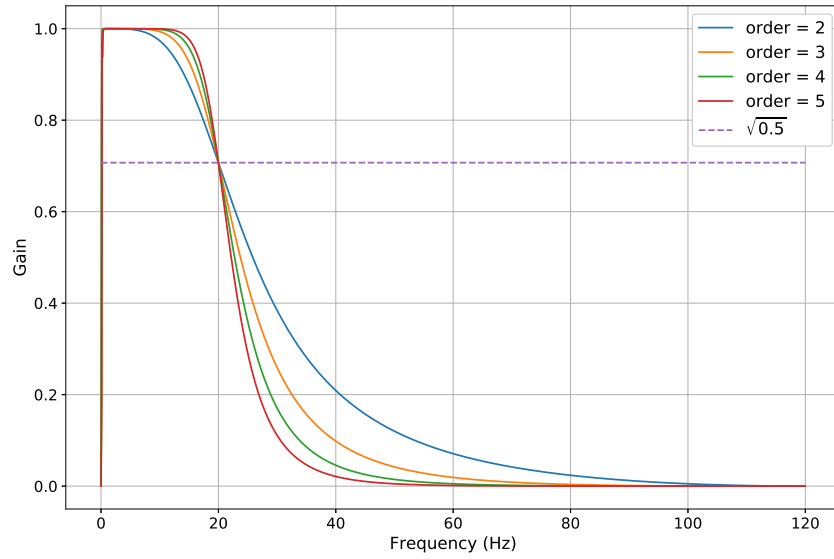


Figure 3.12: The filter response of the Butterworth bandpass filter with cut-off frequencies of $[0.1, 20]$ Hz.

3.4.5 Benefits of Preprocessing

We conduct a practical experiment to compare the accuracy performance benefit when employing artifact removal and filtering. The subjects recruited in our experiments and the employed flashing paradigm are described in details in section 3.5. In this experiment, three subjects' ERP trials were extracted from a 4-symbol P3S. Each subject conducted 30 spelling symbols. Each symbol requires 15 repetitions of 4 single flashes of each symbol on the screen. The inter-stimulus interval was set to 250ms, and each trial is 625ms long. As a result, for each spelling symbol of each subject, there are 60 trials (15 target trials and 45 non-target trials). We used two thirds of the collected data, i.e., 20 symbols per subject, to train an SVM classifier. The SVM classifier was trained by combining all data samples from all three subjects (we have not employed adaptive learning or trained subject-independently to this stage). The unique SVM solution was then used to evaluate the remaining 10 symbols per subject.

Two main evaluation metrics were used to measure the accuracy performance: the symbol-wise accuracy and trial-wise accuracy, as described in section 3.3. The accuracy comparison of each subject with/without artifact removal or filtering is presented in Figure 3.13 and 3.14.

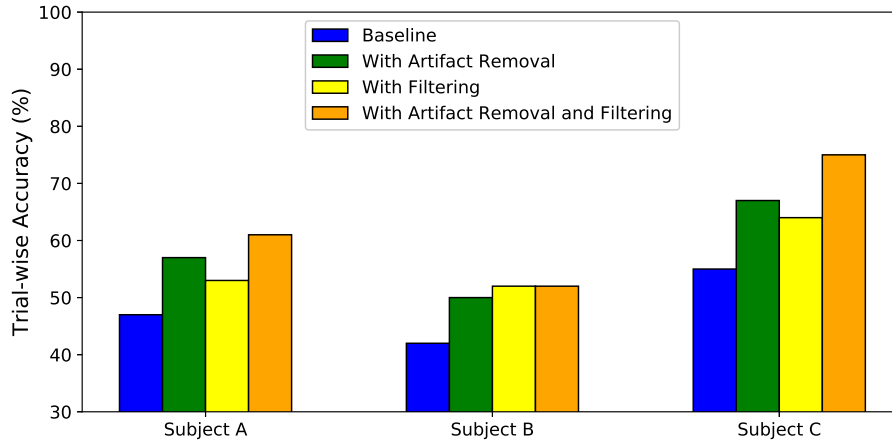


Figure 3.13: The trial-wise accuracy performance with and without preprocessing techniques

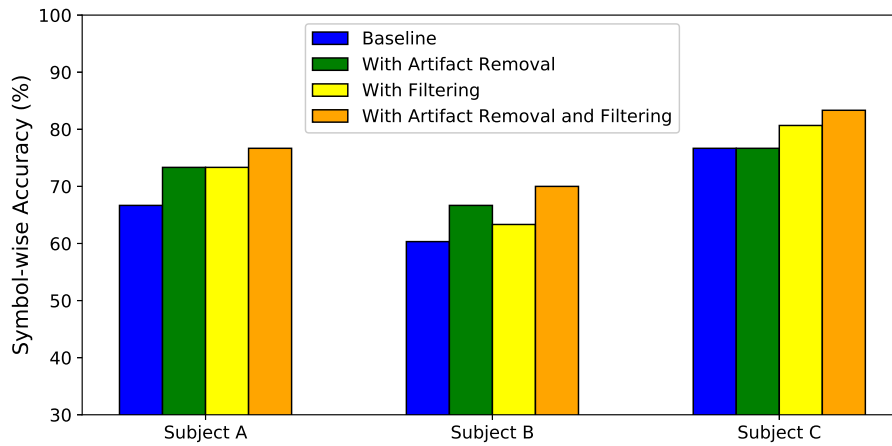


Figure 3.14: The symbol-wise accuracy performance with and without preprocessing techniques

3.4.6 Feature Construction and Analysis

For a given artifact-removed, filtered, and decimated trial, we obtain a digitized matrix representing that single trial, where each row corresponds to an electrode and each column corresponds to a timestamp. Since most machine learning algorithms deal with 1-D vectors as input samples, it is necessary to reshape the samples of matrices into samples of vectors. We do this by flattening the matrices row-wise, i.e., concatenating each row (channel) after another to form a new 1-D feature vector. For instance, after preprocessed, a

trial consists of 14 data points per each channel corresponding to 14 discrete timestamps in the interval from 0 to 625ms post-stimulus. This trial’s dimension is 14×10 . As a result, after being reshaped, the final 1-D feature vector representing this trial will have the dimension of 140.

3.5 Dataset Specifications

In this thesis, we employ two datasets for the subsequent step of classification and evaluation. The first dataset is the Akimpech [16], a public P3S dataset that is comprised of 25 different subjects. As compared to the other public P3S datasets, Akimpech contains a large number of subjects (26, as compared to 1 of [13], 2 of [14], or 9 of [15]). This advantage allows us to carry out our extensive numerical analysis with strong probabilistic aspect. Moreover, each subject in the Akimpech dataset also conducted a large number of experiments (27-39 letters). Each letter session is 15-iteration long. This dataset is named Dataset A as shown in Table 3.3. The second dataset, which is named Dataset B in this thesis, was collected from our three recruited subjects by the Emotiv EPOC EEG headset. We also have reported our ERP experiments on Dataset B in our past publications [43, 190].

In this thesis, we use Dataset A as the major dataset for the subject-independent analysis, which is the main goal of this thesis thanks to the large number of subjects. Dataset A will be used in the subsequent main chapters (Chapter 4, Chapter 5, and Chapter 6). Dataset B, on the other hand, is mainly used for subject-dependent experiments to evaluate preprocessing techniques and analyze ERP features presented in this chapter. Dataset B is also used to evaluate the effects of the ensemble strategy in Chapter 5.

The details of each dataset’s specifications is presented in Table 3.2. The information of every subject in both datasets is described in Table 3.3.

TABLE 3.2: DATASET SPECIFICATIONS

	Dataset A	Dataset B
Source	Akimpech [16]	Own-collected [43, 190]
No. Subjects	25 (aged 19-25)	3 (aged 25)
No. Channels	10	14
Channels	Fz, C4, Cz, C3, P4, Pz, P3, PO8, Pz, PO7	AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4
Device	gtec gUSBamp16	Emotiv EPOC
Sampling rate	256 Hz	128 Hz
ISI	125ms	250ms
Flash time	62.5ms	125ms
Fade time	62.5ms	125ms
P3S settings	6 rows, 6 columns (36 symbols)	2 rows, 2 columns (4 symbols)
Flashing Paradigm	Row/Column	Symbol
Flashes/session	12	4
Target Probability	1/6	1/4
Train subjects	10 (ACS, APM, ASG, ASR, CLL, DCM, DLP, DMA, ELC, FSZ)	3 (A, B, C)
Test subjects	15 (GCE, ICE, IZH, JLD, JLP, JMR, JSC, JST, LAC, LAG, LGP, PGA, WFG, XCL)	3 (A, B, C)
Evaluation Method	Subject-independent	Subject-dependent
Channels used	10 (Fz, C4, Cz, C3, P4, Pz, P3, PO8, Pz, PO7)	6 (F3, F4, FC5, FC6, P7, P8)
Features/channel	14	14
Features/trial	140	84

TABLE 3.3: SUBJECT INFORMATION OF DATASETS IN THIS THESIS

DATASET A						
Subject ID	Date Recorded	Age	Sex	Sleep (hours)	Medicines	Smoke (hours ago, cigars)
PGA	26/05/2009	22	M	8	No	No
CLL	09/06/2009	25	F	9	No	13, 5
JSC	09/06/2009	24	M	8	No	No
LGP	16/06/2009	22	M	7	No	No
DMA	23/06/2009	25	M	6	No	15, 1
GCE	30/06/2009	22	M	7	No	No
JLP	07/07/2009	21	M	7	No	24, 1
ACS	07/07/2009	23	F	7	No	No
ELC	14/07/2009	21	F	8	No	1, 1
ICE	14/07/2009	22	M	8	No	No
DLP	21/07/2009	23	F	7	No	No
JST	21/07/2009	23	M	7.5	No	No
FSZ	11/08/2009	21	F	9.5	No	72, 1
JMR	18/08/2009	23	M	5	No	No
LAG	10/09/2009	23	F	6	Propafenone	No
LAC	17/09/2009	23	F	8	No	No
JLD	23/09/2009	22	M	7	Afrinex	No
APM	24/09/2009	23	M	9	No	No
DCM	25/09/2009	22	M	12	No	No
XCL	30/09/2009	23	M	5	No	No
ASG	02/10/2009	23	F	2	No	No
ASR	06/10/2009	23	M	5	No	No
WFG	08/10/2009	21	M	8	No	No
IZH	13/10/2009	22	F	10	No	No
DATASET B						
Subject ID	Date Recorded DD/MM/YYYY	Age	Sex	Sleep (hours)	Medicines	Smoke (hours ago, cigars)
A	25/02/2014	25	M	5	No	No
B	25/02/2014	25	M	7	No	No
C	25/02/2014	25	F	8	No	No

3.6 Chapter Summary

In this chapter, we have outlined in details the following key sections:

- History, major components of ERPs and the factors influencing their amplitude and latency.
- The data acquisition framework, including how to collect EEG signal using the P3S, two designs of P3S used in this thesis (row/column flash 6-by-6 and symbol flash 2-by-2), and the main P3S evaluation metrics.
- EEG signal preprocessing techniques, including averaging, artifact removal, filtering, decimation, and feature vector construction. Pilot experiments were conducted to demonstrate the benefits of preprocessing techniques.
- Finally, the conditions of subjects prior to experiments, and the specifications of two datasets used in this thesis are also described.

As presented in this chapter, various factors can influence the averaged pattern of target or non-target trials. The patterns of those two classes therefore, greatly vary across multiple subjects. The classifier obtained by crudely learning from a given dataset, as a result, cannot generalize on unseen subjects. We address this problem by developing an adaptive learning mechanism, specifically the adversarial neural network, which will be mentioned in the next chapter.

Chapter 4

SUBJECT-ADVERSARIAL NEURAL NETWORK

In this chapter, an adversarial learning strategy is employed on the preprocessed feature vectors that were produced as outputs from the last chapter, namely the subject-adversarial neural network. First, section 4.1 outlines the issues related to subject-specific ERPs and the motivation of developing an adaptive learning strategy. As a result, section 4.2 reviews the most two widely-used adversarial neural networks: the generative adversarial neural network, and the domain-adversarial neural network. Next, we propose our developed adversarial training which can be employed on multi-subject datasets, namely the subject-adversarial neural network in section 4.3, followed by a simple case study in section 4.4 and the practical experiments on BCI P3 data in section 4.5. Finally, a summary of this chapter is presented in section 4.6.

4.1 Motivation

As described in Chapter 3, ERP patterns significantly vary across different subjects. There are two negative consequences due to this issue with regards to the context of subject-independent BCI. First, any machine learning algorithm that is trained on a provided dataset from past subjects may not generalize on future subjects. Second, the diversity in the distributions of subjects' data samples will surely decrease the discrimination capability of learning algorithms.

The idea of adaptive learning in this thesis is presented as follows. Since

the ERP patterns vary across multiple training subjects, we assume that with a given multi-subject dataset which is subject-statistically-sufficient, an optimal learning algorithm can generalize on future subjects. This assumption is equivalent to saying that we can effectively classify the patterns of a specific future subject if the learning algorithm can:

- Discriminate the subject distributions of training data,
- Classify the patterns of any specific training subject.

Unfortunately due to the large number of subjects, the task of discriminate the distributions of different subjects together with the task of discriminate the patterns in each distribution is infeasible [25], especially for the real-time context of our BCI framework. Therefore, we would need to reformulate the original two required properties into

- Transform the distributions of training subjects into a mixed distribution with maximum representative capability,
- Classify the patterns on the transformed distribution, regardless of subject identity (or subject-independent).

This reformulation relaxes those two original requirements of the learning algorithm. In this thesis, those two relaxed requirements are achieved simultaneously by training a specific neural network architecture, known as the *subject-adversarial neural network* (SANN), a slightly modified version of the domain-adversarial neural network (DANN) [25].

In the subsequent sections, we will present all aspects of adversarial NN, and its application in the BCI context of this thesis. First, section 4.2 reviews the state-of-the-art literature of adversarial, including the two major types: the generative adversarial neural network (GANN), and the domain adversarial neural network (DANN), and the contexts in which they were motivated. We also provided a simple case study of DANN in section 4.4. Next, section 4.3 describes the mathematical theory of SANN, which is a modified version of DANN that is used in this thesis, and its application in this thesis's context. Section 4.5 presents the experimental settings and results of SANN on the P3S

framework. Finally, section 4.6 draws some remarkable conclusions of SANN and summarize the content of this chapter.

4.2 Adversarial Neural Network Review

4.2.1 Generative Adversarial Neural Network (GANN)

Generative adversarial neural network (GANN) is a special architecture of neural network (NN) that allows one to learn deep representations with limited labeled data. GANN was first discovered and developed by Goodfellow, a Google machine learning expert, and his colleagues in 2014 [26]. GANN emerged as a major breakthrough in representative learning at the time and heated up the whole machine learning community by its innovative and smart reasoning, implemented by the NN architecture. The main goal of GANN is to model high-dimensional distributions of data, including both training data and test data. In the original work, GANN was employed in a computer vision application where data images can be retrieved from multiple noisy sources. More specifically, let us imagine the task of digit recognition from various background sources, i.e., background address banners stick on houses. The images of address numbers may have differing fonts, formats, background colors, or background patterns. These factors can be considered as noise and heavily affect the digit recognition performance.

To tackle this problem, GANN tries to generate *synthetic*, or artificial, images to diversify the training data. At the first glance, this job sounds improbable to attain, as one can argue that how can we generate training images out of nothing, or how can we generate images with different distributions as compared to real images. Indeed, GANN does actually generate images from nothing, and the synthetic images are representative different than the real images, hence can boost up the classification performance. The term *nothing* here, in fact, refers to a latent space induced by the adversarial learning procedure. More specifically, GANN uses two networks with opposing training objectives. The first network the *generator*, which generates the synthetic images from a source of noise and the latent space defined by the adversarial learning process. This process will be described in details in the subsequent

sections. The second network is the *discriminator*, which has access to both real and synthetic images, tries to predict the genuineness of each input image. Both networks are trained simultaneously and adversarial, i.e., in competition, with each other.

As described, a crucial aspect of the generator is its inaccessibility to real images. It generates the synthetic images only via its interaction with the discriminator. The discriminator, which has access to both image domains (synthetic and real), is trained to predict the domain of the input image, i.e., predict if the image is fake or real. In other words, the discriminator tries to minimize the domain classification error. On the other hand, the generator is trained to maximize this same error. Intuitively, the generator's goal is to generate fake images that make the discriminator as much uncertain about its decision as possible. This leads to the created fake images that look almost real, and can be used together with real data by any machine learning algorithms to classify the label classes, i.e., the 10 numerical digits in the mentioned example. The general framework of GANN is summarized in Figure 4.1.

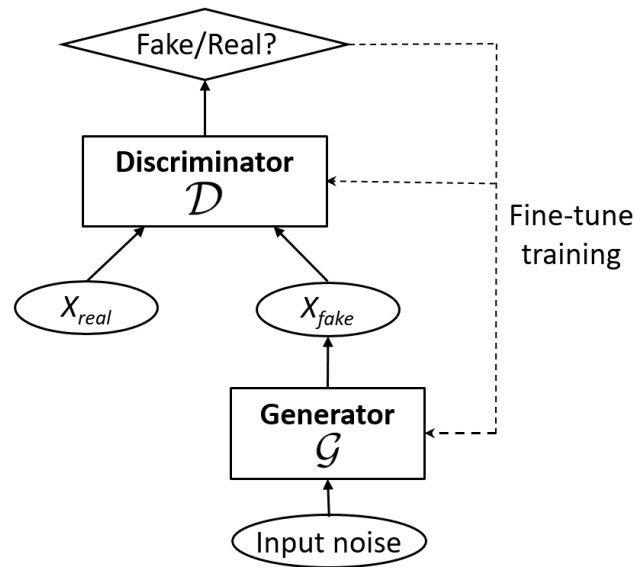


Figure 4.1: The general workflow of GANN.

4.2.2 Applications of GANN

Since its advent, GANN had attracted great interest and is still an active research topic for new applications. GANN is used most in computer vision, more specifically in quantitative assessment of extracted features in unsupervised learning. The training of GANN, after completed, can be reused to extract representative features, with a simple linear regression model on top of these features [191, 192]. The quality of these representations of GANN using deep convolutional NN (or referred to as DCGANN), can be assessed by a regularized L2-SVM classifier. This SVM classifier is applied to an extracted feature vector from the trained discriminator [191]. Some authors also tried to improve the quality of synthetic data by integrating an inference strategy, known as adversarially learned inference (ALI), and obtained higher classification accuracy than DCGANN [193]. GANN was also employed in pose and gaze recognition in [194, 195]. In [194], the authors synthesized images from real data, then used only those fake data to train models (without real data) and achieved competitive results in pose and gaze recognition. Also, in [195] the authors by using a spatio-temporal GANN had obtained state-of-the-art results on gaze prediction tasks. More interestingly, GANN was developed in some image-to-image translation tasks, i.e., converting landscape images from day to night, colorizing gray-scale images, or extracting edges of objects [27]. GANN was also used in super-resolution, i.e., upsampling low-resolution images in [28], synthesize images from textual descriptions [29, 30], making image editing tools [31–33], i.e., editing human hairstyles or making people look younger.

4.2.3 Domain-Adversarial Neural Network (DANN)

In 2015, Ganin and his colleagues developed the domain-adversarial neural network (DANN) [25], which is heavily based on the adversarial training strategy originally proposed in GANN. However, the context motivation and problem formulation of DANN is different, and it excited a large number of subsequent applications after its advent.

As opposed to GANN which tries to synthesize more artificial data to en-

hance the learning process, DANN argues that big training sets, that might be obtained one way or another (i.e., all real or the combination of real and fake), have a high chance to suffer from *shifts* in data distributions. These shifts were usually encountered in the discrepancies of training and test data. In [196–198], the authors demonstrated that although synthetic data can provide abundant labeled training data, the distribution of data in the test time (real images) is significantly different. The motivation of DANN is that it builds a mapping between the training data and the test data, so that a machine learning rule learned from the training data can be applied to the test data in the presence of original distribution shifts.

DANN is also different from GANN in its unified learning framework. More specifically, DANN integrates the tasks of adapting domains and classifying classes into one learning framework. In other words, DANN offers a feature representative space that is discriminative in class distributions and *fuzzy*, i.e., invariant, in domain distributions [25]. This is done by designing a complex neural network with multiple parties, i.e., branches. Each party has its own objective and loss functions, and can be trained simultaneously with other parties in a unified architecture. The general flowchart of DANN is illustrated in Figure 4.2. DANN is comprised of three NN branches, or three parties: the feature encoder, the domain classifier, and the label predictor. The feature encoder learns a feature mapping from the original input space to a latent space. This latent space provides discriminative features to the label predictor and let the label predictor well predict the classes. At the same time, this latent space also provides fuzzy features to the domain classifier and make the domain classifier uncertain about its decision whether the input sample comes from train or test distribution. The core idea of the training strategy is to simultaneously minimize the class prediction errors and maximize domain prediction errors. Since a typical NN architecture can only minimize or maximize one loss function at a time, stacking multiple NN branches allows us for this multiple objective training. During the back-propagation procedure of NN training, it is crucial to inverse the sign of error of the domain classifier’s gradient. This technique will be mentioned in more details in the next section.

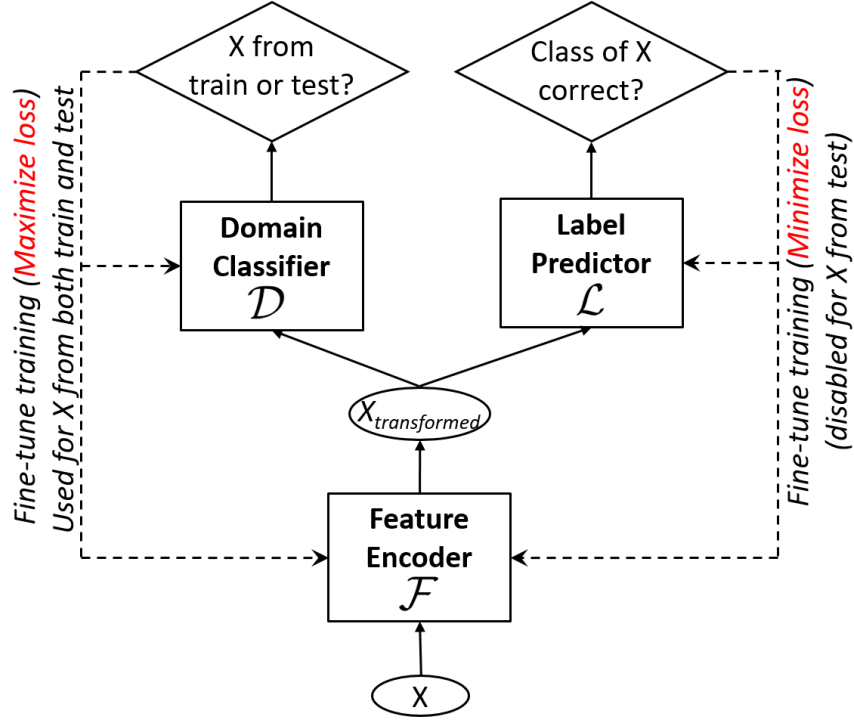


Figure 4.2: The general workflow of DANN.

4.3 Subject-Adversarial Neural Network (SANN)

4.3.1 Context Motivation

In this thesis, we employ the subject adversarial neural network (SANN), which is a modified version of DANN. DANN attempts to find a feature encoder that produces representative features for the label predictor, but fuzzy features for the domain classifier. In the context of our real-time subject-independent P3S framework, the task of learning off-line using both train and test samples cannot be attained. We would need to redefine the objective of the domain classifier of DANN into a slightly different objective. In SANN, we do not use the domain classifier to classify the domain (i.e., train or test) of a sample, but to classify the identity of the subject (i.e., who produces the sample). The domain classifier architecture of SANN therefore, have larger output domain labels corresponding to the number of training subjects. Also, SANN does not use test data samples to enhance the domain adaptation procedure. In the BCI context in this thesis, the differences of SANN as compared to DANN will benefit the BCI framework in two ways. First, we can exclude the test data

from the training process, which will hamper the real-time speed of the BCI which requires fast response. Second, the multi-label output of the domain classifier branch can enhance the subject-independent requirement.

The development of SANN is motivated by two major ideas. First, one needs to possess a sufficiently diverse multi-subject training data which can generalize all ERP patterns. Second, the learning algorithm has the capability not only to identify the domain of each distribution (i.e., which subject does the samples come from), but also to well separate the ERP classes. Within these two assumptions, we can attain the subject-independent characteristic of the BCI.

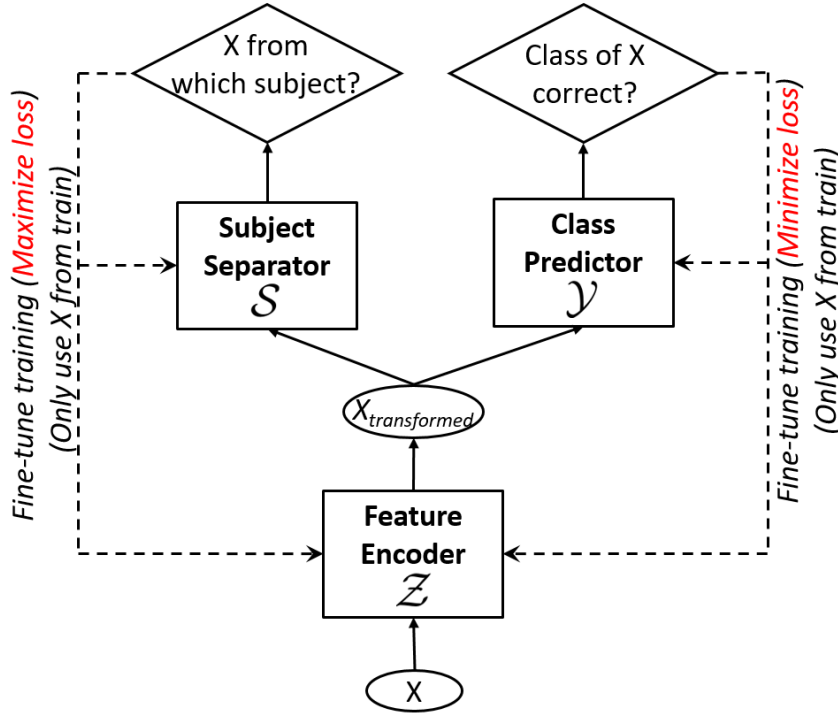


Figure 4.3: The general workflow of SANN.

4.3.2 Theory of SANN

In this section we present the theory of SANN by mainly using notations and equations in the field of statistical learning theory. This section hence serves as a theoretical foundation that supports the innovative aspects of our

proposed SANN method.

In SANN theory, we change some terms that were often used in DANN [25]. First, we use the term *subject separator* ($\mathcal{S}$) as a replacement for the domain classifier ($\mathcal{D}$), since we refer the domain labels as the subject identities. The reason behind changing the notation $\mathcal{D}$ to $\mathcal{Y}$ is to match with $\mathbf{y}$, the notation we use to indicate the class labels. Second, we use the term *class predictor* ($\mathcal{Y}$) as the replacement for the label predictor, since the term *label* is confused as it can refer to the domain label or the class label. The feature encoder ($\mathcal{Z}$) is kept, although the notation is changed from $\mathcal{F}$ to $\mathcal{Z}$ to match with $\mathbf{z}$, the notation we use that refers to the transformed feature vector. The general data workflow of SANN is depicted in Figure 4.3.

Given a training set, denoted as $\mathcal{L}$, which is composed of the samples from D domains (or subjects)

$$\mathcal{L} = \{\mathbf{x}_i, y_i, s_i\}_{i=1}^M, \quad (4.1)$$

where $\mathbf{x}_d^{(i)} \in \mathcal{R}^N$, $\mathbf{y}_d^{(i)} \in [0, 1]^2$, and the notation $|S_d|$ represents the size (i.e., number of samples) of the set S_d . In the context in this thesis, each $\mathbf{x}_d^{(i)}$ is a preprocessed N -dimensional feature vector representing one ERP trial of subject d , while $\mathbf{y}_d^{(i)}$ is the corresponding one-hot encoded label indicating that if $\mathbf{x}_d^{(i)}$ is the target (i.e., $\mathbf{y}_d^{(i)} = [1, 0]$) or the non-target (i.e., $\mathbf{y}_d^{(i)} = [0, 1]$).

For subject-adversarial training, we need to learn a latent representation for $\mathbf{x}$, that cannot only be used to predict $\mathbf{y}$ well, but also invariant to the source domains S_d (hereinafter for notational convenience, $\mathbf{x}$ and $\mathbf{y}$ refer to $\mathbf{x}_d^{(i)}$ and $\mathbf{y}_d^{(i)}$ respectively). The feature encoder is learned in the way that the encoded feature vector is predictive for the conventional label predictor, but detrimental for the domain discriminator. Hence, our framework consists of three different parties which are coupled together in the unified training framework.

Feature Encoder $\mathcal{Z}$:

The feature encoder, denoted as $\mathcal{Z} : \mathcal{R}^N \rightarrow \mathcal{R}^M$, transforms an N -dimensional

$\mathbf{x}$ to an M -dimensional $\mathbf{z}$

$$\mathcal{Z}(\mathbf{x}) = \mathbf{z}. \quad (4.2)$$

Class Predictor $\mathcal{Y}$:

The class predictor, denoted as $\mathcal{Y} : \Omega_z \rightarrow [0, 1]^l$, takes as inputs the encoded feature vector $\mathbf{z}$, and outputs the probabilities of the input encoded sample $\mathbf{z}$ being classified into each class $y \in \Omega_y$ as $Q_{\mathcal{Y}}(y|\mathbf{z})$. In other words, $Q_{\mathcal{Y}}(y_j|\mathbf{z})$ equates the probability of $\mathbf{z}$ being classified into class label $y_j \in \Omega_y$. As a result, we denote $Q_{\mathcal{Y}}(\cdot|\mathbf{z})$ as the estimated probability distribution produced by $\mathcal{Y}$, that is

$$\mathcal{Y}(\mathbf{z}) \sim Q_{\mathcal{Y}}(\cdot|\mathbf{z}). \quad (4.3)$$

Subject Separator $\mathcal{S}$:

The subject (or domain) separator, denoted as $\mathcal{S} : \Omega_z \rightarrow [0, 1]^D$, takes as inputs the encoded feature vector $\mathbf{z} = \mathcal{Z}(\mathbf{x})$ and the true class label distribution $P_y(\cdot|\mathbf{x})$, then outputs the probabilities of the input encoded sample $\mathbf{z}$ being separated into each subject identity $s \in \Omega_s$ as $Q_{\mathcal{S}}(s|\mathbf{z}, P_y(\cdot|\mathbf{x}))$

$$\mathcal{S}(\mathbf{z}, P_y(\cdot|\mathbf{x})) \sim Q_{\mathcal{S}}(\cdot|\mathbf{z}, P_y(\cdot|\mathbf{x})). \quad (4.4)$$

Next, we derive the *class loss* $L_{\mathcal{Y}}$, and *subject loss* $L_{\mathcal{S}}$ by

$$\begin{aligned} L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}) &= \mathbb{E}_{\mathbf{x}, y} \left[-\log Q_{\mathcal{Y}}(y|\mathcal{Z}(\mathbf{x})) \right], \\ L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}) &= \mathbb{E}_{\mathbf{x}, s} \left[-\log Q_{\mathcal{S}}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \right]. \end{aligned} \quad (4.5)$$

In words, the class loss $L_{\mathcal{Y}}$ (or subject loss $L_{\mathcal{S}}$) is the mean of individual losses of all the training samples $\mathbf{x}$, where each individual loss equates the negative logarithmic probability of the corresponding correct class label y (or subject label s). The complete minimization objective function L is followed by

$$L(\mathcal{Z}, \mathcal{Y}, \mathcal{S}) = L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}) - \lambda L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}), \quad (4.6)$$

where $\lambda > 0$ is the regularizer coefficient representing the strength of the subject separator. With $\lambda = 0$, $\mathcal{S}$ is neglected during training, leaving $\mathcal{Z}$ and $\mathcal{Y}$ to freely search for their optimal parameters which can predict y well. In this case, the roles of $\mathcal{Z}$ and $\mathcal{Y}$ are not discriminative, hence the training framework shrinks to a normal fully-connected neural network with two hidden layers. With $\lambda > 0$, $\mathcal{Z}$, $\mathcal{Y}$, and $\mathcal{S}$ are jointly optimized with these following

properties:

- $\mathcal{S}$ attempts to correctly predict the subject domain s of $\mathbf{x}$, given $\mathcal{Z}(\mathbf{x})$ and the optimal $\mathcal{Z}^*, \mathcal{Y}^*$.

$$\mathcal{S}^* = \operatorname{argmin}_{\mathcal{S}} L_{\mathcal{S}}(\mathcal{Z}^*, \mathcal{S}) = \operatorname{argmax}_{\mathcal{S}} L(\mathcal{Z}^*, \mathcal{Y}^*, \mathcal{S}). \quad (4.7)$$

- $\mathcal{Y}$ attempts to predict the true class label y of $\mathbf{x}$, given $\mathcal{Z}(\mathbf{x})$, the optimal feature encoder $\mathcal{Z}^*$, and the optimal subject separator $\mathcal{S}^*$

$$\mathcal{Y}^* = \operatorname{argmin}_{\mathcal{Y}} L_{\mathcal{Y}}(\mathcal{Z}^*, \mathcal{Y}) = \operatorname{argmin}_{\mathcal{Y}} L(\mathcal{Z}^*, \mathcal{Y}, \mathcal{S}^*). \quad (4.8)$$

- $\mathcal{Z}$ attempts to build a feature representation which is predictive for $\mathcal{Y}$ but detrimental for $\mathcal{S}$. This is done by minimizing $L_{\mathcal{Y}}$ and maximizing $L_{\mathcal{S}}$ simultaneously, equivalent to minimizing the total loss L

$$\mathcal{Z}^* = \operatorname{argmin}_{\mathcal{Z}} L(\mathcal{Z}, \mathcal{Y}^*, \mathcal{S}^*). \quad (4.9)$$

By merging the objectives in (4.7), (4.8), (4.9), the training process therefore can be viewed as a three-party optimization problem of $\mathcal{Z}, \mathcal{Y}$, and $\mathcal{S}$. The joint optimal solution $\mathcal{Z}^*, \mathcal{Y}^*$, and $\mathcal{S}^*$ is the one that solves the following optimization problem

$$\min_{\mathcal{Z}} \min_{\mathcal{Y}} \max_{\mathcal{S}} L(\mathcal{Z}, \mathcal{Y}, \mathcal{S}). \quad (4.10)$$

4.3.3 Theory Proofs

In this section, we provide a proof demonstrating that the optimal solution $\mathcal{Z}^*, \mathcal{Y}^*, \mathcal{S}^*$ to the optimization problem (4.10) have these following two desirable properties.

- The subject separator $\mathcal{S}^*$ does not depend on the feature encoder $\mathcal{Z}^*$, that is, $Q_{\mathcal{S}^*}(\cdot | \mathcal{Z}^*(\mathbf{x}), P_y(\cdot | \mathbf{x})) = Q_{\mathcal{S}^*}(\cdot | P_y(\cdot | \mathbf{x}))$. This is equivalent to saying that $\mathcal{Z}^*$ provides an invariant representation with respect to the subject labels s .

- Second, the participation of $\mathcal{S}^*$ does not decrease the best achievable performance of $\mathcal{Y}^*$, that is, $Q_{\mathcal{Y}^*}(\cdot|\mathcal{Z}^*(\mathbf{x})) = Q_{\mathcal{Y}^*}(\cdot|\mathbf{x})$.

Lemma 4.1. Given an encoder $\mathcal{Z}$, the lower bound for the subject loss $L_{\mathcal{S}}$ is the entropy of the subject labels s , conditioned on the encoder's output and the ground-truth class probabilities, that is

$$L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}^*) = \min_{\mathcal{S}} L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}) = \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})). \quad (4.11)$$

Proof. By deriving the KL divergence, from the estimated subject separator $Q_{\mathcal{S}}(\cdot|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x}))$, to its ground-truth distribution $P_s(\cdot|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x}))$, we have

$$\begin{aligned} D_{\text{KL}}(P_s \| Q_{\mathcal{S}}) &= \mathbb{H}(P_s, Q_{\mathcal{S}}) - \mathbb{H}(P_s) \\ &= \mathbb{E}_{\mathbf{x}, s} \left[-\log Q_{\mathcal{S}}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \right] - \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \\ &= L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}) - \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \geq 0. \end{aligned} \quad (4.12)$$

Hence, $L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}) \geq \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x}))$. The equality holds at $\mathcal{S} = \mathcal{S}^*$ if and only if $D_{\text{KL}}(P_s \| Q_{\mathcal{S}^*}) = 0$, or equivalently

$$Q_{\mathcal{S}^*}(\cdot|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) = P_s(\cdot|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \quad (4.13)$$

for all $\mathbf{x}$ satisfying $P_x(\mathbf{x}) \neq 0$. ■

We can also achieve the lower bound for the class loss $L_{\mathcal{Y}}$ as follows.

Lemma 4.2. Given an encoder $\mathcal{Z}$, the lower bound for the class loss $L_{\mathcal{Y}}$ is the entropy of the class labels y , conditioned on the encoder's output, that is

$$L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}^*) = \min_{\mathcal{Y}} L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}) = \mathbb{H}(y|\mathcal{Z}(\mathbf{x})). \quad (4.14)$$

The equality holds at $\mathcal{Y} = \mathcal{Y}^*$ if and only if $Q_{\mathcal{Y}^*}(\cdot|\mathcal{Z}(\mathbf{x})) = P_y(\cdot|\mathcal{Z}(\mathbf{x}))$ for all $\mathbf{x}$ satisfying $P_x(\mathbf{x}) \neq 0$.

Proof. Similar to the proof of Lemma 1. ■

By using Lemma 4.1 and 4.2, the following theorem arrives.

Theorem 4.3. When $\mathcal{Z}$, $\mathcal{Y}$, and $\mathcal{S}$ all attain their optimal solution, $\mathcal{Z}^*$, $\mathcal{Y}^*$, and $\mathcal{S}^*$, the following equalities hold:

$$\begin{aligned} Q_{\mathcal{S}^*}(\cdot|\mathcal{Z}^*(\mathbf{x}), P_y(\cdot|\mathbf{x})) &= Q_{\mathcal{S}^*}(\cdot|P_y(\cdot|\mathbf{x})), \\ Q_{\mathcal{Y}^*}(\cdot|\mathcal{Z}^*(\mathbf{x})) &= Q_{\mathcal{Y}^*}(\cdot|\mathbf{x}). \end{aligned} \quad (4.15)$$

Proof. Assuming that the architecture of $\mathcal{Z}$ and $\mathcal{Y}$ provides enough capacity to attain (4.11) and (4.14) respectively, the optimization problem (4.10) can be restated as

$$\begin{aligned} L^* &= \min_{\mathcal{Z}} \min_{\mathcal{Y}} \max_{\mathcal{S}} L(\mathcal{Z}, \mathcal{Y}, \mathcal{S}) \\ &= \min_{\mathcal{Z}} \min_{\mathcal{Y}} \max_{\mathcal{S}} \left(L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}) - \lambda L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}) \right) \\ &= \min_{\mathcal{Z}} \min_{\mathcal{Y}} \left(L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}) - \lambda L_{\mathcal{S}}(\mathcal{Z}, \mathcal{S}^*) \right) \\ &= \min_{\mathcal{Z}} \min_{\mathcal{Y}} \left(L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}) - \lambda \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \right) \\ &= \min_{\mathcal{Z}} \left(L_{\mathcal{Y}}(\mathcal{Z}, \mathcal{Y}^*) - \lambda \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \right) \\ &= \min_{\mathcal{Z}} \left(\mathbb{H}(y|\mathcal{Z}(\mathbf{x})) - \lambda \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \right). \end{aligned} \quad (4.16)$$

On the one hand, since $\mathcal{Z}(\mathbf{x})$ is a function of $\mathbf{x}$, then $\mathcal{Z}(\mathbf{x})$ and y are conditionally independent given $\mathbf{x}$, i.e., $y \rightarrow \mathbf{x} \rightarrow \mathcal{Z}(\mathbf{x})$ forms a Markov chain. By using the data processing inequalities (see Appendix A), there holds

$$\begin{aligned} 0 &\leq \mathbb{I}(y; \mathbf{x}) - \mathbb{I}(y; \mathcal{Z}(\mathbf{x})) \\ &= \left(\mathbb{H}(y) - \mathbb{H}(y|\mathbf{x}) \right) - \left(\mathbb{H}(y) - \mathbb{H}(y|\mathcal{Z}(\mathbf{x})) \right) \\ &= \mathbb{H}(y|\mathcal{Z}(\mathbf{x})) - \mathbb{H}(y|\mathbf{x}), \end{aligned} \quad (4.17)$$

which leads to

$$\mathbb{H}(y|\mathcal{Z}(\mathbf{x})) \geq \mathbb{H}(y|\mathbf{x}). \quad (4.18)$$

On the other hand, since conditioning reduces entropy (see Appendix A), it holds

$$\mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) \leq \mathbb{H}(s|P_y(\cdot|\mathbf{x})). \quad (4.19)$$

As a result, combining equations (4.16), (4.18), and (4.19) yields

$$L^* \geq \mathbb{H}(y|\mathbf{x}) - \lambda \mathbb{H}(s|P_y(\cdot|\mathbf{x})), \quad (4.20)$$

where the equality holds when (4.18) and (4.19) simultaneously hold, i.e.,

$$\begin{aligned} \mathbb{H}(y|\mathcal{Z}(\mathbf{x})) &= \mathbb{H}(y|\mathbf{x}), \\ \mathbb{H}(s|\mathcal{Z}(\mathbf{x}), P_y(\cdot|\mathbf{x})) &= \mathbb{H}(s|P_y(\cdot|\mathbf{x})). \end{aligned} \quad (4.21)$$

The two equalities can be simultaneously reached when the optimal encoder reaches the best theoretical estimator

$$\mathcal{Z}^*(\mathbf{x}) = P_y(\cdot|\mathbf{x}) \quad (4.22)$$

By employing Theorem A.6 (see Appendix A) on (4.21), we obtain

$$\begin{aligned} Q_{\mathcal{Y}^*}(\cdot|\mathcal{Z}^*(\mathbf{x})) &= Q_{\mathcal{Y}^*}(\cdot|\mathbf{x}), \\ Q_{\mathcal{S}^*}(\cdot|\mathcal{Z}^*(\mathbf{x}), P_y(\cdot|\mathbf{x})) &= Q_{\mathcal{S}^*}(\cdot|P_y(\cdot|\mathbf{x})). \blacksquare \end{aligned} \quad (4.23)$$

4.3.4 SANN Algorithm

In this section, the detailed neural network architecture of each SANN party, and various aspects of SANN training will be described. First, the network architecture of each party is parametrized to be the fully-connected feedforward neural network (as described in section 5.4.2) applied on the input flattened feature vector of size 140×1 (as described in section 3.4.6). The architecture of the feature encoder $\mathcal{Z}$ consists of one hidden layer of 64 ReLu units. The output size of $\mathcal{Z}$ is set to be identical to the input size (140). The class predictor $\mathcal{Y}$ consists of one hidden layers of 64 ReLu units and take the outputs of $\mathcal{Z}$ as its input. The output size of $\mathcal{Y}$ is 2 since we are attempting to perform the binary classification task. Finally, the subject separator $\mathcal{S}$ also consists of one hidden layer of 64 ReLu units, and takes as input the outputs of both $\mathcal{Z}$ and $\mathcal{Y}$. Hence, the input size of $\mathcal{S}$ is 142, and the output size of $\mathcal{S}$ is the total number of training subjects of Dataset A as described in Table 3.2, that is 10. The full architecture of the SANN as described is depicted in Figure 4.4.

In SANN training, $\mathcal{Y}$ and $\mathcal{S}$ perform opposite signs of gradient back-propagation [199]. As a result, the back-propagation flow from the output of $\mathcal{Y}$ to the input of $\mathcal{Z}$ is kept original, while the back-propagation flow from the output of $\mathcal{S}$ to the input of $\mathcal{Z}$ must be sign-reversed at the output of $\mathcal{Z}$. This is done by a *gradient reversal layer* which is inserted at the output of $\mathcal{Z}$ exclusively for back-propagation. In other words, the gradient reversal layer is an extra layer that has no parameter, which perform identity transformation during feedforward and sign reversal during back-propagation [25].

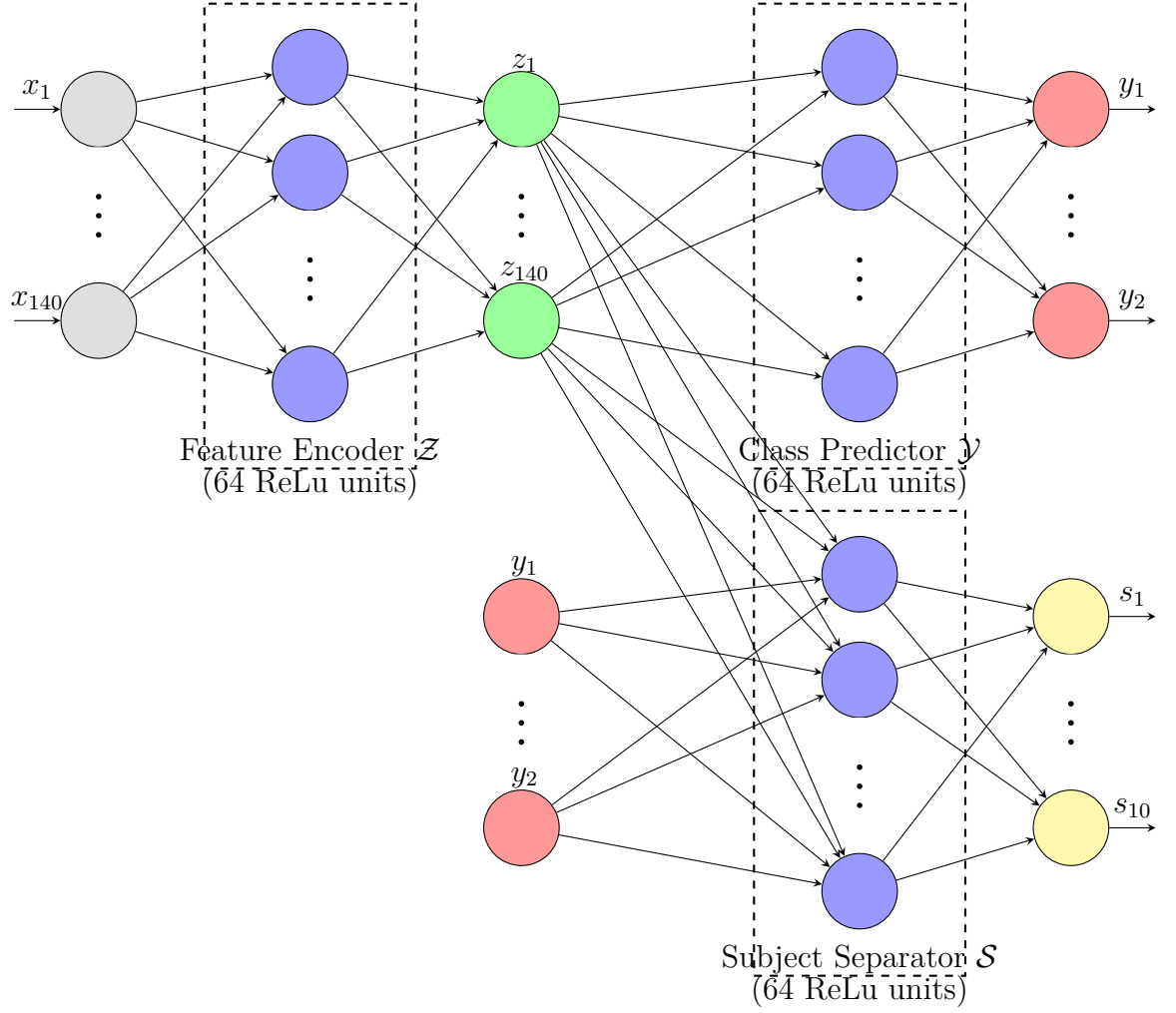


Figure 4.4: The architecture of subject-adversarial neural network (SANN) used in this thesis.

We perform SANN training in batches, or also known as *stochastic gradient descent* (SGD) [199]. The size of each batch is set to be 32. During each batch training, the gradient of the combined loss calculated on all data samples $L = \sum_i L_i$ is back-propagated. The full training SANN training algorithm is presented in Algorithm 1.

4.4 An SANN Case Study

In this section, we present a simple SANN case study on the classification problem. As shown in Figure 4.5, the initial provided data consists samples of two classes (class 1 and class 2) from two subjects distributions (subject 1 and subject 2). All data samples follow the Gaussian distributions. The

Algorithm 1 Subject Adversarial Neural Network Training Algorithm

```

1: Input: Training set  $\mathcal{L} = \{(\mathbf{x}_i, y_i, s_i)\}_{i=1}^M$ ,
2:       Learning rate  $\eta$ .
3:       Domain loss threshold  $\varepsilon_S$ .
4: for number of training batches do
5:   Sample a training batch  $\mathcal{L}' \leftarrow \{(\mathbf{x}_i, y_i, s_i)\}_{i=1}^m$ 
6:   for  $i \in \mathcal{L}'$  do
7:      $L_{\mathcal{Y},i} \leftarrow -\log Q_{\mathcal{Y}}(y_i | \mathcal{Z}(\mathbf{x}_i))$ 
8:      $L_{\mathcal{S},i} \leftarrow -\log Q_{\mathcal{S}}(s_i | \mathcal{Z}(\mathbf{x}_i), Q_{\mathcal{Y}}(\cdot | \mathcal{Z}(\mathbf{x}_i)))$ 
9:      $L_i \leftarrow L_{\mathcal{Y},i} - \lambda L_{\mathcal{S},i}$ 
10:   end for
11:    $\boldsymbol{\theta}_{\mathcal{F}} \leftarrow \boldsymbol{\theta}_{\mathcal{F}} - \eta_{\mathcal{F}} \nabla_{\boldsymbol{\theta}_{\mathcal{F}}} \frac{1}{m} \sum_{i=1}^m L_i$ 
12:    $\boldsymbol{\theta}_{\mathcal{Y}} \leftarrow \boldsymbol{\theta}_{\mathcal{Y}} - \eta_{\mathcal{Y}} \nabla_{\boldsymbol{\theta}_{\mathcal{Y}}} \frac{1}{m} \sum_{i=1}^m L_i$ 
13:   repeat  $\boldsymbol{\theta}_{\mathcal{S}} \leftarrow \boldsymbol{\theta}_{\mathcal{S}} + \eta_{\mathcal{S}} \nabla_{\boldsymbol{\theta}_{\mathcal{S}}} \frac{1}{m} \sum_{i=1}^m L_i$ 
14:   until  $\frac{1}{m} \sum_{i=1}^m L_{\mathcal{S},i} \leq \varepsilon_S$ 
15: end for

```

distribution means of class 1 and class 2 are $[0, 0]$ and $[0, 1]$ respectively for subject 1; and $[1, -1]$ and $[1, 0]$ respectively for subject 2. All distributions have the standard deviations on both dimensions of 0.2. The required task is that we would like to use the data samples of subject 1 to learn a classifier and then apply it to classify the classes of subject 2's samples.

The issue in this case study is that, as obviously illustrated in Figure 4.5, the distributions of two subjects are significantly different. Hence, let us imagine a straight line as a simple linear decision rule (or classifier) that divides the two classes of subject 1, then that classifier cannot well separate the two classes of subject 2.

By employing SANN, we can tackle the issue by transforming the initial feature space into a different feature space where both classes of the two subjects can be separated well. We parametrize all parties in SANN (feature encoder $\mathcal{Z}$, subject separator $\mathcal{S}$, and class predictor $\mathcal{Y}$) as fully-connected feedforward neural networks. The network architectures of the SANN parties in all cases are described in Table 4.1.

First, we analyze the sample distributions when only activating $\mathcal{S}$. The disable of the label predictor $\mathcal{Y}$ is implemented by simply setting the loss func-

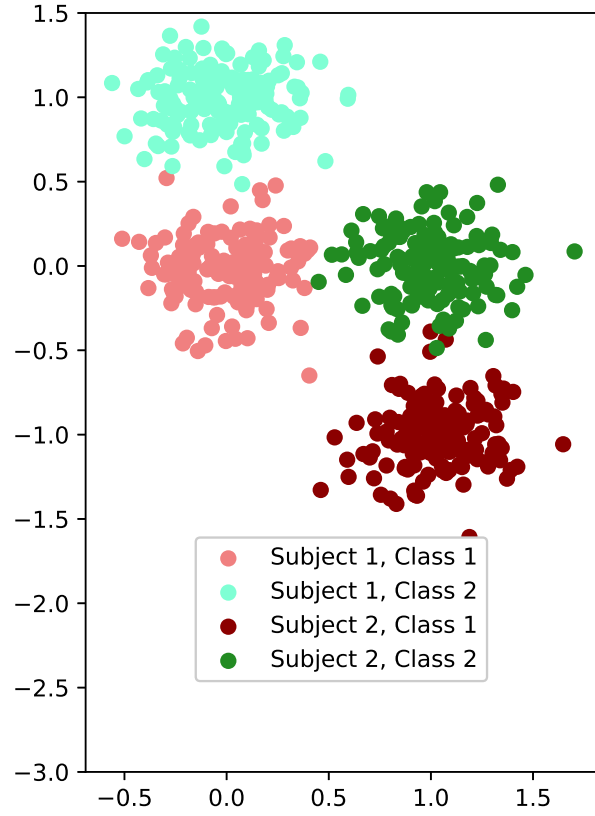


Figure 4.5: The initial distributions of data samples of two subjects mentioned in the case study.

tion of $\mathcal{Y}$ to 0. As shown in Figure 4.6, the samples from each subject are clustered into distinct groups that can leverage the classification by a linear hyperplane, i.e., a straight line in the two-dimensional space. In this case, the subject classification rates are extremely high (99.67%) in both subjects, whereas the class rates are low (50%).

Similarly in case B, we observe the effects of sample distributions when only enabling the class predictor. In this case, one can view the classification process as a single feedforward neural network without domain adaptation. As shown in Figure 4.7, the classes from subject 1 can be extremely well separated by a linear hyperplane, i.e., the vertical line passing through 0. However, this hyperplane fails to classify the classes of subject 2. In this case, the subject classification rates vary in subjects, i.e., extremely high (99.67%) for subject 1, but very low (50.33%) for subject 2.

Finally, the sample distributions when enabling the full SANN architecture

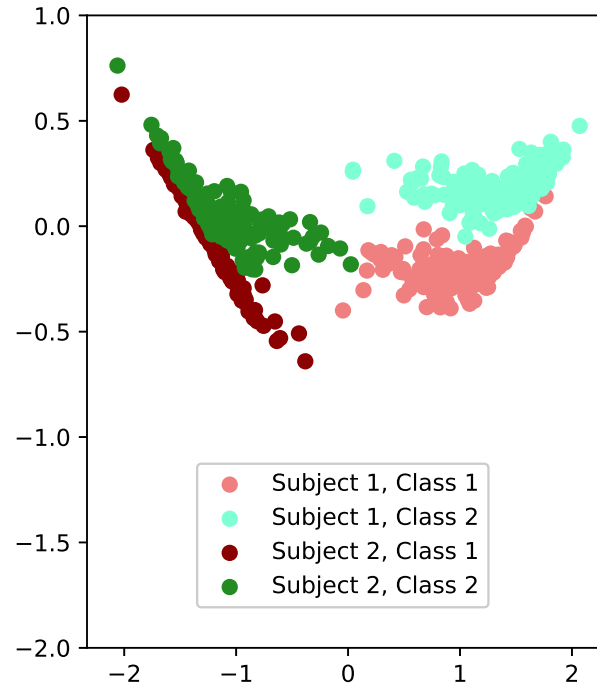


Figure 4.6: The sample distribution of case A in the SANN case study.

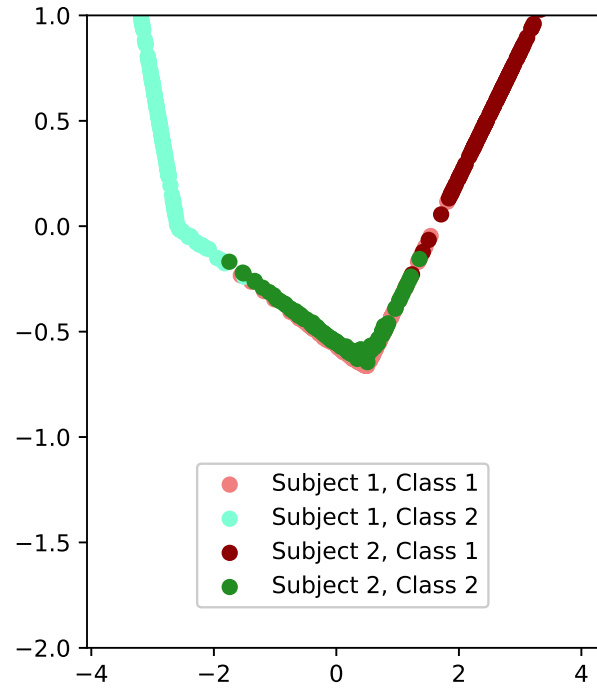


Figure 4.7: The sample distribution of case B in the SANN case study.

are shown is Figure 4.8 and Figure 4.9, corresponding to case C and case D respectively. In case C, the subject rate is approximately 70% in both train

TABLE 4.1: THE SANN ARCHITECTURES OF DIFFERENT CLASSIFICATION CASES IN THE CASE STUDY

	Feature Extractor	Class Predictor	Subject Separator
Case A	1 layer (15 units)	Disabled	1 layer (15 units)
Case B	1 layer (15 units)	1 layer (15 units)	Disabled
Case C	1 layer (15 units)	1 layer (15 units)	1 layer (15 units)
Case D	1 layer (15 units)	1 layer (15 units)	2 layers (15 and 8 units)

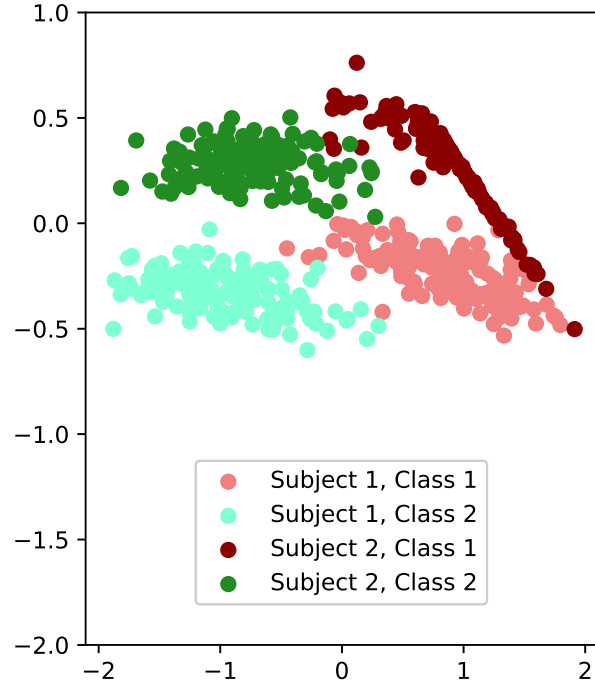


Figure 4.8: The sample distribution of case C in the SANN case study.

and test. As compared to case C, case D employed a deeper domain classifier, resulting in better classification performance in all metrics (above 95%). Let us look at Figure 4.8 and imagine a vertical straight line that can well separate the classes of subject 1, then it can also separate the classes of subject 2 with considerably good performance. In case D, the performance is even better as the distributions of samples are more distinct. We summarize the classification performance of all strategies mentioned in this case study in Figure 4.10.

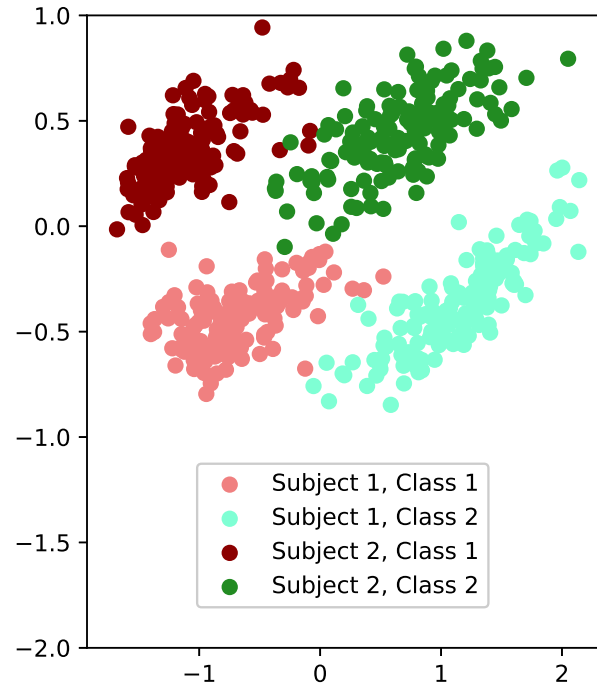


Figure 4.9: The sample distribution of case D in the SANN case study.

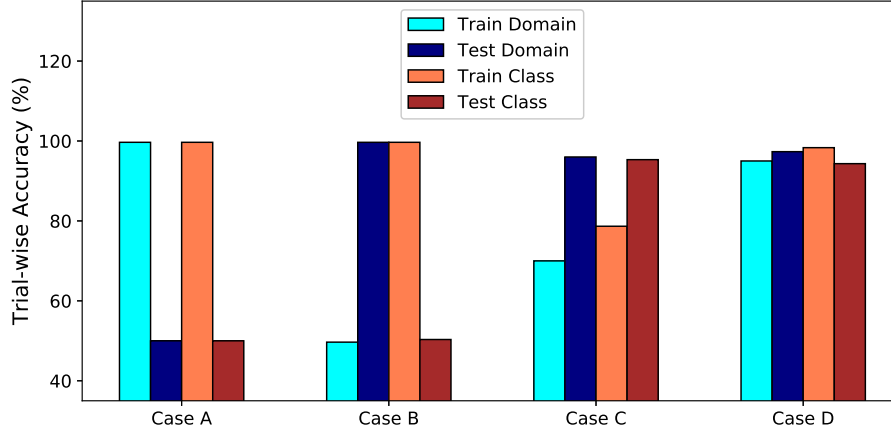


Figure 4.10: The classification performance of 4 cases in the SANN case study. Train domain and test domain refer to the classification accuracy of samples from subject 1 and subject 2, respectively. Train class and test class refer to the classification accuracy of samples from class 1 and class 2, respectively.

4.5 Experiments

4.5.1 t-Distributed Stochastic Neighbor Embedding (t-SNE)

For visualization convenience of data distributions for multi-subject DANN training, we employ t-distributed stochastic neighbor embedding (t-SNE), an award-winning machine learning algorithm for data visualization developed in 2008 by Laurens van der Maaten and his colleagues [200]. The innovation of t-SNE is that it can model each high-dimensional data sample by a representative transformed data point in a two- or three-dimensional point. In the transformed space, the data samples with similar distributions in the original space are close to each other, while dissimilar samples in the original space lies distantly from each other. There exists studies which employed t-SNE for data visualization, including cancer research [201], music analysis [202], computer security [203], bioinformatics [204], or representative learning [205].

Figure 4.11 and Figure 4.12 depicts the t-SNE visualisation of samples of ten first training subjects of Dataset A (as mentioned in Table 3.3) before and after SANN training respectively. The labels are represented by the number of the marker of each point: 1 for target trials and 0 for non-target trials. Each trial belongs to a specific subject, which is depicted as a specific color. As shown, before SANN training the data points of different subjects are distinctively distributed, and the binary class labels are difficult to be separate. After SANN training, in the transformed space the class labels are much easier to be separated, and the distributions of different subjects are mixed, demonstrating the subject-independence aspect of our learning algorithm.

4.5.2 Classification Improvement of SANN

To evaluate the effectiveness of SANN, we view it as an extra feature encoder that encodes the original preprocessed feature vectors (as described in Chapter 3). Hence, only the trained feature encoder of SANN, $\mathcal{Z}$, is used for training and testing other algorithms. The whole dataset is encoded and learned normally by other algorithms, i.e., NN, SVM, FDA. In this chapter, we temporarily skip the review on those algorithms. They will be discussed in details in Chapter 5 (section 5.4).

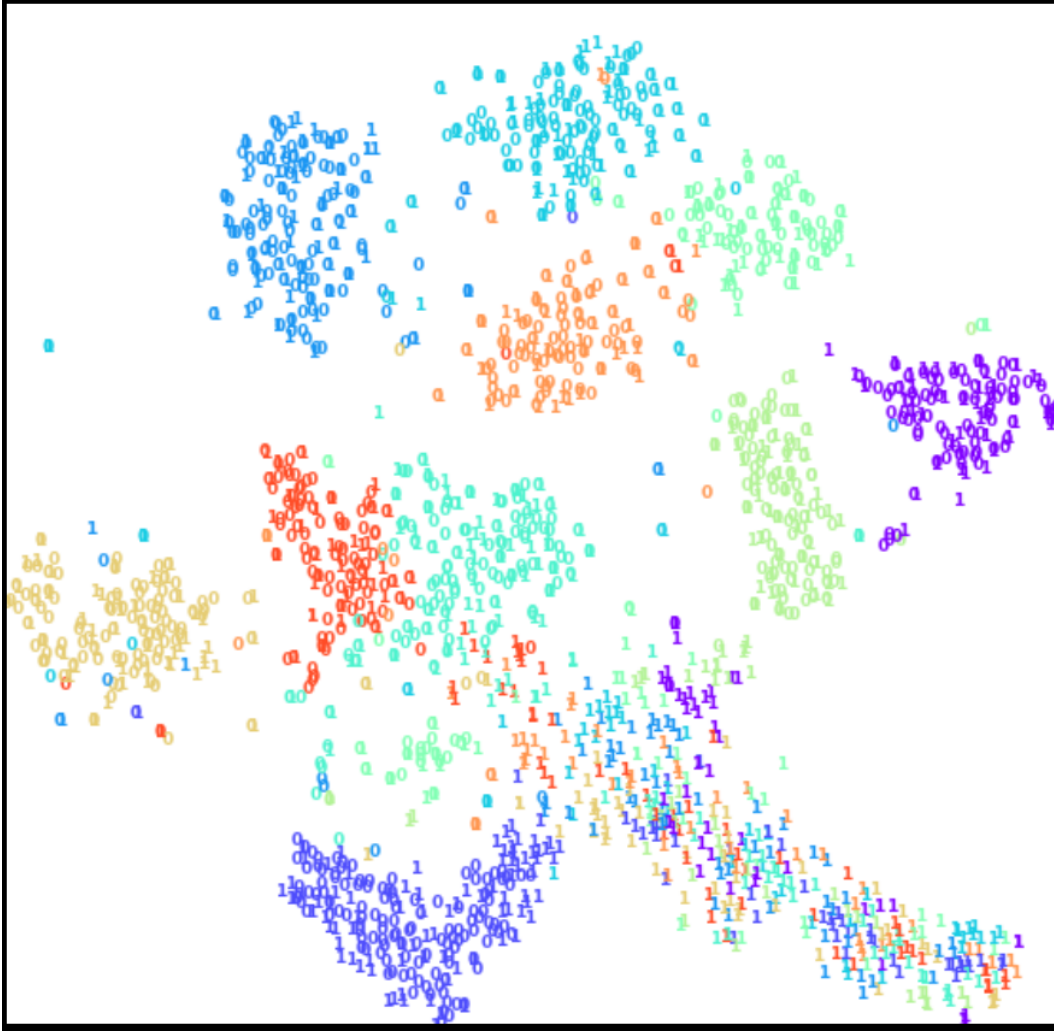


Figure 4.11: The data distribution before SANN learning. The class labels are depicted by marker number (0 for non-target, 1 for target), and the subject labels are depicted by colours.

For better understanding, we plot the full SANN performance evaluation procedure, as depicted in Figure 4.13, and the corresponding accuracy results in Table 4.2. The training and test data is taken from Dataset A as described in Table 3.3). Two measures were evaluated, namely the trial-wise accuracy and symbol-wise accuracy averaged on all test subjects.

As shown in Table 4.2, the accuracies are higher in all algorithms with SANN encoding. Specifically, the average trial-wise accuracy rises from 77.2% to 85.9% by NN, from 79.0% to 83.1% by SVM, and from 54.7% to 56.3% in FDA. Across three algorithms, NN has seen the most increase in symbol-wise accuracy, which rises from 84.4% to 95.0%. This huge leap can be understood

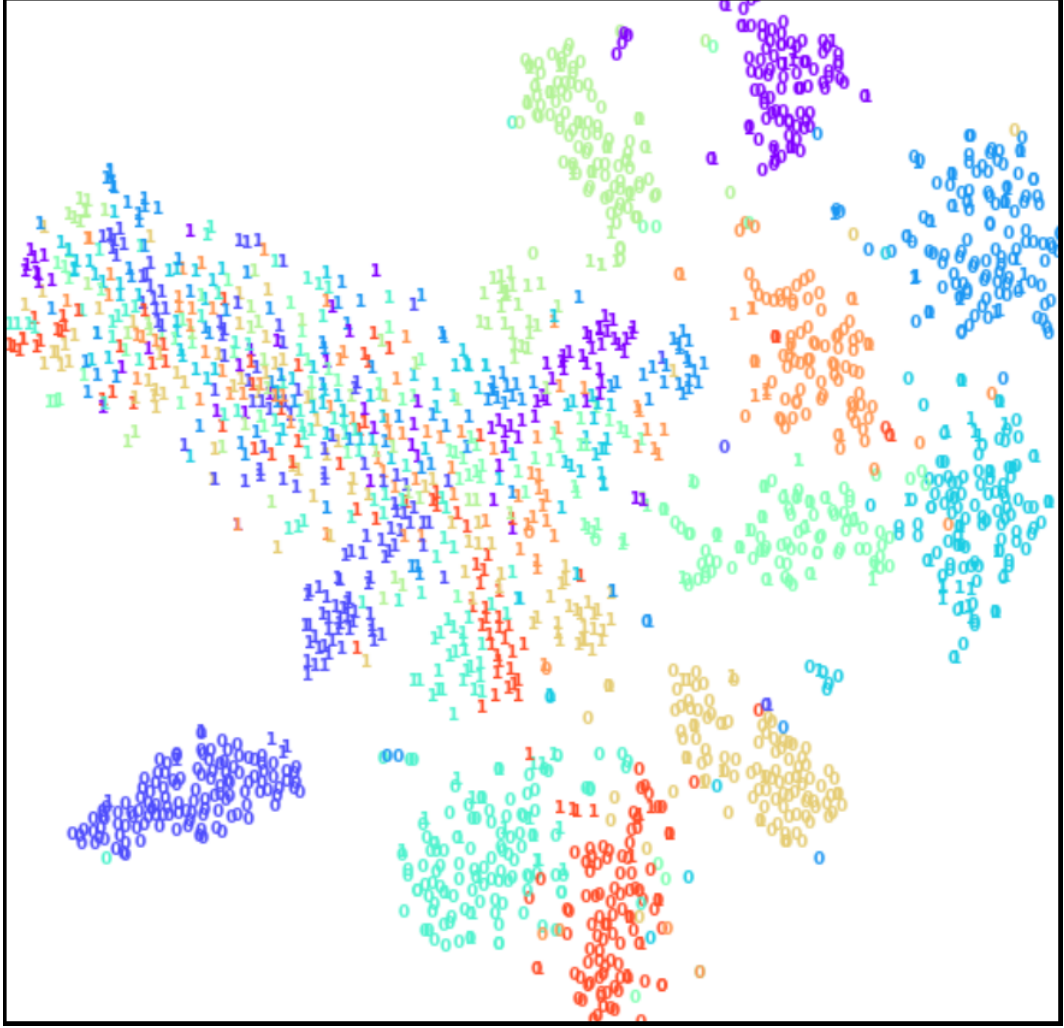


Figure 4.12: The data distribution after SANN learning. The class labels are depicted by marker number (0 for non-target, 1 for target), and the subject labels are depicted by colours.

as the architecture of NN is similar to the class predictor $\mathcal{Y}$ when training the SANN encoder. Therefore an NN classifier can greatly benefit from SANN encoding. On the other hand, the symbol-wise accuracy leap in SVM is 3.9%, while there is no symbol-wise improvement on FDA. This can be explained, since the improvement of trial-wise accuracy on FDA is small, it cannot change the final symbol decision when averaging the FDA scores across multiple repetitions (described in section 3.2).

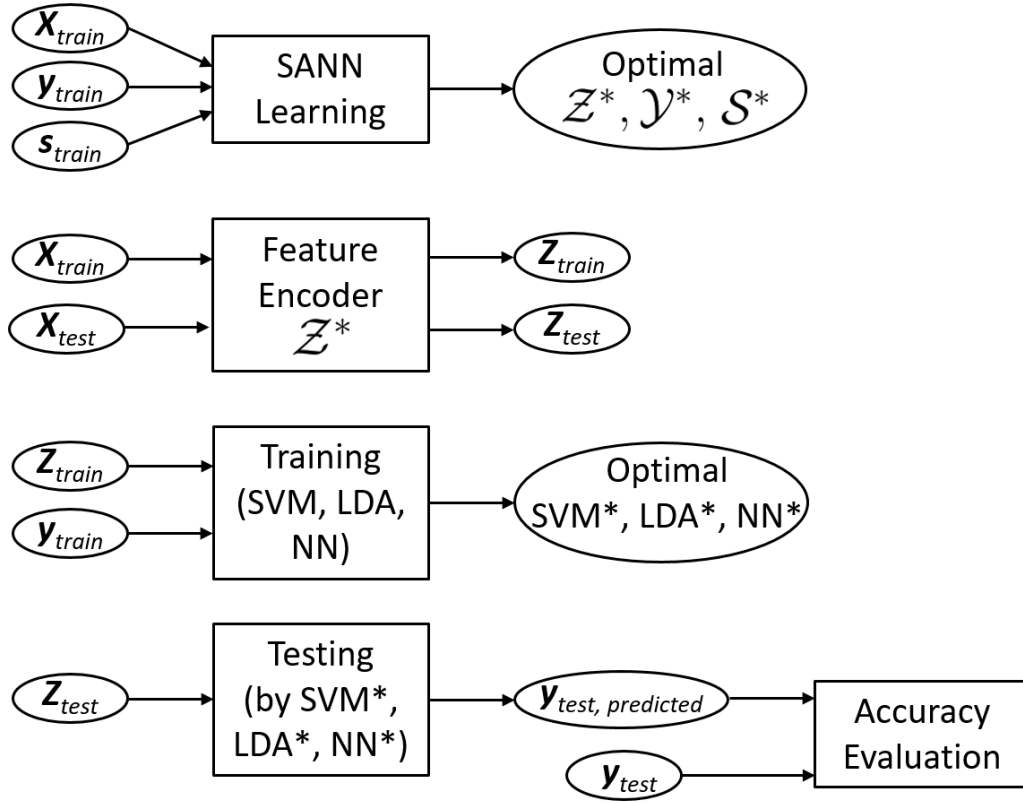


Figure 4.13: The evaluation steps of SANN. This figure is for evaluating the performance with SANN, as shown in Table 4.2. The performance without SANN is achieved by simply directly plugging in $\mathbf{X}_{train}, \mathbf{y}_{train}$ into the training algorithms of SVM, FDA, and NN, equivalent to bypassing the SANN learning and feature encoding steps.

4.6 Chapter Summary

In this chapter, we have presented various aspects of our proposed adaptive subject-independent BCI learning, namely subject adversarial neural network (SANN), which serves as an extra featuring encoding step before classification. The presented contents are

- The motivation of an adaptive learning framework induced by the disadvantages in ERP patterns.
- Literature review on adversarial learning, including the generative adversarial neural network (GANN), and the domain adversarial neural network (DANN).
- The architecture, theory, and properties of SANN, which is a modified version of DANN which can be applied on our ERP-based BCI. We view SANN as a feature encoder which is only used during training.

TABLE 4.2: THE CLASSIFICATION PERFORMANCE WITH AND WITHOUT SANN ON DATASET A.

	Average Trial-wise Accuracy (%)		Average Symbol-wise Accuracy (%)	
	without SANN	with SANN	without SANN	with SANN
NN	77.2	85.9	84.4	95.0
SVM	79.0	83.1	85.8	89.7
FDA	54.7	56.3	76.3	76.3

SANN is different to DANN that the number of domains (subjects) is high, and the identity of the test domain is unknown. Hence, SANN is a complete extra step that can be inserted before the classification step, and can be used with other classification algorithms, i.e., SVM, FDA.

- A simple case study of two subjects, and the practical experiments on the Akimpech dataset (Dataset A).

In the next chapter, we will present the ensemble learning mechanism, which is applied directly on the preprocessed (as described in Chapter 3) and encoded (as described in this Chapter 4) feature vectors. In short, these preprocessed and encoded vectors serve as inputs for various machine learning algorithms, which will be combined in a statistical way.

Chapter 5

ENSEMBLE LEARNING

Ensemble learning is a method to combine different classification or regression models in a specific way, in order to boost up the performance [35,206,207]. Ensemble works more efficiently when there are significant diversity among constituent models, hence overfitting can be drastically reduced [206]. In this thesis, the most two common state-of-the-art ensemble methods are employed in a unified framework, namely *bagging* and *stacking*. Briefly summarised, bagging is the method of dividing the whole training dataset into multiple subsets, where some of them are used for training and the other for validation. By randomly picking the subsets for training multiple times, the resulting averaged predictions are more robust to overfitting, and also can be exploited by the stacking method. Stacking is the action of re-feeding the validation predictions from bagging as extra features, to exploit the strength of the model to learn some implicit relations between the validation prediction and the original features themselves.

This chapter is organized as follows. Section 5.1 and section 5.2 presents the bagging and stacking method respectively. In details, for each of these two sections, a brief review is given, followed by the theoretical explanation on how bagging or stacking can improve the performance, and then the algorithm. The combinations of these two techniques to prevent overfitting is presented in section 5.3. Next, section 5.4 briefly reviews the background of the three base learners used in the ensemble strategy, namely, support vector machine, feedforward neural network, and logistic regression. In this section, we also describe the tuning technique to achieve the optimal hyperparameters for each

base learner. The out-of-fold (OOF) bagging results of these 3 base learners are stacked by a linear regression model, which is described in section 5.5. Finally, section 5.6 presents the performance evaluation results of the whole ensemble strategy used in this thesis, followed by a short chapter summary in section 5.7.

5.1 Bagging

5.1.1 Brief Review of Bagging

Bagging is the abbreviation of *bootstrap aggregating* [207], which is a method based on two key techniques: bootstrap and aggregation. The idea of bagging training is that one can obtain better generalisation error when averaging models that were trained on different subsets of the full dataset [34, 207]. For a given data set, one way to generate many base learners is to divide the original provided data set into numerous non-overlapped subsets, then performing one base learner on each subset. However, this method has a major drawback due to the finiteness of training data. Even for a minimal split (2 subsets), the representative capability of each subset is massively decreased. That leads to poor results in each base learner.

Therefore, bagging employ the distribution of bootstrap to generate overlapped subsets for base learners. The most exhaustive bagging scheme conducted on an m -sample dataset is to generate m subsets, with each subset equates the full set subtracting one unique sample. Then, each base learner can be learned from one subset. However, due to the intractability when dealing with large dataset, one could specify a preferred number of subsets, $K < m$, to increase training time.

The separate results of each bagging base learner (or base model) are combined by a handful of special techniques, such as voting (for classification) or averaging (for regression). For instance, in a classification application, the winning class label is the class label that has the most votes from K base models. An arbitrary winning label is picked if there occurs several labels with equal number of votes, or one can develop a postprocessing algorithm to bet-

ter determine the winning label. In a regression application, each base model produces a soft score for target regression. Therefore one can take the average of all scores from K base models' outputs as the final result. The pseudo-code of bagging for classification and regression is presented in Algorithm 2 and Algorithm 3, respectively.

Algorithm 2 Bagging Algorithm for Classification Applications

- 1: Input: Training set $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$,
 - 2: Number of subsets K ,
 - 3: Base learner algorithm F , dataset as input, output a classifier
 - 4: Draw K subsets with replacement from $\mathcal{L}$, denoted by $\mathcal{L}_k$
 - 5: **for** $k = 1, \dots, K$ **do**
 - 6: $f_k \leftarrow F(\mathcal{L}_k)$
 - 7: **end for**
 - 8: Output: Bagging classifier $f_{\text{bagging}}(\mathbf{x}) = \text{argmax}_y \sum_{k=1}^K \mathbb{I}(f_k(\mathbf{x}) = y)$, where $\mathbb{I}(A) = 1$ if A is true, and $\mathbb{I}(A) = 0$ if A is false.
-

Algorithm 3 Bagging Algorithm for Regression Applications

- 1: Input: Training set $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$,
 - 2: Number of subsets K ,
 - 3: Base learner algorithm F , dataset as input, output a regressor
 - 4: Draw K subsets with replacement from $\mathcal{L}$, denoted by $\mathcal{L}_k$
 - 5: **for** $k = 1, \dots, K$ **do**
 - 6: $f_k = F(\mathcal{L}_k)$
 - 7: **end for**
 - 8: Output: Bagging regressor $f_{\text{bagging}}(\mathbf{x}) = \frac{1}{K} \sum_{k=1}^K f_k(\mathbf{x})$.
-

Another advantage of bagging is that, given m training samples, the probability that each specific sample is selected n times can be approximated by a Poisson distribution with $\lambda = 1$. This is equivalent to saying that, the probability for each sample to appear in bagging sampling at least once is $1 - (1/e) \approx 0.632$. As a result, 36.8% of training samples have not been used for each base learner. This offers us the ability to estimate the generalisation performance of the bagging scheme [207–209].

5.1.2 Bagging reduces generalisation errors

In this section, a proof will be provided to authenticate the following statement: the generalisation errors on test data will be dramatically decreased if

we combine diverse and independent base learners.

Suppose we draw each point $(y, \mathbf{x})$ from the probability distribution P to form a training set $\mathcal{L}$, where y is the scalar of class label and $\mathbf{x}$ is the feature vector. Let $f(\mathbf{x}, \mathcal{L})$ be the learner that is trained on $\mathcal{L}$, then the ensemble bagging learner averaged on all possibilities of $\mathcal{L}$ can be defined as

$$f_{\text{bagging}}(\mathbf{x}, P) = \mathbb{E}_{\mathcal{L}}[f(\mathbf{x}, \mathcal{L})]. \quad (5.1)$$

Let us denote $\mathbf{X}, Y$ as the random variables independent of $\mathcal{L}$ and with distribution P . We calculate the average prediction error e on the weak learning $f(\mathbf{x}, \mathcal{L})$ over all possibilities of $\mathcal{L}$ by

$$\begin{aligned} e &= \mathbb{E}_{\mathcal{L}} \mathbb{E}_{\mathbf{X}, Y} [Y - f(\mathbf{X}, \mathcal{L})]^2 \\ &= \mathbb{E}_{\mathcal{L}} \mathbb{E}_{\mathbf{X}, Y} [Y^2] - 2 \mathbb{E}_{\mathbf{X}, Y} [Y] \mathbb{E}_{\mathcal{L}} [f(\mathbf{X}, \mathcal{L})] + \mathbb{E}_{\mathbf{X}, Y} \mathbb{E}_{\mathcal{L}} [f(\mathbf{X}, \mathcal{L})]^2 \\ &= \mathbb{E}_{\mathbf{X}, Y} [Y^2] - 2 \mathbb{E}_{\mathbf{X}, Y} [Y f_{\text{bagging}}] + \mathbb{E}_{\mathbf{X}, Y} \mathbb{E}_{\mathcal{L}} [f(\mathbf{X}, \mathcal{L})]^2. \end{aligned} \quad (5.2)$$

On the other hand, the error produced by f_{bagging} is

$$\begin{aligned} e_{\text{bagging}} &= \mathbb{E}_{\mathbf{X}, Y} [Y - f_{\text{bagging}}(\mathbf{X}, P)]^2 \\ &= \mathbb{E}_{\mathbf{X}, Y} [Y^2] - 2 \mathbb{E}_{\mathbf{X}, Y} [Y f_{\text{bagging}}] + \mathbb{E}_{\mathbf{X}, Y} [\mathbb{E}_{\mathcal{L}} f(\mathbf{X}, \mathcal{L})]^2. \end{aligned} \quad (5.3)$$

Since $\mathbb{E} Z^2 \geq \mathbb{E}^2 Z$ for all Z we obtain $e \geq e_{\text{bagging}}$, and the margin between e and e_{bagging} depends on the variance of $f_{\text{bagging}}(\mathbf{X}, P)$ as

$$\begin{aligned} e - e_{\text{bagging}} &\geq \mathbb{E}_{\mathbf{X}, Y} [\mathbb{E}_{\mathcal{L}} [f(\mathbf{X}, \mathcal{L})]^2] - \mathbb{E}_{\mathbf{X}, Y} [\mathbb{E}_{\mathcal{L}}^2 f(\mathbf{X}, \mathcal{L})] \\ &= \mathbb{E}_{\mathbf{X}, Y} [\text{Var}_{\mathcal{L}} f(\mathbf{X}, \mathcal{L})]. \end{aligned} \quad (5.4)$$

Hence, we expect to have a good bagging ensemble by drawing and training diverse subsets $\mathcal{L}$ from P . We do so by splitting the originally provided training set into a partition of k separate non-overlapping *folds*, all of which have the equal number of training samples. Each fold of the partition is validated by the learner trained on the samples of all the other folds. This process is called *out-of-fold* (OOF) validation. Another benefit of this bagging method is that it allows us to “try out” for the best hyperparameters of each type of learner, i.e., C and kernel function in SVM, or number of hidden layers and learning rate in NN. This is done by taking the average performance on all folds, and try another hyperparameter settings to see if they perform better. This procedure is wrapped in an outer loop of the original bagging method, where we can specify all possible values of the hyperparameters this loop can validate.

5.2 Stacking

5.2.1 Overview and Algorithm of Stacking

Another powerful ensemble method is *stacking*, where the results of differing learners trained on the same dataset is combined by another learner at a higher level. Stacking was first introduced in [207, 210, 211].

We call the first level of learners the *base learners*, and the learner to combine them the *stack learner*. The key success of stacking is that for each given input sample $\mathbf{x}$, stacking can combine a generalized result using N base learners of totally different nature. Base learners in stacking, can be viewed different to base learners in bagging. In bagging, base learners are learned by a specific algorithm conducted on different subsets of data, while in stacking, base learners are learned from a specific data set, but with different algorithm approaches, i.e., SVM, NN, etc. Stacking is expected to generalise well when its base learners are trained by completely differing nature, i.e., deterministic methods (support vector machines, neural networks) or probabilistic methods (linear regression, random forests). Each base learner's prediction in the first level is one feature for the feature vector of the stack learner. Hence, a better generalisation performance is achieved when one attempts to employ as many base learner's methods as possible. It is also worth to mention that, the stacking process can be employed multiple times, instead of two times as described previously. For instance, all features from the first level are learned by N base methods again, forming the feature vectors of the second level that has the size equal to those of the first levels. This process can be iterated multiple times until the training error reaches a plateau. The pseudo-code for stacking ensemble is provided in Algorithm 4.

5.2.2 Stacked learner performs best if base learners are comparable

In this section, we provide a proof to demonstrate that the stacked learner performs better than each individual base learner, if the base learners are trained by different approaches which yield nearly equal performance. This is

Algorithm 4 A two-level Stacking Algorithm

```

1: Input: Training set  $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ ,
2:          $N$  base learner algorithm  $F_n$ .
3:         1 meta learner algorithm  $F$ .
4: for  $n = 1, \dots, N$  do
5:    $f_n = F_n(\mathcal{L})$ 
6: end for
7: Initialize second-level data set  $D \leftarrow \emptyset$ 
8: for  $i = 1, \dots, m$  do
9:   for  $n = 1, \dots, N$  do
10:     $z_{in} = f_n(\mathbf{x}_i)$ 
11:   end for
12:    $D \leftarrow D \cup \{[z_{i1}, z_{i2}, \dots, z_{iN}], y_i\}$ 
13: end for
14: Train meta model  $f = F(D)$ 
15: Output: Stacking model  $f_{\text{stack}}(\mathbf{x}) = f([f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_N(\mathbf{x})])$ 

```

to ensure that, once base learners are trained by different methods, there is an extremely high probability that the stacked ensemble will produce better generalisation performance.

Suppose each of the N base learners is trained on the training set $\mathcal{L} = \{(\mathbf{x}_m, y_m)\}_{m=1}^{|\mathcal{L}|}$, where $|\mathcal{L}|$ is the *size* of $\mathcal{L}$, i.e., the number of samples in $\mathcal{L}$. We denote the prediction on the sample $\mathbf{x}$ of each base learner that is trained on set $\mathcal{L}$ as $f_n^{(\mathcal{L})}(\mathbf{x}), n = 1 \dots N$. Stacking ensemble combines the predictions from all base learners by assigning non-negative weights to them

$$f_{\text{stack}}(\mathbf{x}) = \sum_{n=1}^N \alpha_n f_n^{(\mathcal{L})}(\mathbf{x}), \quad \alpha_n \geq 0. \quad (5.5)$$

The empirical squared prediction error of f_{stack} on $\mathcal{L}$ is

$$\begin{aligned} e_{\text{stack}} &= \sum_m \left(y_m - \sum_n \alpha_n f_n^{(\mathcal{L})}(\mathbf{x}_m) \right)^2 \\ &= \sum_i \sum_j \alpha_i \alpha_j R_{ij}, \end{aligned} \quad (5.6)$$

where R_{ij} is the *residual* error products of learner f_i and f_j

$$R_{ij} = \sum_m (y_m - f_i^{(\mathcal{L})}(\mathbf{x}_m))(y_m - f_j^{(\mathcal{L})}(\mathbf{x}_m)). \quad (5.7)$$

Breiman et al [35], by using the duality technique in optimisation theory, proved that any base learner n performs only at best as the stacked learner if and only if the following condition holds

$$R_{nn} \leq R_{ni}, \quad \forall i. \quad (5.8)$$

On a statistical viewpoint, the above condition relates to the Pearson correlation $\rho_{A,B}$ between two random variables A and B as

$$\rho_{A,B} = \frac{\mathbb{E}[(A - \mathbb{E}[A])(B - \mathbb{E}[B])]}{\sigma_A \sigma_B}, \quad (5.9)$$

where σ_A, σ_B are the standard deviation of A and B . Considering the residual error, $y_m - f_i^{(\mathcal{L})}(\mathbf{x}_m)$, of any base learner i as a random variable, hence it has zero mean and $\rho_{i,i} = 1$ for all i . The correlation between the residuals of two base learners i, j is computed by

$$\rho_{i,j} = \frac{\sum_m (y_m - f_i^{(\mathcal{L})}(\mathbf{x}_m))(y_m - f_j^{(\mathcal{L})}(\mathbf{x}_m))}{\sqrt{\sum_m (y_m - f_i^{(\mathcal{L})}(\mathbf{x}_m))^2 \sum_m (y_m - f_j^{(\mathcal{L})}(\mathbf{x}_m))^2}}. \quad (5.10)$$

From (5.10), we obtain the equivalent form of the condition in (5.8) as

$$\frac{\sigma_i}{\sigma_n} \leq \rho_{n,i}. \quad (5.11)$$

Intuitively, suppose we have a base learner i that is totally different from the best performing base learner n . If i produces the comparable squared errors as those of n , then their correlation approximates 1, which means learner i must be nearly identical to the learner n . This contradiction yields the final conclusion that no base learner can perform better than the stacked learner if the base learners are of diverse methods with comparable errors.

5.3 Combining Bagging and Stacking: Avoid Overfitting

One issue of stacking ensemble is that $f_i^{(\mathcal{L})}(\mathbf{x}_m)$ will estimate back on $\mathbf{x}_m$ after it was used as a training sample, i.e., $\mathbf{x}_m$ is included in $\mathcal{L}$. As a result overfitting will occur and we have no means to control this problem. One has to therefore divide the original training set into two subsets: one for training and one for validating, then use the validated scores as inputs of the stacking ensemble. Since the validated scores only constitute part of the original training set, we need to have different partitions of divides where the union of validation sets of all partitions is exactly the same as the original training set. This method is identical to bagging ensemble as previously mentioned in Section 5.1.

Let us divide the training set $\mathcal{L}$ into K non-overlapping folds, denoted by $\mathcal{L}_k$, $k = 1, \dots, K$. The number of samples in each fold is made as nearly equal to each other as possible, and the folds must cover all the original dataset, i.e., $\cup_k \mathcal{L}_k = \mathcal{L}$. Each fold k is validated by a base learner n that is trained on the samples combined from all the other folds, denoted by $\bar{\mathcal{L}}_k = \cup_{i \neq k} \mathcal{L}_i$. By further denoting the validated scores of the base learner n on fold k as $\mathbf{s}_{nk}$, then we have the following $K \times N$ sets of scores

$$\mathbf{s}_{nk} = f_n^{(\bar{\mathcal{L}}_k)}(\mathcal{L}_k) = [s_{nk,1} \quad s_{nk,2} \quad \dots \quad s_{nk,|\mathcal{L}_k|}]. \quad (5.12)$$

By treating $\mathbf{s}_{nk}$ as a row vector, we can concatenate all entry-level individual scores into a matrix $\mathbf{S}$ of size $|\mathcal{L}| \times N$ as

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_{11} & \mathbf{s}_{12} & \dots & \mathbf{s}_{1K} \\ \mathbf{s}_{21} & \mathbf{s}_{22} & \dots & \mathbf{s}_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}_{N1} & \mathbf{s}_{N2} & \dots & \mathbf{s}_{NK} \end{bmatrix} = [\mathbf{c}_1 \quad \mathbf{c}_2 \quad \dots \quad \mathbf{c}_{|\mathcal{L}|}]. \quad (5.13)$$

As a result, each column of $\mathbf{S}$, denoted by $\mathbf{c}_m$, $m = 1, \dots, |\mathcal{L}|$, is the combination of scores of all base learners on one specific sample of $\mathcal{L}$. The stacking ensemble is therefore learned from the set of $|\mathcal{L}|$ vectors of dimension N , denoted by $\mathcal{L}_{\text{stack}}$ as

$$\mathcal{L}_{\text{stack}} = \{(\mathbf{c}_m, y_m)\}_{m=1}^{|\mathcal{L}|}. \quad (5.14)$$

In this chapter, we employ linear regression as the stacking ensemble learner thanks to its simplicity and robustness on low-dimensional regression problems [207]. Furthermore, linear regression is tailored to our stacking theoretical assumption and justification described in section 5.2.

5.4 Base Learners

We employ three types of base learners for our ensemble bagging-stacking classifier, namely support vector machine (SVM), Fisher's discriminant analysis (FDA), and fully-connected feedforward neural network (NN).

SVM is a deterministic binary classification algorithm proposed by Vapnik et al in 1995 [212]. Since its advent, SVM gained a huge popularity thanks

to its outstanding generalisation capability on binary classification problems. One advantage of SVM is that the model solution only depends on a subset of the training data, called support vectors. These vectors are determined as the closest points to the decision boundary.

On the other hand, FDA is a probabilistic approach which approximates the distribution of training samples on the feature space, where no vector is treated as special [213]. The totally different natures of SVM and FDA yield some discrepancies on their class probabilities, which is extremely desirable for our ensemble method.

We also inspect the performance of neural network (NN) as the third base learner. Indeed, SVM or FDA can be considered as an NN with one node and an appropriate activation function. NN is theoretically proved to have the capability of constructing the model that fits any arbitrary distribution [214]. With SVM and FDA however, this objective cannot be obtained. Hence in general NN usually performs better than SVM and FDA with the trade-off in training time. The NN used in this chapter is the fully-connected feed-forward architecture with 256 units in the hidden layer followed by ReLu activation functions [215].

5.4.1 Support Vector Machine

Background:

Support vector machine tries to estimate a function $f : \mathbb{R}^N \rightarrow \{\pm 1\}$ by training a dataset consisting of l data samples $\mathbf{x}_i$ with corresponding label y_i

$$\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m, \quad (5.15)$$

where $y_i \in \{\pm 1\}$ is a compulsory binary label classes. Suppose a new example $(\mathbf{x}, y)$ is drawn from the same underlying distribution $P(\mathbf{x}, y)$ as the training data, then the function f , after training, will take $\mathbf{x}$ as input and attempt to predict y . SVM classifiers are constructed based on a class of hyperplanes defined as

$$\mathbf{w} \cdot \mathbf{x} + b = 0, \quad (5.16)$$

where $\mathbf{w} \in \mathbb{R}^N, b \in \mathbb{R}$. This hyperplane is optimal when the margin of separation between two class is maximum. The solution for this optimal hyperplane is unique, and is constructed by solving a quadratic optimisation problem. The decision function f is therefore defined as

$$f(\mathbf{x}) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b). \quad (5.17)$$

Separable Case:

When the samples of two classes of the training data is *separable*, i.e., there exists a hyperplane which can separate all samples from two classes, we can find two parallel hyperplanes with the largest distance, or margin. The margin offset of the optimal hyperplane to the so-called *support vectors*, i.e., the set of vectors lies nearest to the optimal hyperplane, equates $b/\|\mathbf{w}\|$. The equations for these two optimal hyperplanes are $\mathbf{w} \cdot \mathbf{x} + b = 1$, where the support vectors are from class 1, and $\mathbf{w} \cdot \mathbf{x} + b = -1$ for class -1 . As a result, the distance between these two hyperplanes are $2/\|\mathbf{w}\|$. Differently speaking, we want to minimize $\|\mathbf{w}\|$ as a consequence to maximizing this distance. In this separable case, there will be no vector falling into the space gap between these two hyperplanes, as well as there will be no vector misplaced into the wrong hyperplane side. The latter statement can be stated mathematically as

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x} + b &\geq 1, \quad \text{if } y_i = 1, \\ \mathbf{w} \cdot \mathbf{x} + b &\leq -1, \quad \text{if } y_i = -1. \end{aligned} \quad (5.18)$$

Equivalently, the two conditions can be rewritten as the following optimisation problem

$$\begin{aligned} &\text{minimize } \|\mathbf{w}\| \\ &\text{subject to } y_i(\mathbf{w} \cdot \mathbf{x} + b) \geq 1, \quad \forall i. \end{aligned} \quad (5.19)$$

Inseparable Case:

In reality, unfortunately separable cases rarely exist, hence one wants to employ the *soft-margin* hinge-loss function

$$\max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x} + b)). \quad (5.20)$$

Intuitively, the function produces zero if $\mathbf{x}_i$ lies in the correct hyperplane side, and produces a penalty that is proportional to the distance of $\mathbf{x}_i$ to the wrong hyperplane if it lies in the incorrect side. The optimisation problem then can

be restated as

$$\text{minimize } \frac{1}{l} \sum_{i=1}^m \max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x} + b)) + \lambda \|\mathbf{w}\|^2, \quad (5.21)$$

where λ is the positive scalar representing the trade-off between the general separable capacity (i.e., the size of the hyperplanes' margin) and the number of $\mathbf{x}_i$ lying on the correct side. In other words, with small λ , the second term is negligible and the problem behaves like the separable case.

Transforming primal to dual: To solve (5.21), we introduce a variable $\zeta_i = \max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x} + b))$, hence, the primal SVM optimisation problem is rewritten as

$$\begin{aligned} &\text{minimize } \frac{1}{l} \sum_{i=1}^m \zeta_i + \lambda \|\mathbf{w}\|^2 \\ &\text{subject to } y_i(\mathbf{w} \cdot \mathbf{x} + b) \geq 1 - \zeta_i, \quad \forall i, \\ &\quad \zeta_i \geq 0, \quad \forall i \end{aligned} \quad (5.22)$$

The dual problem is derived by using the Lagrangian dual of the primal problem. Its simplified form is

$$\begin{aligned} &\text{maximize } \sum_{i=1}^m c_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m y_i c_i (\mathbf{x}_i \cdot \mathbf{x}_j) y_j c_j, \\ &\text{subject to } \sum_{i=1}^m c_i y_i = 0, \quad \forall i, \\ &\quad 0 \leq c_i \leq \frac{1}{2m\lambda}, \quad \forall i \end{aligned} \quad (5.23)$$

This dual problem can be solved by a quadratic optimisation algorithm [212]. The resulting solution c_i constructs the original solution of the primal problem as

$$\mathbf{w} = \sum_{i=1}^m c_i y_i \mathbf{x}_i. \quad (5.24)$$

Kernel trick and the final form of solution:

One further innovative aspect of SVM is that, it allows one to perform the *kernel trick*, i.e., to transform the input space into a different *feature space* where the problem is more separable by the linear SVM. This transformation maps each input $\mathbf{x}_i$ to $\varphi(\mathbf{x}_i)$, with the corresponding kernel function

$k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$. Consequently and finally, the classifier function which predicts the label of a new sample $\mathbf{x}$ is

$$\hat{y} = \text{sign}(\mathbf{w} \cdot \varphi(\mathbf{x}) + b) = \text{sign}\left(\sum_{i=1}^m c_i y_i k(\mathbf{x}_i, \mathbf{x}) + b\right). \quad (5.25)$$

5.4.2 Neural Network

Background:

Neural network (NN), also called feedforward neural network, or multilayer perceptrons (MLP), is the essential machine learning procedure to learn an implicit function f relating between an input sample $\mathbf{x}$ and corresponding outputs vectors $\mathbf{y}$. For instance, $\mathbf{x}$ can be one vector representing a speech signal segment, or an image, and $\mathbf{y}$ is the corresponding one-hot label (i.e., only one element equals 1 at a specific position indicating the category of $\mathbf{x}$, and the rest elements are all 0).

An important and beautiful aspect of NN is that, it can generate a chain of functions, in which outputs of the current function $f^{(i)}$ will be fed as inputs of the next function $f^{(i+1)}$. In other words, we can construct a multilayer network, or also known as the so-called *deep neural network*. For example, with a 3-layer NN, the final function can be represented by its three constituent layer functions as

$$f(\mathbf{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x}))), \quad (5.26)$$

where $f^{(i)}$ is the function of layer i . The number of layers gives the *depth* of the network. Figure 5.1 depicts the topology of a three-layer fully-connected feedforward NN. In Figure 5.1, the input layer (blue) receives a 3-dimensional vector (i.e., $\mathbf{x} \in \mathbb{R}^3$). The first, second, and third hidden layers (grey) receives the input vector which is the output vector from the previous layer. The output layer (red) is the one-hot encoded label vector $\mathbf{y}$ (i.e., for a binary classification problem, $\mathbf{y} = [1, 0]$ for x in one class or $\mathbf{y} = [0, 1]$ for x in the other class).

Given a training data, NN learns f by producing the network parameters $\boldsymbol{\theta}$. Finding $\boldsymbol{\theta}$ is the ultimate goal of learning the network. Since $\boldsymbol{\theta}$, once achieved, contributes to the classification function f on new samples, one may

want to represent f as a function of both the input sample $\mathbf{x}$ and the learned parameters $\boldsymbol{\theta}$ as

$$\mathbf{y} = f(\mathbf{x}; \boldsymbol{\theta}). \quad (5.27)$$

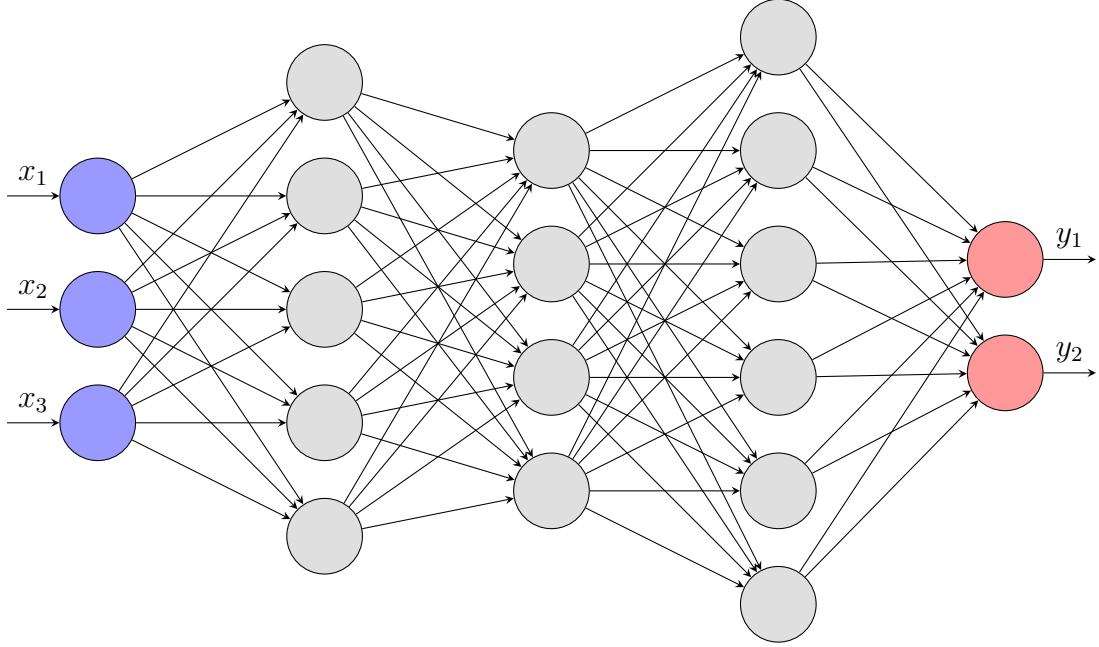


Figure 5.1: A topology of a conventional fully-connected feedforward neural network with 3 hidden layers.

Architecture of a single neuron:

Each single neuron (i.e., a grey square in Figure 5.1) produces a scalar output. It receives as inputs all the outputs of the previous layer's neurons, computes the linear combination of them with weights $\mathbf{w}$, then passes through an *activation function* g . All of these processes can be interpreted as

$$y(\mathbf{x}) = g(\mathbf{w} \cdot \mathbf{x}) = g\left(\sum_{i=1}^n w_i x_i\right) = g(z), \quad (5.28)$$

where n is the size (i.e., the number of neurons) of the preceding layer with regards to the current inspected neuron. The activation g can be selected from a handful of commonly used functions. Some examples are the thresholding function to produce only 0 or 1 based on a specified threshold (5.29), sigmoid function to produce smooth output ranging from 0 to 1 (5.30), or hyperbolic tangent function to produce smooth output ranging from -1 to 1 (5.31)

$$g(z) = \begin{cases} 1 & \text{if } z + t > 0 \\ 0 & \text{if } z + t \leq 0 \end{cases}, \quad (5.29)$$

$$g(z) = \frac{1}{1 + e^{-2s(z+t)}}, \quad (5.30)$$

$$g(z) = \frac{e^{2(s(x+t))} - 1}{e^{2(x(z+t))} + 1}, \quad (5.31)$$

where t is a function parameter that specifies the center of the activation function, and s is the function parameter representing the steepness.

Feedforward:

To execute the NN (i.e., to produce the output layer's prediction), the input vector $\mathbf{x}$ is fed directly into the input layer with the same dimension. Then the feedforward procedure calculates the output of each neuron of the first hidden layer by 5.28. The process is repeated similarly for the second hidden layer, until it reaches the output layer. It should be noted that, different layer can be specified different activation functions, based on the context of each application of the network. However, g must be differentiable for the sake of the backpropagation method used in training.

Backpropagation:

During training, the network parameters $\boldsymbol{\theta}$ is initially randomly generated for the first iteration. Then in an iteration algorithm, $\boldsymbol{\theta}$ is continually updated after each loop using the *backpropagation* method. This iteration will stop if $\boldsymbol{\theta}$ is achieved such that the average output error of all training samples is smaller (or equal) to the predefined threshold value ϵ . In a specific loop, each output neuron k produces the value y_k , resulting in an error value e_k as

$$e_k = d_k - y_k, \quad (5.32)$$

where d_k is the fixed desired output of the output neuron k for every loop. Using this error value, one can calculate δ_k as

$$\delta_k = e_k g'(y_k), \quad (5.33)$$

where g' is the derivative of g . With all values of δ_k of the output layer, the values δ_j of the previous layer can be backpropagated by

$$\delta_j = \eta g'(y_j) \sum_k \delta_k w_{jk}, \quad (5.34)$$

where η is the pre-specified learning rate (i.e., the rate to adjust $\mathbf{w}$), and w_{jk} is the weight that feedforward the neuron j of the previous layer to neuron k of the inspected layer. The weight w_{jk} therefore, can be adjusted by

$$w_{jk} \leftarrow w_{jk} + \Delta w_{jk}, \quad (5.35)$$

where $\Delta w_{jk} = \delta_j y_k$. This backpropagation procedure is repeated for all neurons in all layers until it reaches the input layer.

5.4.3 Fisher's Discriminant Analysis

Linear Discriminant Analysis:

Linear discriminant analysis (LDA) is a low-complexity classification method by searching for an optimum linear combination of features. LDA is strongly related to logistic regression (LgR) and principle component analysis (PCA) [216]. However, LDA is different than LgR or PCA in the manner that LDA is a supervised method that aims to model the difference between data classes given a priori the label for each independent variable (i.e., each data sample is regarded as an independent variable), while PCA is an unsupervised approach which does not require the labels and only attempts to find the features based on differences in the features themselves, and while LgR only attempts to model each class independently (hence LgR is not desirable for separation).

LDA assumes that the conditional probability density function (pdf) of both classes are normally distributed

$$\begin{aligned} p(\mathbf{x}|y=0) &\sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0), \\ p(\mathbf{x}|y=1) &\sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1). \end{aligned} \quad (5.36)$$

where $\boldsymbol{\mu}_0, \boldsymbol{\mu}_1$ are the mean vectors, and $\boldsymbol{\Sigma}_0, \boldsymbol{\Sigma}_1$ are the covariance matrices of the two classes. LDA predicts a sample $\mathbf{x}$ categorized into the second class if the log of the ratio of likelihood is larger than a threshold ϵ

$$(\mathbf{x} - \boldsymbol{\mu}_0)^\top \boldsymbol{\Sigma}_0^{-1} (\mathbf{x} - \boldsymbol{\mu}_0) + \ln |\boldsymbol{\Sigma}_0| - (\mathbf{x} - \boldsymbol{\mu}_1)^\top \boldsymbol{\Sigma}_1^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) - \ln |\boldsymbol{\Sigma}_1| > \epsilon. \quad (5.37)$$

Furthermore, LDA also makes a homoscedasticity assumption that both covariance of two classes are identical, i.e., $\boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}$, and have full rank. As a result, we obtain $\mathbf{x}^\top \boldsymbol{\Sigma}_0^{-1} \mathbf{x} = \mathbf{x}^\top \boldsymbol{\Sigma}_1^{-1} \mathbf{x}$ and $\mathbf{x}^\top \boldsymbol{\Sigma}_i^{-1} \boldsymbol{\mu}_i = \boldsymbol{\mu}_i^\top \boldsymbol{\Sigma}_i^{-1} \mathbf{x}$ ($i = 0, 1$).

Replace these expressions into 5.37, it holds

$$\mathbf{w} \cdot \mathbf{x} > c, \quad (5.38)$$

where $\mathbf{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$ and $c = \frac{1}{2}(\epsilon - \boldsymbol{\mu}_0^\top \Sigma_0^{-1} \boldsymbol{\mu}_0 + \boldsymbol{\mu}_1^\top \Sigma_1^{-1} \boldsymbol{\mu}_1)$. This implies that the prediction function is a linear combination of all features of the input sample. In other words, the condition of an input sample $\mathbf{x}$ being categorized into class y is a projection of $\mathbf{x}$ onto $\mathbf{w}$, translated by the threshold c . This expression is similar to the SVM hyperplanes in 5.18. The main difference of SVM and LDA is that, LDA provides a lightweight and statistical approach that take into account all training samples into its solution. In other words, in LDA any single training sample has an influence in the prediction, while it is not the case in SVM if that sample is not a support vector.

Fisher's Discriminant Analysis:

In this thesis, we indeed employ the Fisher's Discriminant Analysis (FDA), which is a more generalized approach as compared to LDA. The difference of FDA is that it does not make any assumption on the distribution and identical covariances. FDA attempts to maximize the ratio of the variance between classes and the variance within classes

$$S = \frac{\sigma_{between}^2}{\sigma_{within}^2} = \frac{(\mathbf{w} \cdot \boldsymbol{\mu}_1 - \mathbf{w} \cdot \boldsymbol{\mu}_0)^2}{\mathbf{w}^\top \Sigma_1 \mathbf{w} + \mathbf{w}^\top \Sigma_0 \mathbf{w}}. \quad (5.39)$$

Intuitively, S is equivalent to the signal-to-noise ratio for the class labels. The optimum solution $\mathbf{w}$ is proved [217] to be

$$\mathbf{w} \propto (\Sigma_0 + \Sigma_1)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0), \quad (5.40)$$

where $\propto$ is the notation of “proportional to”. In other words, $\mathbf{w}$ is the vector representing the separating hyperplane, i.e., $\mathbf{w}$ is perpendicular to the geometric discriminant hyperplane of the two classes. Under the assumption that the distributions of both classes are identical, the threshold parameter c in $\mathbf{w} \cdot \mathbf{x} > c$ is shown [217] to be

$$c = \mathbf{w} \cdot \frac{1}{2}(\boldsymbol{\mu}_0 + \boldsymbol{\mu}_1) = \frac{1}{2}\boldsymbol{\mu}_1^\top \Sigma_1^{-1} \boldsymbol{\mu}_1 - \frac{1}{2}\boldsymbol{\mu}_0^\top \Sigma_0^{-1} \boldsymbol{\mu}_0. \quad (5.41)$$

5.4.4 Hyperparameters Selection for Base Learners

An important step that one cannot skip before training any supervised algorithm, is to search for the optimal hyperparameters. In this thesis, we

perform hyperparameters searching for NN and SVM, since FDA does not have any influential hyperparameter. In each task, only two most important hyperparameters are chosen to be tuned. For SVM, those are the generalisation coefficient C and the kernel type (linear or rbf). For NN, those are the regularisation parameter α and the activation function (relu and logistic). The specified tuning values (for C and α) that hyperparameters are sequentially assigned to are .001, .002, .005, .01, .02, .05, .1, .2, .5, 1, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, and 5000. For each base learner after tuning, only the best performing hyperparameter is selected for the subsequent ensemble method, which is shown in Table 5.1.

Algorithm 5 Grid Search for Optimal Hyperparameters

```

1: Input: Training set  $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ ,
2:       Number of subsets  $K$ ,
3:       Base learner algorithm  $F$ ,
4:       Set of possible hyperparameters  $\Omega$ .
5: Draw  $K$  subsets with replacement from  $\mathcal{L}$ , denoted by  $\mathcal{L}_k$ 
6: for each  $\omega \in \Omega$  do
7:   Initialize out-of-fold prediction error  $e_\omega \leftarrow 0$ 
8:   for  $k = 1, \dots, K$  do
9:      $\mathcal{L}_{\text{validation}} \leftarrow \mathcal{L}_k$ 
10:     $\mathcal{L}_{\text{train}} \leftarrow \bigcup_{i \neq k} \mathcal{L}_i$ 
11:     $f_k \leftarrow F(\mathcal{L}_{\text{train}}, \omega)$ 
12:    for each  $(\mathbf{x}_i, y_i) \in \mathcal{L}_{\text{validation}}$  do
13:       $e_\omega \leftarrow e_\omega + (y_i - f_k(\mathbf{x}_i))^2$ 
14:    end for
15:  end for
16: end for
17: Output: optimal hyperparameters  $\omega^* = \operatorname{argmin}_\omega e_\omega$ .
```

5.5 Meta Combiner: Linear Regression

In this thesis, the method opted to combine OOF results from the base learners is linear regression (LR). The strength of linear regression in ensemble strategy is demonstrated as it won recent machine learning competitions [36]. Linear regression, when applied on the second or higher level of stacking, provides the improved accuracy by decreasing the variance of error, and guaranteed to be insusceptible to overfitting [199].

LR assumes that the scalar target label y of each corresponding vector $\mathbf{x}$

is a linear combination of all elements in $\mathbf{x}$

$$y = \mathbf{x}^\top \mathbf{w} = x_1 w_1 + x_2 w_2 + \cdots + x_p w_p. \quad (5.42)$$

Given the training dataset $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$, the LR model equivalent to (5.42) for all points in $\mathcal{L}$ can be written as

$$\mathbf{y} = \mathbf{X} \mathbf{w}, \quad (5.43)$$

where $\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \ \mathbf{x}_m]^\top$ and $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_m]^\top$. Practically, the linear system in (5.43) is overdetermined, i.e., the number of data samples is larger than the dimension of $\mathbf{x}_i$, or simply $m > p$. In other words, there are more equations than variables, hence there exists no solution. The goal of LR is therefore to find the linear combination coefficients $\mathbf{w}$ that produce the lowest the mean squared error between each target label y_i and its corresponding LR prediction $\mathbf{x}_i^\top \mathbf{w}$ as

$$\begin{aligned} \hat{\mathbf{w}} &= \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^m (y_i - \mathbf{x}_i^\top \mathbf{w})^2 \\ &= \underset{\mathbf{w}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{X} \mathbf{w}\|^2. \end{aligned} \quad (5.44)$$

To yield the solution of the optimisation problem in (5.44), we solve for where its gradient is $\mathbf{0}$

$$\begin{aligned} &\nabla_{\mathbf{w}} \|\mathbf{y} - \mathbf{X} \mathbf{w}\|^2 \\ &= \nabla_{\mathbf{w}} (\mathbf{y} - \mathbf{X} \mathbf{w})^\top (\mathbf{y} - \mathbf{X} \mathbf{w}) \\ &= \nabla_{\mathbf{w}} (\mathbf{w}^\top \mathbf{X}^\top \mathbf{X} \mathbf{w} - 2 \mathbf{w}^\top \mathbf{X}^\top \mathbf{y} + \mathbf{y}^\top \mathbf{y}) \\ &= 2 \mathbf{X}^\top \mathbf{X} \mathbf{w} - 2 \mathbf{X}^\top \mathbf{y} = \mathbf{0}, \end{aligned} \quad (5.45)$$

which results in

$$\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}. \quad (5.46)$$

5.6 Performance Evaluation

Table 5.2 presents the results of all base learners and ensemble methods. We will go through each scheme's results of each base learner, then discuss the influence of bagging and stacking on each scheme.

For dataset B, SVM and NN output high scores on precision, recall, and F-score (above 90%), while FDA is worse by a very narrow gap on only the

Algorithm 6 The Complete Ensemble Algorithm

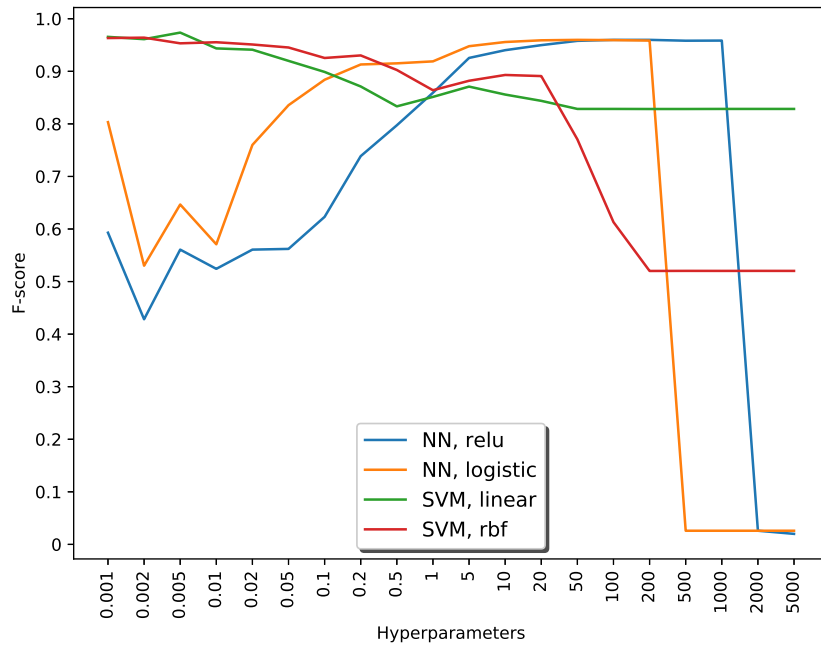
```

1: Input: Training set  $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ ,
2:   Training algorithms:  $F_{\text{SVM}}, F_{\text{FDA}}, F_{\text{NN}}, F_{\text{LR}}$ ,
3:   Hyperparameters  $\omega_{\text{SVM}}^*, \omega_{\text{FDA}}^*, \omega_{\text{NN}}^*$  (from Algorithm 5),
4:   Number of subsets  $K$ .
5: Draw  $K$  non-overlapping subsets (folds) from  $\mathcal{L}$ , denoted by  $\mathcal{L}_k$ 
6: Construct  $K$  corresponding out-of-fold sets  $\bar{\mathcal{L}}_k = \cup_{i \neq k} \mathcal{L}_i$ 
7: Initialize meta data set  $\mathcal{L}_{\text{stack}} \leftarrow \emptyset$ 
8: for  $k = 1, \dots, K$  do
9:    $f_{\text{SVM},k} \leftarrow F_{\text{SVM}}(\bar{\mathcal{L}}_k, \omega_{\text{SVM}}^*)$ 
10:   $f_{\text{FDA},k} \leftarrow F_{\text{FDA}}(\bar{\mathcal{L}}_k, \omega_{\text{FDA}}^*)$ 
11:   $f_{\text{NN},k} \leftarrow F_{\text{NN}}(\bar{\mathcal{L}}_k, \omega_{\text{NN}}^*)$ 
12:  for each  $(\mathbf{x}_i, y_i) \in \mathcal{L}_k$  do
13:     $\mathcal{L}_{\text{stack}} \leftarrow \mathcal{L}_{\text{stack}} \cup \{[f_{\text{SVM},k}(\mathbf{x}_i), f_{\text{FDA},k}(\mathbf{x}_i), f_{\text{NN},k}(\mathbf{x}_i)], y_i\}$ 
14:  end for
15: end for
16: Train meta model  $f_{\text{meta}} = F_{\text{LR}}(\mathcal{L}_{\text{stack}})$ 
17:  $f_{\text{SVM}} \leftarrow \frac{1}{K} \sum_{k=1}^K f_{\text{SVM},k}$ 
18:  $f_{\text{FDA}} \leftarrow \frac{1}{K} \sum_{k=1}^K f_{\text{FDA},k}$ 
19:  $f_{\text{NN}} \leftarrow \frac{1}{K} \sum_{k=1}^K f_{\text{NN},k}$ 
20: Output: Model  $f_{\text{ensemble}}(\mathbf{x}) = f_{\text{meta}}([f_{\text{SVM}}(\mathbf{x}), f_{\text{FDA}}(\mathbf{x}), f_{\text{NN}}(\mathbf{x})])$ .

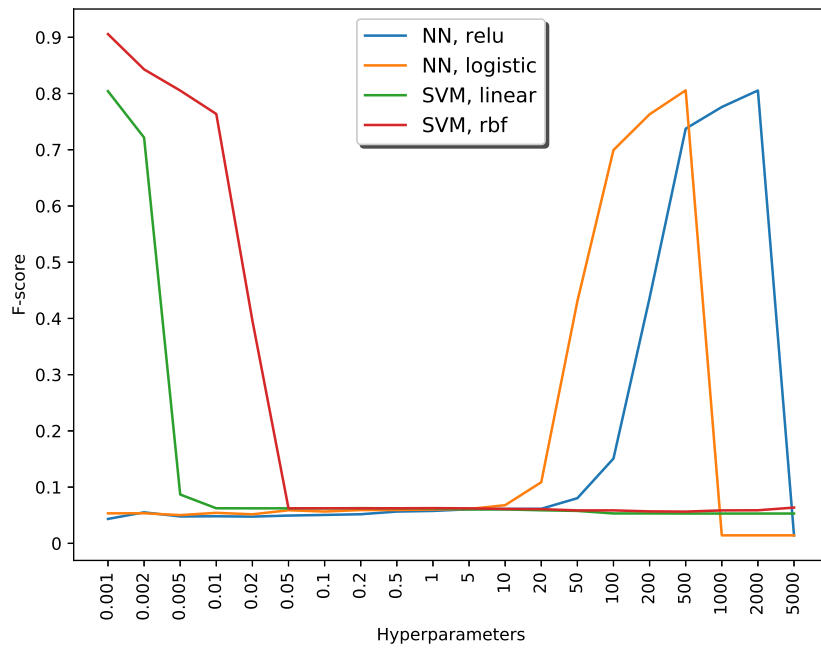
```

precision. Bagging works for SVM, since SVM is equivalent to filtering suspicious samples on different boundaries by learning on different combinations of training folds, hence reducing error variance. Bagging, however, performs only slightly better in NN, but worse in FDA, since it reduces the number of training samples, resulting in reducing probabilistic properties of FDA. The most important observation in dataset B is that it is obviously seen that all base learners perform at considerably equivalent performance. As a result the final stacking ensemble, based on 3 bagging versions of methods, perform significantly better than any single model. Precision and F-score of the best bagging model (SVM-bag) is boosted from 95.46% to 96.97%, and from 97.55% to 98.18%, respectively.

For dataset A, each base learner performs totally different. F-score measurements for SVM, NN, and FDA are 90.67%, 80.51%, and 47.78% respectively. SVM is still the best in terms of all metrics. The significant inferiority of FDA on dataset A causes the stacking performance to be a little bit worse than the best base learner (SVM). Since the performance gap between base models is large, F-score decreased from 92.63% to 91.72% as compared to the



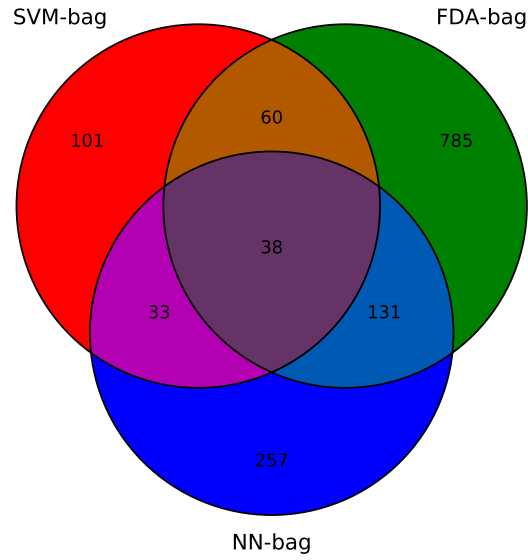
(a) Dataset A



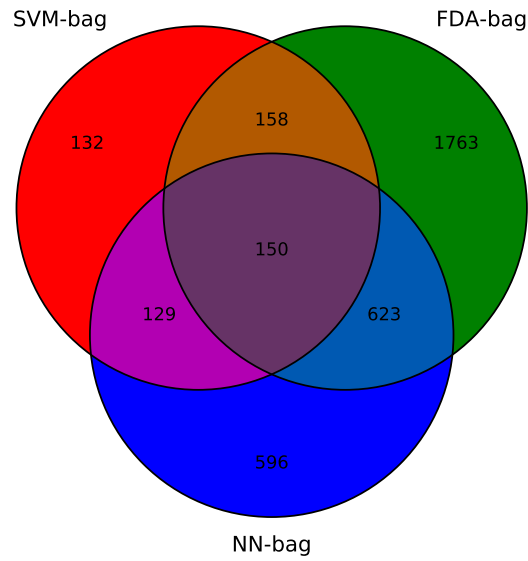
(b) Dataset B

Figure 5.2: The Hyperparameter Tuning Performance on two datasets

best performing model (SVM-bag). Similarly, the number of false counts when combining FDA-bag with SVM-bag, or with NN-bag increases due to the low performance of FDA, as depicted in Figure 5.3. However, this effect is negli-



(a) Dataset A



(b) Dataset B

Figure 5.3: The number of False Counts of Various Ensemble Combinations across 3 Base Learners.

gible. This discussion, which emphasizes on the relationship between stacking performance and the performance gap of base learners, is perfectly matched with what we have presented in the theoretical section.

TABLE 5.1: HYPERPARAMETERS GRID SEARCH FOR BASE LEARNERS.

Base learner	Dataset	Best Hyperparameter	F-score
SVM	A	RBF kernel with $C = 0.001$	90.67
SVM	B	Linear kernel with $C = 0.005$	97.23
NN	A	Relu activation with $\alpha = 2000$	80.51
NN	B	Relu activation with $\alpha = 100$	96.08

TABLE 5.2: THE CLASSIFICATION PERFORMANCE OF BAGGING AND STACKING.

	Dataset A			Dataset B		
	<i>pr</i>	<i>re</i>	<i>fs</i>	<i>pr</i>	<i>re</i>	<i>fs</i>
SVM	85.84	96.08	90.67	94.85	99.73	97.23
SVM-bag	99.25	86.84	92.63	95.46	99.73	97.55
NN	69.08	96.46	80.51	92.80	99.59	96.08
NN-bag	70.56	96.46	81.50	92.75	99.57	96.04
FDA	31.61	97.80	47.78	84.46	99.69	91.44
FDA-bag	31.30	97.69	47.41	84.27	99.69	91.33
Stack & Bag	99.28	85.23	91.72	96.97	99.43	98.18

5.7 Chapter Summary

In this section, we have presented the following key aspects of our proposed ensemble learning strategy

- The description of bagging ensemble, algorithms of bagging, and the proof demonstrating that bagging can reduce generalisation errors.
- The description of stacking ensemble, algorithms of stacking, and the proof to show that stacked learner can outperform individual base learners if they are comparable to each other.
- The strategy to combine bagging and stacking into a unified ensemble framework to avoid overfitting.
- The brief theoretical review of three base learners employed in the ensemble, namely SVM, NN, and FDA.
- A simple grid search scheme to search for optimal hyperparameters for each base learner.
- Linear regression as the meta combiner that gets as inputs the individual soft scores of all base learners.

- The practical experiments to show the effectiveness of ensemble in P3S data.

In the next chapter, we will move to the final two major techniques that can massively enhance the information transfer rate, i.e., speed, of our proposed BCI: dynamic stopping and incremental (adaptive) learning. These two steps directly use the final ensemble soft scores that are the outputs of this current chapter.

Chapter 6

DYNAMIC STOPPING AND ADAPTIVE LEARNING

In this chapter, we will describe the final two major techniques employed after retrieving the ensemble’s score as presented in the last chapter. First, section 6.1 describes the dynamic stopping (DS) method, which is based on two proposed probabilistic parameters. The goal of DS is to dynamically stop the P3S flashing paradigm whenever the DS algorithm is sufficiently confident about its output decision. Then, section 6.2 presents the adaptive (or incremental) learning (AL) method using the incremental SVM (ISVM) algorithm to integrate the newly-classified test samples into the SVM classifier solution. The DS and AL methods must be employed together as a unified framework. The practical experiments are presented in section 6.3, including the dataset review, classification schemes, DS parameters grid search result, classification performance, discussion of the influence of DS and AL, and comparisons of our method with other studies. Finally, the chapter is summarised in section 6.4.

6.1 Dynamic Stopping

During each spelling session, the desired letter can be correctly elicited by using only a few first iterations when the classifiers’ score is sufficiently confident about its decision, hence dramatically reduce the spelling time. The confidence level checking process is implemented continually after each iteration, and is almost based on the Bayesian updating scheme [10, 11, 21–23, 218]. However, since those schemes do not work with our adaptive SVM, we propose a method which based on the SVM scores. Our preliminary result had been

reported in [219]. In this paper, we make some modifications to the criteria in [219] to better suits the real-time adaptive SVM. As compared to [219], the DS criteria in this report are less computational. This is the desirable aspect for a real-time system. Moreover, we also propose an analytical grid search algorithm to obtain the optimal DS parameters. With this algorithm one can specify the DS thresholds which can yield pre-defined accuracies on purpose.

We attempt to embed a bagging strategy into the DS scheme to boost up the accuracy. The bagging theory is described in details in Chapter 5. It should be noted that the bagging ensemble applied to DS in this chapter is completely different and independent to the bagging ensemble of the main classification algorithm described in Chapter 5. The difference between them is that, this section employs bagging ensemble as an extra tool to enhance the postprocessing steps (DS and AL) performance on the SVM algorithm, while the ensemble mentioned in Chapter 5 refers to the main classification procedure conducted on all algorithms (SVM, NN, FDA). To let the reader understand the ensemble in this chapter better, we will describe in details as follows.

6.1.1 Dynamic Stopping Framework

For reading convenience, in this section we will re-describe the flashing paradigm of the P3S with extra steps related to DS. An example of the row-column flashing paradigm for the P3S used in our work is presented in Figure 6.1. As shown in Figure 6.1 The subject is focusing on the letter ‘K’ of the P3S, hence the target row is row 2 and the target column is column 11 of the P3S matrix. The main signal stream is depicted by sequential flashes of numbered rows and columns (from 1 to 12 in shuffled order), interleaved with unflashed intervals (numbered 0). In any iteration, the post-stimulus feature vectors associated to row 2 and column 11 contain the P3 responses. Specifically, the target flashes are shaded blue in the stream of Iteration 1, and their corresponding feature vectors are denoted by $\mathbf{x}_8$ and $\mathbf{x}_2$. These are ERPs occurs approximately 300 ms (P300 delay) after the target stimuli onset.

The diagram of our DS framework is shown in Figure 6.2. As shown in Fig-

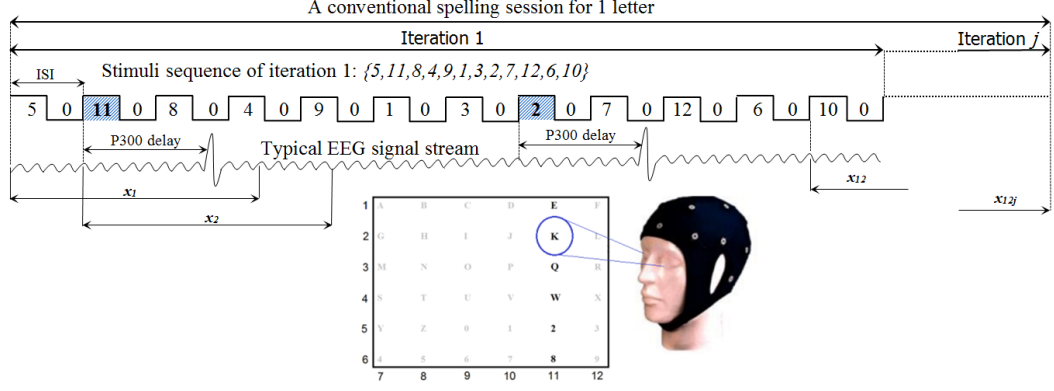


Figure 6.1: The EEG signal timeline and row-column flashing paradigm.

Figure 6.2, the training (or learning) stage consists of the following steps: (T1) The training data is divided into P partitions, denoted by T_p (as in (6.1)). (T2) The partitioned data is used to construct P ensemble classifiers. (T3, T4) The remaining unused learning data is fed as input for the grid search algorithm (as presented in Algorithm 7) to search for DS parameters $\tilde{\theta}_1$ and $\tilde{\theta}_2$. (T5) The resulted parameters are used as the dynamic stopping criteria (as presented in Section 6.1) for the validation stage.

The validation (or testing) stage consists of the following steps: (V1) The test data is collected through the row-column P3S flashing paradigm after each iteration (which equals to 12 trial vectors). (V2) The preprocessed vectors (as described in Chapter 3) are scored by the ensemble scheme, as in (6.5) and (6.6). (V3) The resulting scores are used to compute the dynamic stopping variable θ_1 and θ_2 , to perform the DS check, as in (6.8). If the DS criteria are satisfied, the current letter session is stopped and the output is shown. (V4) If the criteria are not satisfied, an adaptive learning stage is implemented (as presented in section 6.2), and the next flashing iteration is performed.

6.1.2 Dataset Partitioning

We denote P training sets of P different subjects as

$$T_p = \left\{ (\mathbf{x}_{p,i}, y_{p,i}) \in \mathbb{R}^D \times \{-1, 1\} \right\}_{i=1}^N, \quad (6.1)$$

where $p \in \{1, \dots, P\}$, $\mathbf{x}_{p,i}$ are the preprocessed training feature vectors of length D , $y_{p,i}$ are the binary labels or classes, and N is the number of training

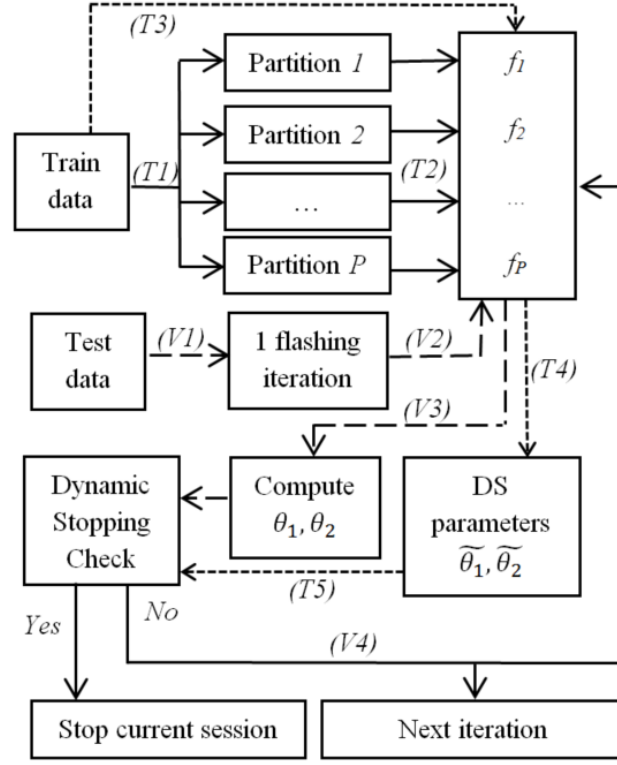


Figure 6.2: The main diagram of our proposed dynamic stopping framework for a one-letter P3S session.

samples (of both classes) in each training set. Hereinafter we call each training set T_p a *partition*. In this thesis, we select SVM as the baseline learning algorithm to apply DS and AL. SVM had been employed with good results in various BCI studies [39, 40, 220, 221]. The crucial reason of choosing SVM is that we want to exploit its lightweight computational requirement for DS, and its availability of the incremental toolbox (ISVM) [222, 223] for AL. The integration of DS and AL can be viewed as postprocessing stages, and can significantly improve the performance as compared to other methods, which will be presented in section 6.3.7.

Given P training partitions denoted by T_p , the SVM classifier solution (α_p, b_p) for partition p is obtained by solving a quadratic convex constrained optimization problem in the duality space [212]. We have outlined the SVM method more detailedly in section 5.4. For a new vector $\mathbf{x}$, its *classifier score*

generated by partition T_p is computed by

$$f_p(\mathbf{x}) = \sum_{i \in T_p} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b. \quad (6.2)$$

where $K(\cdot, \cdot)$ is the homogeneous polynomial kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$. We employ the quadratic solving program provided by [224]. Since C , the SVM penalty parameter, can be freely chosen and can heavily affect the classification performance, we perform the OOF cross validation process (as described in Chapter 5) on P classifiers to determine the optimum C_p for each partition to yield the best classification performance on the ensemble training set. The possible values for C_p are 0, 0.01, 0.02, 0.05, 0.1, 0.25, 0.5, 0.75, and 1.

Let the set of 12 preprocessed feature vectors of length D , belong to the j -th iteration, denoted by

$$X^{(j)} = \{\mathbf{x}_{12(j-1)+1}, \mathbf{x}_{12(j-1)+2}, \dots, \mathbf{x}_{12j}\}. \quad (6.3)$$

For each set $X^{(j)}$ we have the associated set of flash codes

$$L^{(j)} = \{\ell_{12(j-1)+1}, \ell_{12(j-1)+2}, \dots, \ell_{12j}\}, \quad (6.4)$$

where $\ell_i \in \{1, 2, \dots, 12\}$ are the row/column labels as depicted in Figure 6.1. We define the sum (on all p partitions) and accumulated (after iteration j) row/column scores associated with each flash code as

$$s_r^{\text{row}(j)} = \sum_{p=1}^P \sum_{\substack{i'=1..12j \\ \ell_{i'}=r}} f_p(\mathbf{x}_{i'}), \quad (6.5)$$

$$s_c^{\text{col}(j)} = \sum_{p=1}^P \sum_{\substack{i'=1..12j \\ \ell_{i'}=c+6}} f_p(\mathbf{x}_{i'}). \quad (6.6)$$

where $r, c \in \{1, 2, \dots, 6\}$. After each iteration, a candidate pair of row/column which yields the highest scores is derived to determine the letter output:

$$\hat{r}, \hat{c} = \underset{r, c}{\operatorname{argmax}} \mathbf{M}_{rc}. \quad (6.7)$$

where $\mathbf{M}_{rc} = \mathbf{M}_{rc}^{(j)} = s_r^{\text{row}(j)} + s_c^{\text{col}(j)}$ ($r, c \in \{1, 2, \dots, 6\}$) is the score matrix of all P3S letters. Unless stated, hereinafter we refer s_r^{row} as $s_r^{\text{row}(j)}$, and s_c^{col} as $s_c^{\text{col}(j)}$ for notational convenience.

6.1.3 Proposed Criteria for Dynamic Stopping

Our framework has the aspect that the final letter output is decided based on every individual post-stimulus trial's score $f_p(\mathbf{x})$ of a specific preprocessed trial vector $\mathbf{x}$. We exploit this aspect, as well as the computational strength, to develop the real-time stopping criteria. Instead of estimating the output only after a fixed iteration, we propose a criteria checking method to dynamically stop the flashing at any iteration, once the classifier is sufficiently confident about its decision. We propose two criteria, one based on the classifiers' scores and one based on the classifiers' votes. Our DS method thereby can be presented as follows: The flashing of a single letter spelling session is terminated after a specific iteration once both of the following criteria are met

$$\begin{cases} \theta_1 \geq \tilde{\theta}_1 \\ \theta_2 \geq \tilde{\theta}_2 \end{cases}. \quad (6.8)$$

6.1.3.1 Criterion 1

Let $\Delta s^{\text{row}(j)}$ and $\Delta s^{\text{col}(j)}$ denote the margin between the highest and the second-highest scores of all rows/columns after iteration j as

$$\Delta s^{\text{row}(j)} = s_{\hat{r}}^{\text{row}} - \max_{r \neq \hat{r}}(s_r^{\text{row}}), \quad (6.9)$$

$$\Delta s^{\text{col}(j)} = s_{\hat{c}}^{\text{col}} - \max_{c \neq \hat{c}}(s_c^{\text{col}}). \quad (6.10)$$

We define θ_1 as the parameter for qualifying criterion 1 as

$$\theta_1 = \frac{\Delta s^{\text{row}} + \Delta s^{\text{col}}}{1/6 \times (\sum_r \hat{s}_r^{\text{row}} + \sum_c \hat{s}_c^{\text{col}})}, \quad (6.11)$$

where

$$\hat{s}_r^{\text{row}} = s_r^{\text{row}} - \min_{r'}(s_{r'}^{\text{row}}), \quad (6.12)$$

$$\hat{s}_c^{\text{col}} = s_c^{\text{col}} - \min_{c'}(s_{c'}^{\text{col}}). \quad (6.13)$$

The intuitive idea behind criterion 1 is that the ensemble accumulated margin between two highest-scored row/column candidates represents the confidence level of the classifier about its decision.

Lemma 6.1. *The expected value of θ_1 increases with respect to j .*

Proof. Lemma 1 demonstrates the validity of criterion 1, as the confidence of the classifier will increase with respect to the number of iteration j . We can easily observe that the denominator in (6.11) is the mean of scaled row/column scores and stays as a constant with respect to j . With regards to the nominator, we denote $\Delta s^{(j)}$ (where $\Delta s^{(j)}$ can stand for either $\Delta s^{\text{row}(j)}$ or $\Delta s^{\text{col}(j)}$) as the subtraction between the averaged accumulated score of the target row (or column) response, $S^{+(j)}$, and the maximum score among 5 averaged accumulated scores of 5 non-target responses, $S_k^{-(j)}$, ($k = 1 \dots 5$), after j iterations. Here, the index k of 5 non-target responses is not necessarily relevant to the non-target row/column numbering on the P3S, since we only concern about the probabilistic properties of the general target and non-target scores. $S^{+(j)}$ and $S_k^{-(j)}$ can be decomposed into random samples s_i^+ and s_i^- of size j ($i = 1 \dots j$), respectively, drawn from some unknown distribution with the expected values given by μ^+, μ^- and variances given by σ^+, σ^- , respectively. According to the central limit theorem, regardless of the distribution of s_i^+ and s_i^- , their sum, $S^{+(j)}$ and $S_k^{-(j)}$, tend towards a Gaussian distribution

$$S^{+(j)} \sim \mathcal{N}(\mu^{+(j)}, \sigma^{+(j)}); \quad S_k^{-(j)} \sim \mathcal{N}(\mu^{-(j)}, \sigma^{-(j)}). \quad (6.14)$$

As a result we can write the expected value of Δs as

$$\mathbb{E}[\Delta s^{(j)}] = \mathbb{E}[S^{+(j)} - \max_k S_k^{-(j)}] = \mu^+ - \mathbb{E}[\max_k S_k^{-(j)}]. \quad (6.15)$$

The variances of $S^{+(j)}$ and $S_k^{-(j)}$ decreases when their sample size of them (j) increase as

$$\begin{aligned} (\sigma^{+(j)})^2 &= \text{Var}\left[\frac{1}{j} \sum_{i=1}^j s_i^+\right] = \frac{1}{j^2} \sum_{i=1}^j \text{Var}[s_i^+] = \frac{(\sigma^+)^2}{j}, \\ (\sigma^{-(j)})^2 &= \text{Var}\left[\frac{1}{j} \sum_{i=1}^j s_i^-\right] = \frac{1}{j^2} \sum_{i=1}^j \text{Var}[s_i^-] = \frac{(\sigma^-)^2}{j}. \end{aligned}$$

The effect of the reduction in the variance of $S^{+(j)}$ does not affect $\Delta s^{(j)}$ as its mean is independent from its variance ($\mu^{+(j)} = \mu^+, \mu^{-(j)} = \mu^-$). However $S_k^{-(j)}$ does affect the mean of $\max_k S_k^{-(j)}$ by its variance reduction. As in [225], $\max_k S_k^{-(j)}$ is Gumbel-distributed with its probability density function given by

$$f_{\max S_k^{-(j)}}(x) = \frac{1}{\sigma^{-(j)}} \exp\left(\frac{x - \mu^{-(j)}}{\sigma^{-(j)}}\right) \exp\left(-\exp\left(\frac{x - \mu^{-(j)}}{\sigma^{-(j)}}\right)\right).$$

The corresponding expected value of $\max_k S_k^{-(j)}$ is

$$\mathbb{E}[\max_k S_k^{-(j)}] = \mu^{-(j)} + \gamma \sigma^{-(j)} = \mu^- + \gamma \frac{(\sigma^-)}{\sqrt{j}}, \quad (6.16)$$

where $\gamma \simeq 0.5772$ is the Euler-Mascheroni constant. With regards to (6.16), since μ^- , σ^- , and γ are all constants, then $\mathbb{E}[\max_k S_k^{-(j)}]$ decreases with respect to j . As a result, $\mathbb{E}[\Delta s^{(j)}]$ increases, hence $\mathbb{E}[\theta_1]$ also increases with respect to j . ■

6.1.3.2 Criterion 2

We define the accumulated score of row r or column c , which is validated on partition p after j iterations as

$$s_{r,p}^{\text{row}(j)} = \sum_{\substack{i'=1..12j \\ \ell_{i'}=r}} f_p(\mathbf{x}_{i'}), \quad (6.17)$$

$$s_{c,p}^{\text{col}(j)} = \sum_{\substack{i'=1..12j \\ \ell_{i'}=c+6}} f_p(\mathbf{x}_{i'}). \quad (6.18)$$

where $r, c \in \{1, 2, \dots, 6\}$. Let $V_{rc}^{(j)}$ denote the voting count for the letter on row r and column c as

$$V_{rc}^{(j)} = \sum_p v_p^{(j)}(r, c) \quad (6.19)$$

where $v_p^{(j)}$ is the voting function of classifier p

$$v_p^{(j)}(r, c) = \begin{cases} 1, & \text{if } r, c = \underset{r', c'}{\operatorname{argmax}} (s_{r', p}^{\text{row}(j)} + s_{c', p}^{\text{col}(j)}) \\ 0, & \text{otherwise} \end{cases}. \quad (6.20)$$

Criterion 2 is characterized by the parameter θ_2 which is defined as

$$\theta_2 = \frac{\max_{r, c} V_{rc}^{(j)}}{P}. \quad (6.21)$$

Intuitively, criterion parameter θ_2 represents the number of constituent classifiers voting for the same output letter, which is also increasing with respect to j as similar to θ_1 .

Lemma 6.2. *For any $n \in \{1, \dots, P-1\}$, the probability $\mathcal{P}(\theta_2 \geq n)$ increases with respect to j , and $\lim_{j \rightarrow \infty} \theta_2 = 1$.*

Proof. Let r^* and c^* denote the true target row and column, then it can be observed that $v_p^{(j)}(r^*, c^*)$ is Bernoulli-distributed with probability ρ_p for value 1 and $1 - \rho_p$ for value 0, where ρ_p is the probability of correct output letter, produced by classifier f_p , validated on one iteration. As a result, the sum of $v_p^{(j)}(r^*, c^*)$ over P classifiers, $V_{r^*c^*}^{(j)}$, has the Poisson binomial distribution with the expectation given by

$$\mathbb{E}[V_{r^*c^*}^{(j)}] = \sum_{p=1}^P \rho_p^{(j)}, \quad (6.22)$$

where $\rho_p^{(j)}$ is the accumulated probability of the correct output letter after j iterations.

As $\lim_{j \rightarrow \infty} \rho_p^{(j)} = 1 \forall p$ when $r = r^*, c = c^*$. Then $V_{r^*c^*}^{(j)}$ converges to P over j as

$$\lim_{j \rightarrow \infty} \mathbb{E}[V_{r^*c^*}^{(j)}] = \lim_{\rho_p^{(j)} \rightarrow 1} \sum_{p=1}^P \rho_p^{(j)} = P. \blacksquare \quad (6.23)$$

6.1.4 Estimation $\tilde{\theta}_1$ and $\tilde{\theta}_2$ through dynamic training

We also perform an estimation for $\tilde{\theta}_1$ and $\tilde{\theta}_2$ based on the learning sets T_p . Our aim is to provide a reliable method to obtain the criteria parameters which satisfy our specific desirable accuracy. The parameters are obtained through an extensive grid search analysis with respect to the changes in $\tilde{\theta}_1$ and $\tilde{\theta}_2$.

Let $A_p(\hat{\theta}_1, \hat{\theta}_2)$ denote the accuracy when performing the validation stage on set T_p with a pair of parameters $\tilde{\theta}_1 = \hat{\theta}_1$ and $\tilde{\theta}_2 = \hat{\theta}_2$. Let $S(a)$ denote the set of parameter pairs which can attain the optimal accuracy decreased by a margin of $a\%$. The set $S(a)$ can be presented as

$$S(a) = \{(\hat{\theta}_1, \hat{\theta}_2) | A(\hat{\theta}_1, \hat{\theta}_2) \geq \max_{\substack{\theta'_1 \in R_1 \\ \theta'_2 \in R_2}} A(\theta'_1, \theta'_2) - a\}, \quad (6.24)$$

where

$$A(\hat{\theta}_1, \hat{\theta}_2) = \frac{1}{P} \sum_p A_p(\hat{\theta}_1, \hat{\theta}_2). \quad (6.25)$$

The tuning range of $\hat{\theta}_2$, denoted by R_2 , drops within the interval $[0, \dots, 1]$, as

$$\max_{i,j} \theta_2 = \sum_{p=1}^P \max_{r,c} v_p^{(j)}(r, c) / P = (\sum_{p=1}^P 1) / P = 1. \quad (6.26)$$

Algorithm 7 Iterative grid search for $\tilde{\theta}_1, \tilde{\theta}_2$

```

1: Input:  $a$ 
2: Output:  $\tilde{\theta}_1, \tilde{\theta}_2$ 
3: Initialization:  $\eta_1, \eta_2$ 
4:  $\hat{\theta}_1 \leftarrow 0, \hat{\theta}_2 \leftarrow 0, A_{max} \leftarrow 0, \text{STOP\_FLAG} \leftarrow 0, S \leftarrow \emptyset$ 
5: while  $\text{STOP\_FLAG} \neq 1$  do
6:   for  $\hat{\theta}_2 = 0 : 1$  (step  $\eta_2$ ) do
7:     Compute and save  $A(\hat{\theta}_1, \hat{\theta}_2)$ 
8:     if  $A(\hat{\theta}_1, \hat{\theta}_2) > A_{max}$  then  $A_{max} \leftarrow A(\hat{\theta}_1, \hat{\theta}_2)$ 
9:     end if
10:  end for
11:  if  $A(\hat{\theta}_1, 0) = A_{max}$  and  $A(\hat{\theta}_1, 0) = A(\hat{\theta}_1 - \eta_1, 0)$  then
12:     $\text{STOP\_FLAG} \leftarrow 1$ ;
13:  end if
14:   $\hat{\theta}_1 \leftarrow \hat{\theta}_1 + \eta_1$ 
15: end while
16: for each  $\hat{\theta}_1, \hat{\theta}_2$  do
17:   if  $A(\hat{\theta}_1, \hat{\theta}_2) \geq A_{max} - a$  then add  $(\hat{\theta}_1, \hat{\theta}_2)$  to  $S$ 
18:   end if
19: end for
20: Search for  $\hat{\theta}_1, \hat{\theta}_2$  in  $S$ 
21: return  $\hat{\theta}_1, \hat{\theta}_2$ 

```

We can also find the tuning range for $\hat{\theta}_1$, denoted by R_1 , by computing $\lim_{j \rightarrow \infty} \theta_1$, but this is unnecessary. Since R_2 is found, for each specific value of $\hat{\theta}_2$ we can perform the search by iteratively increasing $\hat{\theta}_1$ until the maximum accuracy is achieved. The step for iteratively increasing $\hat{\theta}_1$ and $\hat{\theta}_2$, denoted by η_1 and η_2 respectively, is freely chosen (i.e., $\eta_2 = 0.25$, and $\eta_1 = 0.2$).

To obtain the least possible iterations (for fastest performance), a unique corresponding pair of $\tilde{\theta}_1, \tilde{\theta}_2$ can be achieved by

$$\tilde{\theta}_1, \tilde{\theta}_2 = \underset{(\hat{\theta}_1, \hat{\theta}_2) \in S}{\operatorname{argmin}} \{A(\hat{\theta}_1, \hat{\theta}_2)\}. \quad (6.27)$$

The algorithm of the iterative grid search method is presented in Algorithm 7.

6.2 Adaptive Learning using Incremental SVM

6.2.1 Adaptive Learning as a Black Box

Adaptive learning (AL), or also known as incremental learning, is a method of continually incorporating new test samples that have been classified during

the on-line experiments into the existing classifier. This method can make the classifier that was trained on different subjects to get adapted with the ongoing subject's patterns, thereby increase the classification accuracy. The arising problem of AL is that we do not know if the new samples' label are correctly classified, as if we wrongly label them they can negatively affect the existing classifier. As a result, the effects of AL are heavily governed by the performance of the DS method. The influence and relationship of DS and AL are further discussed in Section 6.3.5 and Section 6.3.6.

There have been various approaches on adaptive SVM [222, 223, 226, 227]. We employ the method, called incremental SVM (ISVM) that was inspired by [222] because it was well-developed to integrate new training samples without re-solving the quadratic program over the whole training set. ISVM is an incremental learning method with *discriminative* manner, i.e., analytically tuning the existing classifier using new data sample-by-sample. As compared to the *population-based* methods of the papers in Table 6.3 (mostly used Bayesian approach), ISVM has a significantly better generalization ability and accuracy [228]. especially when adjusting the existing classifier with some few newly-classified samples. However, one drawback of ISVM is the complicated algorithm and time-consuming learning time.

Suppose we need to integrate the set of N_1 newly-classified test trials, denoted by $\mathcal{T}$, into the existing SVM solutions that were learned from p sets (T_p) of N training samples. Cauwenberghs [222] proposed an incremental method by adding new samples from $\mathcal{T}$, one at a time, into the existing solution $\{\alpha_p, b_p\}$, by analytically tuning the solution to retain the KKT conditions on all samples. In short, the method attempts to find the final solution $\{\alpha_{p,\text{new}}, b_{p,\text{new}}\}$ of the set $T_p \cup \mathcal{T}$ that can be expressed in terms of $\{\alpha_p, b_p\}$ and the samples of $\mathcal{T}$ as

$$(\alpha_{p,\text{new}}, b_{p,\text{new}}) = \text{ISVM}(T_p, \alpha_p, b_p, \mathcal{T}). \quad (6.28)$$

In this thesis, we assign new labels for new validation samples of $\mathcal{T}$ based on the letter decision of the whole previous letter session, rather than the individual output of each vector. The benefit of this approach is that once

the letter of the previous session is correctly classified, it will increase the next session's performance. However, a misclassified letter may negatively affect the accuracy of later sessions. This effect will be discussed in Section 6.3.5.

The number of samples in $\mathcal{T}$ may vary from 1 to 15 iterations (corresponding to 12 samples to 180 samples), depending on the DS performance of the previous session. Algorithm 8 presents the detailed updating process after each spelling session. The computational time required for this process will be further discussed in Section 6.3.1.

Although at this stage ISVM can be seen as a black-box algorithm as described above, in the following subsections we will elaborate ISVM method in a more detailed way. We also propose our additional theoretical proofs to bridge some gaps that we confronted during the usage of ISVM in practice.

Algorithm 8 On-line Incremental Learning Update for Continual P3S Sessions

```

1: Initialization:  $k \leftarrow 0, \boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha}_{\mathcal{L}}, b \leftarrow b_{\mathcal{L}}$ 
2: while USER_STOP  $\neq 1$  do
3:   Implement flashing paradigm of the  $k$ -th session
4:   Implement Dynamic Stopping
5:   Obtain  $s, \tilde{i}_s, \tilde{j}_s$  using solution  $(\boldsymbol{\alpha}, b)$ 
6:   Input: USER_STOP (1 for stop)
7:   Assign new labels for  $y_{\mathcal{T},t}$  according to  $\tilde{i}_s, \tilde{j}_s$ 
8:   Add new  $N_2 = s \times 12$  samples to  $\mathcal{T} = \{(\mathbf{x}_{\mathcal{T},t}, y_{\mathcal{T},t})\}_{t=1}^{N_2}$ 
9:   Increment solution  $(\boldsymbol{\alpha}, b) \leftarrow \text{ISVM}(\mathcal{L}, \boldsymbol{\alpha}, b, \mathcal{T})$ 
10: end while

```

6.2.2 Brief Review on SVM

Given an input training dataset $\mathcal{T} = \{\mathbf{X}, \mathbf{y}\}$ consisting of $|\mathcal{T}|$ data samples from 2 classes, denoted by

$$\begin{aligned} \mathbf{X} &= [\mathbf{x}_1, \dots, \mathbf{x}_{|\mathcal{T}|}], \\ \mathbf{y} &= [y_1, \dots, y_{|\mathcal{T}|}]^T. \end{aligned} \tag{6.29}$$

where $\mathbf{x}_i \in \mathcal{R}^{n \times 1}, \forall i \in \{1, \dots, |\mathcal{T}|\}$. Each sample comes with a corresponding scalar label, $y_i = 1$ if $\mathbf{x}_i$ belongs to one class, or $y_i = -1$ if $\mathbf{x}_i$ belongs to the other. A hyperplane with maximum distance to data samples from both

classes, characterized by primal coefficients $(\mathbf{w}, b) \in \mathcal{R}^r \times \mathcal{R}^1$, is achieved by solving the following quadratic optimization problem in the feature space

$$\begin{aligned} \underset{\mathbf{w}, b, \boldsymbol{\xi}}{\text{minimize}} \quad & P = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i^p, \\ \text{subject to} \quad & \mathbf{y} \odot (\mathbf{w}^\top \phi(\mathbf{X}) + b \mathbf{1}_m) \geq \mathbf{1}_m - \boldsymbol{\xi}, \end{aligned} \quad (6.30)$$

where $\phi : \mathcal{R}^n \rightarrow \mathcal{R}^r$ is a mapping into the r -dimensional feature space, $\phi(\mathbf{X}) = [\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_m)]^\top \in \mathcal{R}^{r \times m}$ is the feature matrix, $\boldsymbol{\xi} \in \mathcal{R}^m$ are slack variables for soft-margin SVM [212], $\mathbf{1}_m$ is a m -dimensional vector of ones, and $\odot$ is the element-wise Hadamard product.

Solving (6.30) involves constraints... The associated Lagrangian form of the primal problem can be solved in the dual space for a much simpler constraints as

$$\begin{aligned} \underset{\boldsymbol{\alpha}, b}{\text{maximize}} \quad & D = \frac{1}{2} \boldsymbol{\alpha}^\top \mathbf{Q} \boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \mathbf{1} + b \boldsymbol{\alpha}^\top \mathbf{y}, \\ \text{subject to} \quad & \mathbf{0} \leq \boldsymbol{\alpha} \leq C \mathbf{1}, \end{aligned} \quad (6.31)$$

where $\mathbf{Q}$ is the label-augmented kernel matrix

$$\mathbf{Q} = (\mathbf{y} \mathbf{y}^\top) \odot \mathbf{K} = \begin{bmatrix} y_1 y_1 K_{11} & y_1 y_2 K_{12} & \dots & y_1 y_m K_{1m} \\ y_2 y_1 K_{21} & y_2 y_2 K_{22} & \dots & y_2 y_m K_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ y_m y_1 K_{m1} & y_m y_2 K_{m2} & \dots & y_m y_m K_{mm} \end{bmatrix} \quad (6.32)$$

The dot product in dual space is identical to the positive definite kernel function in the direct space

The classifier discriminant function to predict a sample $\mathbf{x}_{\text{new}}$ is

$$f(\mathbf{x}_{\text{new}}) = K(\mathbf{x}, \mathbf{x}_{\text{new}})^\top (\boldsymbol{\alpha} \odot \mathbf{y}) + b \quad (6.33)$$

6.2.3 ISVM Theory

6.2.3.1 Analytical Integration of New Samples

In this section, we re-formulate the ISVM method proposed in [222]. The purpose of our reformulation is to develop our theoretical proofs for bridging the gaps in ISVM, which will be described in the next section.

The goal of ISVM is to analytically integrate unlearned samples into the existing solution. The principle idea is to maintain the Karush-Kuhn-Tucker (KKT) conditions on all samples by transferring learned samples across 3 SVM vector categories. Trained vectors are assigned into 3 categories, namely support vectors ($\mathcal{S}$), error vectors ($\mathcal{E}$), and reserved vectors ($\mathcal{R}$) which satisfy the KKT conditions

$$g_i = \frac{\partial D}{\partial \alpha_i} = \sum_{j=1}^m Q_{ij} \alpha_j + y_i b - 1 \begin{cases} > 0, \text{ if } \alpha_i = 0 \\ = 0, \text{ if } 0 \leq \alpha_i \leq C \\ < 0, \text{ if } \alpha_i = C \end{cases} \quad (6.34)$$

$$\forall i \in \mathcal{T},$$

$$h = \frac{\partial D}{\partial b} = \boldsymbol{\alpha}^\top \mathbf{y} = 0.$$

Given a new unlearned set $\mathcal{U} = \{\mathbf{X}_{\mathcal{U}}, \mathbf{y}_{\mathcal{U}}\}$ with the corresponding parameters $\boldsymbol{\alpha}_{\mathcal{U}}$, it is evident that any sample from $\mathcal{U}$ will join $\mathcal{R}$ immediately if $g_u > 0$. Hence hereinafter we refer $\mathcal{U}$ to the unlearned set where $\mathbf{g}_{\mathcal{U}} \leq \mathbf{0}$, implying that prior to the increment process all unlearned samples with $g_u > 0$ are integrated into $\mathcal{T}$. During incremental learning the existing solution $(\boldsymbol{\alpha}, b)$ needs to be tuned in parallel with $\boldsymbol{\alpha}_{\mathcal{U}}$. Assuming with small $\boldsymbol{\alpha}_{\mathcal{U}}$ no sample changes between categories, the difference in $\mathbf{g}$ and h are computed as

$$\begin{aligned} \Delta \mathbf{g} &= \mathbf{Q}_{\mathcal{T}\mathcal{U}} \Delta \boldsymbol{\alpha}_{\mathcal{U}} + \mathbf{Q}_{\mathcal{T}\mathcal{S}} \Delta \boldsymbol{\alpha}_{\mathcal{S}} + \mathbf{y} \Delta b, \\ \Delta h &= \mathbf{y}_{\mathcal{U}}^\top \Delta \boldsymbol{\alpha}_{\mathcal{U}} + \mathbf{y}_{\mathcal{S}}^\top \Delta \boldsymbol{\alpha}_{\mathcal{S}}, \end{aligned} \quad (6.35)$$

where $\mathbf{Q}_{AB} \in \mathcal{R}^{|A| \times |B|}$ is the label-augmented kernel matrix conducted on set A and set B (as a result, $\mathbf{Q} = \mathbf{Q}_{\mathcal{T}\mathcal{T}}$). Similarly a set subscript can be embedded to any parameter to indicate the elements of that parameter derived from the subscripted set. Under the assumption, $\Delta \mathbf{g}_{\mathcal{S}} = \mathbf{0}$ and $\Delta h = 0$, which lead to the following system of $|\mathcal{S}| + 1$ equations

$$\begin{aligned} \mathbf{Q}_{\mathcal{S}\mathcal{U}} \Delta \boldsymbol{\alpha}_{\mathcal{U}} + \mathbf{Q}_{\mathcal{S}\mathcal{S}} \Delta \boldsymbol{\alpha}_{\mathcal{S}} + \mathbf{y}_{\mathcal{S}} \Delta b &= \mathbf{0}, \\ \mathbf{y}_{\mathcal{U}}^\top \Delta \boldsymbol{\alpha}_{\mathcal{U}} + \mathbf{y}_{\mathcal{S}}^\top \Delta \boldsymbol{\alpha}_{\mathcal{S}} &= 0. \end{aligned} \quad (6.36)$$

For the unlearned samples it is much more simpler to increment all elements in $\Delta \boldsymbol{\alpha}_{\mathcal{U}}$ by the same scalar value represented by Δd . This will create freedom for $\alpha_s, \forall s \in \mathcal{S}$ and b to be adjusted with different element-wise sensitivities β_s

of the adjusted sample i . Expressing in matrix form we have

$$\begin{aligned} \Delta \alpha_{\mathcal{U}} &= \Delta d \mathbf{1}_{|\mathcal{U}|}, \\ \begin{bmatrix} \Delta b \\ \Delta \alpha_{\mathcal{S}} \end{bmatrix} &= \begin{bmatrix} \beta_0 \\ \beta_{\mathcal{S}} \end{bmatrix} \Delta d = \beta \Delta d. \end{aligned} \quad (6.37)$$

Let us denote $\tilde{\mathbf{Q}}_X$ as the augmented kernel matrix between any set X (can be composed of a single sample) and the support set $\mathcal{S}$ as

$$\mathbf{A}_X = \begin{bmatrix} \mathbf{y}_X^\top \\ \mathbf{Q}_{SX} \end{bmatrix}, \quad (6.38)$$

then by plugging (6.37) into (6.36), the sensitivity coefficients β can be solved by

$$\beta = -\mathbf{R} \mathbf{A}_{\mathcal{U}} \mathbf{1}_{|\mathcal{U}|}, \quad (6.39)$$

where

$$\mathbf{R} = \begin{bmatrix} 0 & \mathbf{y}_{\mathcal{S}}^\top \\ \mathbf{y}_{\mathcal{S}} & \mathbf{Q}_{SS} \end{bmatrix}^{-1}. \quad (6.40)$$

6.2.3.2 Searching for Δd

The updating process requires the obtainment of Δd in each step, given that β has been found from (6.39). From (6.35) each element in $\Delta \mathbf{g}$ can be expressed explicitly as

$$\Delta g_i = \mathbf{Q}_{i\mathcal{U}} \mathbf{1}_{|\mathcal{U}|} \Delta d + \mathbf{Q}_{i\mathcal{S}} \Delta \alpha_{\mathcal{S}} + y_i \Delta b, \forall i \in \mathcal{T} \quad (6.41)$$

Hence, the relation between Δg_i and Δd can be derived using (6.37) to yield

$$\Delta g_i = \gamma_i \Delta d, \quad (6.42)$$

where

$$\gamma_i = \begin{cases} \mathbf{Q}_{i\mathcal{U}} \mathbf{1}_{|\mathcal{U}|} + \mathbf{A}_i^\top \beta, & \text{if } i \notin \mathcal{S} \\ 0, & \text{if } i \in \mathcal{S} \end{cases}. \quad (6.43)$$

Based on the changes in g_i and α_i which cause only one sample to move across $\mathcal{S}, \mathcal{E}, \mathcal{R}$ and $\mathcal{U}$, the possibly largest Δd is desired. There are 6 cases where one sample may change the category:

- Case 1. A sample s moves from $\mathcal{S}$ to $\mathcal{R}$: α_s decreases from positive to 0. From (6.37) we obtain $\Delta d = -\alpha_s / \beta_s$. Since Δd must be positive, the condition for the sample s to be potentially become a case 1 move is that $\beta_s < 0$.

- Case 2. A sample s moves from $\mathcal{S}$ to $\mathcal{E}$: α_s increases to C . Similarly to case 1, $\Delta d = (C - \alpha_s)/\beta_s$ with the condition $\beta_s > 0$.
- Case 3. A sample r moves from $\mathcal{R}$ to $\mathcal{S}$: g_r decreases from positive to 0. From (6.42) we obtain $\Delta d = -g_r/\gamma_r$ with the condition $\gamma_r < 0$.
- Case 4. A sample e moves from $\mathcal{E}$ to $\mathcal{S}$: g_e increases from negative to 0. Similar to case 3, $\Delta d = -g_e/\gamma_e$ with the condition $\gamma_e > 0$.
- Case 5. A sample u moves from $\mathcal{U}$ to $\mathcal{S}$: the foremost condition is that $g_u < 0$, since otherwise it will be automatically assigned into $\mathcal{R}$. To have u moves to $\mathcal{S}$, g_u increases from negative to 0, hence $\Delta d = -g_u/\gamma_u$ with the condition $\gamma_u > 0$.
- Case 6. A sample u moves from $\mathcal{U}$ to $\mathcal{E}$: similarly to case 5, there must hold $g_u < 0$. To have u moves to $\mathcal{S}$, α_u increases to C , hence from the first equation in (6.37) we obtain $\Delta d = C - \alpha_u$.

6.2.3.3 Technique to Expand or Shrink $\mathbf{R}$

After each iteration, the dimension of the square matrix $\mathbf{R}$ will be expanded by 1 if the sample i is added to $\mathcal{S}$, or shrunk by 1 if i is removed from $\mathcal{S}$. Special techniques are required to efficiently derive $\mathbf{R}^{\text{new}}$ based on $\mathbf{R}$ and the sensitivity coefficient β of i . We consider the adding case first, described in the following lemma.

Lemma 6.3: When a sample i is added into $\mathcal{S}$, the matrix $\mathbf{R}$ is expanded as

$$\mathbf{R}_{\text{new}} = \begin{bmatrix} \mathbf{R} & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{\delta} \begin{bmatrix} \boldsymbol{\mu} \\ 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu} \\ 1 \end{bmatrix}^{\top} \quad (6.44)$$

where $\boldsymbol{\mu} = -\mathbf{R}\mathbf{A}_i$ and $\delta = Q_{ii} - \mathbf{A}_i^{\top} \mathbf{R} \mathbf{A}_i$.

Proof.

We explicitly write $\mathbf{R}^{\text{new}}$ as

$$\mathbf{R}_{\text{new}} = \begin{bmatrix} 0 & \mathbf{y}_{\mathcal{S}}^{\top} & y_i \\ \mathbf{y}_{\mathcal{S}} & \mathbf{Q}_{\mathcal{SS}} & \mathbf{Q}_{\mathcal{Si}} \\ y_i & \mathbf{Q}_{\mathcal{Si}}^{\top} & Q_{ii} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{R}^{-1} & \mathbf{A}_i \\ \mathbf{A}_i^{\top} & Q_{ii} \end{bmatrix}^{-1}. \quad (6.45)$$

Using the matrix inversion lemma (Appendix 1), we obtain

$$\mathbf{R}_{\text{new}} = \begin{bmatrix} \mathbf{R} + \delta^{-1} \mathbf{R} \mathbf{A}_i \mathbf{A}_i^\top \mathbf{R} & -\delta^{-1} \mathbf{R} \mathbf{A}_i \\ -\delta^{-1} \mathbf{A}_i^\top \mathbf{R} & \delta^{-1} \end{bmatrix}. \quad (6.46)$$

Let $\boldsymbol{\mu} = -\mathbf{R} \mathbf{A}_i$ we achieve the desired result. ■

6.2.3.4 Bridging the Gaps

During the practical usage of ISVM, two major issues appeared due to the theoretical gaps in ISVM. The first issue is the presence of duplicate training samples, which can break the convergence of the algorithm. The second issue relates to the zero updating procedure which potentially creates an infinite updating cycle.

Duplicate Learning Samples:

Learning error will occur when duplicate samples exist. If they inadvertently become support or error vectors, their corresponding α_i must be uniformly divided. This exact but cumbersome approach complicates the learning process. A more delicate solution is to make active only one of them, and the others are marked as *duplicate* and excluded from the learning process. The procedure of checking for duplicate samples needs to be performed prior to the learning stage. It should also be noticed that samples marked as duplicate are not equivalent to reserved vectors. Future incremental learning based on the solution including these clones cannot bring them into calculation. As a result duplicate samples can be categorized into a fourth category, namely duplicate vectors, denoted by $\mathcal{D}$, which will be ruled out entirely from the learning process with $\boldsymbol{\alpha}_{\mathcal{D}} = 0$.

Zero Update Cycle If the sample i moves from $\mathcal{E}$ or $\mathcal{R}$ to $\mathcal{S}$, the update process for α_i is not taken place since $\beta_{\mathcal{E}}$ and $\beta_{\mathcal{R}}$ do not exist, hence $\alpha_i^{\text{new}} = \alpha_i^{\text{old}} = C$ (if i moves from $\mathcal{E}$ to $\mathcal{S}$) or $\alpha_i^{\text{new}} = \alpha_i^{\text{old}} = 0$ (if i moves from $\mathcal{R}$ to $\mathcal{S}$). As a consequence in the next iteration, a zero update on that sample, equivalently expressed as $\Delta d = 0$, may occur on that sample to transfer i back to $\mathcal{E}$ or $\mathcal{R}$ immediately. To guarantee this cycle does not happen, β_i^{new} must be positive if $\alpha_i = C$, or negative if $\alpha_i = 0$, which will be proved in the following

Lemma.

Lemma 6.4: The last element of β_{new} , which corresponds to the sensitivity coefficient of the newly-added sample i , has the opposite sign with γ_i .

Proof. From (6.39), we obtain the following expression of β_{new}

$$\beta_{\text{new}} = -\mathbf{R}_{\text{new}} \begin{bmatrix} \mathbf{A}_{\mathcal{U}} \\ \mathbf{Q}_{i\mathcal{U}} \end{bmatrix} \mathbf{1}_{|\mathcal{U}|}. \quad (6.47)$$

Therefore the sensitivity coefficient of the added sample i is the last element of β_{new} and is computed as

$$\begin{aligned} \beta_{\text{new},i} &= -\delta^{-1} \begin{bmatrix} \boldsymbol{\mu} \\ 1 \end{bmatrix}^{\top} \begin{bmatrix} \mathbf{A}_{\mathcal{U}} \\ \mathbf{Q}_{i\mathcal{U}} \end{bmatrix} \mathbf{1}_{|\mathcal{U}|} = -\delta^{-1} (\boldsymbol{\mu}^{\top} \mathbf{A}_{\mathcal{U}} + \mathbf{Q}_{i\mathcal{U}}) \mathbf{1}_{|\mathcal{U}|} \\ &= -\delta^{-1} (\mathbf{Q}_{i\mathcal{U}} \mathbf{1}_{|\mathcal{U}|} - \mathbf{A}_i^{\top} \mathbf{R} \mathbf{A}_{\mathcal{U}} \mathbf{1}_{|\mathcal{U}|}) \\ &= -\delta^{-1} (\mathbf{Q}_{i\mathcal{U}} \mathbf{1}_{|\mathcal{U}|} + \mathbf{A}_i^{\top} \boldsymbol{\beta}) = -\delta^{-1} \gamma_i. \end{aligned} \quad (6.48)$$

The quadratic representation of δ is derived from (6.44) as

$$\begin{aligned} \delta &= Q_{ii} - \mathbf{A}_i^{\top} \mathbf{R} \mathbf{A}_i = Q_{ii} - 2\mathbf{A}_i^{\top} \mathbf{R} \mathbf{A}_i + \mathbf{A}_i^{\top} \mathbf{R} \mathbf{R}^{-1} \mathbf{R} \mathbf{A}_i \\ &= \boldsymbol{\mu}^{\top} \mathbf{R}^{-1} \boldsymbol{\mu} + \mathbf{A}_i^{\top} \boldsymbol{\mu} + \boldsymbol{\mu}^{\top} \mathbf{A}_i + Q_{ii} \\ &= \begin{bmatrix} \boldsymbol{\mu} \\ 1 \end{bmatrix}^{\top} \mathbf{R}_{\text{new}}^{-1} \begin{bmatrix} \boldsymbol{\mu} \\ 1 \end{bmatrix} > 0, \end{aligned} \quad (6.49)$$

since $\mathbf{R}_{\text{new}}^{-1}$ is positive-definite. Referring to the process of searching for Δd , the condition for a sample i to move from $\mathcal{S}$ to $\mathcal{R}$ (or $\mathcal{E}$) immediately after it was transferred from $\mathcal{R}$ (or $\mathcal{E}$) to $\mathcal{S}$, is that $\gamma_i \beta_{\text{new},i} > 0$. This is the condition for zero update to occur, and is also a contradiction to (6.48) and (6.49). Hence, we conclude that the provided algorithm is guaranteed to converge and avoid zero updating. ■

6.3 Experiments

6.3.1 Influence of Partitioning on Adaptive Learning

Figure 6.3 and Figure 6.4 illustrate the evolution of the time needed for updating the existing classifiers with respect to the number of new learning

samples (from 12 samples to 180 samples). Each plot consists of 4 different line graphs associated to different numbers of existing samples in the classifier pool (i.e., 540, 1080, 1620, and 2160). Without partitioning, the AL updating time drastically increases with respect to the number of new learning samples, as shown in Figure 6.3. For instance, to integrate 180 new samples into 1080 existing samples, we need roughly 5 seconds. But with the same number of new samples it takes over 17 seconds to integrate them into 2160 existing samples. However, just by dividing the 2160 existing samples into 3 partitions (720 each) as shown in Figure 6.4, we can drastically reduce the AL time from 17 seconds to 8 seconds. This proves the huge benefit of partitioning in terms of AL time reduction.

One desired job in our proposed method is to determine the appropriate number of samples and partitions of the existing ensemble classifier to best fit the AL stage. Typically, the P3S gives the subjects a short break (from 5 to 10 seconds) between letter sessions to help them relax and find the next target letter to focus. This break time can be exploited by the AL stage to update the classifier between sessions. As a result each AL stage must not last too long (i.e., over 10 seconds). Assuming the worst performance is performed by the DS stage (i.e., we need the maximum of 15 iterations to output the decision), we plot the average time needed (in seconds) to integrate 180 new samples (15 iterations) into 1080, 2160, and 3240 existing samples of the classifier, as shown in Figure 6.5. It is observed from the figure that using 3240 existing samples is not an option, since it takes longer than 10 seconds in all partitioning schemes. On the other hand, when updating 1080 and 2160 existing samples (with partitioning), the AL time is below 5 seconds and 10 seconds, respectively, which satisfies our break time criterion. We choose 2160 to be the fixed number of learning samples, as more existing samples undoubtedly result in better performance. As a result, we make further analysis on the accuracies and iterations of different partitioning schemes using 2160 samples, as described in Table 6.1.

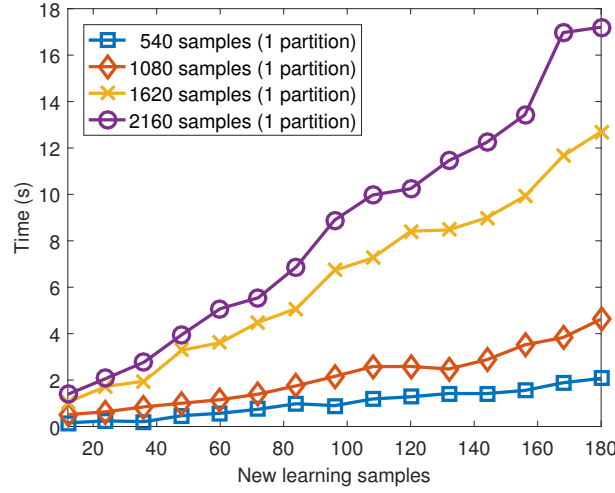


Figure 6.3: Time vs. new learning samples, with no partitioning. 4 lines indicate 4 different numbers of existing samples in the available classifier solution.

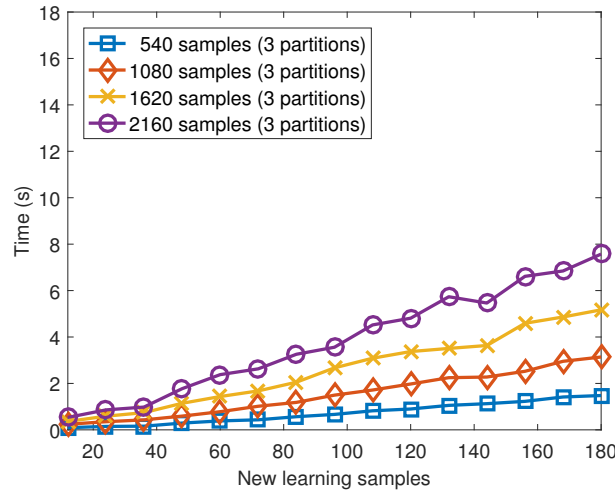


Figure 6.4: Time vs. new learning samples, with partitioning. 4 lines indicate 4 different numbers of existing samples in the available classifier solution.

6.3.2 Classification Schemes

In this thesis, we conduct our analysis of DS and AL on Dataset A (as described in Table 3.3). We use the data of 12 *learning subjects* (namely, ACS, APM, ASG, ASR, CLL, DCM, DLP, LGP, ELC, JCR, FSZ, GCE) of Dataset A to implement the learning process. The first phase is to construct four classifiers (ENS-2, ENS-3, ENS-4, ENS-6) from the first six subjects (ACS, APM, ASG, ASR, CLL, and DCM), as presented in Table 6.1. The data of six other subjects (with 9 letters each) are used to evaluate the DS parameters $\tilde{\theta}_1$ and $\tilde{\theta}_2$.

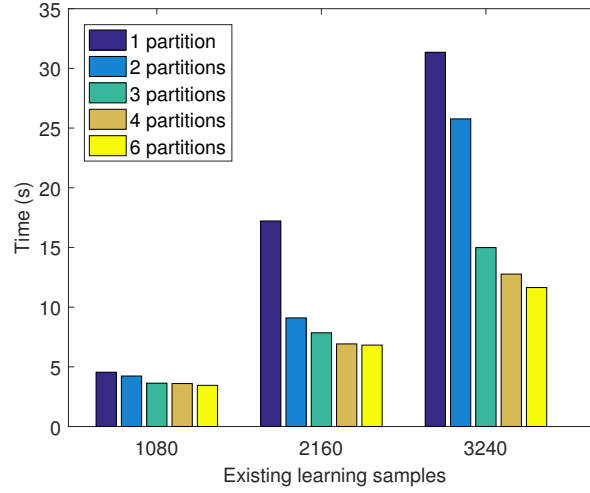


Figure 6.5: Time vs. existing samples in the classifier solution and classifier partitions.

As shown in Table 6.1, P is the number of total partitions (each partition has learning samples of only 1 subject), L is the number of learning letters used in each partition. The number of samples for each scheme is 2160 (explained in the previous section), equals to 12 learning letters.

The resulting parameters are then used to validate the large test set of the other 14 *validation subjects* (FSZ, GCE, ICE, IZH, JLD, JLP, JMR, JSC, JST, LAC, LAG, PGA, WFG, XCL), with 12 to 24 letters per subject.

TABLE 6.1: CLASSIFICATION SCHEMES' SETTINGS.

Scheme	P	L	Learning Subjects
ENS-2	2	6	ACS, APM
ENS-3	3	4	ACS, APM, ASG
ENS-4	4	3	ACS, APM, ASG, ASR
ENS-6	6	2	ACS, APM, ASG, ASR, CLL, DCM

6.3.3 Grid Search Result

Figure 6.6 and Figure 6.7 respectively illustrate the expected accuracy and the number of iterations validated from the learning stage using different values of DS parameters $\tilde{\theta}_1$ and $\tilde{\theta}_2$. Generally, the numbers of iterations vary very slightly given a specific pair of DS parameters over the 4 schemes. Similarly, the expected accuracies heat maps of the 4 schemes are also similar as they never reach above 80%. From Figure 6.6 and Figure 6.7, we extract the searching

results for $\tilde{\theta}_1$ and $\tilde{\theta}_2$, given 3 different values of expected accuracy reduction a (as in Algorithm 7), in Table 6.2.

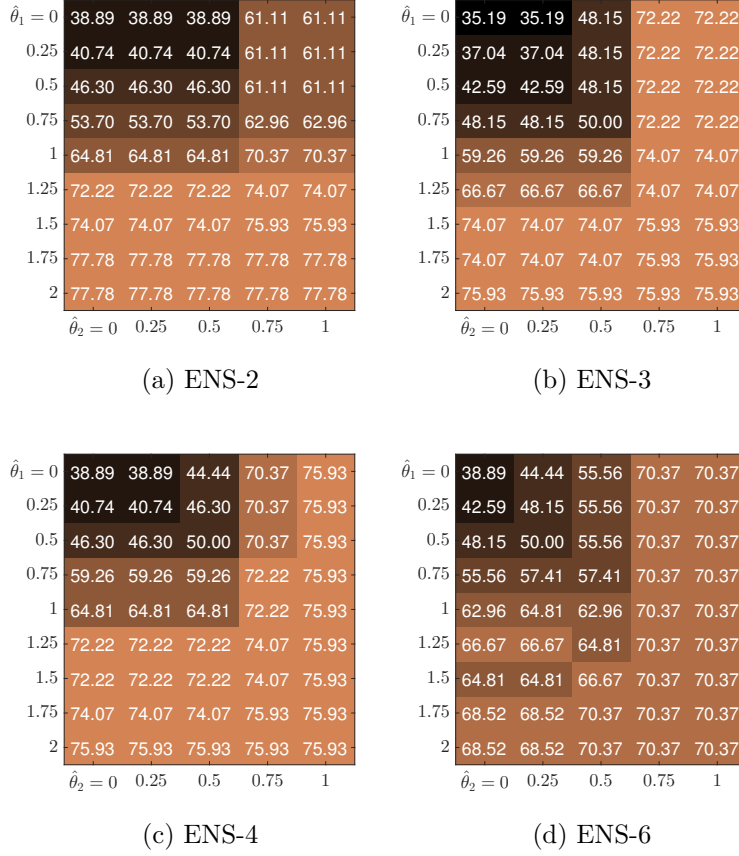


Figure 6.6: The accuracies of 4 schemes on the learning set with different values of DS parameters $\hat{\theta}_1$ and $\hat{\theta}_2$.

TABLE 6.2: THE GRID SEARCH RESULTS OF 4 CLASSIFICATION SCHEMES.

Scheme	a	$\tilde{\theta}_1$	$\tilde{\theta}_2$	Expected Accuracy	Expected Iteration
ENS-2	0	1.75	0.50	77.78	13.17
	5	1.25	0.75	74.07	9.91
	10	1.00	0.75	70.37	8.39
ENS-3	0	1.50	0.75	75.93	11.70
	5	0.50	0.75	72.22	9.15
	10	1.25	0.25	66.67	8.11
ENS-4	0	0.50	1.00	75.93	13.28
	5	1.25	0.25	72.22	9.04
	10	0.25	0.75	70.37	8.94
ENS-6	0	0.50	0.75	70.37	12.32
	5	1.25	0.00	66.67	9.35
	10	1.00	0.00	62.96	7.00

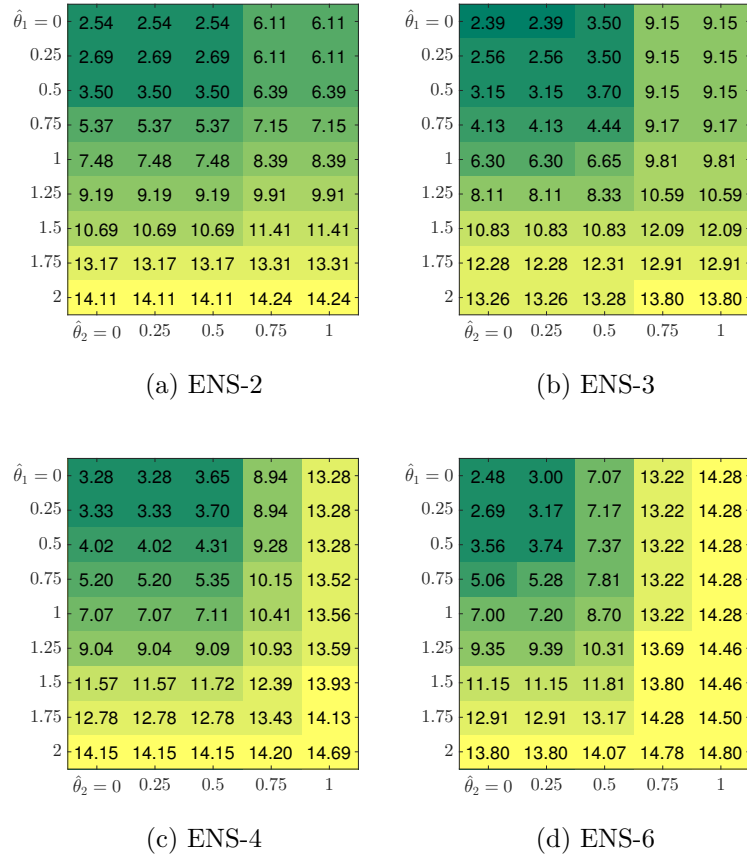


Figure 6.7: The number of iterations of 4 schemes on the learning set with different values of DS parameters $\hat{\theta}_1$ and $\hat{\theta}_2$.

6.3.4 Validation Stage Performance

In this section, for the reader's convenience we re-mention the evaluation metrics used exclusively in this chapter, that had been described in a general context in Chapter 3. First, the term *Accuracy* for each subject (used in this chapter) is simply computed as

$$Accuracy = \frac{\text{correct classified letters}}{\text{total letters spelled}},$$

and the term *Iteration* of a subject is the number of iterations taken averaged over all his/her spelling letters. The term *ITR*, which stands for information transfer rate, is the harmonized metric between *Accuracy* and *Time*, and is calculated by

$$ITR = \frac{\log_2 36 + A \log_2 A + (1 - A) \log_2 (1 - A)}{Iteration \times ISI \times 12/60},$$

where A is the accuracy and ISI is the inter-stimulus interval (seconds). Finally, the term *MaxTime* is defined as the maximum AL time (in seconds) for a subject taken over all the spelling sessions. More specifically, for each subject we take the maximum AL time among multiple letters spelled by that subject, then we take the average of maximum AL time of all subjects. Using those 4 evaluation metrics, we present the average *Accuracy*, average *Iteration*, average *ITR*, and average *MaxTime* taken on all validation subjects (as described in Section 6.3.2) in Table 6.4.

It should be noticed that given the existing 2160 samples of the classifier at the beginning, we should not employ AL for all approaching testing sessions, as the classifier size will be excessively accumulated which leads to increasing AL time. Since each subject has 12 to 24 letters to validate, we only use their first 0, 2, 4, 6, or 8 new letter sessions for the AL stage, then analyze the changes in accuracies and iterations with respect to the numbers of new letters learned. As shown in Table 6.4, for all input values of a , there are huge leaps in accuracies when employing the AL process. For instance, when *Letters for AL* equals 2 ($a = 0$) we can increase the accuracy from around 80% to around 90% in all 4 schemes ENS-2, ENS-3, ENS-4 and ENS-6. However when increasing the letters for AL, the accuracies increase more slowly and the iterations also drop more slowly in all schemes (for $a = 0$ and $a = 5$), but the AL time increases much faster. This effect is not desirable in our real-time system. For that reason we only need to use the AL stage for a very first few letters (optimally 2 to 4 letters) of a new subject, which can balance between the accuracy and AL time.

6.3.5 Influence of Expected Accuracy Parameter a of the Grid Search Algorithm

When using a low expected accuracy parameter ($a = 10$), it is observed from Table 6.4 that the accuracies do not increase but fluctuate (in ENS-4 and ENS-6) with respect to the number of AL letters. The explanation for this effect is that the performance is hugely influenced by the correctness of the previous classified letters. A wrongly-classified letter will negatively affect the existing classifier and cause the deterioration in accuracies of the following

sessions. The safe tuning range for a is therefore quite narrow (from 0 to 5). If the highest accuracy is desired, $a = 0$ is the optimal choice, whereas $a = 5$ will slightly decrease the accuracies but also reduce the number of iterations significantly. This choice depends on user-purposed system specifications.

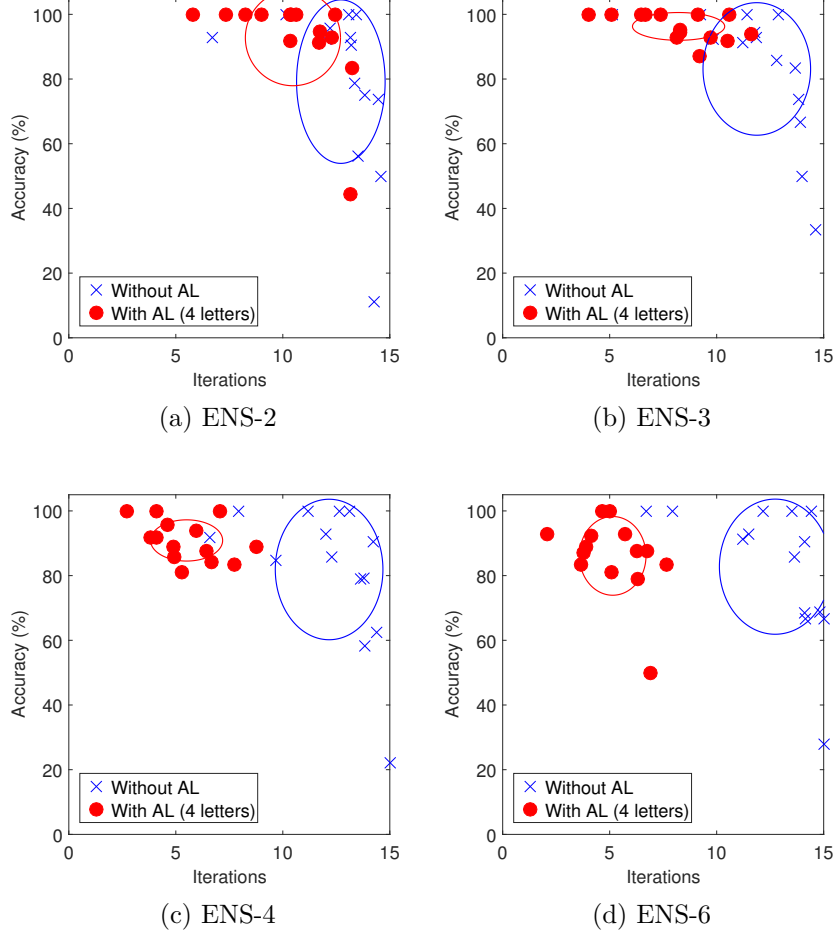


Figure 6.8: The subject-wise results for all test subjects using different input values of dynamic stopping parameters $\hat{\theta}_1$ and $\hat{\theta}_2$.

6.3.6 Influence of DS and AL on the Performance of Each Subject

Figure 6.8 plots the subject-wise performance of all schemes using 4 AL letters with $a = 0$. Each individual point represents the performance of one subject. The ellipses represent the average area of each scheme, with the centers of the ellipses being the mean and the radii being the standard deviations of the accuracy and iterations on the whole scheme. As shown, we achieve

a significant reduction in iterations in all schemes. The regions of AL points are narrower, implying that the performances of all subjects will converge to a specific accuracy and iteration given a fixed setting. There are points (without AL) over the subplots which have extremely poor accuracies (below 60%). This problem is caused by the mismatch between the brain patterns of the validation subject and the learning subjects. However with adaptive learning, this problem is solved.

6.3.7 Comparisons with Related Studies

We have re-implemented the method used in other related studies [10, 17, 19, 21–24], using the equivalent classifier and dataset as specified in our work. Our target is that we want to provide a most reliable comparison between our work with the others. The comparison is shown in Table 6.3.

TABLE 6.3: COMPARISON OF OUR STUDY AND RELATED STUDIES, CONDUCTED ON THE SUBJECT-INDEPENDENT BASIS.

Method	Setting	Accuracy (%)	Iterations
[10, 21–24]	ENS-2	60.85	6.56
	ENS-3	64.07	4.84
	ENS-4	65.24	4.90
	ENS-6	65.65	5.06
[19]	ENS-2	52.63	3.00
	ENS-3	57.90	3.00
	ENS-4	52.55	3.05
	ENS-6	61.23	5.32
[17]	ENS-2	48.95	9.36
	ENS-3	56.41	8.82
	ENS-4	47.56	9.45
	ENS-6	51.45	8.64
Our method	ENS-2	95.96	9.25
	ENS-3	96.58	8.52
	ENS-4	90.87	5.50
	ENS-6	88.30	3.50

Overall we outperformed the others in both accuracy and iterations. Liu *et al.* [19] took the sum over the last 3 iterations of row/column candidates and compared them with the threshold $N \times d$, where N is the free parameter and d is the average distance of target responses to the classifier solution, derived from the training set. We employed their method using $N = 1$. Jin

et al. [17] used 2 dynamic stopping parameters. N_1 is the number of consecutive iterations which output the same letter, and N_2 is the beginning iteration to start checking N_1 . As suggested in [17] we conducted their method using $N_1 = 2$ and $N_2 = 3$. More recently, a probabilistic approach for DS stage was used in [10, 21–24]. Although those mentioned studies have different research problems, they however were employed using the similar DS method first proposed by [24]. This method calculates the probability of each letter candidate after each iteration via a Bayesian updating basis and compares them to the threshold t . To make a comparison with our method we also re-conducted their approach in our schemes, using $t = 0.95$. A trade-off clearly occurs between the accuracy and implementation time. Our fastest setting (ENS-6) is approximately three times faster than our best setting in accuracy. However overall, our proposed method offers a significantly better performance than the related studies thanks to its complexity in ensemble modelling and adversarial feature encoding.

6.4 Chapter Summary

. In this chapter, we have outlined in details the following key sections:

- The dynamic stopping method on the P3S, with theoretical proofs backing the supportive aspects of the two criteria DS parameters.
- The adaptive learning method using ISVM, with theoretical foundation and how to resolve the ISVM gaps that we confronted during practical usage.
- The detailed experiments with thorough analysis on various aspects of DS and AL on the overall performance.

The outputs from Chapter 5, which are the final output score of the classification algorithm, direct serve as inputs to the DS framework in this chapter. Hence DS and AL can be employed independently to the ensemble classification stage as presented in Chapter 5. Therefore this chapter can serve as a separate enhancement method for the P3S in this thesis's context, or more generally, for the classification of various other BCI applications.

TABLE 6.4: CLASSIFICATION RESULT AVERAGED ON ALL VALIDATION SUBJECTS.

		$a = 0$				
Letters for AL		0	2	4	6	8
ENS-2	<i>Accuracy</i>	79.20	90.98	92.74	94.89	95.96
	<i>Iteration</i>	12.71	11.31	10.47	9.87	9.25
	<i>ITR</i>	7.57	10.78	12.06	13.36	14.57
	<i>MaxTime</i>	0.00	6.52	9.82	11.27	13.18
ENS-3	<i>Accuracy</i>	83.11	93.99	96.27	96.18	96.58
	<i>Iteration</i>	11.87	9.23	8.22	7.84	7.61
	<i>ITR</i>	8.78	14.03	16.50	17.27	17.94
	<i>MaxTime</i>	0.00	3.48	6.01	6.92	8.52
ENS-4	<i>Accuracy</i>	81.89	89.90	90.87	90.04	89.46
	<i>Iteration</i>	12.16	6.63	5.50	4.90	4.60
	<i>ITR</i>	8.36	18.01	22.13	24.43	25.73
	<i>MaxTime</i>	0.00	3.45	4.79	4.92	5.97
ENS-6	<i>Accuracy</i>	82.75	88.30	86.10	83.93	81.83
	<i>Iteration</i>	12.73	6.14	5.14	4.50	4.29
	<i>ITR</i>	8.12	18.84	21.54	23.55	23.67
	<i>MaxTime</i>	0.00	3.50	4.95	5.42	6.24
		$a = 5$				
Letters for AL		0	2	4	6	8
ENS-2	<i>Accuracy</i>	75.72	86.04	90.58	91.21	92.32
	<i>Iteration</i>	9.84	8.18	7.81	6.80	6.47
	<i>ITR</i>	9.07	13.51	15.50	18.02	19.36
	<i>MaxTime</i>	0.00	4.24	7.28	9.85	10.23
ENS-3	<i>Accuracy</i>	80.76	84.98	85.09	85.43	83.21
	<i>Iteration</i>	8.78	5.81	4.70	4.19	3.92
	<i>ITR</i>	11.31	18.63	23.08	26.07	26.64
	<i>MaxTime</i>	0.00	2.12	3.27	4.08	4.15
ENS-4	<i>Accuracy</i>	76.30	84.70	87.83	89.25	90.12
	<i>Iteration</i>	8.95	6.81	6.47	6.09	6.02
	<i>ITR</i>	10.10	15.80	17.71	19.35	19.92
	<i>MaxTime</i>	0.00	2.81	4.65	5.95	7.15
ENS-6	<i>Accuracy</i>	75.91	86.00	88.63	87.15	87.24
	<i>Iteration</i>	8.75	6.75	6.59	6.20	6.01
	<i>ITR</i>	10.25	16.37	17.67	18.24	18.85
	<i>MaxTime</i>	0.00	3.12	4.60	6.31	7.26
		$a = 10$				
Letters for AL		0	2	4	6	8
ENS-2	<i>Accuracy</i>	71.48	79.86	84.53	86.46	87.41
	<i>Iteration</i>	8.26	6.66	6.12	5.26	5.28
	<i>ITR</i>	9.83	14.63	17.53	21.20	21.52
	<i>MaxTime</i>	0.00	3.25	5.70	8.10	8.94
ENS-3	<i>Accuracy</i>	77.94	88.82	88.96	90.09	90.94
	<i>Iteration</i>	8.56	7.17	6.53	6.09	6.04
	<i>ITR</i>	10.93	16.30	17.95	19.68	20.18
	<i>MaxTime</i>	0.00	2.86	4.70	5.61	6.62
ENS-4	<i>Accuracy</i>	76.18	77.67	70.89	68.93	69.51
	<i>Iteration</i>	8.31	3.83	3.29	2.98	2.90
	<i>ITR</i>	10.85	24.30	24.37	25.71	26.78
	<i>MaxTime</i>	0.00	1.94	2.33	2.48	2.62
ENS-6	<i>Accuracy</i>	68.19	78.87	73.26	76.25	76.38
	<i>Iteration</i>	6.63	5.61	4.94	4.72	4.56
	<i>ITR</i>	11.35	17.02	17.12	19.13	18.63
	<i>MaxTime</i>	0.00	2.17	3.34	5.51	5.86

Chapter 7

CONCLUSIONS AND FUTURE WORK

This thesis focuses on the ERP-based BCI, especially the P3S with the row-column flashing paradigm. Various machine learning techniques were proposed to attain the subject-independent requirement. In other words, new users of the BCI are exempted from the procedural, uncomfortable, and time-consuming training stage. In this chapter, we will summarize the innovations of this thesis in section 7.1, followed by the proposals of future work in section 7.2.

7.1 Innovations in the Thesis

The main innovations of this thesis are summarised as follows.

7.1.1 Application of Adversarial Neural Network Training on an ERP-based BCI

One innovation in this thesis is the particular adversarial learning architecture, as presented in chapter 4, which designed specifically for the multi-subject dataset of an ERP-based BCI. As compared to the DANN architecture [25], our SANN network uses the domain discriminator as a subject separator, hence converting a binary domain classification into a multi-class subject classification using the subject separator. The experiments were conducted on two stages. The first stage is to train the whole SANN by the training data, and the second stage is the testing of the feature encoder of SANN using different machine learning algorithms. Beside the t-SNE visualisation which confirms the effects of SANN on test data, the result had shown that the test aver-

age trial-wise accuracy is boosted from 77.2% to 85.9% in NN, from 79.0% to 83.1% in SVM, and from 54.7% to 56.3% in FDA. These correspond to significant boosts on the average symbol-wise accuracy performance: from 84.4% to 95.0% in NN, from 85.8% to 89.7% in SVM, and no improvement in FDA. The related research work is under review by the IEEE Transactions of Biomedical Engineering in 2018.

7.1.2 An Ensemble Strategy of Bagging and Stacking

By winning various machine learning competitions [36], the meta-ensemble strategy of multiple algorithms had proved its strong practical capability in real-time applications. In this thesis we apply that ensemble strategy in the context of P3S BCI. We also provided the theoretical demonstration of the improvement of bagging and stacking. In our method, each base learner (SVM, FDA, NN) is learned normally from the training data, with the extra step of hyperparameter grid search. The final test performance had shown that the precision, recall, and F-score can all be boosted by bagging and stacking, as reported detailedly in Table 5.2 of Chapter 5. Our related works were published in the IEEE Conference of Engineering in Medicine and Biology Society (EMBC) in 2017, and the IEEE International Symposium on Medical Information and Communication Technology (ISMICT) in 2017.

7.1.3 Dynamic Stopping and Adaptive Learning in Real-time

We have provided a complete solution to integrate DS and incremental AL based on SVM to boost the ITR and accuracy of the ERP-based BCI, especially on the real-time fashion. Our experimental results suggest that, instead of achieving the average (taken over 4 schemes with $a = 0$, as presented in Chapter 6) accuracy of 75.00% with 12.62 iterations for each spelling session, we can boost it up to 91.26% with just 6.78 iterations. The trade-off for this improvement is the huge computational resources for updating the existing SVM classifier. However, with thorough benchmarking and analysis, we also propose a method to achieve the most appropriate DS parameters and AL settings. As a consequence, under our control the additional computation does not affect the system since the free time between letter

sessions is efficiently exploited. To emphasize the merit of our proposal, we also re-implement the methods from related works using the same subject-independent setting as in ours. Our method outperformed most of the related studies in terms of accuracy. With an appropriate selection on the ensemble scheme, the experimental speed is also significantly boosted. Our source code is provided online at <https://github.com/voanhkha/Subject-Independent-ERP-based-BCI>. We have published the related works in the IEEE Transactions on Neural Systems and Rehabilitation Engineering (TNSRE) in 2018, the ACM/IEEE Conference on Connected Health: Applications, Systems and Engineering (CHASE) in 2018, and the IEEE Conference of Global Communications (GLOBECOM) in 2017.

7.2 Future Work

Due to the limited time and funding of the author, this thesis still leaves a handful of valuable research areas for future work, which will be pointed out as follows.

7.2.1 Perform SANN of Deep Architectures

In this thesis we employed the shallow feedforward NN network architecture for all SANN parties (feature encoder, class predictor, and subject separator) due to the limitations of graphics processing units (GPU) resources. For future work, these SANN architectures can be parametrised as deep neural networks, or even deep convolutional neural networks (CNN). For CNN, the feature vector therefore must be transformed into 2-dimensional, one dimension for the temporal domain and one dimension for the spatial domain. For doing so, we will be able to employ existing state-of-the-art deep learning toolboxes, i.e., Tensorflow, easily. As reported in [25], deep architectures result in significantly better performance.

7.2.2 Optimal Segmentation of EEG Data in Real-time

Recently we have published our work related to the optimal segmentation of ECG arrhythmia data [20]. This algorithm is guaranteed to find the optimal segmentation of a long 1-D data, which consists of similar segments.

The innovation of our developed algorithm is that we do not need to specify the number of segments in the signal. The only constraints that need to be specified are the minimum and maximum length of a single segment. Considering the potential of the application of this algorithm on an ERP-based BCI, we find that the flashing paradigm can be relaxed in some ways to exploit the dynamic capability of the segmentation algorithm. More specifically, if we are able to design a different design of P3S in terms of flashing patterns and flashing intervals, the optimal segmentation algorithm will benefit. This will create opportunities for real-time biomedical supportive devices, such as language translator, or intention indicator, for disabled patients.

7.2.3 Language Model

Some postprocessing steps can be embedded into the BCI framework to further enhance the performance. The most promising area to research is language models [125, 126, 140] using hidden Markov models to estimate the probability of the next letter based on the spelling recent history. Moreover, if we view language model in a more nuanced context, it can be applied on the trial level, where the probability of the next trial can be estimated by the history of the last estimated trials.

Appendix A

Derivations of Some Expressions in Chapter 4

In this Appendix, the derivations of some expressions in Chapter 4 are presented. It should be noted that, except Theorem A.6, the derivations in all of the other theory and corollaries are borrowed from literature [229].

Theorem A.1. (Non-negativity of KL Divergence):

Given two probability distributions $p(x)$ and $q(x)$, $x \in X$, then the following expression holds

$$D_{\text{KL}}(p||q) \geq 0 \quad (\text{A.1})$$

with equality holds if and only if $p(x) = q(x)$ for all x .

Proof. To prevent dividing zero in the following expressions, let A be the support set of $p(x)$, i.e., $A = \{x|p(x) > 0\}$. From the definition of $D_{\text{KL}}(p||q)$, we have

$$\begin{aligned} -D_{\text{KL}}(p||q) &= \sum_{x \in A} p(x) \log \frac{q(x)}{p(x)} \\ &\stackrel{(*)}{\leq} \log \sum_{x \in A} p(x) \frac{q(x)}{p(x)} \\ &= \log \sum_{x \in A} q(x) \\ &\stackrel{(**)}{\leq} \log \sum_{x \in X} q(x) \\ &= \log 1 = 0, \end{aligned} \quad (\text{A.2})$$

where $(*)$ follows from Jensen's inequality [230]. Since $-\log z$ is a strictly convex function, the equality in $(*)$ holds if and only if $q(x)/p(x) = 1$ everywhere, i.e., $p(x) = q(x)$ for all x . Also, when $p(x) = q(x)$, the equality in $(**)$ also holds. ■

Corollary A.2. (Non-negativity of Mutual Information):

The mutual information of two random variables X and Y is always non-negative

$$\mathbb{I}(X; Y) \geq 0 \quad (\text{A.3})$$

with equality holds if and only if X and Y are independent.

Proof. From the definition of $\mathbb{I}(X; Y)$, we have

$$\begin{aligned} \mathbb{I}(X; Y) &= \sum_y \sum_x p(x, y) \log \left(\frac{p(x, y)}{p(x)p(y)} \right) \\ &= D_{\text{KL}}(p(x, y) \| p(x)p(y)) \\ &\geq 0. \quad (\text{see Theorem A.1}) \end{aligned} \quad (\text{A.4})$$

The equality holds if and only if $p(x, y) = p(x)p(y)$, or in other words X and Y are independent.

Theorem A.3. (Data Processing Inequalities):

Suppose we have a Markov chain [231] describing a probability model as

$$X \rightarrow Y \rightarrow Z \quad (\text{A.5})$$

where X and Z conditionally uncorrelated given Y . In other words, knowledge of the previous state is all that is necessary to determine the probability distribution of the current state. Then the following expression holds

$$\mathbb{I}(X; Y) \geq \mathbb{I}(X; Z). \quad (\text{A.6})$$

Proof. By using the chain rule, we can express $\mathbb{I}(X; (Y, Z))$ as

$$\begin{aligned} \mathbb{I}(X; (Y, Z)) &= \mathbb{I}(X; Z) + \mathbb{I}(X; Y|Z) \\ &= \mathbb{I}(X; Y) + \mathbb{I}(X; Z|Y). \end{aligned} \quad (\text{A.7})$$

Since X and Z conditionally independent given Y , it holds $\mathbb{I}(X; Z|Y) = 0$. As a result, $\mathbb{I}(X; Z) + \mathbb{I}(X; Y|Z) = \mathbb{I}(X; Y)$. Also, we have $\mathbb{I}(X; Y|Z) \geq 0$ since mutual information is always non-negative (see Theorem A.2), hence

$$\mathbb{I}(X; Z) \leq \mathbb{I}(X; Y). \quad \blacksquare$$

Theorem A.4. :

For all random variables X, Y , it holds

$$\mathbb{I}(X; Y) = \mathbb{H}(X) - \mathbb{H}(X|Y). \quad (\text{A.8})$$

Proof. From the definition of mutual information, we have

$$\begin{aligned} \mathbb{I}(X; Y) &= \sum_{x \in X, y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \\ &= \sum_{x \in X, y \in Y} p(x, y) \log \frac{p(x|y)}{p(x)} \\ &= - \sum_{x \in X, y \in Y} p(x, y) \log p(x) + \sum_{x \in X, y \in Y} p(x, y) \log p(x|y) \quad (\text{A.9}) \\ &= - \sum_{x \in X, y \in Y} p(x) \log p(x) - \left(- \sum_{x \in X, y \in Y} p(x, y) \log p(x|y) \right) \\ &= \mathbb{H}(X) - \mathbb{H}(X|Y). \quad \blacksquare \end{aligned}$$

Corollary A.5. (Conditioning reduces entropy):

For all random variables X, Y , it holds

$$\mathbb{H}(X|Y) \leq \mathbb{H}(X) \quad (\text{A.10})$$

with equality if and only if X and Y are independent.

Proof. By applying Theorem A.2 and Theorem A.4, we have $0 \leq \mathbb{I}(X; Y) = \mathbb{H}(X) - \mathbb{H}(X|Y)$. The equality holds if and only if $\mathbb{I}(X; Y) = 0$, or X and Y are independent (see Corollary A.2). $\blacksquare$

Theorem A.6.

Given three random variables X, Y, Z satisfying $\mathbb{H}(X|Y) = \mathbb{H}(X|Z)$. Then the following expression must hold

$$p(x|y) = p(x|z). \quad (\text{A.11})$$

Proof. Since $\mathbb{H}(X|Y) = \mathbb{H}(X|Z)$, then $\mathbb{H}(X) - \mathbb{H}(X|Y) = \mathbb{H}(X) - \mathbb{H}(X|Z)$, i.e., $\mathbb{I}(X; Y) = \mathbb{I}(X; Z)$. By employing the data processing inequalities (Theorem A.3), this is equivalent to the occurrences of the following two Markov chains

$$X \rightarrow Y \rightarrow Z, \quad (\text{A.12})$$

and

$$X \rightarrow Z \rightarrow Y. \quad (\text{A.13})$$

Now note that

$$\begin{aligned} p(x, y, z) &= p(y, z)p(x|y, z) \\ &= p(y, z)p(x|y), \end{aligned} \quad (\text{A.14})$$

where the last equality holds from (A.12), i.e., X is independent of Z when conditioned on Y . Similarly, we also note that

$$\begin{aligned} p(x, y, z) &= p(y, z)p(x|y, z) \\ &= p(y, z)p(x|z), \end{aligned} \quad (\text{A.15})$$

where the last equality holds from (A.13), i.e., X is independent of Y when conditioned on Z . Then it follows from (A.14) and (A.15) that

$$p(x|y) = p(x|z). \blacksquare \quad (\text{A.16})$$

Bibliography

- [1] J. Vidal, "Toward direct brain-computer communication," Annual Review of Biophysics and Bioengineering, vol. 2, no. 1, pp. 157–80, 1973.
- [2] E. Niedermeyer and F. da Silva, "Electroencephalography: basic principles," Clinical applications and related fields, Wolters Kluwer Health, 2005.
- [3] E. Burmistrov, A. Matlashov, H. Sandin, L. Schultz, P. Volegov, and M. Espy, "Optimization and Configuration of SQUID Sensor Arrays for a MEG-MRI System," IEEE Transactions on Applied Superconductivity, vol. 23, no. 3, pp. 1601304–1601304, 2013.
- [4] J. Mellinger, G. Schalk, C. Braun, H. Preissl, W. Rosenstiel, N. Birbaumer, and A. Kubler, "An MEG-based brain-computer interface (BCI)," NeuroImage, vol. 36, no. 3, pp. 581–593, 2007.
- [5] H. Berger, "Ueber das Elektrenkephalogramm des Menschen," Archives fur Psychiatrie Nervenkrankheiten, vol. 87, pp. 527–570, 1929.
- [6] A. Kubler, V. Mushahwar, L. Hochberg, and J. Donoghue, "BCI meeting 2005-workshop on clinical issues and applications," IEEE Trans. Neural Syst. Rehabil. Eng., vol. 14, no. 2, pp. 131–134, 2006.
- [7] J. Wolpaw and E. Wolpaw, "Brain-computer interfaces: principles and practice," Oxford University Press, 2012.
- [8] P.-J. Kindermans, H. Verschore, D. T. W. Verstraeten, and B. Schrauwen, "A P300 BCI for the masses: prior information enables instant unsupervised spelling," 2012.
- [9] P.-J. Kindermans, D. Verstraeten, and B. Schrauwen, "A Bayesian Model for Exploiting Application Constraints to Enable Unsupervised Training of a P300-based BCI," PLOS ONE, vol. 7, no. 4, p. e33758, 2012.
- [10] P. J. Kindermans, H. Verschore, and B. Schrauwen, "A Unified Probabilistic Approach to Improve Spelling in an Event-Related Potential-Based Brain-Computer Interface," IEEE Transactions on Biomedical Engineering, vol. 60, no. 10, pp. 2696–2705, 2013. 10.
- [11] K. Pieter-Jan, T. Michael, M. Klaus-Robert, and S. Benjamin, "Integrating dynamic stopping, transfer learning and language models in an adaptive zero-training ERP speller," Journal of Neural Engineering, vol. 11, no. 3, p. 035005, 2014.
- [12] "EEG/ERP data available for free public download," 2018. Available: https://sccn.ucsd.edu/arno/fam2data/publicly_available_EEG_data.html.

- [13] M. Kaper, P. Meinicke, U. Grossekhoefer, T. Lingner, and H. Ritter, "BCI competition 2003-data set IIb: support vector machines for the P300 speller paradigm," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1073–1076, 2004.
- [14] B. Blankertz, K. R. Muller, D. J. Krusienski, G. Schalk, J. R. Wolpaw, A. Schlogl, G. Pfurtscheller, J. R. Millan, M. Schroder, and N. Birbaumer, "The BCI competition III: validating alternative approaches to actual BCI problems," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 153–159, 2006.
- [15] U. Hoffmann, J.-M. Vesin, T. Ebrahimi, and K. Diserens, "An efficient P300-based brain-computer interface for disabled subjects," *Journal of Neuroscience Methods*, vol. 167, no. 1, pp. 115–125, 2008.
- [16] O. S. and, L. Bougrain, C. Saavedra, E. Bojorges, and G. Gentiletti, "P300-speller public-domain database," 2011. Available: <http://akimpech.izt.uam.mx/p300db>.
- [17] J. Jin, Z. A. Brendan, W. S. Eric, B. Clemens, H. Petar, W. Xingyu, and N. Christa, "An adaptive P300-based control system," *Journal of Neural Engineering*, vol. 8, no. 3, p. 036006, 2011.
- [18] A. Lenhardt, M. Kaper, and H. J. Ritter, "An Adaptive P300-Based On-line Brain-Computer Interface," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 16, no. 2, pp. 121–130, 2008.
- [19] L. Tao, G. Leslie, G. Shangka, and H. Bo, "An online brain-computer interface using non-flashing visual evoked potentials," *Journal of Neural Engineering*, vol. 7, no. 3, p. 036003, 2010.
- [20] V. Kha and D. Eryk, "Optimal Length-Constrained Segmentation and Subject-Adaptive Learning for Real-time Arrhythmia Detection," in *ACM/IEEE Conference on Connected Health: Applications, Systems, and Engineering (CHASE), Big Data for Health Workshop*, 2018.
- [21] B. O. Mainsah, L. M. Collins, K. A. Colwell, E. W. Sellers, D. B. Ryan, K. Caves, and C. S. Throckmorton, "Increasing BCI communication rates with dynamic stopping towards more practical use: an ALS study," *Journal of Neural Engineering*, vol. 12, no. 1, p. 016013, 2015.
- [22] B. O. Mainsah, K. A. Colwell, L. M. Collins, and C. S. Throckmorton, "Utilizing a Language Model to Improve Online Dynamic Data Collection in P300 Spellers," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 837–846, 2014.
- [23] J. M. Clements, E. W. Sellers, D. B. Ryan, K. Caves, L. M. Collins, and C. S. Throckmorton, "Applying dynamic data collection to improve dry electrode system performance for a P300-based brain-computer interface," *Journal of Neural Engineering*, vol. 13, no. 6, p. 066018, 2016.
- [24] C. S. Throckmorton, K. A. Colwell, D. B. Ryan, E. W. Sellers, and L. M. Collins, "Bayesian Approach to Dynamically Controlling Data Collection in P300 Spellers," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, no. 3, pp. 508–517, 2013. 23529202[pmid] *IEEE Trans Neural Syst Rehabil Eng*.

- [25] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, Domain-Adversarial Training of Neural Networks, pp. 189–209. Cham: Springer International Publishing, 2017.
- [26] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” Advances in Neural Information Processing Systems, pp. 2672–2680, 2014.
- [27] P. Isola, J. Zhu, T. Zhou, and A. Efros, “Image-to-image translation with conditional adversarial networks,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [28] C. Ledid, F. L. Theis, Huszar, J. Caballero, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, “Photo-realistic single image super-resolution using a generative adversarial network,” in IEEE Conference on Computer Vision and Pattern Recognition, 2017.
- [29] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, “Generative adversarial text to image synthesis,” in International Conference on Machine Learning, 2016.
- [30] S. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, “Learning what and where to draw,” Advances in Neural Information Processing Systems, pp. 217–225, 2016.
- [31] S. Gurumurthy, R. Sarvadevabhatla, and V. Radhakrishnan, “Deligan: Generative adversarial networks for diverse and limited data,” in IEEE Conference On Computer Vision and Pattern Recognition (CVPR), 2017.
- [32] J. Zhu, P. Krahenbuhl, E. Shechtman, and A. Efros, “Generative visual manipulation on the natural image manifold,” in European Conference on Computer Vision, pp. 597–613, Springer, 2016.
- [33] A. Brock, T. Lim, J. Ritchie, and N. Weston, “Neural photo editing with introspective adversarial networks,” in 6th International Conference on Learning Representations (ICLR), 2017.
- [34] Y. Grandvalet, “Bagging Equalizes Influence,” Machine Learning, vol. 55, no. 3, pp. 251–270, 2004.
- [35] L. Breiman, “Stacked regressions,” Machine Learning, vol. 24, no. 1, pp. 49–64, 1996.
- [36] H. J. D. Le, and A. Segnini, “Kaggle Ensemble Guide,” 2018. Available: <https://mlwave.com/kaggle-ensembling-guide/>.
- [37] H. Cecotti and A. Graser, “Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 3, pp. 433–445, 2011. 57.
- [38] H. Cecotti, M. P. Eckstein, and B. Giesbrecht, “Single-Trial Classification of Event-Related Potentials in Rapid Serial Visual Presentation Tasks Using Supervised Spatial Filtering,” IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 11, pp. 2030–2042, 2014. 10.

- [39] N. Jrad, M. Congedo, R. Phlypo, S. Rousseau, R. Flamary, F. Yger, and A. Rakotomamonjy, “sw-SVM: sensor weighting support vector machines for EEG-based brain-computer interfaces,” Journal of Neural Engineering, vol. 8, no. 5, p. 056004, 2011.
- [40] A. Rakotomamonjy and V. Guigue, “BCI Competition III: Dataset II- Ensemble of SVMs for BCI P300 Speller,” IEEE Transactions on Biomedical Engineering, vol. 55, no. 3, pp. 1147–1154, 2008.
- [41] K. Vo, D. Nguyen, H. H. Kha, and E. Dutkiewicz, “Subject-Independent P300 BCI using Ensemble Classifier, Dynamic Stopping and Adaptive Learning,” in IEEE Global Communications Conference (GLOBECOM), 2017.
- [42] V. A. Kha, T. Pham, D. Nguyen, H. H. Kha, and E. Dutkiewicz IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2018.
- [43] H. H. Kha, K. Vo, and D. Q. Hung, “Brainwave-controlled applications with the Emotiv EPOC using support vector machine,” in 2016 3rd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), pp. 106–111, 2016.
- [44] K. Vo, D. N. Nguyen, H. H. Kha, and E. Dutkiewicz, “Real-time analysis on ensemble SVM scores to reduce P300-Speller intensification time,” in 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 4383–4386, 2017.
- [45] M. Bear, B. Connors, and A. Michael, Neuroscience: exploring the brain 3rd Edition. Lippincott Williams and Wilkins, 2007.
- [46] B. Biswal, M. Mennes, X. Zuo, S. Gohel, C. Kelly, S. Smith, C. Beckmann, J. Adelstein, R. Buckner, and S. Colcombe, “Toward discovery science of human brain function,” in Proceedings of the National Academy of Sciences, vol. 107, pp. 4734–4739, 2010.
- [47] J. Nolte, The human brain: An introduction to its functional anatomy. 2002.
- [48] B. He, S. Gao, H. Yuan, and J. Wolpaw, “Brain-Computer Interfaces,” Neural Engineering, pp. 87–151, 2013.
- [49] S. Mason and G. Birch, “A general framework for brain-computer interface design,” IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 11, no. 1, pp. 70–85, 2003.
- [50] J. Wolpaw, D. McFarland, T. Vaughan, and G. Schalk, “The Wadsworth Center brain-computer interface (BCI) research and development program,” IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 11, no. 2, pp. 1–4, 2003.
- [51] L. Hochberg, M. Serruya, G. Friehs, J. Mukand, M. Saleh, A. Caplan, A. Branner, D. Chen, R. Penn, and J. Donoghue, “Neuronal ensemble control of prosthetic devices by a human with tetraplegia,” Nature, vol. 442, no. 7099, pp. 164–171, 2006.
- [52] L. Nicolas-Alonso and J. Gomez-Gil, “Brain computer interfaces, a review,” Sensors, vol. 12, no. 2, pp. 1211–1279, 2012.

- [53] J. Wolpaw, G. Loeb, B. Allison, E. Donchin, O. Nascimento, W. Heetderks, F. Nijboer, W. Shain, and J. Turner, "BCI meeting 2005: workshop on signals and recording methods," IEEE Trans. Neural Systems and Rehabilitation Engineering, vol. 14, no. 2, pp. 138–141.
- [54] W. Grill and J. Mortimer, "Neural and connective tissue response to long-term implantation of multiple contact nerve cuff electrodes," Journal of biomedical materials research, vol. 50, no. 2, pp. 215–226, 2000.
- [55] V. Polikov, P. Tresco, and W. Reichert, "Response of brain tissue to chronically implanted neural electrodes," Journal of Neuroscience Methods, vol. 148, no. 1, pp. 1–18, 2005.
- [56] S. Musallam, B. Corneil, B. Greger, H. Scherberger, and R. Andersen, "Cognitive control signals for neural prosthetics," Science, vol. 305, no. 5681, pp. 258–262, 2004.
- [57] L. Hochberg, D. Bacher, B. Jarosiewicz, N. Masse, J. Simeral, J. Vogel, S. Haddadin, J. Liu, S. Cash, and S. P, "Reach and grasp by people with tetraplegia using a neurally controlled robotic arm," Nature, vol. 485, no. 7398, pp. 372–375., 2012.
- [58] C. Henle, M. Schuettler, J. Rickert, and T. Stieglitz, "Towards Electro-corticographic Electrodes for Chronic Use in BCI Applications," Towards Practical Brain-Computer Interfaces, Springer, pp. 85–103, 2013.
- [59] W. Freeman, L. Rogers, M. Holmes, and D. Silbergeld, "Spatial spectral analysis of human electrocorticograms including the alpha and gamma bands," Journal of Neuroscience Methods, vol. 95, no. 2, pp. 111–121, 2000.
- [60] M. Slutzky, L. Jordan, T. Krieg, M. Chen, D. Mogul, and L. Miller, "Optimal spacing of surface electrode arrays for brain-machine interface applications," Journal of Neural engineering, vol. 7, no. 2, p. 026004, 2010.
- [61] T. Ball, M. Kern, I. Mutschler, A. Aertsen, and A. Schulze-Bonhage, "Signal quality of simultaneously recorded invasive and non-invasive EEG," NeuroImage, vol. 46, no. 3, pp. 708–716, 2009.
- [62] C. Gaona, M. Sharma, Z. Freudenburg, J. Breshears, D. Bundy, J. Roland, D. Barbour, G. Schalk, and E. Leuthardt, "Nonuniform high-gamma (60-500 Hz) power changes dissociate cognitive task and anatomy in human cortex," The Journal of Neuroscience, vol. 31, no. 6, pp. 2091–2100, 2011.
- [63] J. Wilson, E. Felton, P. Garell, G. Schalk, and J. Williams, "ECoG factors underlying multimodal control of a brain-computer interface," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 14, no. 2, pp. 246–250, 2006.
- [64] E. Leuthardt, K. Miller, G. Schalk, R. Rao, and J. Ojemann, "Electrocorticography-based brain computer interface-the Seattle experience," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 14, no. 2, pp. 194–198, 2006.
- [65] G. Schalk, J. Kubanek, K. Miller, N. Anderson, E. Leuthardt, J. Ojemann, D. Limbrick, D. Moran, L. Gerhardt, and J. Wolpaw, "Decoding

- two-dimensional movement trajectories using electrocorticographic signals in humans,” Journal of Neural engineering, vol. 4, no. 3, p. 264, 2007.
- [66] N. Ramsey, M. van de Heuvel, K. Kho, and F. Leijten, “Towards human BCI applications based on cognitive brain systems: an investigation of neural signals recorded from the dorsolateral prefrontal cortex,” IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 14, no. 2, pp. 214–217, 2006.
- [67] S. Fazli, J. Mehnert, J. Steinbrink, G. Curio, A. Villringer, K. Muller, and B. Blankertz, “Enhanced performance by a hybrid NIRS-EEG brain computer interface,” NeuroImage, vol. 59, no. 1, pp. 519–529, 2012.
- [68] A. Villringer, J. Planck, C. Hock, L. Schleinkofer, and U. Dirnagl, “Near infrared spectroscopy (NIRS): a new tool to study hemodynamic changes during activation of brain function in human adults,” Neuroscience Letters, vol. 154, no. 1, pp. 101–104, 1993.
- [69] S. Coyle, T. Ward, and C. Markham, “Brain-computer interface using a simplified functional near-infrared spectroscopy system,” Journal of Neural engineering, vol. 4, no. 3, p. 219, 2007.
- [70] S. Ranganatha, Y. Hoshi, and C. Guan, “Near infrared spectroscopy based braincomputer interface,” International Society for Optics and Photonics, pp. 434–442, 2005.
- [71] M. Naito, Y. Michioka, K. Ozawa, M. Kiguchi, and T. Kanazawa, “A communication means for totally locked-in ALS patients based on changes in cerebral blood volume measured with near-infrared light,” IEEE transactions on information and systems, vol. 90, no. 7, pp. 1028–1037, 2007.
- [72] S. Ogawa, T. Lee, A. Kay, and D. Tank, “Brain magnetic resonance imaging with contrast dependent on blood oxygenation,” Proceedings of the National Academy of Sciences, vol. 87, no. 24, pp. 9868–9872, 1990.
- [73] B. Sorger, J. Reithler, B. Dahmen, and R. Goebel, “A Real-Time fMRI-Based Spelling Device Immediately Enabling Robust Motor-Independent Communication,” Current Biology, vol. 22, no. 14, pp. 1333–1338, 2012.
- [74] D. Hermes, K. Miller, M. Vansteensel, E. Aarnoutse, F. Leijten, and N. Ramsey, “Neurophysiologic correlates of fMRI in human motor cortex,” Human brain mapping, vol. 33, no. 7, pp. 1689–1699, 2012.
- [75] S. Cherry and M. Phelps, “Imaging brain function with positron emission tomography,” Brain mapping: The methods, New York, 2002.
- [76] T. Holly, B. Abbott, M. Al-Mallah, D. Calnon, M. Cohen, F. DiFilippo, E. Ficaro, M. Freeman, R. Hendel, and D. Jain, “Single photonemission computed tomography,” Journal of nuclear cardiology, vol. 17, no. 5, pp. 941–973, 2010.
- [77] R. Van Heertum, R. Tikofsky, and M. Ichise, “Functional cerebral SPECT and PET imaging 3rd edn,” Lippincott Williams and Wilkins., 2000.
- [78] J. Long, Y. Li, T. Yu, and Z. Gu, “Target Selection With Hybrid Feature for BCI-Based 2-D Cursor Control,” IEEE Transactions on Biomedical Engineering, vol. 59, no. 1, pp. 132–140, 2012.

- [79] T. Carlson and J. d. R. Millan, "Brain-Controlled Wheelchair: A Robotic Architecture," IEEE Robotics and Automation Magazine, vol. 20, no. 1, pp. 65–73, 2013.
- [80] A. Haider and R. Fazel-Rezai, "Application of P300 Event-Related Potential in Brain-Computer Interface," Event-Related Potentials and Evoked Potentials, 2017.
- [81] B. Allison, D. McFarland, G. Schalk, S. Zheng, M. Jackson, and J. Wolpaw, "Towards an independent brain-computer interface using steady state visual evoked potentials," Clinical Neurophysiology, vol. 119, no. 2, pp. 399–408, 2008.
- [82] Y. Zhang, J. Jin, X. Qing, B. Wang, and X. Wang, "LASSO based stimulus frequency recognition model for SSVEP BCIs," Biomedical Signal Processing and Control, vol. 7, no. 2, pp. 104–111, 2012.
- [83] M. Middendorf, G. McMillan, G. Calhoun, and K. Jones, "Brain-computer interfaces based on the steady-state visual-evoked response," IEEE Transactions on Rehabilitation Engineering, vol. 8, no. 2, pp. 211–214, 2000.
- [84] E. Lalor, S. Kelly, C. Finucane, R. Burke, R. Smith, R. Reilly, and G. McDarby, "Steady-state VEP-based brain-computer interface control in an immersive 3D gaming environment," EURASIP journal on applied signal processing, vol. 2005, pp. 3156–3164, 2005.
- [85] X. Zhang, L. Yuhong, F. Zhang, J. Ren, Y. Sun, Q. Yang, and H. Huang, "On Design and Implementation of Neural-Machine Interface for Artificial Legs," IEEE Transactions on Industrial Informatics, vol. 8, no. 2, pp. 418–429, 2012.
- [86] P. Martinez, H. Bakardjian, and A. Cichocki, "Fully online multicommand braincomputer interface with visual neurofeedback using SSVEP paradigm," Computational intelligence and neuroscience, vol. 2007, pp. 13–13, 2007.
- [87] K. Muller, C. Anderson, and G. Birch, "Linear and nonlinear methods for brain-computer interfaces," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 11, no. 2, pp. 165–169, 2003.
- [88] L. Trejo, R. Rosipal, and B. Matthews, "Brain-computer interfaces for 1-D and 2-D cursor control: designs using volitional control of the EEG spectrum or steady-state visual evoked potentials," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 14, no. 2, pp. 225–229, 2006.
- [89] N. Beuchat, R. Chavarriaga, S. Degallier, and J. Millan, "Offline decoding of upper limb muscle synergies from EEG slow cortical potentials," in 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 3594–3597, 2013.
- [90] N. Birbaumer and L. Cohen, "Brain-computer interfaces: communication and restoration of movement in paralysis," The Journal of Physiology, vol. 579, no. 3, pp. 621–636, 2007.
- [91] H. Shibasaki and M. Hallett, "What is the Bereitschaftspotential?," Clinical Neurophysiology, vol. 117, no. 11, pp. 2341–2356, 2006.

- [92] N. Birbaumer, T. Hinterberger, A. Kubler, and N. Neumann, "The thoughttranslation device (TTD): neurobehavioral mechanisms and clinical outcome," IEEE Trans. Neural Systems and Rehabilitation Engineering, vol. 11, no. 2, pp. 120–123, 2003.
- [93] T. Hinterberger, N. Neumann, M. Pham, A. Kubler, A. Grether, N. Hofmayer, B. Wilhelm, H. Flor, and N. Birbaumer, "A multimodal brain-based feedback and communication system," Experimental Brain Research, vol. 154, no. 4, pp. 521–526, 2004.
- [94] M. Pham, T. Hinterberger, N. Neumann, A. Kubler, N. Hofmayer, A. Grether, B. Wilhelm, J. Vatine, and N. Birbaumer, "An auditory brain-computer interface based on the self-regulation of slow cortical potentials," Neurorehabilitation and Neural Repair, vol. 19, no. 3, pp. 206–218, 2005.
- [95] A. Kubler and K. Muller, "2007," Toward Brain-Computer Interfacing, The MIT Press, vol. 1-25, An Introduction to Brain-Computer Interfacing.
- [96] O. Bai, V. Rathi, P. Lin, D. Huang, H. Battapady, D. Fei, L. Schneider, E. Houdayer, X. Chen, and M. Hallett, "Prediction of human voluntary movement before it occurs," Clinical Neurophysiology, vol. 122, no. 2, pp. 364–372, 2011.
- [97] E. Friedrich, D. McFarland, C. Neuper, T. Vaughan, P. Brunner, and J. Wolpaw, "A scanning protocol for a sensorimotor rhythm-based brain-computer interface," Biological Psychology, vol. 80, no. 2, pp. 169–175, 2009.
- [98] P. Boord, A. Craig, Y. Tran, and H. Nguyen, "Discrimination of left and right leg motor imagery for brain-computer interfaces," Medical and biological engineering and computing, vol. 48, no. 4, pp. 343–350, 2010.
- [99] M. Grosse-Wentrup and B. Scholkopf, "A Review of Performance Variations in SMR-Based Brain Computer Interfaces (BCIs)," Brain-Computer Interface Research, Springer Berlin Heidelberg, pp. 39–51, 2013.
- [100] B. Blankertz, C. Sannelli, S. Halder, E. Hammer, A. Kubler, K. Muller, G. Curio, and T. Dickhaus, "Neurophysiological predictor of SMR-based BCI performance," NeuroImage, vol. 51, no. 4, p. 1303, 2010.
- [101] G. Muller-Putz, C. Pokorny, D. Klobassa, and P. Horki, "A single-switch BCI based on passive and imagined movements: Toward restoring communication in minimally conscious patients," International Journal of Neural Systems, vol. 23, no. 2, 2013.
- [102] B. Graimann, J. Huggins, S. Levine, and G. Pfurtscheller, "Visualization of significant ERD/ERS patterns in multichannel EEG and ECoG data," Clinical Neurophysiology, vol. 113, no. 1, pp. 43–47, 2002.
- [103] D. McFarland, L. McCane, S. David, and J. Wolpaw, "Spatial filter selection for EEG-based communication," Electroencephalography and clinical Neurophysiology, vol. 103, no. 3, pp. 386–394, 1997.
- [104] G. Pfurtscheller, G. Muller-Putz, A. Schlogl, B. Graimann, R. Scherer, R. Leeb, C. Brunner, C. Keinrath, F. Lee, and G. Townsend, "15 years of BCI research at Graz university of technology: current projects," IEEE

- Transactions on Neural Systems and Rehabilitation Engineering, vol. 14, no. 2, pp. 205–210, 2006.
- [105] J. Wolpaw, “Brain-computer interface research comes of age: traditional assumptions meet emerging realities,” Journal of motor behavior, vol. 42, no. 6, pp. 351–353, 2010.
 - [106] F. Cincotti, D. Mattia, F. Aloise, S. Bufalari, G. Schalk, G. Oriolo, A. Cherubini, M. Marciani, and F. Babiloni, “Non-invasive brain-computer interface system: Towards its application as assistive technology,” Brain research bulletin, vol. 75, no. 6, pp. 796–803, 2008.
 - [107] A. Royer, A. Doud, M. Rose, and B. He, “EEG control of a virtual helicopter in 3-dimensional space using intelligent control strategies,” IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 18, no. 6, pp. 581–589, 2010.
 - [108] K. Muller and B. Blankertz, “Toward noninvasive brain-computer interfaces,” IEEE Signal Processing Magazine, vol. 23, no. 5, pp. 126–128, 2006.
 - [109] G. Muller-Putz, R. Scherer, G. Pfurtscheller, and R. Rupp, “EEG-based neuroprosthesis control: A step towards clinical practice,” Neuroscience Letters, vol. 382, no. 1-2, pp. 169–174, 2005.
 - [110] G. Pfurtscheller, G. Muller, J. Pfurtscheller, H. Gerner, and R. Rupp, “Thought: control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia,” Neuroscience Letters, vol. 351, no. 1, pp. 33–36, 2003.
 - [111] R. Leeb, D. Friedman, G. Muller-Putz, R. Scherer, M. Slater, and G. Pfurtscheller, “Self-paced (asynchronous) BCI control of a wheelchair in virtual environments: a case study with a tetraplegic,” Computational intelligence and neuroscience, pp. 1–8, 2007.
 - [112] B. Allison and C. Neuper, “Could Anyone Use a BCI,” Brain-Computer Interfaces, Applying our Minds to Human-Computer Interaction, vol. Springer London, pp. 35–54, 2010.
 - [113] M. Conson, S. Sacco, M. Sara, F. Pistoia, D. Grossi, and L. Trojano, “Selective motor imagery defect in patients with locked-in syndrome,” Neuropsychologia, vol. 46, no. 11, pp. 2622–2628, 2008.
 - [114] E. Curran, P. Sykacek, M. Stokes, S. Roberts, W. Penny, I. Johnsrude, and A. Owen, “Cognitive tasks for driving a brain-computer interfacing system: a pilot study,” IEEE Trans. Neural Syst. Rehabil. Eng., vol. 12, no. 1, pp. 48–54, 2004.
 - [115] G. Birch, Z. Bozorgzadeh, and S. Mason, “Initial on-line evaluations of the LF-ASD brain-computer interface with able-bodied and spinal-cord subjects using imagined voluntary motor potentials,” IEEE Trans. Neural Syst. Rehabil. Eng., vol. 10, no. 4, pp. 219–224, 2002.
 - [116] C. Anderson and J. Bratman, “Translating thoughts into actions by finding patterns in brainwaves,” Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems, pp. 1–6, 2008.
 - [117] D. Craig and H. Nguyen, “Adaptive EEG Thought Pattern Classifier for Advanced Wheelchair Control,” in Proc. IEEE 29th Annu. Int. Conf. Eng. Med. Biol. Soc., pp. 2544–2547, 2007.

- [118] E. Friedrich, R. Scherer, and C. Neuper, "The effect of distinct mental strategies on classification performance for brain-computer interfaces," International Journal of Psychophysiology, vol. 84, no. 1, pp. 86–94, 2012.
- [119] M. Teli and C. Anderson, "Nonlinear dimensionality reduction of electroencephalogram (EEG) for Brain Computer interfaces," in Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 2486–2489, 2009.
- [120] E. W. Sellers and E. Donchin, "A P300-based brain-computer interface: Initial tests by ALS patients," Clinical Neurophysiology, vol. 117, no. 3, pp. 538–548, 2006.
- [121] T. W. Picton, "The P300 wave of the human event-related potential," J. Clin. Neurophysiol., vol. 9, pp. 456–479, 1992.
- [122] R. C. Panicker, S. Puthusserypady, and Y. Sun, "An Asynchronous P300 BCI With SSVEP-Based Control State Detection," IEEE Transactions on Biomedical Engineering, vol. 58, no. 6, pp. 1781–1788, 2011. 55.
- [123] K. Tobias and K. Andrea, "Beyond maximum speed-a novel two-stimulus paradigm for brain-computer interfaces based on event-related potentials (P300-BCI)," Journal of Neural Engineering, vol. 11, no. 5, p. 056004, 2014.
- [124] E. Yin, T. Zeyl, R. Saab, T. Chau, D. Hu, and Z. Zhou, "A Hybrid Brain-Computer Interface Based on the Fusion of P300 and SSVEP Scores," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 23, no. 4, pp. 693–701, 2015. 8.
- [125] W. Speier, C. Arnold, J. Lu, A. Deshpande, and N. Pouratian, "Integrating Language Information With a Hidden Markov Model to Improve Communication Rate in the P300 Speller," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 22, no. 3, pp. 678–684, 2014. 9.
- [126] W. Speier, C. Arnold, and N. Pouratian, "Integrating language models into classifiers for BCI communication: a review," Journal of Neural Engineering, vol. 13, no. 3, p. 031002, 2016.
- [127] T. Zeyl, E. Yin, M. Keightley, and T. Chau, "Adding Real-Time Bayesian Ranks to Error-Related Potential Scores Improves Error Detection and Auto-Correction in a P300 Speller," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 24, no. 1, pp. 46–56, 2016.
- [128] B. Wittevrongel and M. M. Van Hulle, "Faster P300 Classifier Training Using Spatiotemporal Beamforming," International Journal of Neural Systems, vol. 26, no. 03, p. 1650014, 2016.
- [129] G. Townsend and V. Platsko, "Pushing the P300-based brain-computer interface beyond 100 bpm: extending performance guided constraints into the temporal domain," Journal of Neural Engineering, vol. 13, no. 2, p. 026024, 2016.
- [130] G. Dornhege, B. Blankertz, G. Curio, and K. R. Muller, "Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms," IEEE Transactions on Biomedical Engineering, vol. 51, no. 6, pp. 993–1002, 2004.

- [131] L. A. Farwell and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials," Electroencephalography and Clinical Neurophysiology, vol. 70, no. 6, pp. 510–523, 1988.
- [132] S. Sutton, M. Braren, J. Zubin, and E. R. John, "Evoked-Potential Correlates of Stimulus Uncertainty," Science, vol. 150, no. 3700, p. 1187, 1965.
- [133] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain computer interfaces for communication and control," Clinical Neurophysiology, vol. 113, no. 6, pp. 767–791.
- [134] U. Hoffmann, G. Garcia, J. Vesin, K. Diserens, and T. Ebrahimi, "A Boosting Approach to P300 Detection with Application to Brain-Computer Interfaces," 2005.
- [135] N. Birbaumer and L. G. Cohen, "Brain-computer interfaces: communication and restoration of movement in paralysis," The Journal of Physiology, vol. 579, no. Pt 3, pp. 621–636, 2007.
- [136] R. C. Panicker, S. Puthusserypady, and Y. Sun, "Adaptation in P300 Brain-Computer Interfaces: A Two-Classifer Cotraining Approach," IEEE Transactions on Biomedical Engineering, vol. 57, no. 12, pp. 2927–2935, 2010.
- [137] B. Blankertz, S. Lemm, M. Treder, S. Haufe, and K.-R. M  ijller, "Single-trial analysis and classification of ERP components - A tutorial," NeuroImage, vol. 56, no. 2, pp. 814–825, 2011.
- [138] S. Martijn, H. Johannes, B. Benjamin, H. Stefan, D. Thorsten, and T. Michael, "Optimizing event-related potential based brain-computer interfaces: a systematic evaluation of dynamic stopping methods," Journal of Neural Engineering, vol. 10, no. 3, p. 036025, 2013.
- [139] I. Kathner, S. Wriessnegger, G. Muller-Putz, A. Kubler, and S. Halder, "Effects of mental workload and fatigue on the P300, alpha and theta band power during operation of an ERP (P300) brain-computer interface," Biological Psychology, vol. 102, pp. 118–129, 2014.
- [140] W. Speier, C. W. Arnold, A. Deshpande, J. Knall, and N. Pouratian, "Incorporating advanced language models into the P300 speller using particle filtering," Journal of Neural Engineering, vol. 12, no. 4, p. 046018, 2015.
- [141] E. D. Adrian and B. H. C. Matthews, "The Berger rhythm: Potential changes from the occipital lobes in man," Brain, vol. 57, pp. 355–385, 1934.
- [142] H. H. Jasper and L. Carmichael, "Electrical potentials from the intact human brain," Science, vol. 81, pp. 51–53, 1935.
- [143] F. A. Gibbs, H. Davis, and W. G. Lennox, "The electro-encephalogram in epilepsy and in conditions of impaired consciousness," Archives of Neurology and Psychiatry, vol. 34, pp. 1133–1148, 1935.
- [144] Wikipedia, "Hans Berger," 2018. Available: https://en.wikipedia.org/wiki/Hans_Berger.

- [145] S. Slobounov, "An Introduction to the Event-Related Potential Technique. Cognitive Neuroscience. By Steven J Luck," The Quarterly Review of Biology, vol. 81, no. 2, pp. 201–202, 2006.
- [146] H. Davis, P. A. Davis, A. L. Loomis, E. N. Harvey, and G. Hobart, "Electrical reactions of the human brain to auditory stimulation during sleep," Journal of Neurophysiology, vol. 2, pp. 500–514, 1939.
- [147] W. G. Walter, R. Cooper, V. J. Aldridge, W. C. McCallum, and A. L. Winter, "Contingent negative variation: An electric sign of sensorimotor association and expectancy in the human brain," Nature, vol. 203, pp. 380–384, 1964.
- [148] S. Sutton, M. Braren, J. Zubin, and E. R. John, "Evoked potential correlates of stimulus uncertainty," Science, vol. 150, pp. 1187–1188, 1965.
- [149] M. G. H. Coles and M. D. Rugg, "Event-related potentials: An introduction," Electrophysiology of Mind, pp. 1–26, 1995.
- [150] S. A. Hillyard and T. W. Picton, "Electrophysiology of cognition," Handbook of Physiology: Section 1. The Nervous System: Volume 5. Higher Functions of the Brain, Part 2, pp. 519–584, 1987.
- [151] T. W. Picton, O. G. Lins, and M. Scherg, The recording and analysis of event-related potentials, vol. 10 of Handbook of Neuropsychology. New York: Elsevier, 1995.
- [152] R. Naatanen and T. W. Picton, "N2 and automatic versus controlled processes," Cerebral Psychophysiology: Studies in Event-Related Potentials, 1986.
- [153] D. Regan, "Human Brain Electrophysiology: Evoked Potentials and Evoked Magnetic Fields in Science and Medicine," 1989.
- [154] D. A. Jeffreys and J. G. Axford, "Source locations of pattern-specific components of human visual evoked potentials," Experimental Brain Research, vol. 16, pp. 1–21, 1972.
- [155] F. Di Russo, A. Martinez, and S. A. Hillyard, "Source analysis of event-related cortical activity during visuo-spatial attention," Cerebral Cortex, vol. 13, pp. 486–499, 2003.
- [156] S. A. Hillyard and L. Anllo-Vento, "Event-related brain potentials in the study of visual selective attention," Proceedings of the National Academy of Sciences USA, vol. 95, pp. 781–787, 1998.
- [157] F. Di Russo, A. Martinez, M. I. Sereno, S. Pitzalis, and S. A. Hillyard, "Cortical sources of the early components of the visual evoked potential," Human Brain Mapping, vol. 15, pp. 99–111, 2002.
- [158] E. K. Vogel and S. J. Luck, "The visual N1 component as an index of a discrimination process," Psychophysiology, vol. 37, pp. 190–123, 2000.
- [159] S. J. Luck, S. A. Hillyard, M. Mouloua, M. G. Woldorff, V. P. Clark, and H. L. Hawkins, "Effects of spatial cuing on luminance detectability: Psychophysical and electrophysiological evidence for early selection," Journal of Experimental Psychology: Human Perception and Performance, vol. 20, pp. 887–904, 1994.

- [160] G. R. Mangun and S. A. Hillyard, "Modulations of sensory-evoked brain potentials indicate changes in perceptual processing during visual-spatial priming," Journal of Experimental Psychology: Human perception and performance, vol. 17, pp. 1057–1074, 1991.
- [161] G. R. Mangun, "Neural mechanisms of visual selective attention," Psychophysiology, vol. 32, pp. 4–18, 1995.
- [162] E. Wascher, S. Hoffman, J. Sanger, and M. Grosjean, "Visuo-spatial processing and the N1 component of the ERP," Psychophysiology, vol. 46, pp. 1270–1277, 2009.
- [163] R. Freunberger, W. Klimiesch, M. Doppelmayr, and Y. Holler, "Visual P2 component is related to theta phase-locking," Neuroscience Letters, vol. 426, pp. 181–186, 2007.
- [164] S. J. Luck and S. A. Hillyard, "Electrophysiological correlates of feature analysis during visual search," Psychophysiology, vol. 32, pp. 291–308, 1994.
- [165] J. R. Folstein and C. Van Petten, "Influence of cognitive control and mismatch on the N2 component of the ERP: A review," Psychophysiology, vol. 45, pp. 152–170, 2008.
- [166] B. A. Eriksen and C. W. Eriksen, "Effects of noise letters upon identification of a target letter in a non- search task," Perception and Psychophysics, vol. 16, pp. 143–149, 1974.
- [167] A. Pfefferbaum, J. M. Ford, W. B. J., and B. S. Kopell, "ERPs to response production and inhibition," Electroencephalography and Clinical Neurophysiology, vol. 60, pp. 423–434, 1985.
- [168] J. Polich, "Updating P300: An integrative theory of P3a and P3b," Clinical Neurophysiology, vol. 118, no. 10, pp. 2128–2148, 2007.
- [169] N. K. Squires, K. C. Squires, and S. A. Hillyard, "Two varieties of long-latency positive waves evoked by unpredictable auditory stimuli," Electroencephalography and Clinical Neurophysiology, vol. 38, pp. 387–401, 1975.
- [170] R. Johnson and Jr, "A triarchic model of P300 amplitude," Psychophysiology, vol. 23, pp. 367–384, 1986.
- [171] W. S. Pritchard, "Psychophysiology of P300," Psychology Bulletin, vol. 89, pp. 506–540, 1981.
- [172] T. W. Picton, "The P300 wave of the human event-related potential," Journal of Clinical Neurophysiology, vol. 9, pp. 456–479, 1992.
- [173] J. Polich, "Clinical application of the P300 event-related brain potential," Physical Medicine and Rehabilitation Clinics of North America, vol. 15, pp. 133–161, 2004.
- [174] J. Polich and A. Kok, "Cognitive and biological determinants of P300: An integrative review," Biological Psychology, vol. 41, pp. 103–146, 1995.
- [175] C. C. Duncan-Johnson and E. Donchin, "On quantifying surprise: The variation of event-related potentials with subjective probability," Psychophysiology, vol. 14, pp. 456–467, 1977.

- [176] R. Johnson and Jr, "P300: A model of the variables controlling its amplitude," Annals of the New York Academy of Sciences, vol. 425, pp. 223–229, 1984.
- [177] J. B. Isreal, G. L. Chesney, C. D. Wickens, and E. Donchin, "P300 and tracking difficulty: Evidence for multiple resources in dual-task performance," Psychophysiology, vol. 17, pp. 259–273, 1980.
- [178] M. Kutas, G. McCarthy, and E. Donchin, "Augmenting mental chronometry: The P300 as a measure of stimulus evaluation time," Science, vol. 197, pp. 792–795, 1977.
- [179] A. Magliero, T. R. Bashore, M. G. H. Coles, and E. Donchin, "On the dependence of P300 latency on stimulus evaluation processes," Psychophysiology, vol. 21, pp. 171–186, 1984.
- [180] Wikipedia, "Signal Averaging," 2018. Available: https://en.wikipedia.org/wiki/Signal_averaging.
- [181] C. M. Gray, P. Konig, A. K. Engel, and W. Singer, "Oscillatory responses in cat visual cortex exhibit inter-columnar synchronization which reflects global stimulus properties," Nature, vol. 338, pp. 334–337., 1989.
- [182] C. Woody, "Characterization of an adaptive filter for the analysis of variable latency neuroelectric signals," Medical and Biological Engineering, vol. 5, pp. 539–553, 1967.
- [183] S. Makeig, "Auditory event-related dynamics of the EEG spectrum and effects of exposure to tones," Electroencephalography and Clinical Neurophysiology, vol. 86, pp. 283–293, 1993.
- [184] C. Tallon-Baudry, O. Bertrand, C. Delpuech, and J. Pernier, "Stimulus specificity of phase-locked and non-phase-locked 40 Hz visual responses in humans," Journal of Neuroscience, vol. 16, pp. 4240–4249, 1996.
- [185] M. Iwasaki, C. Kellinhaus, A. Alexopoulos, R. Burgess, A. Kumar, Y. Han, H. O. Luders, and J. Leigh, "Effects of eyelid closure, blinks, and eye movements on the electroencephalogram," Clinical Neurophysiology, vol. 116, no. 4, pp. 878–885, 2005.
- [186] G. Oliver Roesler, Stuttgart, "EEG Eye State Data Set," 2013.
- [187] Wikipedia, "Emotiv," 2018. Available: <https://en.wikipedia.org/wiki/Emotiv>.
- [188] Wikipedia, "Nyquist-Shannon sampling theorem," 2018. Available: https://en.wikipedia.org/wiki/Nyquist-Shannon_sampling_theorem.
- [189] A. Widmann, E. Schr  ger, and B. Maess, "Digital filter design for electrophysiological data - a practical approach," Journal of Neuroscience Methods, vol. 250, pp. 34–46, 2015.
- [190] H. H. Kha and V. Kha, "Real-time brainwave-controlled interface using P300 component in EEG signal processing," in 2016 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), pp. 235–240, 2016.

- [191] A. Radford, L. Metz, and â. i. P. o. t. S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” in 5th International Conference on Learning Representations (ICLR), 2016.
- [192] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” Advances in Neural Information Processing Systems, vol. 2226-2234, 2016.
- [193] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and . A. Courville, “Adversarially learned inference,” in Proceedings of the International Conference on Learning Representations, 2017.
- [194] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, “Learning from simulated and unsupervised images through adversarial training,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [195] M. Zhang, K. T. Ma, J. H. Lim, Q. Zhao, and J. Feng, “Deep future gaze: Gaze anticipation on egocentric videos using adversarial networks,” in IEEE Conference on Computer Vision and Pattern Recognition, pp. 4372–4381, 2017.
- [196] J. Liebelt and C. Schmid, “Multi-view object class detection with a 3d geometric model,” in Conference on Computer Vision and Pattern Recognition (CVPR), 2010.
- [197] M. Stark, M. Goesele, and B. Schiele, “Back to the future: Learning shape models from 3d CAD data,” British Machine Vision Conference (BMVC), pp. 1–11, 2010.
- [198] B. Sun and K. Saenko, “From virtual to reality: Fast adaptation of virtual object detectors to real domains,” in British Machine Vision Conference (BMVC), 2014.
- [199] I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning, MIT Press,” 2016.
- [200] L. Maaten and G. Hinton, “Visualizing Data Using t-SNE,” Journal of Machine Learning Research, vol. 9, pp. 2579–2605, 2008.
- [201] A. Jamieson, M. Giger, K. Drukker, H. Lui, Y. Yuan, and N. Bhooshan, “Exploring Nonlinear Feature Space Dimension Reduction and Data Representation in Breast CADx with Laplacian Eigenmaps and t-SNE,” Medical Physics, vol. 37, no. 1, pp. 339–351, 2010.
- [202] P. Hamel and D. Eck, “Learning Features from Music Audio with Deep Belief Networks,” in Proceedings of the International Society for Music Information Retrieval Conference, pp. 339–344, 2010.
- [203] I. Gashi, V. Stankovic, C. Leita, and O. Thonnard, “An Experimental Study of Diversity with Off-the-shelf AntiVirus Engines,” in Proceedings of the IEEE International Symposium on Network Computing and Applications, pp. 4–11, 2009.
- [204] I. Wallach and R. Liliean, “The Protein-Small-Molecule Database, A Non-Redundant Structural Resource for the Analysis of Protein-Ligand Binding,” Bioinformatics, vol. 25, no. 5, pp. 615–620, 2009.

- [205] C. Olah, “Visualizing Representations: Deep Learning and Human Beings Christopher Olah’s blog, 2015,” 2015. Available: <https://colah.github.io/posts/2015-01-Visualizing-Representations/>.
- [206] Z.-H. Zhou, Ensemble Methods: Foundations and Algorithms. Chapman & Hall/CRC, 2012.
- [207] L. Breiman, “Bagging Predictors,” Machine Learning, vol. 24, no. 2, pp. 123–140, 1996.
- [208] R. Tibshirani, “Regression Shrinkage and Selection via the Lasso,” Journal of the Royal Statistical Society. Series B (Methodological), vol. 58, no. 1, pp. 267–288, 1996.
- [209] D. H. Wolpert and W. G. Macready, “An Efficient Method To Estimate Bagging’s Generalization Error,” Machine Learning, vol. 35, no. 1, pp. 41–55, 1999.
- [210] D. H. Wolpert, “Stacked generalization,” Neural Networks, vol. 5, no. 2, pp. 241–259, 1992.
- [211] P. Smyth and D. Wolpert, “Linearly Combining Density Estimators via Stacking,” Machine Learning, vol. 36, no. 1, pp. 59–83, 1999.
- [212] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A Training Algorithm for Optimal Margin Classifiers,” in Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT ’92) (D. Haussler, ed.), pp. 144–152, ACM Press, New York, NY, USA, 1992.
- [213] M. Welling, “Fisher Linear Discriminant Analysis,” Technical Report, University of Toronto, 2000.
- [214] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” Mathematics of Control, Signals and Systems, vol. 2, no. 4, pp. 303–314, 1989.
- [215] V. Nair and G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines,” 2010.
- [216] A. M. Martinez and A. C. Kak, “PCA versus LDA,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 2, pp. 228–233, 2001.
- [217] Wikipedia, “Linear Discriminant Analysis,” 2018. Available: https://en.wikipedia.org/wiki/Linear_discriminant_analysis.
- [218] B. O. Mainsah, L. M. Collins, and C. S. Throckmorton, “Using the detectability index to predict P300 speller performance,” Journal of Neural Engineering, vol. 13, no. 6, p. 066007, 2016.
- [219] K. Vo, D. Nguyen, H. H. Kha, and E. Dutkiewicz, “Dynamic Stopping Using eSVM Scores Analysis for Event-Related Potential Brain-Computer Interfaces,” in 11th International Symposium on Medical Information and Communication Technology (ISMICT), 2017.
- [220] Y. Li, C. Guan, H. Li, and Z. Chin, “A self-training semi-supervised SVM algorithm and its application in an EEG-based brain computer interface speller system,” Pattern Recognition Letters, vol. 29, no. 9, pp. 1285–1294, 2008.

- [221] M. Spuler, W. Rosenstiel, and M. Bogdan, "Adaptive SVM-Based classification increases performance of a MEG-Based brain-computer interface (BCI)," in Proceedings of the 22nd international conference on Artificial Neural Networks and Machine Learning - Volume Part I, (2407034), pp. 669–676, Springer-Verlag, 2012.
- [222] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," in Proceedings of the 13th International Conference on Neural Information Processing Systems (NIPS'00), (3008808), pp. 388–394, MIT Press, 2000.
- [223] C. P. Diehl and G. Cauwenberghs, "SVM incremental learning, adaptation and optimization," in Proceedings of the International Joint Conference on Neural Networks, 2003., vol. 4, pp. 2685–2690 vol.4, 2003.
- [224] L. o. Bottou and C.-J. Lin, "Support Vector Machine Solvers," in Large Scale Kernel Machines (L. o. Bottou, O. Chapelle, Dennis, and J. Weston, eds.), pp. 301–320, MIT Press, 2007. e decoste.
- [225] A. Fougeres, "Multivariate Extremes," Extreme Values in Finance, Telecommunications, and the Environment, Monographs on Statistics and Applied Probability 99, Chapman and Hall/CRC, 2004.
- [226] M. Palaniswami and A. Shilton, "Adaptive support vector machines for regression," in Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on, vol. 2, pp. 1043–1049 vol.2, 2002.
- [227] R. Herbrich and J. Weston, "Adaptive margin support vector machines for classification," in 1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470), vol. 2, pp. 880–885 vol.2, 1999.
- [228] V. Losing, B. Hammer, and H. Wersing, "Incremental on-line learning: A review and comparison of state of the art algorithms," Neurocomputing, vol. 275, no. Supplement C, pp. 1261–1274, 2018.
- [229] J. Thomas and T. Cover, "Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing), Wiley-Interscience," 2006.
- [230] Wikipedia, "Jensen's Inequalities," 2018. Available: <https://en.wikipedia.org/wiki/Jensen>
- [231] Wikipedia, "Markov Chain," 2018. Available: https://en.wikipedia.org/wiki/Markov_chain.