Effective EEG Analysis for Advanced AI-Driven Motor Imagery BCI Systems

Natasha Padfield

Centre for Signal and Image Processing

Department of Electronic and Electrical Engineering

February 2022

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

Signed: Natasha Padfield

Date: 15/02/2022

Abstract

Developing effective signal processing for brain-computer interfaces (BCIs) and brain-machine interfaces (BMIs) involves factoring in three aspects of functionality: classification performance, execution time, and the number of data channels used. The contributions in this thesis are centered on these three issues. Contributions are focused on the classification of motor imagery (MI) data, which is generated during imagined movements.

Typically, EEG time-series data is segmented for data augmentation or to mimic buffering that happens in an online BCI. A multi-segment decision fusion approach is presented, which takes consecutive temporal segments of EEG data, and uses decision fusion to boost classification performance. It was computationally lightweight and improved the performance of four conventional classifiers. Also, an analysis of the contributions of electrodes from different scalp regions is presented, and a subset of channels is recommended.

Sparse learning (SL) classifiers have exhibited strong classification performance in the literature. However, they are computationally expensive. To reduce the test-set execution times, a novel EEG classification pipeline consisting of a genetic-algorithm (GA) for channel selection and a dictionary-based SL module for classification, called GABSLEEG, is presented. Subject-specific channel selection was carried out, in which the channels are selected based on training data from the subject. Using the GA-recommended subset of EEG channels reduced the execution time by 60% whilst preserving classification performance.

Although subject-specific channel selection is widely used in the literature, effective subject-independent channel selection, in which channels are detected using data from other subjects, is an ideal aim because it leads to lower training latency and reduces the number of electrodes needed. A novel convolutional neural network (CNN)-based subject-independent channels selection method is presented, called the integrated channel selection (ICS) layer. It performed on-a-par with or better than subject-specific channel selection. It

was computationally efficient, operating 12-17 times faster than the GA channel selection module. The ICS layer method was versatile, performing well with two different CNN architectures and datasets.

Acknowledgements

I am very grateful to the University of Strathclyde and its scholarship scheme for the valuable opportunity to carry out this research. My time at Strathclyde has been character-building, and a wonderful experience I will cherish.

Firstly, I would like to thank Prof. Jinachang Ren for his help and direction as first supervisor for over two years. I appreciate the research skills you have helped me build. Thank you for pushing me to investigate every avenue, and for the extensive feedback which led to the publication of three papers, at the time of writing, related to this PhD. I would also like to thank Dr Paul Murray for his guidance and encouragement as first supervisor during the final year of my PhD, and particularly whilst writing this Thesis. I also appreciate the feedback and suggestions he provided during his time as second supervisor. I thank Prof. Stephen Marshall for taking up the role of second supervisor, and for providing thought-provoking feedback during both of my end-of-year reviews. Finally, I thank Dr Lykourgos Petropoulakis for his feedback during my end-of-year reviews.

I also thank my colleagues at CeSIP for being so kind and welcoming, and for the discussions and knowledge transfer. I would also like to show my appreciation for the work of the Strathclyde Doctoral School and the opportunity to present my research through them.

I would also like to thank my family, who have been a source of strength throughout this whole process, even if they did not always understand what my research was about. I would especially like to thank my parents for their support throughout the Covid-19 pandemic, and my Uncle Bernard for encouraging my love of science and being the first person to suggest I pursue a doctoral degree.

I would also like to thank my supportive friends, especially those who always encouraged me to pursue a research degree. Thank you, Rubi, for always being so interested by my research and so encouraging from day one. And finally, thank you Alex, for your support through all stages of this process, for your generosity, and your unwavering encouragement at the toughest moments.

List of Tables

List of Figures

List of Acronyms

AAR	Adaptive Autoregressive
ACS	Automatic Channel Selection
ANOVA	Analysis of Variance
AR	Autoregressive
BCI	Brain-Computer Interface
BMI	Brain-Machine Interface
BP	Bereitschafts-Potentials
CCS	Correlation Coefficient Channel Selection
CNMF	Covariance and Non-Negative Matrix Factorization
CNN	Convolutional Neural Network
CSP	Common Spatial Patterns
CSSP	Common Spatio-Spectral Patterns
CSSSP	Common Sparse Spatio-Spectral Patterns
CWT	Continuous Wavelet Transform
DL	Deep Learning
DWT	Discrete Wavelet Transform
EEG	Electroencephalogram
EMD	Empirical Mode Decomposition
ELM	Extreme Learning Machine
EMG	Electromyogram
ERD	Event-Related Desynchronization

ERS	Event-Related Synchronization
EWT	Empirical Wavelet Transform
FBCSP	Filterbank Common Spatial Patterns
FD	Frequency-Domain
FFT	Fast Fourier Transform
FN	False Negatives
FP	False Positives
FPGA	Field-Programmable Gate Array
GA	Genetic Algorithm
GABSLEEG	Genetic Algorithm Band Power Sparse Learning Classification System for EEG
GPU	Graphical Processing Unit
h-CNN	Hybrid CNN
HN	Number of Neurons in Hidden Layer
ICS	Integrated Channel Selection
IEEG	Integrated Electroencephalogram
k-NN	k-Nearest Neighbour
kSVD	k Single Value Decomposition
LDA	Linear Discriminant Analysis
LR	Logistic Regression
LSTM	Long Short-Term Memory
MI	Motor imagery
MLP	Multilayer Perceptron

MPSO	Modified Particle Swarm Optimization
MTL	Mixed Data Transfer Learning
NB	Naïve Bayes
NMF	Non-Negative Matrix Factorization
OMP	Orthogonal Matching Pursuit
PSO	Particle Swarm Optimization
RBF	Radial Basis Function
RCSP	Regularized Common Spatial Patterns
RF	Random Forest
RMS	Root-Mean-Square
RMSD	Root-Mean-Square Deviation
RTL	Retraining Transfer Learning
SCSP	Sparse Common Spatial Patterns
SFFS	Sequential Floating Forward Search
SL	Sparse learning
STFT	Short-Time Fourier Transform
SVM	Support Vector Machine
TD	Time-Domain
TFD	Time-frequency domain
TN	True Negatives
TP	True Positives
TQWT	Tunable Q-Factor Wavelet Transform
WPD	Wavelet Packet Decomposition

Table of Contents

Chapter 1 : Introduction

This chapter opens with a discussion of the motivations and aims of the PhD research, and then goes on to list the main contributions. It closes with a summary of the layout of the rest of the thesis.

1.1 Motivations and Aims

1.1.1 EEG Technology for Brain-Computer Interfaces

Brain-computer interfaces (BCIs) have enabled humans to control computers and machines using just their brains. Some works in the literature refer to these systems as brain-machine interfaces (BMIs). The recording equipment used can be invasive, with electrodes being placed within the skull via a surgical procedure, or non-invasive, with electrodes or recording equipment remaining external to the body [1], [2]. Non-invasive techniques present a lower risk to the user, are faster to set up, can be used more widely with immediate effect and, currently, may be more viable for commercial systems for consumers who do not want to carry out surgery [1].

Of the non-invasive recording techniques available, electroencephalogram (EEG) recording has a relatively high time resolution, and it is also one of the cheapest, most portable, and easiest of the options to set up [1]. EEG signals are comprised of multi-channel time-series data, recoded using electrodes placed on the scalp, with each electrode producing a channel signal [3]. Other recording techniques include magnetoencephalography and functional magnetic resonance imaging, which have better spatial resolution than EEG, but require machinery that is expensive, high-power, and bulky, making them commercially unviable for many practical BCIs [1]. Functional near infrared spectroscopy, which monitors blood flow in brain tissues, is a more portable recording technique used in BCIs

and provides good spatial localization[1], [4]. It is often used in conjunction with EEG to exploit the time-domain resolution of EEG signals [4].

Signal processing and classification techniques enable EEG signals to be used in BCIs. Artificial intelligence techniques, particularly machine learning and deep learning, have been widely used to classify EEG signals [5]–[11]. Classification can be carried out on raw EEG data [6]–[9], however many studies extract features or time-frequency domain (TFD) images from EEG signals for classification [5], [10], [12]–[15].

This thesis is focused on classification techniques for motor imagery (MI) EEG signals. MI activity is generated in the brain when the subject imagines a movement, and MI EEG-based BCIs can be used for the intuitive control of prosthetics or robotic vehicles, neurorehabilitation, physical therapy, and gaming [16]–[20]. Machine learning and deep learning techniques have been widely used to classify MI EEG signals [8], [10], [11], [21]–[24]. This thesis is focused on investigating how the effectiveness of BCI classification systems based on machine learning and deep learning can be improved. In this work, the concept of 'effectiveness' is explored in terms of classification performance, user comfort, hardware investment and execution times. Whilst some core contributions in this thesis improve the classification accuracy of different BCI processing pipelines, other contributions improve computational times or reduce the number of electrodes needed for classification. These contributions are important because for BCIs to become widely used and commercially viable, they must have a high classification performance, but also acceptable computational latencies for training and during real-time use. They should also use the least electrodes possible to reduce cost and improve practicality. The rest of this section highlights the core motivations of this work.

1.1.2 Issues in Machine Learning Classification Pipelines

Many MI EEG classification systems segment data in the time domain during pre-processing. Sometimes, segmentation can be used to prune transitional activity

[7] or for data augmentation [9], [14], but it can also mimic a buffer in a real-time system [19], [25]. Relatively few studies in MI EEG classification have investigated how majority voting-based time-domain decision fusion between adjacent segments could be used to improve performance [8], [26]. However, time-domain decision fusion has been found to significantly improve classification accuracy in electromyography (EMG) signal classification [27]. EMG signals record electrical activity within muscles and are non-linear and non-stationary like EEG signals [28].

Many conventional MI EEG classification systems also use a static subset of EEG channels that is arbitrarily chosen from the full cohort of channels available in a dataset [10], [11], [22], [29]–[31]. In this thesis, a 'static' subset of EEG channels refers to a subset of channels selected using an unautomated method. A 'static' subset may be based on channels chosen 'arbitrarily', usually selected based on heuristics. It is common in the literature for studies to choose EEG channels for analysis without any justification for the selection of the channels beyond the fact that they are located in the vicinity of the scalp region associated with the motor region [21], [22], [31]. There is no formal name in the literature for these kinds of channel subsets, so in this thesis they are referred to as 'static' subsets, and the selection of channels is referred to as 'arbitrary'. Despite the fact that arbitrary channel selection is widely used [10], [11], [22], [29]–[31], the choice of MI EEG channels in the subset can significantly impact the classification performance [32], [33]. Discussion in the literature has been focused on whether increasing or decreasing the number of EEG channels in a static subset can improve performance [32], [33]. However, different scalp regions have been associated with different mental activities [34]–[37], such as motivated attention [34], problem-solving [38], planning and control of movement [38], and idling [36]. Although distinct MI EEG activity is observed on the central scalp region at the crown of the head [34], scalp areas associated with other mental activities such as concentration may also impact classification performance in a MI BCI. Despite this, the distinct contributions of channel groups surrounding the motor cortex to classification performance have not been studied. To elaborate, in the

literature reviewed, a study had not been conducted that involved adding and removing groups of electrodes that make up the regions neighboring the motor cortex, in order to investigate the influence of these electrode groups on performance. These regions consist of the central-parietal and central-frontal regions in this thesis. Furthermore, whilst studies have compared the performance of different conventional classifiers using fixed channel subsets [5], [14], [31], to the knowledge of this author no study has compared the performance of different classifiers across different EEG channel subsets to obtain a more generalized summary of performance.

1.1.3 Sparse Learning

Among machine learning techniques commonly used in the literature, sparse learning (SL) techniques have exhibited promising performance [11], [15], [39]–[42]. In particular, dictionary-based learning systems, which construct a sub-dictionary for each class using training segments of EEG data, then sparse encode test samples over the dictionaries and classify the samples based on the reconstruction error associated with each sub-dictionary, have been shown to have a strong classification performance [11], [39]. However, the performance of promising SL systems in the literature have not been assessed when subjected to constraints that might be found in practical systems, such as a reduced training data size, reduced numbers of EEG electrodes being available, or the introduction of the idle state class, which occurs when the user is not carrying out MI activity [11], [15], [39]–[42]. Furthermore, the computational times of successful dictionary-based SL systems have not been assessed in recent papers [11], [39], even though some encoding techniques that can be used, such as orthogonal matching pursuit (OMP), are known to be computationally expensive [43]. Computational times can affect user experience at all stages of BCI use [44]: longer training times can increase user fatigue and decrease practicality, whilst long classification times on the test set can make the system slow and affect the perceived ‘real-time’ performance users expect from computer interfaces [45].

1.1.4 Automated Channel Selection

Automated channel selection has been used in the literature to maintain the performance of an MI EEG classification system whilst improving test set computational times [44]. This is achieved by applying a channel selection algorithm to the training dataset to obtain a reduced subset of EEG channels. Only channels within the recommended subset are used during the testing phase, leading to improved computational times as a result of having less data to process [44]. There are two main categories of automated channel selection techniques: filter and wrapper techniques [44]. Filter techniques use statistical or analytical measures to rank EEG channels for selection [13], [44], [46], [47]. Wrapper techniques iteratively test candidate channel groups, searching for an optimal subset by evaluating the classification performance with each candidate [44], [48]–[51]. Wrapper techniques are more computationally expensive than filter techniques, but generally produce better results [44]. Metaheuristic wrapper techniques [49]–[51] use heuristic algorithms to speed up the channel selection process, at the expense of solutions that are possibly less-than-optimal, but often still acceptable [44], [49]–[51]. Metaheuristic techniques offer a trade-off between the high performance of wrapper techniques and faster processing [44]. Automated channel selection could be used to reduce the number of channels in a dictionary-based SL classifier to improve its computational times. Automated channel selection, and in particular metaheuristic channel selection, could ensure that the strong performance of the dictionary-based system is preserved when using fewer electrodes.

Many authors opt to use subject-specific channels, which are selected using the target training data [13], [23], [46]–[49], and a previous study has found that channels selected in a subject-independent (or 'cross-subject') manner performed worse than when selecting channels using subject-specific data [52]. Ideally, however, channels are selected in a way that is independent of the target subject, since this would enable fewer electrodes to be used with the target from the onset, possibly resulting in faster training times as well as faster testing times. Furthermore, using fewer electrodes with target subjects reduces hardware

costs, makes system set-up faster, and could make the system more comfortable for the end user. Since EEG data can experience great intra-subject variability [53], to obtain a generalizable subset, the channel selection process would involve using data from as many source subjects as possible. Moreover, due to computational constrains, it may be challenging to carry out subject-independent channel selection with some classification pipelines. For example, dictionary-based SL systems are memory intensive, requiring the whole dictionary to be stored in memory, as well as requiring memory for the encoding computations. Constructing a dictionary based on data from many source subjects, as well as carrying out the memory-expensive encoding calculations would be time-consuming and may occupy unacceptable amounts of memory [54], [55]. Furthermore, SL dictionaries are typically constructed based on features such as wavelet energy, CSP features and signal power features [11], [56], [57], but the frequency bands in which MI activity can occur can vary between subjects.

1.1.5 Deep Learning and Channel Selection

Deep learning systems, specifically convolutional neural networks (CNNs) [6], are being used at the forefront of subject-independent training for BCIs [7], [58]–[60]. CNNs have excelled at complex classification problems involving large and diverse training sets because they are able to represent data at a high level of abstraction [6], [61]–[63]. Possibly, this generalizable representation extracted from CNNs could facilitate improved subject-independent channel selection. Research on carrying out MI EEG channel selection within CNN frameworks is relatively sparse [23], [24], and many recent leading studies into CNNs for classification of MI EEG have not delved into the area of channel selection [7]–[9], [59], [64], [65]. In one study, Mzurikwao et al. [23] developed a CNN network architecture in which the weights of the network could be related back to the individual input data channels. After training the network, the channels associated with the greatest weights were selected. Then, the CNN was retrained with just the selected channels. However, not all CNN networks have architectures that can facilitate this kind of weight association with the input channels [7], [8], [65], and performance may be linked to the architecture of the

whole CNN. Zhang et al. [24] added an automatic channel selection (ACS) module to the start of a CNN classification network. The ACS module forces sparsity on the input data channels, with the aim of suppressing the contribution of redundant channels. Using the ACS module improved the classification performance, however all EEG channels still had to be input to the CNN, thus not leading to any of the benefits associated with using fewer electrodes. Furthermore, Mzurikwao et al. [23] and Zhang et al. [24] did not explicitly investigate the problem of subject-independent channel selection, and they do not test the effectiveness of their channel selection techniques on different CNN architectures.

1.1.6 Main Research Aims

Based on this discussion, the main aims of the research reported in this thesis were as follows:

- To study whether using time-domain decision fusion based on segmented EEG data can improve the classification accuracy of different machine learning pipelines. EEG data segmentation is typically carried out in the literature either for augmentation of the training dataset [9], [58], [66], or to mimic buffering in a real-time system, which would receive segments of EEG data [19], [25]. However, in the literature reviewed, there has been no extensive investigation into the impact of time-domain decision fusion of these segments across different classifiers.
- To investigate the contributions of different scalp regions (by association - channel subsets) to classification performance and compare the performance of popular classifiers across different static channel subsets. This is because signals linked to MI activity could appear in scalp regions neighbouring the central region [67] (pp. 87-91), which is the traditional motor area of the scalp [68].
- To investigate the effectiveness of a dictionary-based SL classifier within the context of practical constraints.

- To apply automated subject-specific channel selection to the SL classifier to improve test-set computational times.
- To develop a versatile approach for carrying out subject-independent channel selection within a CNN system.

1.2 Main Contributions in this Thesis

The contributions in this thesis can be summarized as follows:

1) **Implementation of a majority voting-based multi-segment decision fusion framework for improved classification performance.**
    - The framework led to a statistically significant improvement in classification accuracy for four different kinds of conventional classifiers. An extensive analysis was carried to assess the impact of the length of the segmentation window and the overlap between consecutive windows on classification performance.
    - A classification pipeline using the framework outperformed some other conventional classification systems presented in the literature.
    - Additional novel analysis was also carried out:
        - The contributions of electrodes from the central-parietal and central-frontal scalp regions were investigated. Static subsets were proposed based on this analysis.
        - The performances of various conventional classifiers were compared across a variety of static channel subsets.

2) **Implementation of a novel dictionary-based SL classifier with a metaheuristic channel selection module.**
    - A novel MI EEG classification pipeline comprising of a dictionary-based SL classifier and a metaheuristic (genetic algorithm (GA)) channel selection module, called GABSLEEG, is presented. Subject-specific channel selection is carried out.

- The GA selected a channel subset that preserved the classification accuracy whilst improving the test-set computational times of the system.
- The GABSLEEG system outperformed five different benchmarking and comparison approaches.
- Additional novel findings:
    - When comparing classifiers without GA channel selection, the SL classifier outperformed three machine learning classifiers.
    - The GA channel selection module was also effective in maintaining or improving the classification accuracy of the three benchmarking classifiers.

3) **An innovative Integrated Channel Selection Layer for subject-independent channel selection in CNNs**
    - A novel custom CNN layer for channel selection was implemented. This integrated channel selection (ICS) layer can be used as a preface to existing CNN networks during the training stage. Then, a post-training weight analysis is carried out to rank the EEG channels, with larger weights in the ICS layer being associated with more important channels.
    - When using the proposed new ICS layer, there was no statistically significant difference in performance when using either 1) channel subsets that were selected in a subject-independent fashion [52] or 2) channel subsets that were selected in a subject-specific way [47], [49]. Importantly, this indicates that the proposed ICS method is effective in selecting generalizable channels from subject-independent data.
    - Channels selected using the new ICS layer always obtained a higher classification accuracy than those selected using two other state-

of-the-art techniques [13], [47]. In 68% of instances, the improvement was statistically significant.

- The ICS layer method was found to be computationally efficient when compared to state-of-the-art filter techniques, and more computationally efficient than a wrapper channel selection method.
- Using transfer learning helped to improve performance when fewer EEG channels were used, leading to performance that was comparable to using the full cohort of electrodes.

1.3 Publications Resulting from this Thesis

Articles Published in Peer-Reviewed Journals

1. **N. Padfield**, J. Zabalza, H. Zhao, V. Masero and J. Ren, "EEG-Based Brain-Computer Interfaces Using Motor-Imagery: Techniques and Challenges," *Sensors,* 2019.
2. **N. Padfield**, J. Ren, C. Qing, P. Murray, H. Zhao and J. Zheng, "Multi-segment Majority Voting Decision Fusion for MI EEG Brain-Computer Interfacing,". *Cognitive Computation*, 2021.
3. **N. Padfield**, J. Ren, P. Murray, and H. Zhao, "Sparse learning of band power features with genetic channel selection for effective classification of EEG signals," *Neurocomputing*, *463*, pp.566-579, 2021.

Articles in Preparation for Submission

1. "A Versatile CNN-based Subject-Independent Channel Selection Method for MI EEG"

1.4 Layout of Thesis

The rest of this thesis is organized as follows:

- Chapter 2 provides relevant background information about EEG recording and the nature of EEG signals, as well as details about the datasets used in the contribution chapters.
- Chapter 3 presents a review of the literature focused on MI EEG classification, highlighting conceptual gaps that were explored and exploited in the contribution chapters. It also provides technical details of the machine learning and deep learning systems used in the contribution chapters.
- Chapter 4 introduces a majority voting-based multi-segment decision fusion approach for boosting classification performance. An analysis into the contribution of electrodes from different scalp regions was also carried out, and a comparison of classifiers using different electrode configurations is also included.
- Chapter 5 presents a dictionary-based SL classification pipeline with subject-specific channel selection.
- Chapter 6 proposes a versatile custom layer for subject-independent channel selection for CNN-based MI EEG classifiers. It also investigates how transfer learning can be used to improve performance.
- Chapter 7 summarizes the key points discussed in this thesis and concludes with a discussion of possible future work.
- The Appendix contains supplementary information for Chapter 6.

Chapter 2 : Background in EEG Signal Analysis

This chapter provides background information relevant to the contributions made in this thesis. It introduces EEG signals, including their physiological nature, recording techniques used, and characteristics relevant to signal processing. It also summarizes the datasets used for experimentation, including the recording protocols used.

2.1 EEG Data: Physiological Aspects and Recording Methods

2.1.1 EEG Signal Generation and Recording

The brain is constructed from approximately 100 billion nerve cells called neurons. These microscopic cells communicate with one another via electrical signals [69](pp. 1-41). The brain has distinct regions of cells which are associated with different activities, such as the sensorimotor region in the upper-central part of the brain which is responsible for planning and execution of movements, and the occipital cortex towards the back of the head which is involved in processing visual information [69](pp. 1-41)[34]. During complex, conscious tasks there is activity within a dominant brain source associated with the task, but there is continuous two-way communication with other brain regions [69](pp. 1-41). For example, during movement the dominant brain source is the sensorimotor region, however the occipital cortex provides visual information to guide movements [69](pp. 1-41).

The neuronal signals do not just travel within the brain, they also travel up towards the scalp. These signals are conducted from the brain source, up through other areas of the brain, through the blood-brain barrier, through the skull and onto the scalp. The skull, being a poor conductor, attenuates the brain

waves, resulting in microvolt signals reaching the scalp [70]. The blood-brain barrier, skull and skin produce perturbation and mixing within the signals known as volume conduction, and this presents challenges when tracing scalp signals to the original brain sources [70]. The scalp signals can be recorded non-invasively using sensors placed on the skin. In EEG research, multiple sensors are typically placed in contact with the scalp to record the signals across various scalp regions [34]. The number of sensors used varies, with open-access datasets like those used in this thesis typically using between 3 and 118 sensors [6], [71]–[74].

Figure 2.1 shows the extended 10-20 electrode placement scheme used for recording EEG data. The scalp is divided into 7 main regions, namely: the frontal (F), parietal (P), central (C), occipital (O), anterior (A), temporal (T) and frontopolar (Fp) regions. There are also regions which are transitional between major regions, such as the central-parietal (CP) region [34]. Electrodes with even numbers are located on the right side and those with odd numbers are located on the left side. Figure 2.2 shows a colour-coded map of the electrodes in the extended 10-20 layout, with each colour being associated with a different scalp region. Bold colours denote distinct regions, whereas pastel colours denote

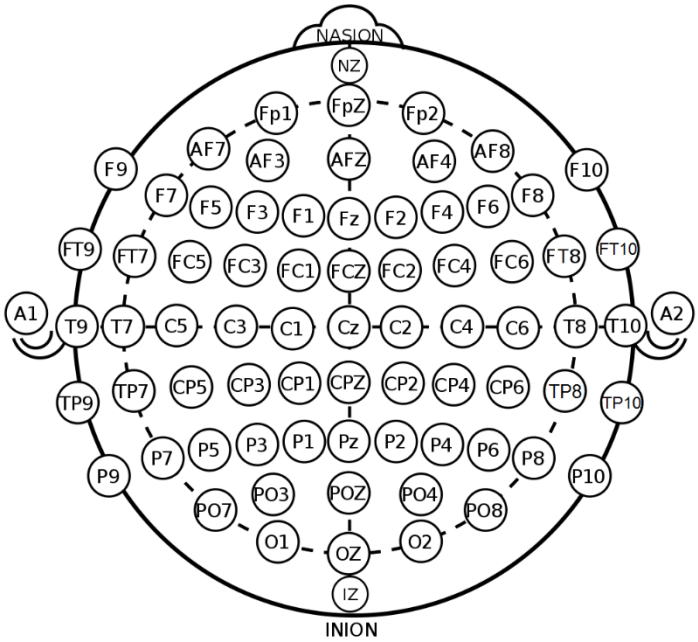

Figure 2.1: The electrode layout for the extended 10-20 EEG recording montage.

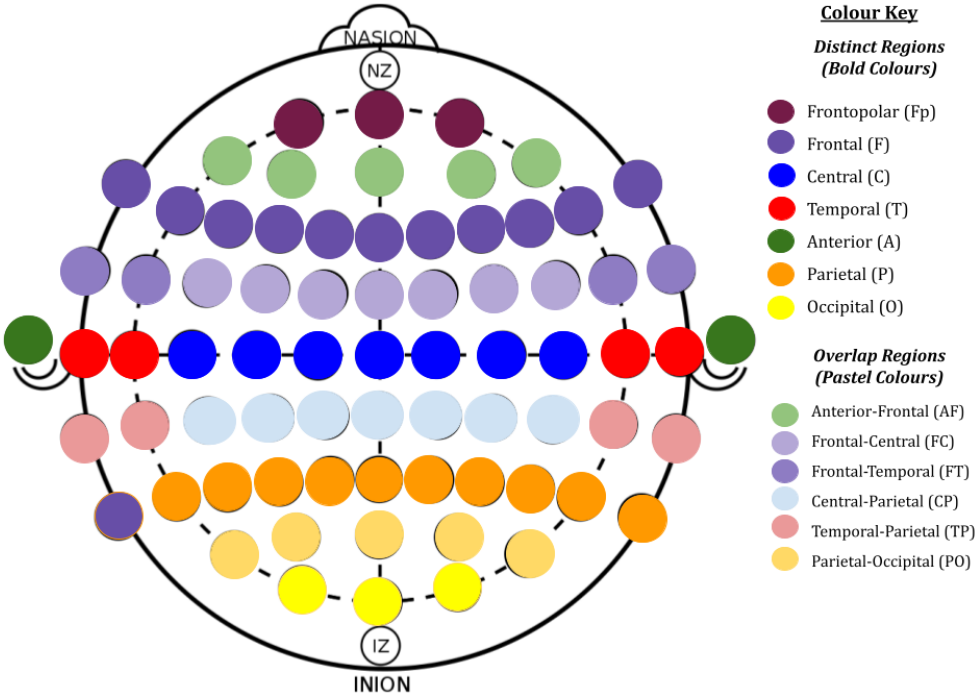

Figure 2.2: Colour-coded electrode map for the extended 10-20 layout, with each colour denoting a particular region.

transitional/overlap regions. The sensors are arranged in a cap which is placed on the subject, aligning the Cz electrode with the top-centre point of the head. Figure 2.3 shows an EEG cap with electrodes in place, being used by a test subject within a practical BCI set-up. In this thesis, the 'EEG recording montage' refers to the selection of electrodes placed on the scalp during recording. During discussions, the electrode signals are often referred to as 'channel signals'.

The 7 scalp regions of EEG electrodes mentioned previously, namely C, P, F, O T, A, and Fp are broadly associated with different brain activities, summarized in the following list:

- **C**: The central region is strongly associated with the sensorimotor cortex in the brain, and predominantly captures activity related to the execution and imagination of motor movements [34].
- **P:** The parietal region is associated with motivated attention and control [34].

Figure 2.3: An EEG cap with electrodes being used in a typical BCI set-up.

- **F:** The frontal region is associated with problem solving, emotions, memory and planning or control of voluntary movement [38].
- **O:** The occipital region is associated with visual processing. It is also the source of the posterior dominant rhythm, an alpha-band wave which occurs during relaxed, idle behaviour [36].
- **T:** The temporal lobe is associated with audio processing [37], and certain seizures [75].
- **A:** anterior region EEG has been associated with sleep, anesthesia, and drug use [76].
- **Fp:** related to behaviour [77] and mental health [78], including anxiety and depression.

EEG data has five clinically relevant frequency bands. These are the: delta band (<4)Hz, theta band (4-7)Hz, alpha band (8-12)Hz, beta band(12-30)Hz and gamma band (>30)Hz [34]. These frequency bands can vary slightly between research papers, but the variation is generally of less than 1Hz [67](pp. 36). EEG activity recorded at any one time will have a mixture of data in all these frequency

bands. Lower frequency bands, in particular the delta and theta bands, are generally associated with subconscious, low-level mental processes including sleep, whereas higher frequency bands tend to be associated with conscious thought and actions, or higher-level cognitive processing [67](pp. 36). The alpha band has been associated with both subconscious and conscious mental processes [34], [36].

One of the core advantages of EEG signals is that they provide a high temporal resolution at a relatively cheaper cost and through equipment with more portability when compared to other brain imaging techniques such as functional magnetic resonance imaging and magnetoencephalography, which both require powerful magnets, bulky machinery and high electricity input [18], [34]. However, EEG signals are considered to have a poorer spatial resolution when compared to these other brain imaging techniques [34].

2.1.1.1 Mathematical Modelling of EEG Signals

There is no straightforward, linear relationship between the EEG scalp electrodes and the underlying brain sources [70]. This sub-section discusses a mathematical model known as the forward model, which is used to explain the nature of scalp EEG signals. Although no contribution was made to this area in the thesis, this model is useful for explaining the non-trivial nature of EEG signals. The forward problem models the movement of electrical brain activity from the cortical source (dipole) within the brain, through the head and to the electrode on the scalp. Equation (2.1)[67](pp. 86-87) shows the forward problem:

$$\boldsymbol{V} = \boldsymbol{GJ} + \boldsymbol{n} \tag{2.1}$$

where:

- $\boldsymbol{V}$ is the matrix containing the scalp voltage measurements from N electrodes at T time samples: $\boldsymbol{V} = \begin{bmatrix} V(r_1,1) & \dots & V(r_1,T) \\ . & \dots & . \\ V(r_N,1) & \dots & V(r_N,T) \end{bmatrix}$, where r_x is the position of the scalp electrode.

- $\boldsymbol{G}$ is the gain matrix, which describes the potential measured at a scalp position r_x and which originated from a dipole at position r_{dip_y} with moment d. Given that p is the number of dipoles in the cortical model and e is a unit vector, then:

$$\boldsymbol{G} = \begin{bmatrix} g(r_1, r_{dip\_1}, e_{d1}) & \dots & g(r_1, r_{dip\_p}, e_{dp}) \\ . & \dots & . \\ g(r_N, r_{dip\_1}, e_{d1}) & \dots & g(r_N, r_{dip\_p}, e_{dp}) \end{bmatrix}.$$

- $\boldsymbol{J}$ is the matrix of dipole magnitudes, containing the magnitude of p dipoles over T time samples: $\boldsymbol{J} = \begin{bmatrix} ||d_{1,1}|| & \dots & ||d_{1,T}|| \\ . & \dots & . \\ ||d_{p,1}|| & \dots & ||d_{p,T}|| \end{bmatrix}$.

- $\boldsymbol{n}$ is the noise matrix.

Thus, the voltage at any scalp point can be decomposed into two additive parts: a brain signal produced by the combination of different sources ($\boldsymbol{GJ}$) and additive noise ($\boldsymbol{n}$). The 'pure' brain signal can be decomposed further into [67](pp. 83-91):

$$\boldsymbol{GJ} = \begin{bmatrix} g(r_1, r_{dip1}, e_{d1}) & \dots & g(r_1, r_{dip1}, e_{dp}) \\ . & \dots & . \\ g(r_N, r_{dip1}, e_{d1}) & \dots & g(r_N, r_{dip1}, e_{dp}) \end{bmatrix} \begin{bmatrix} ||d_{1,1}|| & \dots & ||d_{1,T}|| \\ . & \dots & . \\ ||d_{p,1}|| & \dots & ||d_{p,T}|| \end{bmatrix} \quad (2.2)$$

This 'pure' part of the brain signal produced at any point on the scalp is composed of a summation of the contributions from each dipole. The dipole's influence diminishes the further away it is located from the electrode's position on the scalp [67], [70](pp. 83-91). Therefore, essentially, the biological mixing which produces EEG signals is additive, based on the principle of superposition.

The core issue is that there are billions of neurons within the brain in continuous communication.

Solving the forward problem involves calculating the $\boldsymbol{G}$ matrix coefficients, which depend on the recording electrodes, the dipole configuration, and the attributes of the volume conductor model. The volume conductor model represents the electrical properties of the brain, skull, and scalp that the signals travel through before being recorded using EEG [67] (pp. 87-91). There are two standard models for representing volume conduction: a simplified spherical head model and a more accurate realistic head model [67] (pp. 87-91).

EEG signals are mixed with noise from a multitude of sources within and outside the body that produce signal anomalies known as artifacts. The human body is driven by electrical signals which control eye movements, heartbeats and muscle movements, and these signals produce electrocular, electrocardio and myoelectric artifacts within EEG recordings, respectively [79]. These are known as physiological artifacts. Artifacts can also be produced by the subject themselves through movements both voluntary and involuntary, which can disturb the electrical contact between the electrodes and the scalp [79]. EEG signals are also affected by the surrounding environment, including the 50Hz power line hum in recording equipment, light sources, and electrical equipment [79]. With good subject and environmental preparation, non-physiological artifacts can be reduced, whilst signal filtering and additional processing can be carried out to remove physiological artifacts generated by signals within the body [79]. Automated artifact detection is an active area of research. Some studies choose to use artifact-free data which has had trials containing artifacts removed following visual inspection by an expert [2], [9], [38].

2.1.2 Motor Imagery EEG

Motor imagery (MI) is the action of imagining movements. The accurate classification of EEG signals recorded during different imagined movements can be used in brain-computer interfaces (BCIs) for the control of prosthetic limbs,

graphical user interfaces, neurorehabilitation, artistic ventures, and gaming, to name a few examples [1], [18]–[20], [80]. Typically, MI of the hands, arms, legs, tongue, or joints such as elbow or wrist, are studied [7], [10], [23], [59], [81].

MI is characterized by two phases: i) event-related desynchronization (ERD), which is a decrease in activity in the alpha and beta frequency bands at the start of a MI event, and ii) event-related synchronization (ERS), an increase in the activity within these bands at the end of a MI event [68]. MI is contra-lateral, thus movements imagined on the right-hand side of the body manifest in brain signals recorded on the left-hand side of the scalp, and vice-versa [34]. Different subjects can have different sub-bands within the alpha and beta frequency bands within which ERD and ERS are most predominant [82]. Also, there is a human reaction time delay between a subject seeing a cue to perform MI and starting the task, generally of 0.7s [82] to 1s [81]. The gamma frequency band has also exhibited ERD/ERS activity but has not been widely used within MI EEG classification since this frequency band usually contains strong myoelectric noise from muscles [34], [83].

ERD/ERS activity occurs within the sensorimotor region of the brain, and is most noticeable on the central electrodes [34], [68]. Channels C3 and C4 have been found to have highly discriminative signals [34], however electrodes outside of the central region, particularly in the parietal and frontal regions, have also been found to be discriminative for MI EEG processing [47], [48], [84].

ERD changes may not be the only way to detect the onset of MI EEG activity. Bereitschafts-potentials (BPs) are low-amplitude spikes of activity indicating planning, preparation and/or commencement of conscious acts and can be observed prior to MI activity [85]. However, popular EEG datasets do not have these potentials marked, and for non-experts it could be challenging to differentiate them from noise [2], [86]–[89].

MI EEG activity is generally recorded from subjects in a structured data recording session. Subjects are usually informed when to start imagining a movement and when to stop imagining a movement through a graphical user

interface or audial signal such as a beep [2], [86], [89]. Each interval of recorded MI is called a 'trial'. Within one session subjects are generally asked to imagine several MI tasks in different trials, with a short break of a few seconds between each trial [2], [86], [89]. The ground truth in these recordings is the trigger signal used to control the graphical user interface or audial signal.

2.1.3 EEG Signal Features

EEG signals are stochastic and non-stationary [67](pp. 21)[90]. However, short EEG segments [67](pp. 21) of length 1.5s or less have been considered approximately stationary for signal processing purposes [91], [92].

There is also wide inter-subject variability within EEG signals for normal brain activities [81], [82]. Consider the action of imagining a right-hand movement. For any healthy subject this will cause a decrease in the power within the alpha and beta bands in the left sensorimotor region. However, there is inter-subject variability in the prominence of the power decrease, the latency between the imagined movement and the desynchronization, and the most predominant frequency sub-bands in which the desynchronization can occur [81], [82].

Figure 2.4 and Figure 2.5 demonstrate this inter-subject variability. Both images are based on EEG data generated during imagined right-hand movements for two subjects, labelled 'al' and 'ay' from Dataset IVa, which was used at BCI Competition III [86]. During a data recording trial, a cue was used to indicate when the subject is to start imagining the movement, and subjects carried out the task for 3.5s. In total, 140 right-hand MI trials were recorded per subject. Results for channel C3 were chosen in the figures since this channel is on the left hemisphere, which is where MI activity related to the right-hand side can be expected to manifest. Plots related to subject al are in blue and those related to subject ay are in red. More information on this dataset, and the other datasets used in this thesis, can be found in Section 2.2 of this chapter.

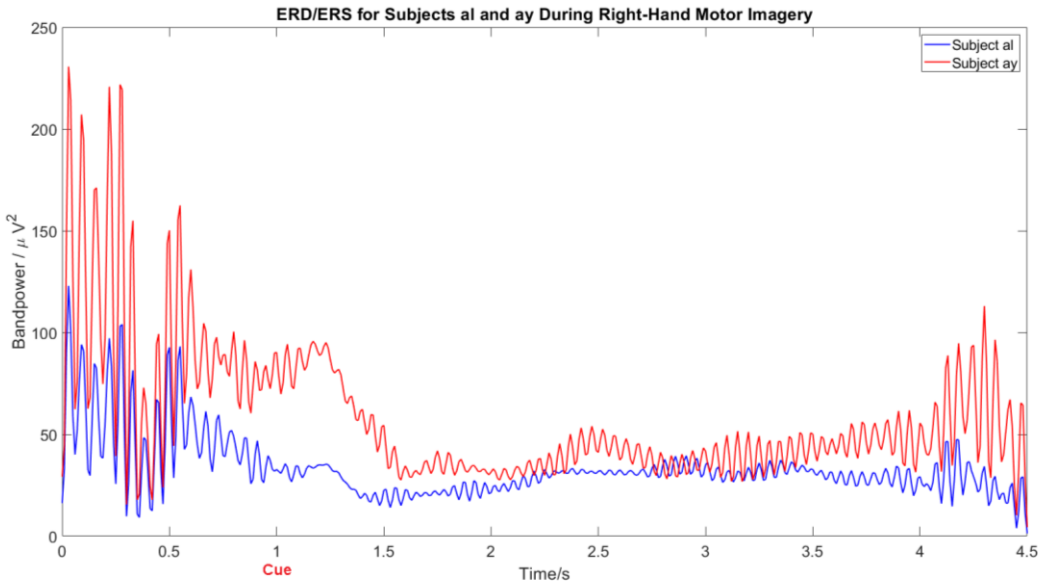

Figure 2.4: Plots of band power against time illustrating the ERD behaviour occurring during right-hand motor imagery for subjects al and ay from Dataset IVa from the BCI Competition III. The subjects are initially in the idle state, then at time 1s the cue to imagine the movement is given.

The plots in Figure 2.4 show the ERD behavior for both subjects. The plots were generated by first re-referencing the EEG trials using common average referencing (which involves subtracting the mean value of all the EEG channels in the trial from every sample on every channel [93]) and then using a standard method [68] to extract ERD. This method involves bandpass filtering the signals, squaring to obtain the band power, and then finding the average over the trials. This method generates the average ERD curves for each subject, and the results can be used for demonstrative purposes [68]. In this example, the ERD dynamics in the alpha band were studied, so the filter had a passband of (8-12)Hz [68]. For the first 1s of data, the subjects are in the idle state, then the cue is shown at the 1s mark, as indicated in the diagram with the label 'Cue'. From 1s to 4.5s the subjects imagined the right-hand movement. The ERD manifests as a decrease in band power that occurs after the 1s mark for both subjects. The plots capture inter-subject variability in MI EEG: the idle state signals prior to the cue have different amplitudes, with the signal associated with subject ay having a greater amplitude. After the cue occurs, both subjects experience a decrease in band power, but the decrease for subject al ends around 1.48s, whereas the decrease for subject ay ends around 1.58s, which is a ~0.1s difference.

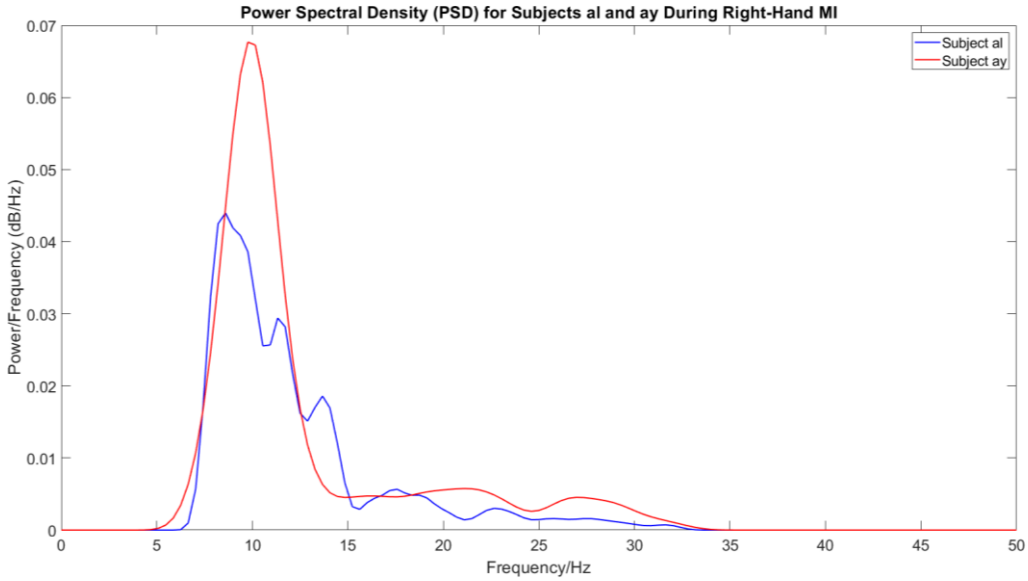

Figure 2.5: Power spectral density plots showing the differing frequency content during a right-hand MI task for subjects al and ay from Dataset IVa from the BCI Competition III.

Figure 2.5 shows power spectral density (PSD) plots for subjects al and ay when right hand MI was carried out. The signals were first bandpass filtered in the frequency range (8-32) Hz to capture content in the alpha and beta bands. The plots were obtained by finding the Welch periodogram for all trials of a given subject, then the periodograms were averaged across all the trials for the subject to plot the mean PSD plots. These plots capture the inter-subject variability in frequency content: the PSD plot for subject ay has a singular peak in the alpha band and greater frequency content in the beta band than subject al. Subject al has a spectrum with a wider spread and multiple peaks. The greatest peak of the spectrum for al is at a lower frequency than that of ay. These plots illustrate that the PSD in the bands normally associated with MI can vary between subjects.

EEG signals also exhibit intra-subject variability [53]. This variability can be due to naturally occurring microstates within the brain and due to signal drift during a single recording session [53]. Figure 2.6 illustrates intra-trial differences for subject aa from Dataset IVa when right-hand MI was carried out. Again, results for channel C3 are shown. The figure shows PSD plots for four different trials within the same recording session, namely trials 25, 50, 75, and 100. These trials were chosen to be dispersed equally over the recording session. The plots were

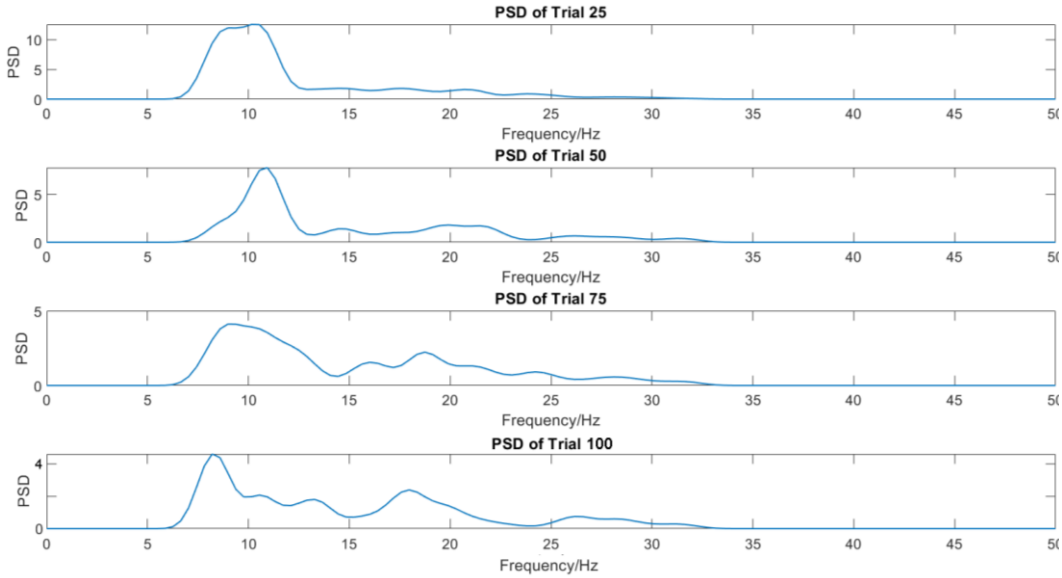

Figure 2.6: The PSD plots of four different trials of right-hand MI for subject aa.

obtained by first band pass filtering the signals in the range (8-32) Hz, then the Welch periodogram was calculated. From the figure it is evident that, even for the same subject, the frequency content of the MI signal can vary widely, with the peak frequency, width of the peak, and the level of beta-band frequency content changing noticeably between trials.

It has been shown that prior sleep [94], posture [95] or mood [78] can all significantly affect the EEG signals recorded. These issues could also result in recordings taken on different days suffering from poor intra-subject consistency, although the poor consistency could also be caused by changes in the placement of electrodes when the EEG cap is placed on the subject. In general, poor intra-subject consistency is due to classification algorithms not being able to adapt as opposed to a fault of the user. Also, drying of the electroconductive gel which forms a connection between the scalp and electrodes can affect signal quality [96].

These issues of inconsistencies over time, low repeatability and data which is unusable due to artifacts are not unique to EEG data and are experienced in other signals within the biomedical engineering sphere, such as electromyogram (EMG) signals [28], [97], which are recorded from muscles.

EEG is a non-invasive, cost-effective, and portable technology for brain activity recording [1]. It gives insight into neurological and mental processes which can provide invaluable information for researchers [34], [36]–[38], [75]. Data generated during MI tasks is widely studied [7], [10], [11], [30], [31], [98], [99] and is the focus of this thesis because it can be used in BCIs for applications including prosthetics [100], brain-controlled vehicles [19], [20], neurorehabilitation [1], [16], and gaming [16]. Effective EEG signal processing and classification is necessary for BCI technology [3], [16], [34]. However, EEG data presents processing challenges due to its non-stationary and non-linear nature, as well as its intra- and inter-subject variability [53], [81], [82]. Recent research has focused on improving signal processing [14], [30], [101], machine learning [10], [11], [13], [51], [99], and deep learning [6], [65], [102] techniques for MI EEG classification. This thesis makes contributions to machine learning and deep learning for MI EEG classification, and the following section introduces technical concepts from these areas used in the contribution chapters.

2.2 Datasets

This section describes the open-access datasets used in this thesis. There is no standard protocol used for MI EEG recording across datasets, and research protocol design is an active area of research [103].

2.2.1 BCI Competition III Dataset IVa

This dataset was presented by Dornhege et al. [86]. This is a two-class dataset, with MI of the right hand (class 1) and right foot (class 2). Data from five healthy subjects, labelled as *aa*, *al*, *av*, *aw* and *ay* is available. The data can be downloaded from [73]. Information about the recording protocol and data was obtained from [73] and [86]. This dataset is used in Chapter 4 and Chapter 5.

In the recording protocol, subjects were instructed, through a visual cue, to carry out a MI activity. The possible activities were MI of the left hand, right hand or right foot. However, in the dataset available for download [73], only MI of the right hand or right foot is available. The length of the MI trials is 3.5s because this is how long the cue was displayed on-screen. Between each trial,

subjects were asked to relax for an interval of between 1.75s and 2.25s. The duration of these breaks was varied randomly. Two types of visual cues were used:

- Type 1: A letter appears on-screen behind a fixation cross. The letter is associated with the MI activity that should be carried out. This may introduce small eye movements correlated with the target.
- Type 2: A randomly moving object on-screen indicates the MI activity to carry out. This may produce target-uncorrelated eye movements.

Data was recorded over 4 sessions, with 35 trials per class being recorded during each session. This means a total of 140 trials per class are available in the dataset. For subjects *al* and *aw* two sessions using each type of visual cue were recorded. For the other three subjects, one session of Type 1 and two sessions of Type 2 were recorded. Typically, the cue type is not factored when using the dataset for BCI research, unless a special focus on the effect of the cue type on the data recorded is being investigated [11], [86], [104]. In this thesis the research was not focused on the effect of cue type, so this information was not used when processing data.

The data was recorded using 118 Ag/AgCl EEG channels in the extended 10-20 system. The data was recorded using a 1000Hz sampling frequency at a 16 bit (0.1μV) resolution. The data was bandpass filtered between 0.05Hz and 200Hz to remove noise. The data was then down sampled to 100Hz by retaining every tenth sample. The creators of the dataset recommend using this version of the dataset [73]. Since frequencies above the down sampled folding frequency of 50Hz are outside the frequency bands of interest for MI EEG classification, the down sampled data was used in this thesis. Table 2.1 shows the number of trials

Table 2.1: The number of training and testing trials available for each subject.

Subject	No. Training Trials	No. Testing Trials
aa	168	112
al	224	56
av	84	196
aw	56	224
ay	28	252

marked for training and testing for each subject in the original competition for which the dataset was published [73], [86]. These partitions are available with the dataset and have been used in the literature [12], [15], [105], [106]. They are also used in Chapter 4. This dataset was chosen because it has been widely used in the literature for conventional [10], [49], [104] and deep learning [106], [107] systems.

2.2.2 BCI Competition IV Dataset I

The BCI Competition IV dataset I [108] was recorded specifically for classification of MI EEG data and the idle state. For this reason, it is used in Chapter 5 since one aim of that chapter is to assess the performance of the proposed system with data including the idle data. This dataset contains data from four healthy subjects, labelled 1a, 1b, 1f and 1g. The data was downloaded from [109] and information about the dataset was obtained from [2], [109] and [108].

The dataset has two MI classes, with subjects imagining movements from two of three possible classes, namely: right hand, left hand, or foot. Subjects 1a and 1f imagined left hand and foot movements, whilst subjects 1b and 1g imagined left-hand and right-hand movements. For the 'foot' class subjects could imagine movement of one foot or both feet at the same time, but it is not specified in the literature what specific subjects imagined. The dataset also has idle state data, thus presenting a three-class classification problem consisting of two MI states and the idle state.

Data was recorded using 59 Ag/AgCl electrodes that were distributed over and around the central part of the scalp, which is associated with sensorimotor activity. The data was recorded at a sampling frequency of 1000Hz at a 16-bit (0.1μV) resolution and was bandpass filtered in the range 0.05Hz to 200Hz. The data was then down sampled to 100Hz using a two-step process: first the data was filtered using a low-pass Chebychev Type-II filter (stopband frequency of 49Hz, stopband ripple of 50dB and order 10) and then the average of consecutive blocks on 10 samples were calculated to produce the down sampled signal. The down sampled signals are used for computational efficiency.

The dataset consists of two sub-datasets: the calibration dataset and the evaluation dataset. The calibration dataset consists of 4s long MI trials interspersed with 4s periods of the idle state. During recording of the calibration dataset, a visual cue appeared on-screen to indicate which MI activity the subject should carry out. These cues were arrows pointing towards the left, right or downwards to indicate left-hand, right-hand or foot MI, respectively. During the idle state, the subjects observed a blank screen for 2s and a fixation cross for 2s. In the calibration set there were 100 trials per class. The evaluation dataset was designed to mimic asynchronous BCI data, with intervals of MI between 1.5s and 8s long interspersed with similar length intervals of the idle state. Subjects were given a quiet acoustic cue which indicated when they should start imagining a particular MI activity. These cues were the words 'left', 'right' or 'foot'. The end of a period of MI activity was indicated by the acoustic cue 'stop'. The acoustic cues were quiet to prevent auditory evoked potentials, which are spikes in EEG that can occur when listening to audial stimuli [110]. Distractions in the form of music or videos were played during certain intervals to further replicate distractions that can occur during the use of a practical BCI [2]. For each individual subject, the data available in the evaluation dataset is: 24 minutes for subject 1a, 33 minutes for subject 1b, 32 minutes for subject 1f, and 32 minutes for subject 1g. This means an average of 30 minutes of data were recorded for each subject.

This dataset also contains artificially generated EEG data, which was labelled as subjects 1c, 1d and 1e. This data was not included in any research in this thesis since the focus was on developing classification approaches for data recorded from human subjects.

2.2.3 Graz 2A Dataset

The Graz 2A dataset [72], [111] has data recorded from 9 healthy subjects, labelled A1-A9. All information about this dataset was retrieved from [72]. EEG signals were recorded using 22 Ag/AgCl electrodes in locations that are part of the standard 10-20 system [72]. Three channels of EOG channels were also

recorded for artifact removal, however these were not used directly in this thesis. However, the dataset contains markings by an expert which indicate trials that have artifacts, and the expert used this EOG data to visually assess the data.

Data was recorded for four MI classes, namely: left hand, right hand, feet, and tongue. 288 training trials and 288 testing trials were recorded per subject, with a balanced representation of classes. During a trial, subjects were instructed when to carry out MI activities via visual cues on a screen. The recording protocol was as follows: i) a fixation cross appears on-screen and a brief beep also sounds to indicate that a MI trial will begin soon; ii) after 2s the visual cue appears on-screen and the subject starts the MI activity and continues the activity even when the cue disappears 1.25s later; iii) 4s after the cue appeared on-screen the fixation cross appears, indicating that the trial has ended and the subject can relax; iv) the screen goes blank as the subject is given a short break of 1.5s. The cues consist of arrows pointing left, right, up or down to indicate which MI task to carry out. Data was recorded during two sessions which took place on different days. A session was comprised of six runs, with each run consisting of 48 trials equally distributed between the four classes. Between each run, subjects were given a brief break.

Data was recorded at a sampling frequency of 250Hz and bandpass filtered between 0.5Hz and 100Hz. A 50Hz notch filter was used to remove power line noise. During recording, the amplifier sensitivity was set 100μV.

Trials containing artifacts were not included in analysis and Table 2.2 shows the numbers of training and testing trials that were used for each subject. On average, there are 262 trials for training and 277 for testing.

This dataset was used in Chapter 6 due to its frequent use in the literature in conjunction with deep learning systems, particularly the architectures used in Chapter 6 [7]. Furthermore, the dataset has relatively more training data samples available on average per subject that the BCI Competition III Dataset IVa and the BCI Competition IV Dataset I, making it more suitable for using with a deep learning system.

Table 2.2: A breakdown of the Graz 2A Dataset, detailing the training and testing samples available per subject after artifact removal.

Subject Number	Total Number of Training Trials	Total Number of Testing Trials
A1	281	288
A2	283	288
A3	273	288
A4	244	192
A5	246	288
A6	215	288
A7	277	288
A8	271	288
A9	264	288
Average	*262*	*277*

2.2.4 HG Dataset

The HG dataset [8], [112] contains MI data from 14 healthy subjects, labelled H1-H14, and each EEG trial is 4s long. A 64-channel recording montage was used, however in the supplementary material and code provided by the authors a subset of 44 central-associated EEG channels are recommended for use [112], and this subset was used in this chapter as the full EEG montage. This subset contains all the EEG channels surrounding the motor region of the scalp. The information in this section was obtained from [113]

The four MI classes are: left hand, right hand, feet, and the idle state (called 'rest' in the original paper [8]). During data recording, a fixation point appeared on-screen which indicated to the subjects that a trial would soon begin. Then, an arrow appeared on-screen, with the direction of the arrow indicating which MI action to execute. The cue remained on-screen for a four second period, during which subjects had to execute the MI task. When the arrow disappeared, subjects could rest. Breaks of three to four seconds were given between trials. Subjects were instructed to keep muscle artifact-generating actions like blinking and swallowing to a minimum during trials and carry them out only during the breaks. Data was recorded over multiple runs, with each run consisting of 80 trials. Between runs, subjects were given breaks.

The experimental setup was designed to reduce high-frequency noise in the data. Precautions involved using active electromagnetic shielding and a

shielded EEG cap. These precautions ensured that even gamma band frequencies could be studied, however this feature of the dataset was not used in this thesis.

The HG dataset was recorded at 1000Hz, at a resolution of 24 bits/sample. This data was down sampled to 128Hz using the MATLAB *resample* function. The data was high pass filtered with a passband of 4Hz. The software accompanying the HGD dataset [112] removes trials having spikes of 800mV or greater since these are likely to contain artifacts. Table 2.3 contains the total number of training and testing trials available for each subject after artifact removal. When carrying out experiments, data for subjects H2-H14 was used since testing samples for subject H1 were not available for download at the time this work was carried out. Thus, for experimental purposes, data from 13 subjects is available. Subjects had an average of 705 trials for training and 160 for testing.

This dataset was used in Chapter 6 due to its frequent use in the literature with deep learning systems [58], and because when compared to the other datasets discussed, this dataset has the most EEG training trials available.

Table 2.3: A breakdown of the HG Dataset, detailing the training and testing samples available per subject after artifact removal.

Subject Number	Total Number of Training Trials	Total Number of Testing Trials
H1	319	-
H2	811	160
H3	880	160
H4	895	160
H5	872	160
H6	835	160
H7	654	159
H8	880	160
H9	880	160
H10	813	160
H11	880	160
H12	877	160
H13	800	159
H14	800	160
Average	*705**	*160*

* Average excluding subject 1 since their data was not used in experiments.

2.3 Conclusion

This chapter discussed the nature of EEG signals and EEG recording, with a special focus on MI EEG and EEG characteristics relevant to signal processing. The

chapter also provided in-depth summaries of the datasets used in this thesis. The next chapter contains an extensive review of the literature, highlighting the conceptual gaps that were investigated, exploited, and addressed in the contribution chapters of this thesis.

Chapter 3 : Technical Background for EEG Signal Processing

This chapter presents a literature review of different EEG classification approaches, with a focus on MI EEG classification. The discussions are based, in part, on those in a review paper [18] published in *Sensors* as part of the PhD project. Throughout this chapter, a thorough technical background of important approaches used in this thesis is also provided. This technical background introduces common spatial pattern (CSP) feature extraction, conventional classifiers, sparse encoding, deep learning concepts, and genetic algorithms.

The chapter opens with a summary of conventional feature extraction methods and classifiers, then goes on to mention different sparse representation-based classification methods in the literature. It then discusses the state-of-the art in CNN processing systems for EEG. Static and automatic channel selection techniques are then tackled. Finally, the chapter closes with a discussion of how EEG time series are segmented in different BCI applications. The conclusion at the end of the chapter then gathers the gaps in the literature that were explored in the contribution chapters of this thesis.

3.1 Conventional Feature Extraction

A variety of approaches have been used for feature extraction for MI EEG classification. This section discusses the most prominent techniques, namely time-domain (TD), frequency-domain (FD), time-frequency domain (TFD), and common spatial pattern (CSP) techniques.

3.1.1 Time-Domain Feature Extraction

A predominant TD feature extraction technique involves autoregressive (AR)-type models. In the traditional AR approach, popular in novel approaches in the

1990s and 2000s [114]–[116], the model is fitted to a segment of EEG data and the AR coefficients or spectrum are used as features [117], [118]. Adaptive AR (AAR) models fit an adaptive model to segments of EEG data [114]–[116], with the adaptive parameters being estimated using least-mean squares [114], recursive least-squares [115] or Kalman filters [116]. Although AR and AAR techniques can be computationally advantageous [118], they can be effected by artifacts [117], which are common in EEG data. Furthermore, AR models are linear models meaning they can provide limited information on non-linear EEG data. This may be a reason why AR-type models have fallen out of popularity in recent years.

Hjorth features were developed in the 1970s [119] to model EEG data using three parameters: 'activity', which is the variance of the signal, 'mobility', which is the average frequency, and 'complexity', which captures change in frequency [120]. These parameters are derived through TD differentiation. All three parameters are computationally inexpensive to obtain [120], have been used recently in the literature [120]–[122], and have outperformed AAR features [120]. Quaternions are another modelling technique used for feature extraction [101]. This technique enables multichannel EEG data to be represented within a three-dimensional space, because quaternions can model orientation and rotation. Although they have exhibited promising performance [101], they have not been directly compared to other feature extraction techniques.

The TD techniques discussed thus far were parametric modelling techniques. TD feature extraction based on analysis and statistical features has also been used [123]. In 2014 Hamedi et al. [123] compared using root-mean-square (RMS) and integrated EEG (IEEG) features. IEEG measures the power within the EEG signal using the equation: $\sum_{j=1}^{M}|\boldsymbol{x}_j|$ where $\boldsymbol{x}$ is a vector representing a segment of EEG data and M is the number of samples in the segment. RMS features provided a better average classification accuracy than IEEG features, by 3.42%. Selective bandpower [124] is a power-focused feature like IEEG, but extracts the average power within specific frequency bands, as opposed to the absolute power, and uses the square of the signal instead of the

absolute value. A 2018 paper found selective band power to outperform TD template matching and TD statistical moments as a feature extraction method for MI EEG classification [124]. Since selective band power features were found to be computationally effective as well as being associated with good classification accuracy [124], the SL classifier presented in Chapter 5 is based on selective band power features.

3.1.2 Frequency-Domain Techniques

FD techniques are commonly based on the Fourier transform [5], [25], [120], [124], [125]. The fast Fourier transform (FFT) is the most primitive FD feature extraction technique used in the literature [25], [120], [124]. Analytical features extracted from the FFT magnitude signal include the relative power spectrum [124], energy in the alpha and beta bands [120], median frequency [25], mean peak frequency [25], and total power [25]. These features were effective, with Samuel et al. [25] recording a classification accuracy of 99.79% on a private dataset with three amputee subjects. The Welch method has also been used to extract power spectral density which was used directly for classification [5]. The Welch method aims to reduce the variance in the spectrum when compared to using the FFT, at the cost of a poorer frequency resolution.

3.1.3 Comparing Time and Frequency Domain Techniques

There is some disagreement in the literature over whether FD features can provide a significantly better classification performance than TD features [120], [124]. Consider two studies both focused on the same public dataset [120], [124]. In [120], FFT-based features were found to significantly outperform AAR and Hjorth features for MI EEG classification. In another study, Arnin et al. [124] compared FFT feature extraction to three TD feature extraction techniques, namely template matching, statistical moments and selective band power. Although the FFT features did provide an improved performance, the improvement was not statistically significant. They also found that FFT feature

extraction had significantly greater computational complexity, making it less favorable than the TD techniques. Based on the results of Arnin et al. [124], the sparse-learning classifier in Chapter 5 is based on TD as opposed to FD features.

3.1.4 Time-Frequency Domain Techniques

TFD techniques have gained popularity for MI EEG classification over the last five years [12], [22], [30], [118], [122], [126]–[129] They aim to provide a richer analysis than FD techniques by capturing dynamic behaviour at different frequencies over time. TFD approaches could also be more suitable than TD and FD features for non-stationary and non-linear data [130] such as EEG signals. However, these techniques come with their own characteristic trade-offs between time and frequency resolution.

TFD transformations have been applied in three different ways for feature extraction: i) the transformation coefficients are used as features [122]; ii) statistical features extracted from the transformed signals, such as mean, mode, standard deviation, skewness and kurtosis are used to construct a feature vector which is passed onto a classifier [22], [30], [118]; or iii) a TFD image is obtained and used as input to deep-learning (DL) classifiers [12], [126], [127].

The spectrogram has been widely used for visual presentation of MI within the literature [14], [131] and it is effective for identifying ERD and ERS [132]. The short-time Fourier transform (STFT) can provide improved time-domain resolution by dividing the EEG signal into segments, obtaining the spectrum of each segment, and then concatenating the images to form the STFT image [130]. The segmentation step in the STFT can improve the representation of signals that are non-stationary. The STFT has been used to convert the EEG time-series to a TFD image prior to DL-based classification [12], [126], [127]. The STFT is given in Equation (3.1) [12]:

$$Y(t,f) = \int y(\tau)h(\tau - t)e^{-j2\pi ft}d\tau \tag{3.1}$$

where $y(t)$ is the signal on the EEG channel, $Y(t, f)$ is the TFD representation, f is the frequency, t is time and $h(t)$ is the windowing function used for segmentation. However, the STFT creates a trade-off between the time and frequency resolutions in the image.

Wavelet-based feature extraction techniques are more flexible than the STFT, offering a representation at multiscale and multiresolution [12], [130]. The continuous wavelet transform (CWT) aims to solve some of the resolution issues of the STFT by using a scaling factor for more localized signal representation [12], and has gained popularity in MI EEG classification [12], [128], [129]. The CWT is summarized by Equation (3.2) [12]:

$$W_{(a,b)}[y(t)] = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} y(t) \varphi^* \left(\frac{t-b}{a}\right) dt \, , a > 0 \tag{3.2}$$

where $\varphi(t)$ is the wavelet basis function, a is the scaling parameter and b is the time-shift parameter. To solve the heavy computational demands of the CWT, the discrete wavelet transform (DWT) was developed [130].

A variation of the DWT is wavelet packet decomposition (WPD), in which coefficients excluded during DWT decomposition are retained. In 2017, Kevric and Subasi [22] compared the feature extraction capabilities of the WPD, DWT and the time-domain decomposition technique empirical mode decomposition (EMD), obtaining accuracies of 94.5%, 81.1%, and 62.8%, respectively. This analysis indicated that the data retained by WPD could contain salient information for MI classification. Later, Mumtaz et al. [122] also found that wavelet transform features outperformed TD features.

In Chapter 5, a sparsity-based classification system using time-domain band pass features is presented. The method was found to outperform a similar classification pipeline using wavelet-based features. The implementations of Kevric and Subasi [22] and Mumtaz et al. [122] were based on feature extraction followed by classification either using a k-NN classifier or a hidden Markov model classifier. The classification approach in Chapter 5 is different, being based on the dictionary-based reconstruction, which is discussed in more depth in Section 3.3.

The results in Chapter 5 may suggest that the question of whether TD or TFD features are better for MI EEG representation may depend on the classifier used, although this would need to be verified through more rigorous investigation.

Finally, the empirical wavelet transform (EWT) is a wavelet decomposition technique offering another degree of freedom through adaptive wavelets [30], [118]. Univariate [118] and multivariate [30] versions of the empirical wavelet transform have been effective for MI EEG feature extraction, with accuracies of 95% and 97% being obtained for each approach. These features performed on a par with the literature, however they required extensive tuning for individual subjects [30], [118], unlike the selective band power features used in Chapter 5. A variety of TFD approaches have been discussed in this section because systems based on TFD feature extraction have been used for comparison in contribution Chapter 4 and Chapter 5.

3.1.5 Common Spatial Patterns

CSP-based features are one of the most popular feature extraction techniques used in MI EEG processing [10], [15], [26], [46], [48], [104], [106], [133]–[135]. This approach uses spatial filtering to transform the data in such a way that the variance of data in one class is maximized whilst the variance in the other class is minimized. Features are then extracted based on the filtered data. Since the variance in the filtered EEG signals corresponds to the power in the frequency bands within the signal, CSP has been popular for MI EEG BCIs [10], [14], [136], [137].

Standard CSP features have been widely used in the last five years [82], [104], [106], [107], [133], [134], [138], particularly in studies presenting novel channel selection or classification techniques [104], [106], [107], [133], [134], [138], or for benchmarking [82]. Standard CSP feature extraction is implemented based on the theory in [10], [139], and is summarized in the following steps:

1. Training phase:

a. Calculate the spatial covariance, $\boldsymbol{C}^{(0)}$, of the multichannel data, $\boldsymbol{X}$, in each trial, τ, using: $\boldsymbol{C}_{X_\tau}^{(0)} = \frac{1}{T}\boldsymbol{X}_\tau \boldsymbol{X}'_\tau$ where T is the length of trial and $'$ denotes the transpose. The columns of $\boldsymbol{X}$ represent the different channels.

b. Obtain the standardised covariance matrix [139], $\boldsymbol{C}_{X_\tau}^{(1)}$:

$\boldsymbol{C}_{X_\tau}^{(1)} = \frac{\boldsymbol{C}_{X_\tau}^{(0)}}{\mathrm{Tr}(\boldsymbol{C}_{X_\tau}^{(0)})/N_{\boldsymbol{X}}}$, where $N_{\boldsymbol{X}}$ is the number of channels and Tr() is the trace of the matrix. This compensates for trials having different instantaneous power content.

c. Estimate the class-conditional covariance matrices:

$\widehat{\sum}_{\boldsymbol{X}|C_j}^{(1)} = \frac{1}{N_{C_k}} \sum_{\tau:C_\tau = C_j} \boldsymbol{C}_{\boldsymbol{X}_\tau}^{(1)}$, for $k = 1, \ldots, K$ where C_j denotes class j, C_τ denotes the class associated with trial τ and K denotes the total number of classes.

d. Calculate the weight vector, $\boldsymbol{W}$, which is used to filter the data:

   i. Obtain the eigenvectors, $\boldsymbol{V}$, from the eigenvalue problem:

   $\sum_{\mathbf{X}|C_1} \boldsymbol{V} = \sum_{\mathbf{X}|C_2} \boldsymbol{V}\, \lambda$

   ii. Extract the $\frac{p}{2}$ highest and lowest eigenvectors to create $\boldsymbol{W}$:

   $\boldsymbol{W} = \left[\boldsymbol{V}\left(1{:}\frac{p}{2}\right),\ \boldsymbol{V}\left(N_{\boldsymbol{X}} - \frac{p}{2}{:}N_{\boldsymbol{X}}\right)\right]$, where p is the dimensionality of the subspace used for the filtering transformation.

e. Carry out spatial filtering to obtain the transformed covariance matrix, $\boldsymbol{C}_{Y_\tau}$, for each trial: $\boldsymbol{C}_{Y_\tau} = \boldsymbol{W}'\, \boldsymbol{C}_{X_\tau}^{(0)}\, \boldsymbol{W}$, $\tau = 1, \ldots, N_\tau$.

f. Obtain the feature vector for the trial, $\boldsymbol{f}_\tau$: $\boldsymbol{f}_\tau = \frac{\log(\mathrm{diag}(\boldsymbol{C}_{Y_\tau}))}{\mathrm{sum}(\mathrm{diag}(\boldsymbol{C}_{Y_\tau}))}$

2. Testing phase: Carry out steps 1a), 1b), 1e) and 1f) on each test trial to obtain the test feature vectors.

Standard CSP features capture wideband frequency behaviour. Since subjects may experience changes in different frequency bands during MI, standard CSP may sometimes provide sub-optimal features [140]. Optimizing the filter band

used can improve classification accuracy, but the process is computationally expensive, leading to increased training times [140].

A branch of research has been devoted to exploring technical alterations to traditional CSP to improve performance. For example, common spatio-spectral patterns (CSSPs) integrate a finite impulse response filter into the CSP processing steps to improve performance [141]. Common sparse spatio-spectral patterns (CSSSPs) are a sparse extension of the CSSP method, and aim to extract spatial patterns which occur on all channels, not just on individual channels [142]. In filterbank CSP (FBCSP), EEG data is filtered within different sub-bands, features are extracted from each band, then feature selection is carried out [46], [140], [143], [144]. FBCSP can lead to improved performance compared to CSP, CSSP and CSSSP, at the expense of increased computation [140], [143].

Regularized CSP (RCSP) is an alternative approach to improving CSP which is not focused on tackling the problem of wideband CSP feature extraction [13], [145]. RCSP introduces regularization coefficients to the CSP algorithm for improved generalizability which has led to better results, but at the cost of high computational demands [13], [145]. A simplified regularization algorithm, proposed by Jin et al. [13] in 2019, has outperformed traditional CSP methods with a lower computational load compared to traditional RCSP. In a different approach, Olias et al. [10] used power normalization to improve the classification accuracy of CSP features by enabling more homogenous feature extraction across trials.

Although these novel approaches to CSP feature extraction are promising [10], [13], [46], [140], [143], [144], standard CSP is still favored in the literature for studies that implement novel algorithms [104], [106], [107], [133], [134], [138]. For example, Baig et al. [104] presented a novel feature selection approach, whilst She et al. [107] presented a novel hierarchical extreme learning machine classifier, and both studies used standard CSP for feature extraction. Standard CSP feature extraction is still used because it is a reliable, robust and computationally efficient feature extraction approach [107]. In Chapter 4, a novel temporal decision fusion approach for EEG classification is presented, and in-

keeping with this trend in the literature [104], [106], [107], [133], [134], [138], standard CSP features were used.

3.2 Conventional Classifiers

Classifiers can be separated into two types, those based on supervised learning and those based on unsupervised learning. Supervised learning [39] involves the use of labelled data for training, whereas unsupervised learning [39], [146] uses unlabeled sample data and learns decision boundaries in an exploratory way. Supervised learning is widely used when training classifiers for MI EEG classification [8], [10], [11], [14], [21], [22], [30], [31], [65], [101], [104], [133], [147], [148] because the training data is recorded during controlled experimental conditions [71], and thus the labels are known. In this thesis, supervised learning is used in all three contribution chapters.

This section discusses the different classifiers used in the literature for MI EEG classification. The discussion is focused on conventional machine learning approaches, with DL approaches being tackled later.

Various conventional classification approaches have been used for EEG classification, namely support vector machines (SVMs) [13], [14], [21], [31], [104], [140], linear discriminant analysis (LDA) [10], [14], [21], [133], k-nearest neighbour (k-NN) [14], [22], [31], [101], logistic regression (LR) [10], [31], decision trees [101], [104], random forests (RFs) [148] and naïve Bayes (NB) [104], [147] classifiers. Table 3.1 summarizes several salient studies focused on conventional classification techniques. The table includes technical details including features extracted, classifier type, the main contribution of the study and the classification accuracy obtained with the technique. There are also columns devoted to the number of EEG channels used, which is relevant to discussions later, in Section 3.5.1. The works in Table 3.1 were selected to present a wide variety of examples from the literature that cover topics discussed in this section [10], [11], [14], [22], [26], [31]–[33], [101], [134].

The SVM and LDA classifiers are widely popular in the literature [10], [13], [14], [21], [31], [104], [133], [140], however there is no clearly superior classifier

Table 3.1: A table comparing different studies that have used conventional machine learning classification techniques.

Paper	Dataset	No. EEG Channels	Features	Classifier	Main Contribution	Accuracy
Asenio-Cubero et al. [26] (2011)	BCI Competition III, Dataset IVa	118[1]	CSP	LDA	Time segmentation methods for improved classification	60.14%
Batres-Mendoza et al. [101] (2016)	Proprietary dataset	14[1]	Quaternion-based features	Decision Tree	Extraction quaternion-based features	84.75%
Ilyas et al. [31] (2016)	BCI Competition IV, Dataset I	11	FFT features	Logistic Regression	Comparing logistic regression, SVM, k-NN and MLP classifiers.	73.03%
Kevric and Subasi [22] (2017)	BCI Competition IV, Dataset I	118 for pre-processing /3 for classification	Wavelet packet decomposition	k-NN	Finding wavelet packet decomposition was a better feature compared to the state-of-the-art.	94.50%
Siuly et al. [33] (2017)	BCI Competition III, Dataset IVa	118[1]	Cross-correlation-based features	Least-Squares SVM	Found that using all 118 channels gave better performance than just motor-related (central) EEG channels.	97.96%
Yang et al. [32] (2019)	Proprietary dataset	3	Welch spectrum features	LDA	Using an optimized time window for segmentation can improve performance.	87.63%[2]
Oilias et al. [10] (2019)	BCI Competition III, Dataset IVa	118[1]	Normalized CSP	Tangent Space Logistic Regression	Improved covariance estimation compared to traditional CSP.	79.62%
Sreeja et al. [11] (2019)	BCI Competition III, Dataset IVa	30	Wavelet energy	Dictionary-based sparse learning classifier based on reconstruction error	Improving classification using a weighted dictionary.	97.98%
Hekmatmanesh et al. [14] (2020)	BCI Competition III, Dataset IVa	118[1]	Discriminative filterbank CSP	Soft-margin SVM	Using a discriminative sensitive learning vector quantization for discriminative filterbank CSP	92.70%

[1]The full number of EEG channels available in the dataset.

[2]The accuracy was obtained by interpreting a graph.

for MI EEG. Some papers report that SVM classifiers outperformed other classifiers, namely LDA, k-NN, RF and NB [5], [31], [104]. However, other studies have reported different results, for example Batres-Mendoza et al. [101] reported that decision tree and k-NN classification consistently outperformed SVM

classification. Ilyas et al. [31] also reported that LR, although not widely used, performed on a par with SVM. Different studies may report conflicting results due to different features or hyperparameter tuning techniques being used. This discussion of conventional classifiers illustrates the importance of experimenting with different classifiers when developing novel pipelines.

The SVM classifier has some technical features which could make it mathematically attractive [149]. SVM classifiers have noise robustness in-built into their learning algorithm [149], unlike k-NN classifiers which build local boundaries and depend only on the size of the smoothing kernel to deal with noise and outliers [150](pp. 124-127). Although k-NN training is much faster than SVM training, it demands the whole of the training dataset to be present in memory for classification to take place, making it very memory expensive and possibly slower to assign labels to test-set samples for training large datasets [150](pp. 291-292). However, SVM classifier training is computationally demanding, leading some BCI researchers to opt for less computationally expensive classifiers, such as LDA [151]. Furthermore, because SVM classifiers rigidly process multichannel EEG data, they can be ineffective at identifying spatial relationships between EEG channels which can vary between subjects [152]. Although some studies have identified SVM as giving better performance [5], [104], [140], it is not the default classifier in the literature. In fact, many other studies still include a variety of classifiers when assessing the effectiveness of features [5], [21], classification methodology [14] or channel/feature selection approach [104].

In this thesis, six different conventional classifiers have been used. In Chapter 4, the effect of time-domain decision fusion was investigated on SVM, LDA, NB, RF, and MLP classifiers. In Chapter 5, the performance of the proposed sparse learning and genetic algorithm channel selection approach is compared to that of SVM, k-NN and RF classifiers. These classifiers were chosen for Chapter 4 and Chapter 5 because they have all been used as part of state-of-the-art classification systems [10], [14], [21], [104], [148] and they provide good performance, as indicated in Table 3.1 and the discussions earlier in this section.

Furthermore, together they provide a range of classification approaches. This variety is particularly important in Chapter 4, which aims to investigate the impact of the decision fusion approach on various classifiers.

In Chapter 5, a comparison between four classifiers, namely a novel SL system, as well as RF, k-NN and SVM classifiers was carried out. The SL classifier was found to be more computationally efficient than the SVM classifier, and outperformed the other classifiers in terms of accuracy, sensitivity, and specificity. This comparison is a novel contribution to this area of the literature because, to the best of the authors' knowledge, this mix of classifiers has not been compared before under the experimental conditions in Chapter 5. Specifically, the comparison in Chapter 5 includes the idle state as a class for classification and investigates the performance of the classifiers as the training data size was reduced, which is not common in the literature reviewed [5], [31], [101], [104]. However, this kind of analysis is important because an ideal classifier must accurately identify the idle state and should be able to operate at high accuracy with minimal training data.

There are some common issues in how papers report experimentation with conventional classifiers. Firstly, many papers do not explain how classifier hyperparameters are tuned, what values were used for some hyperparameters, or whether these hyperparameters were optimized on a validation set or on the test set [5], [14], [104], [153]. Furthermore, some papers do not mention the kind of kernel used for SVM classification, even though the kernel type (linear, polynomial, or radial-basis function (RBF)) has an impact on how the decision boundaries are formed [104], [140], [153]. In Chapter 4 and Chapter 5 of this thesis conventional classifiers are used, and the hyperparameter tuning process, as well as the kernels used with SVM classifiers, are clearly described.

Furthermore, comparisons between different classifiers are often carried out within the context of a fixed ('static') channel subset [5], [14], [31]. However, the static channel subset could have an impact on classification performance because it could affect the dynamics captured, the noisy channels included and the number of correlated channels [14]. In Chapter 4 of this thesis, the

performance of various conventional classifiers is compared when using channel subsets comprised of electrodes from different scalp regions.

The rest of this section discusses technical details of each of the six conventional classifiers used in this thesis.

3.2.1 Support Vector Machines

SVMs classify data points within a hyper-dimensional space by constructing a hyperplane that acts as a decision boundary. SVMs classify using *soft* margins, meaning that the hyperplane is constructed to classify most of the training datapoints correctly, but allows a small proportion to be misclassified, thus reducing the risk of the decision boundary overfitting to noise or outliers in the training data [149]. The group of data points from different classes closest to the decision boundary is called the 'support'. To construct the decision boundary, the SVM classifier aims to maximise the margin between the support points and the boundary itself. The distance between the support points and the decision boundary is measured perpendicular to the boundary [149]. Figure 3.1 shows an example of a feature space with two classes, dark blue and dark green, separated by a linear SVM decision boundary, denoted by the solid line. The light blue and light green shaded areas denote the class spaces in the feature space as interpreted by the SVM classifier. The dotted lines denote the maximum margin, and the data points lying on the dotted lines denote the support. A soft margin was used, with a single green datapoint near the bottom right-hand corner of the space being misclassified. This example is for a 2D feature space, but SVMs can classify data in multi-dimension feature spaces.

In this thesis, SVM classifiers with non-linear decision boundaries are used. In non-linear SVMs, kernels are used to transform the data to a new training feature space, and the decision boundary hyperplane is created in this space, resulting in a non-linear boundary in the original space [149]. In this thesis, polynomial kernels, and radial basis function (RBF) function kernels are used.

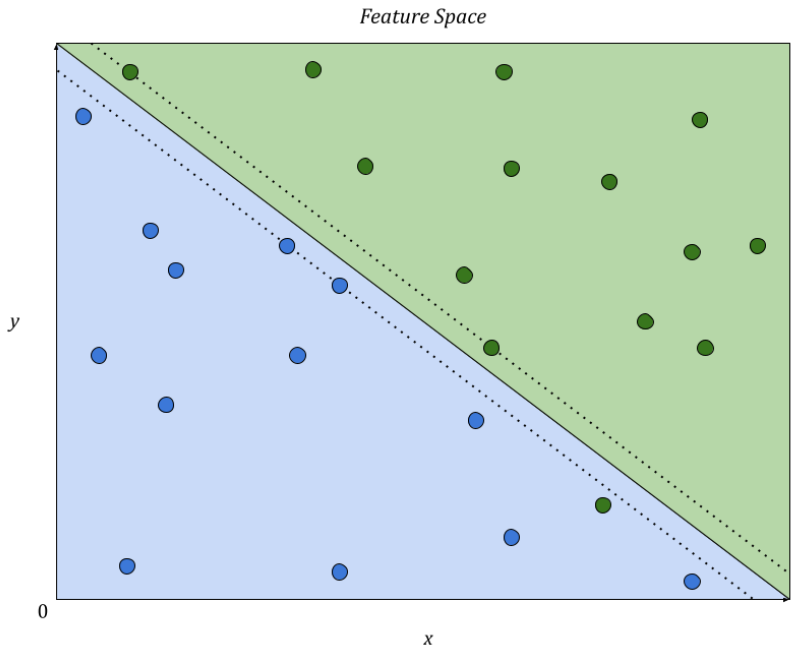

Figure 3.1: Forming an SVM decision boundary hyperplane (solid line) between two classes in a feature space. The dotted lines denote the maximum margin, with datapoints lying on those lines forming the support.

The chosen kernel can be scaled using the parameter g. The Gram matrix, which is the inner product of two vectors, is used to perform the transformation. Thus, if the kernel function is $\emptyset$, then the Gram matrix of a set of input vectors $\{x_1, x_2, \dots, x_n\}'$ is $G(x_j, x_i) = \langle \emptyset(x_j) | \emptyset(x_i) \rangle$ [149], [154].

Consider the case of linear SVM classification. The score function used for linear SVM classification is given by: $f(x) = x'\beta + b$, where x is the row vector of observations, β is the row vector of coefficients describing the hyperplane, and b is a bias.

In EEG classification, linear classification cannot be carried out since the classes are not perfectly separable. To develop the boundary for non-linear problems, the SVM classifier uses the slack variable, ξ_j, which penalizes the objective function for datapoints which are on the incorrect side of the margin for their class. The primal form of the objective function which is optimized by the SVM classifier is: $0.5\|\beta\|^2 + C\sum \xi_j$, which is optimized with respect to β, b and ξ_j, given that $\sum \xi_j > 0$ for $j = 0, \dots, n$ where n is the number of training points. The

box constraint, C, is a positive integer which controls the severity and quantity of violations of the margin. The optimization problem is solved using Lagrange multipliers [149], [154].

The dual formalization for non-linear SVM classification involves minimizing [154]:

$$\frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n}\alpha_j\alpha_k y_j y_k G(x_j, x_k) - \sum_{j=1}^{n}\alpha_j \tag{3.3}$$

with respect to the Lagrange multipliers $\alpha_1, \dots, \alpha_n$ and subject to the constraint: $\sum \alpha_j\, y_j = 0, 0 \le \alpha_j \le C\ \forall\, j = 1, \dots, n$ and the Karsh-Khun-Tucker (KKT) constraints, which are general conditions which must be satisfied in non-linear optimization problems [154]. For an SVM classifier the KKT constraints manifest as [154]:

$$\begin{cases} \alpha_j f(x_j) - 1 + \xi_j \\ \xi_j(C - \alpha_j) = 0 \end{cases},\ \forall\, j = 1, \dots, n \tag{3.4}$$

where $f(x_j) = \emptyset(x_j)'\beta + b$. Both complementary conditions must be satisfied. Thus, the score function for the non-linear SVM classifier with non-separable classes is:

$$\hat{f}(x) = \sum_{j=1}^{n}\hat{\alpha}_j y_j\, G(x, x_j) + \hat{b} \tag{3.5}$$

where $\hat{b}$ is the estimate of the bias and $\hat{\alpha}_j$ is the estimate of the jth Lagrange multiplier [154].

Thus, two hyperparameters control SVM classification: the regularization parameter, C, and the kernel scale parameter, g. C is a non-negative scalar value which controls the amount of error allowed when fitting the SVM model to the training dataset, and thus prevents overfitting [155]. g is a positive scalar value that influences the spread of the kernel: a larger value shrinks the kernel size, thus enabling the decision boundary to respond to local variations in the data, but limits the number of points contributing to boundary construction in a region [150]. In the case of SVM with a polynomial kernel, the order of the polynomial is also tuneable. Parameters can be tuned using a grid-search method, where different combinations of parameters are considered, and their efficacy evaluated

on a validation dataset. The parameters that give the best classification accuracy are then used on the data. Chapter 4 and Chapter 5 explain in more detail how the classifiers used in this thesis were tuned.

3.2.2 Linear Discriminant Analysis

The LDA classifier assumes that the data has a Gaussian distribution and constructs a linear decision boundary by characterising the training data associated with each class in terms of a mean vector and a covariance matrix [150]. Each class has its own mean, but the covariance matrix, $\hat{\boldsymbol{\Sigma}}$, is calculated over the whole training data. Figure 3.2 shows an example of decision-boundary construction for the LDA classifier for a two-class problem. The classes are represented by the data point colours (dark blue/dark green) and the black line

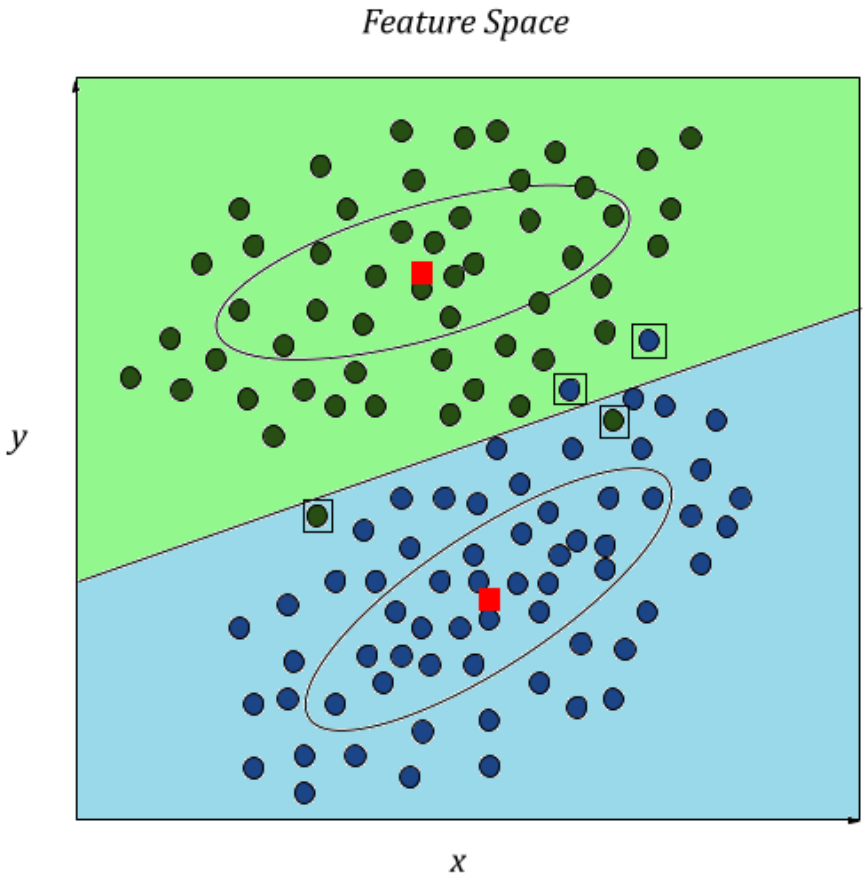

Figure 3.2: Constructing decision boundaries of an LDA classifier for a two-class problem. The classes are represented by dark green and dark blue data points, and the decision boundary is the black line. The red boxes denote the means of the Gaussian models, and the ellipses represent the variance of the models. Data points outlined with a black box are outliers lying on the incorrect sided of the decision boundary.

represents the decision boundary. The red squares show the class means, and the ellipses show the variance of the Gaussian distributions. Outliers, which lie on the incorrect side of the decision boundary, are indicated using black squares.

To avoid overfitting, the regularized covariance matrix, $\widehat{\boldsymbol{\Sigma}}_y$, is used, where:

$$\widehat{\boldsymbol{\Sigma}}_y = (1-\gamma)\widehat{\boldsymbol{\Sigma}} + \boldsymbol{\gamma} diag(\widehat{\boldsymbol{\Sigma}}) \tag{3.6}$$

γ is the regularization coefficient and can have a scalar value from 0 to 1. It influences the amount of regularization used during the estimation of the covariance matrices [150], [156].

The LDA classifier generates a predicted class label, $\hat{y}$, using [156]:

$$\hat{y} = arg \min_{y=1,\dots,K} \sum_{k=1}^{K} \hat{P}(k|x) C(y|k) \tag{3.7}$$

where K is the number of classes, $\hat{P}(k|x)$ is the posterior probability that observation x is part of class k and $C(y|k)$ is the cost associated with classifying y as class k.

To perform prediction, latent variables are generated from a linear combination of the input data. The weights which are used to generate these variables are known as discriminant coefficients [150], [156]. The linear coefficient threshold, delta (Δ), is a non-negative scalar [156] which governs which features influence the decision boundary: if the discriminant coefficient associated with a certain predictor is smaller than Δ, then the value of the coefficient is set to zero, effectively removing the influence of that predictor. Thus, increasing the value of Δ increases the likelihood that more predictors will be excluded [150], [156]. Thus, in the LDA classifier, both γ and Δ are tunable parameters.

3.2.3 k-Nearest Neighbour

k-NN classification is a non-parametric method, meaning that it does not use any parameters to explicitly model the data. Classification of a test sample is carried out by a vote based on the k nearest training samples within the feature space:

whichever class most of the nearest neighbours form a part of is the class assigned to the test sample. Therefore, the posterior probability that sample $\boldsymbol{x}$ is a part of class $C_m$is [150](pp. 125-127):

$$P(C_m|\boldsymbol{x}) = \frac{K_m}{k} \tag{3.8}$$

where K_m is the number of nearest neighbours forming part of class $C_m$and k is a positive, non-zero number denoting the total number of nearest neighbours being considered. During experiments parameter k was tuned.

Figure 3.3 shows an example of the k-NN method applied to a three-class problem. The true datapoints for the three classes are denoted by dark blue, green, and purple circles. Datapoints circled in red are from the test-set and those not circled are from the training set. The training set points were used to delineate the decision boundaries which are shown as black lines in the feature space. The pale blue, green and purple shadings show the class regions in the

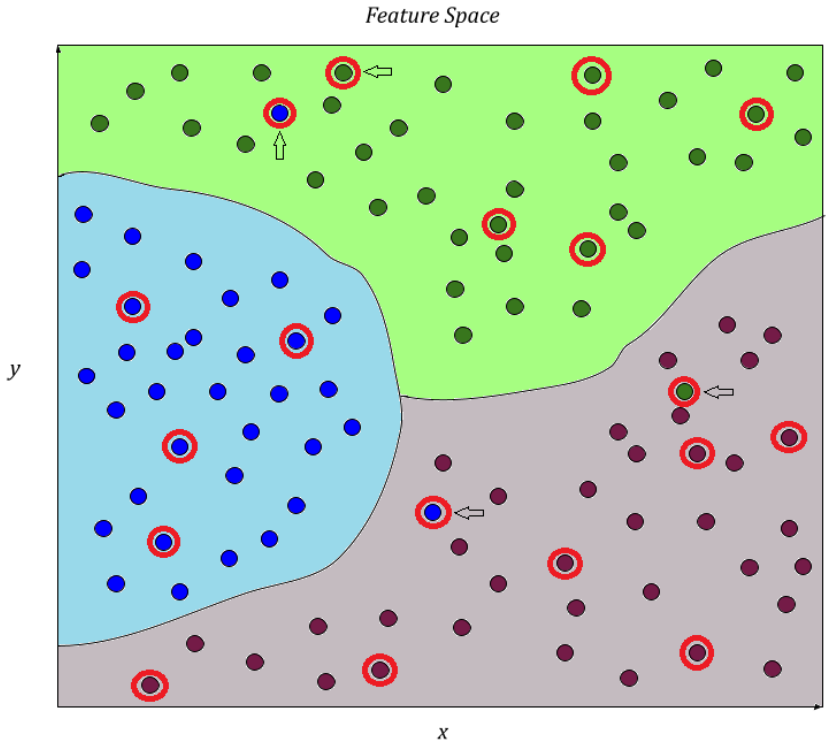

Figure 3.3: Classification using the k-NN method. The dark blue, green, and purple data points represent data from three different classes. The pale shadings represent the class regions as understood by the classifier. Red circled data points are from the test set, the rest are from the training set. Misclassified points are highlighted with arrows.

feature space as understood by the classifier. Misclassified test set data points are shown using arrows. For example, there is a dark blue data point which is found in the pale green shaded region, meaning that it was misclassified as the green class. This is because this blue data point was closer to training samples from the green class than the blue class.

Parametric methods such as SVM and LDA, which explicitly model the distribution of the input data, require extensive tuning to prevent overfitting or underfitting. The *k*-NN method does not suffer these issues since it considers the actual training data distribution for classification. However, this requires the whole training data set to be retained in memory for classification to be carried out, making it very memory expensive [150](pp. 125-127). Furthermore, although non-parametric methods like the *k*-NN can be fast to train because they do not need to learn a model of the data, they can be slower to classify test set samples when the datasets are relatively large. Tree-based search methods, such as the ones underlying the RF classifier discussed next, can attempt to approximate nearest-neighbour searches without having to process the entirety of the dataset [150](pp. 125-127).

3.2.4 Random Forest

Decision trees have been used for classification and regression for different problems, including EEG classification [101], [157]. A decision tree is comprised of nodes (splits) and branches, similar to a flowchart, as shown in Figure 3.4 [158]. The blue ovals represent split nodes, the arrows represent the branches, and the orange ovals represent the leaf nodes, where final decisions are made. The example in Figure 3.4 is a decision tree for the process of buying a book, which represents a two-class problem: either buy the book, or do not buy the book. At each split node, a decision about which branch to take is made, eventually leading to a final decision at the terminal nodes, known as leaf nodes.

When applied to numerical problems, the input to the decision tree is a set of predictors, which in this work consists of a feature vector. At each node,

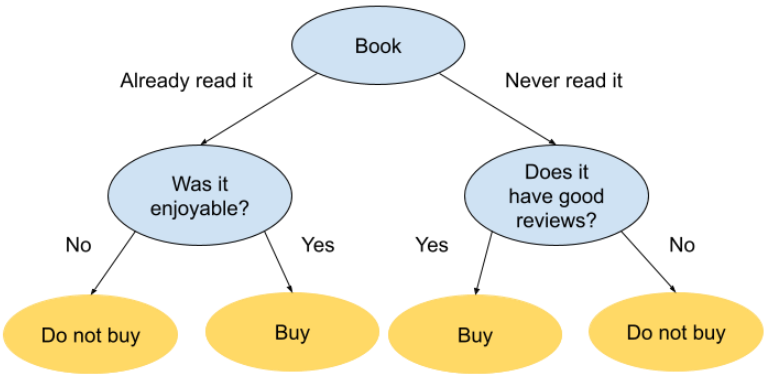

Figure 3.4: An example of a decision tree for buying a book. The ovals represent nodes, with the blue ones representing split nodes and the orange ones representing leaf nodes. The arrows represent branches.

several predictors are randomly selected, and a statistical test is carried out to determine which branch (outcome) should be taken. Leaf nodes are terminals and are used to assign a decision label. Leaves output the probability that the input feature vector belongs to each class. The leaf size is the number of predictors at the leaf node that are available to determine the probability [158]. The minimum leaf size determines how many predictors are required for a node to be determined as a leaf node. It controls the depth of the tree, with larger leaf sizes leading to more complex trees which run the risk of overfitting, whilst smaller leaf sizes lead to shallower trees which risk underfitting [159]. Thus, within a decision tree, the number of predictors sampled at each node, and the minimum leaf size are two hyperparameters that can be tuned.

Using just one decision tree for classification can lead to overfitting and/or a high variance in results when training multiple decision tree classifiers on the same problem [158]. RF classification is an ensemble learning approach in which several decision trees are used to classify a trial. On average, RFs reduce the variance in results and tend to give better classification performance than using just a single decision tree [158], [160]. To train a RF classifier, the training data is divided into different subsets, and each tree is trained on a different subset of data [158]. When classifying a test sample, the class probabilities output by each tree are averaged to obtain the classification labels [160]. The class with the

highest average probability is the class the feature vector is assigned to [160]. In RF classifiers, the number of trees in the forest is a tunable hyperparameter. All the trees within the forest have the same number of predictors at each node and minimum leaf size, which can also be tuned.

3.4.5 Naïve Bayes

The NB classifier uses a maximum *a posteriori* decision rule, which means that it assigns test set samples to the class which has the highest probability. Figure 3.5 shows a simple example of a NB classification boundary for a two-class problem. Classification involves three steps [161]:

1. Model the probability density functions of the predictors within the context of each class. For modelling, a multinomial distribution based on kernel functions was used in this thesis. Specifically, Normal, Box, Epanechnikov, and Triangular kernels were considered during model tuning. The kernels also have a smoothing window with a tunable width.

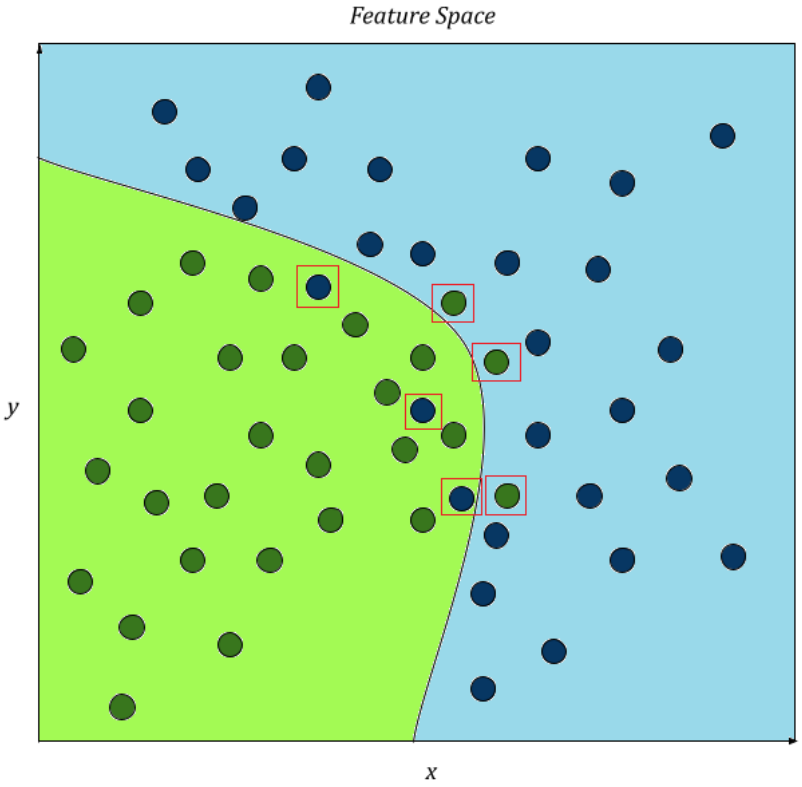

Figure 3.5: Decision boundary formation of a naïve-Bayes classifier for a two-class problem (dark green vs dark blue). The circles denote the data points, and the black curve denotes the decision boundary. Outliers are indicated with red boxes.

2. Use Bayes' rule to calculate the posterior probability for each class, given by [162]:

$$\hat{P}\left(Y = k \middle| X_1 \dots X_q\right) = \frac{\pi(Y=k)\prod_{j=1}^{q} P(X_j|Y=k)}{\sum_{k=1}^{K} P(X_j|Y=k)} \tag{3.9}$$

where $X_1 \dots X_q$ are the predictors, q is the number of predictors, Y is the class label assigned, K is the number of classes and $\pi(Y = k)$ is the prior probability that the class label is k.

3. Assign the class label based on the class that gave the greatest *a posteriori* probability.

The NB classifier makes the broad assumption that all the predictors are conditionally independent in terms of probability distributions. Although in practice this assumption is often violated, the NB classifier has exhibited strong performance [99], [104], [147], [161] and was thus considered as well in this work.

3.2.6 Multilayer Perceptron

The MLP is a classical neural network classifier. It consists of multiple, fully connected feed-forward layers, as shown in Figure 3.6. In this work, the input layer is an identity layer and has the same size as the input feature vector. The initial layer is followed by one or more hidden layers which can have a varying number of neurons. In this work, the output layer has a single neuron which outputs the classification result. There can be more than one output neuron in the final layer, for example if the classes are one-hot encoded (for example, for a two-class problem, class 1 is encoded as 10 and class 2 is encoded as 01). The number of hidden layers and the number of neurons in the hidden layers are tuneable parameters linked to the complexity of the transformation which maps the input data to the output value.

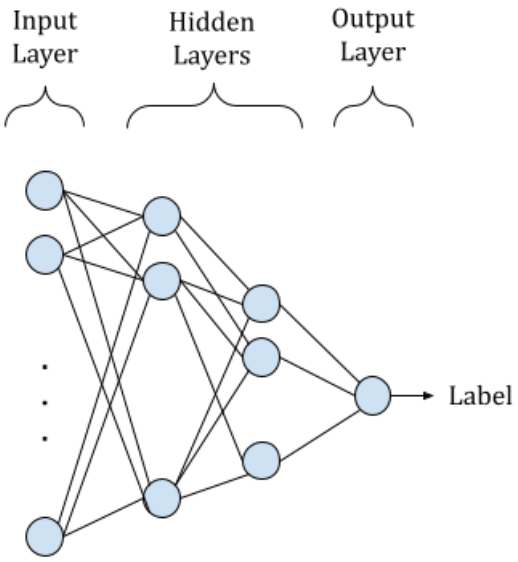

Figure 3.6: A generic multilayer perceptron classifier.

Figure 3.7 shows a generic artificial neuron. The hidden layers are made up of multiple artificial neurons. Each neuron in the hidden layer maps the values in the input data vector, $\boldsymbol{x}$, to an output value $y(\boldsymbol{x})$, via an activation function, σ where [150]: $y(\boldsymbol{x}) = \sigma(\boldsymbol{w}^T x + \boldsymbol{w}_o)$. $\boldsymbol{w}$ is a weight vector learnt during the training phase and $\boldsymbol{w}_o$ is a fixed bias [150]. The activation function thus determines the decision surface used by each neuron and is another hyperparameter also

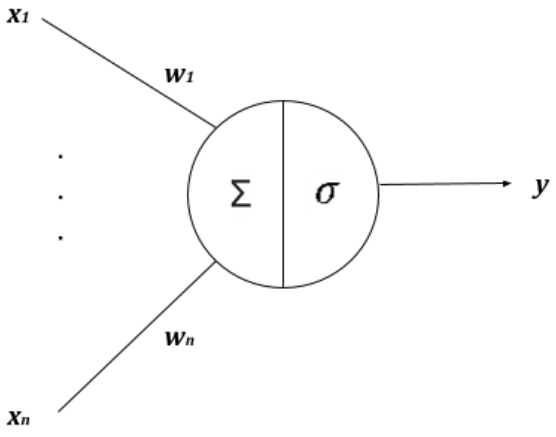

Figure 3.7: A generic artificial neuron. Σ represents a summation and σ *is the activation function*

affecting the mapping carried out in the MLP. Commonly used activation functions in EEG processing are the hyperbolic tangent (tanh) function: $f(\boldsymbol{x}) = \tanh \boldsymbol{x}$ [163], the rectified linear unit (relu) function: $f(\boldsymbol{x}) = \max(0, \boldsymbol{x})$ [164], and the logistic sigmoid function: $f(\boldsymbol{x}) = \frac{1}{1+e^{-x}}$ [165].

Whilst the number of neurons in the hidden layer and the activation function are related to the architecture of the MLP, other hyperparameters can affect the training process. Stochastic gradient descent is a widely used algorithm for the iterative training of weights in a neural network [3], [150], and uses the following equation : $\boldsymbol{w}^{(\tau+1)} = w^{\tau} - \eta \nabla \boldsymbol{E}_N$ where τ is the iteration number, η is the learning rate, and $\nabla \boldsymbol{E}_N$ is the sum of the error over the data points using the current weight vector, $\boldsymbol{w}^{\tau}$ [150]. Thus, the learning rate is a hyperparameter which controls the influence of the error in the previous iteration on the update of the weights, whilst the maximum number of iterations is another hyperparameter controlling the number of iterations which can be carried out in the training process. Regularization can also be added to the training process to prevent overfitting, with L2 regularization, also known as ridge regression, often used [150]. The hyperparameter α is used to control the strength of regularization, with a larger value of α corresponding to stronger regularization, thus reducing overfitting. However, if α is too large, underfitting can occur [150].

Momentum is a popular optimization technique which can be added to stochastic gradient descent to encourage the training of the weights to accelerate by pushing the weight gradient vectors in the optimum direction [166]. Equation (3.10) shows the weight changes in each iteration without the momentum term, whilst Equation (3.11) denotes the weight changes with the momentum term [166]:

$$\Delta \boldsymbol{w}_{ab}^{\tau} = \eta \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{w}_{ab}^{\tau}} \tag{3.10}$$

$$\Delta \boldsymbol{w}_{ab}^{\tau} = \eta \frac{\partial E}{\partial \boldsymbol{w}_{ab}} + \beta \Delta \boldsymbol{w}_{ab}^{\tau-1} \tag{3.11}$$

where w_{ab}^{τ} is the value of the weight connecting node a to node b in iteration τ, E is the error rate in the weight, and β is the momentum term.

3.3 Sparse Representation

Sparse representation involves simplifying data with the aim of reducing redundancy or noise in data [167]. This representation can also provide additional insight into the content of the signal. Sparse representation has been applied to the MI EEG classification problem in three distinct ways:

a. Classification based on a SL dictionary and residual reconstruction error [11], [39];
b. Using the sparse coefficients for classification with a classifier [15], [40];
c. Using sparse representation for channel or feature selection [41], [42].

Each of these applications will be discussed in more detail in the following sub-sections.

3.3.1 Classification Based on Reconstruction Error

This approach [11], [39] involves constructing a dictionary of examples from the training data segments. A dictionary-based SL approach by Sreeja et al. [11] was included in Table 3.1. In this work, the dictionary is based on sub-dictionaries for each MI class, built using EEG training segments for that class. Thus, if the classification problem involves left-hand vs right-hand MI, a sub-dictionary is constructed based on left-hand MI training segments, and another is constructed based on right-hand MI training segments. The training segments are typically converted to a feature vector before being inserted into the dictionary [11], [39]. Examples of features used are CSP [39] and wavelet energy [11], [56].

To classify a test feature vector, $\boldsymbol{y}$, it is encoded over the sub-dictionary, $\boldsymbol{D}$, to obtain the sparse coefficient vector, $\boldsymbol{c}$. The residual reconstruction error, e, is calculated as [11], [42]:

$$e = \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2 = \|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{c}\|^2 \tag{3.12}$$

where $\hat{\boldsymbol{y}}$ is the estimate of $\boldsymbol{y}$ based on the sparse encoding, and $\|.\|^2$ is the Euclidean norm. The test feature vector is assigned to the class whose sub-dictionary gave the lowest value of e.

State-of-the-art dictionary-based classification systems for MI EEG typically use orthogonal matching pursuit (OMP) for sparse encoding [11], [56], [57]. OMP is a linear modelling algorithm that can be used to encode a vector over a dictionary [54]. Due to its frequent use in the state-of-the-art, OMP was used in this thesis as part of a sparse learning classifier in Chapter 5.

The OMP algorithm encodes the vector over the dictionary according to a sparsity constraint [11], [54]. This constraint is assigned by the user and limits the number of non-zero coefficients in the encoding. It is a parameter that can be tuned. Letting the vector be v and the dictionary be matrix $\boldsymbol{D}$, the OMP optimization problem can be summarized as follows [54]:

$$\underset{z}{argmin}\|v - \boldsymbol{D}z\|_2^2 \ , where \ \|z\|_o \leq K \tag{3.13}$$

where z is the encoding and K is the number of non-zero coefficients. The OMP algorithm aims to construct a linear reconstruction of v through the matrix multiplication $\boldsymbol{D}z$, where $v \approx \boldsymbol{D}z$. It achieves this using a greedy forward search, assigning non-zero coefficients the dictionary entries (atoms) which have the highest correlation with the residual reconstruction error [54].

An overview of the OMP algorithm is shown in Algorithm 1 [54]. The OMP works iteratively until the required number of non-zero coefficients, K, are obtained in z. Note that d_q^T is the transpose (T) of the dictionary atom at q and r is the residual, therefore in line 6 the dictionary atom that is most highly correlated with the residual is selected. This is the greedy step. S is the support which is built up of the indexes of the atoms selected in line 6. In line 8, the signal

Algorithm 1: Orthogonal Matching Pursuit

1. Inputs: Dictionary $\boldsymbol{D}$, input vector v and sparsity constraint K
2. Outputs: Sparse encoding z
3. Variables: S is the support set, q is an index to add to the support
4. Initialization: S = [], $r = q$
5. **while** $\|z\|_o \; != \; K$ **do:**
6. q = argmax $\left|d_q^T r\right|$ (Greedy step)
7. Append q to S
8. $z_S = (\boldsymbol{D_S})^+ v$ (Orthogonalization step, where $()^+$ is the Penrose-Moore Inverse)
9. r = q - $\boldsymbol{D_S}\, z_S$
10. **end**

is projected orthogonally over the atoms selected, and then the residual is calculated in line 9. The orthogonalization step is important because it ensures that there is linear independence in the atoms chosen [54]. The equation in line 8 can be expanded as [54]:

$$z_S = (\boldsymbol{D_S})^{+} v$$
$$= (\boldsymbol{D}^T \boldsymbol{D}_S)^{-1} \boldsymbol{D}_S^T v \quad (3.14)$$

Explicitly calculating the inverse in Equation (3.14) is computationally expensive, and it is instead calculated using the Cholesky factorization method [54].

There are different implementations of OMP [54], [55], [168], and in this thesis the approach by Rubinstein et al. [54] was used, because it is more computationally efficient than traditional OMP [54] and forms part of the *Scikit-Learn* package in Python [169] . The efficiency of the algorithm by Rubinstein et al. is a result of how the Cholesky factorization is calculated for large signal sets, and more details on the mathematics of this can be found in their paper [54]. Essentially, Rubinstein et al. [54] use pre-calculations to speed up computation times.

Since non-stationarity within the EEG samples could impact the effectiveness of the sparse dictionary, some papers in this area try to adapt the dictionary to the test dataset [11], [39]. In their 2015 paper, Shin et al. [39] tested supervised and unsupervised methods of calibrating the dictionary. First, an incoherence measure was used to compare the dictionary to the test data, and the dictionary entries which were most discordant were identified. Assuming an online scenario, past test samples can be used to update the dictionary in either a supervised or unsupervised manner. The accuracy was increased from 82.90% to 85.60% when using the adaptive updating. In their 2020 paper, Sreeja et al. [11] tried to compensate for non-stationarity using an approach which introduced locality to the dictionary. Before encoding the feature vector over the dictionary, Sreeja et al. derived a multiplicative weight for each dictionary atom based on the Euclidean distance n_j between the test feature vector and the j^{th}

dictionary atom. The weight, w_j, for the dictionary atom was calculated using a Gaussian formula: $w_j = e^{\frac{-n_j}{2\sigma^2}}$, where σ is the mean distance of the dictionary atoms. Without the weighted correction, the accuracy on the BCI Competition III dataset IVa [86], a two-class classification problem, was 96.91%, and with the correction it increased to 97.98%.

In their promising recent works, Sreeja et al. [11], [56], [57] have typically built the dictionary using features such as wavelet energy [56][57], discrete wavelet transform coefficient features [11], and the frequency domain band-power after CSP filtering have been used [56]. Research by Arnin et al. [124] discussed previously in Section 3.1.3 has suggested that frequency-based approaches to feature extraction are more computationally expensive than using time-domain based approaches such as time-domain band-power, and may not significantly improve classification performance. In Chapter 5, a SL-based classification approach is presented, and the combined alpha and beta time-domain band power is used for feature extraction. The combined power is used since subjects can experience changes in both the alpha and beta frequency bands during MI [34]. Moreover, in a short analysis summarized in Chapter 5, the combined power was found to be more effective than extracting separate features for the alpha and beta frequency bands. To the best of the author's knowledge, this feature has not been used for dictionary construction in this kind of classification system before.

Furthermore, Sreeja et al. [11], [56], [57] do not investigate the effect of sparsity level, or the effect of the window size used to segment EEG data for dictionary construction. These design parameters were tuned for the system in Chapter 5, and the tuning results are demonstrated graphically.

3.3.2 Sparse Representation and Classification

Sparse representation has been used as a feature extraction technique, with the sparse encoding being passed onto a classifier [15], [40]. In 2018 She et al. [15] used CSP feature extraction to construct the dictionary and then used an extreme learning machine (ELM) for classification based on the sparse encoding. Later,

Taran and Bajaj [40] obtained a sparse representation using a FD technique called the tunable Q-factor wavelet (TQFW) and classified feature vectors using a least-squares SVM classifier. On BCI Competition III dataset IVa [86], a two-class classification problem, the implementation by She et al. [15] obtained an accuracy of 87.54% whilst that of Taran and Bajaj [40] obtained an accuracy of 96.89%.

3.3.3 Sparse Representation for Channel or Feature Selection

The sparse representation coefficients can provide information about the test feature vector. Since dictionary atoms are designed to capture characteristics of the test feature vector, the highest valued encoding coefficients indicate the characteristics most associated with the test signal, whilst zero-valued coefficients indicate which characteristics are not associated with the test signal. This sub-section discusses two implementations which use this aspect of sparse representation, one for channel selection [41] and the other for feature selection [42].

In 2011, Arvaneh et al. [41] applied sparse representation to an EEG channel selection problem. They used sparse CSP (SCSP) feature extraction as the basis of their optimization problem, which involved finding the smallest subset of EEG channels that would provide the same or improved classification accuracy when compared to using all EEG channels. Sparse spatial filters were constructed, with a sparse coefficient being associated with each EEG channel. In the search for the smallest subset, channels associated with zero-valued coefficients were discarded and the remaining channels were ranked according to the sparse coefficient values. An SVM-RBF classifier was used to classify the spatially filtered signals. This channel selection approach produced better performance than other common channel selection approaches, including the Fisher criterion, mutual information, and traditional CSP channel selection. For the two-class problem in BCI Competition III dataset IVa [86], using this channel selection approach resulted in a mean classification accuracy of 82.28%, compared to 73.56% when using all EEG channels.

Later, Zhang et al. [42] extracted features using FBCSP, and then used the sparse representation coefficients to select the most salient features to be used for classification with an SVM-Linear classifier. Using the same dataset as Arvaneh et al. [41], they obtained an average classification error rate of 7.95% with the proposed feature selection method, which was an improvement from using just FBCSP, which gave an error rate of 9.50%.

3.4 Deep Learning for MI EEG Classification

Research into using DL for MI EEG classification has been growing in recent years [6]–[9], [12], [29], [58]–[60], [65], [102], [138], [164], [170]–[176]. In their 2021 paper, Al-Saegh et al. [6] performed a broad review of 40 papers related to DL for MI classification, and found that 73% of systems were based on CNNs, 14% were based on other DL systems such as recurrent neural networks, deep belief networks, ELMs or stacked autoencoders, and the remaining 13% of the literature involved hybrid CNNs (h-CNNs), which consist of a CNN module or CNN layers together with other DL structures, such as long short-term memory (LSTM) units or an autoencoder for post-processing. There is also variation in pre-processing applied to the EEG time-series before it is input to a DL classifier: 31.7% of systems use time-series as input, 31.7% use images - typically TFD images from the CWT or STFT- and 36.6% used calculated features such as CSP, FBCSP or EMD [6].

Since the state-of-the art for DL systems applied to MI EEG classification has shown that CNNs and h-CNNs that process multichannel time-series data are strong candidates [6], and DL -related research in Chapter 6 of this thesis has been focused on CNN systems. I

The following section provides a technical introduction to CNN-based architectures in general. This is followed by a discussion of state-of-the-art architectures.

3.4.1 Convolutional Neural Networks

This section first introduces general concepts related to CNNs and then goes on to discuss two salient CNN architectures in the literature, ShallowConvNet [8] and EEGNet [7]. These architectures are explained in detail here since they have been used either for benchmarking or inspiration in this thesis.

3.4.1.1 General Introduction to Convolutional Neural Networks

CNNs are deep-learning approaches inspired by the visual cortex. They are built using consecutive layers, as shown in Figure 3.8. CNNs consider the 2D nature of the input data, making them more powerful and versatile than simple MLP classifiers, which classify data based on feature vectors. The depth of a CNN is determined by how many layers it has, and shallower layers typically extract more generic features, whilst deeper layers extract progressively more abstract features [150] (pp. 267-269). CNNs are typically used for image processing [150] (pp. 267-269) and have been used to classify time-frequency domain images such as spectrograms for EEG classification [12]. However, CNNs have also been applied to segments of time-domain multichannel EEG signals, as shown in Figure 3.8 [7], [8], [60]. In these cases, the input data to a CNN will be referred to as a time-series segment of size $(1 \times N \times T)$, where N is the number of channels and

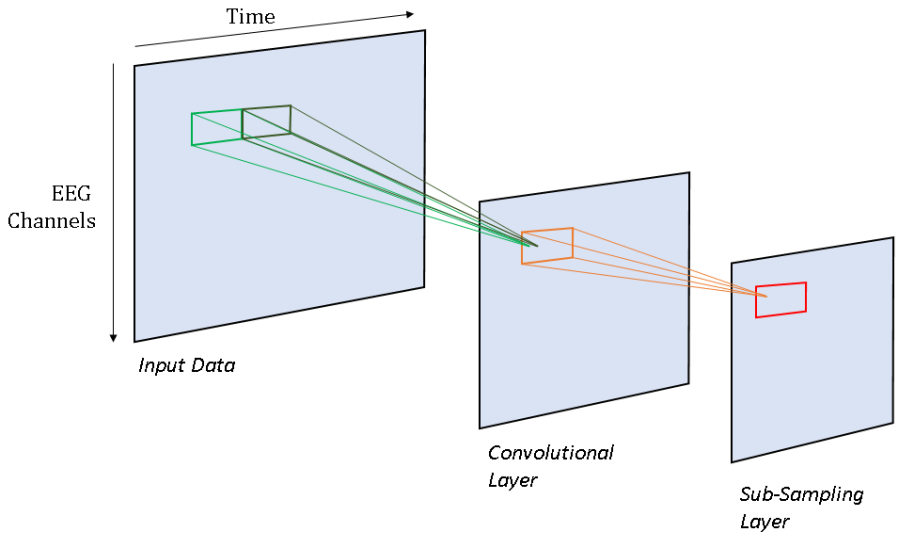

Figure 3.8: Part of a generic CNN for EEG classification. The input data is a time-series segment, which is first processed in the convolutional layer and then a sub-sampling layer.

T is the number of time-samples. This input time series segment is treated in the same way as an image by the CNN.

CNN Building Blocks

CNNs are designed to extract local features from input images [150] (pp. 267-269), as shown in the relationship between the input data layer and the convolutional layer in Figure 3.8. For simplicity, the convolutional layer and sub-sampling layer have been shown to have one plane. However, in general, each convolutional and sub-sampling layer has multiple planes, with each plane representing different features in a feature map.

Three core concepts underly CNN functionality: i) local receptive fields, ii) weight sharing, and iii) sub-sampling [150] (pp. 267-269). Consider the simple network in Figure 3.8. The convolutional layer extracts a series of feature maps from the input data. Each unit within a feature map is extracted from a local region within the input image. Each feature map has a particular set of weights associated with it, which are learnt during the training phase. These weights map the local data at the input to the corresponding unit in the feature map. Weight sharing ensures that the transformation weights for each individual feature map are the same for the whole input image, such that the process of extracting feature maps can be considered equivalent to convolving the input data with a kernel with those weights [150] (pp. 267-269). These convolutions can be considered a type of filtering, thus the number of 'filters' used in a convolutional layer corresponds to the number of feature maps extracted. Soft weight sharing uses regularization to provide some more flexibility than traditional, strict, weight sharing [150] (pp. 267-269). Convolutional layers typically have an activation layer (not shown in Figure 3.8) at their output [7], [8]. The activation layer performs similarly to the activation function in the multilayer perceptron, but at a larger scale on whole feature maps. Activation functions considered in this work are the elu [7] and square-log functions [8], given as:

$$\text{Elu: } \begin{cases} x, & x \geq 0 \\ \alpha(e^x - 1), & x < 0 \end{cases} \tag{3.15}$$

$$\text{Square-log: } log_{10} x^2 \tag{3.16}$$

where α is a tunable decay parameter. The log function can also be clipped to prevent very large or very small numbers being output [8]. These functions were chosen based on the recommendations in [7], [8] for the CNN architectures used.

Sub-sampling layers, also known as pooling layers, typically follow convolutional layers. Each feature map is transformed to a new plane in the sub-sampling layer. The sub-sampling layer considers local regions within the feature map and transforms the data to a single unit in the new plane by means of a pooling function [150] (pp. 267-269). This pooling function is typically a simple mathematical function, such as the average or maximum of the points in the local field [7], [8], [60]. Sub-sampling helps to decrease the number of parameters which need to be trained, leading to faster networks, and the feature map smoothing can help prevent overfitting. However, when applied poorly, sub-sampling layers can over-smooth the data, leading to underfitting [150] (pp. 267-269).

A typical CNN can consist of multiple convolutional and sub-sampling layers [150] (pp. 267-269). At the end of the network, the final feature maps are flattened using a 'Flatten' layer, which simply unravels and concatenates the data in the feature maps to create a final feature vector [7], [8], [60]. This feature vector is then input to one or more fully-connected layers like those in the MLP discussed in Section 3.2.6. These layers are typically known as 'Dense' layers [7], [8], [60] In this work, classes are encoded using one-hot encoding, so the output layer consists of n neurons where n is the number of classes in the classification problem [7], [8], [60]. The final layer uses a SoftMax function, which outputs a number from 0 to 1 on each output neuron, with the value denoting the probability that the input sample is part of that class [7], [8], [60].

Depthwise and Separable Layers

Depthwise layers and separable convolution layers are special CNN layers designed to improve computational complexity [177], [178]. They were originally

used in computer vision, but have also been applied to EEG data classification [7], [65], [102]. In a depthwise layer, each input channel is processed individually using the convolutional kernel [177]. Thus, unlike a basic 2D convolutional layer, there is no mixing of data between different channels. This layer has a parameter called the depthwise multiplier, D, that determines how many feature maps are derived from each input channel [7], [177]. In a separable convolution layer, depthwise spatial convolution is carried out on each individual input channel, and then the results are mixed using a pointwise convolution [178].

Preventing Overfitting

To prevent overfitting, modern CNNs [7], [8], [60] typically use batch normalization layers [179] and dropout layers [180]. During training, batch normalization layers ensure that the output distribution from the layer has approximately zero mean and unit standard deviation [181]. During classification of a test sample, known as the inference stage, the layer normalizes its output using a moving average which corrects the output relative to the mean and standard deviation of the training data [181]. This means that if the testing data is very different from the training data, this layer can disrupt performance. A dropout layer operates during the training stage, and randomly ignores a percentage of the outputs from the previous layer [180], [182]. It is a type of regularization that can help the network learn more robust features and can be viewed as either adding training noise or forcing sparsity into the representation. The outputs which are ignored are changed every training epoch [180]. Dropout layers do not force any drops in connections during inference [182].

3.4.1.2 Discussion of CNN-Based Classification Approaches in the Literature

Table 3.2 shows a selection of CNN-based classification approaches that are most relevant to the contributions in Chapter 6 [7], [8], [65], [102], [106], [171], [174]. The table provides information on the special name given to the architecture (where applicable), as well as the main contribution of each study, and the classification accuracy obtained. The table summarizes the performance of various salient architectures and the main contributions of each paper. The table

Table 3.2: A table summarizing salient CNN-based classification architectures.

Paper	Dataset	Architecture Name	Main Contribution	Classification Accuracy
Schirrmeister et al. [8] (2017)	HG dataset	ShallowConvNet	Investigating in-depth the potential of CNNs to be applied to MI EEG time-series data for classification.	93.90%
Lawhern et al. [7] (2018)	Graz 2A dataset	EEGNet	Applied depthwise and separable convolution layers to a CNN architecture for MI EEG classification.	~70.0%[1] (ShallowConvNet: ~70.0%[1])
Kumar et al. [106] (2017)	BCI Competition III, Dataset IVa	N/A	Using CSP features with a deep neural network	~ 91.00%[1]
Zhao et al. [171] (2019)	Graz 2A Dataset	3D CNN	Multi-branch 3D CNN classifier	75.02%
Wu et al. [174] (2019)	HG dataset	MSFBCNN (multiscale filterbank CNN)	Design of a MSFBCNN based on CSP features. Used retraining transfer learning for fine-tuning.	89.30%
Huang et al. [102] (2020)	Graz 2A Dataset	S-EEGNet	Applied Hilbert-Huang transform pre-processing, and using bilinear interpolation to add a displacement variable to the CNN	77.90%
Roots et al. [65] (2020)	Physionet MI Dataset	EEGNet-Fusion	Multi-branch CNN based on EEGNet, trained using mixed data transfer learning.	83.80% (EEGNet: 65.80% ShallowConvNet: 77.00%)

[1]Results were obtained by interpreting a graph.

indicates that in recent years there has been increased interest in transfer learning [65], multi-branch CNNs [65] and 3D CNNs [171]. In Chapter 6, transfer learning is explored in more depth and a comparison between retraining transfer learning and mixed data transfer learning is presented. It is also notable that the performance of the same architecture can vary depending on the dataset used, for example ShallowConvNet [66] had an accuracy of 93.90% when tested with the HG dataset, but an accuracy of ~70.00% when tested on the Graz 2A dataset. This discrepancy could be due to the HG dataset being recorded in an environment with electromagnetic shielding, resulting in low noise content in the signals. Conversely, the Graz 2A dataset was recorded in average lab conditions, without special shielding, meaning the data can be expected to have greater noise content. Each of these architectures is discussed in more detail throughout this section, but since the architectures in the table are largely inspired by EEGNet [7] and ShallowConvNet [8], these architectures are discussed next.

3.4.1.3 ShallowConvNet and EEGNet

The ConvNet [8] and EEGNet [7] architectures, published in 2017 and 2018 respectively, are landmark CNN classification systems which take EEG time-series data as input. The ConvNet paper [8] introduces ShallowConvNet and Deep ConvNet, which, as the names imply, are shallow and deep CNN networks for EEG classification. The DeepConvNet architecture was found to work poorly on one of the datasets used in Chapter 6 [7] and was not included in work in this thesis.

ShallowConvNet

ShallowConvNet (2017)[8] is a CNN used in BCI studies [7], [65]. The structure of ShallowConvNet is shown in Table 3.3. As input it takes an array with shape (1, Nº Channels, Samples), with the 1-dimension accommodating for the extraction of the feature maps in the first convolutional layer, Nº Channels being the number of EEG channels in the segment and Samples being the length of the EEG segment. Typically, the length is the segment is fixed to 2s worth of samples [7], [8]. The EEG data is first passed through a convolutional layer performing temporal convolution on each channel. The kernel size, W, determines the number of transformations that can be performed in the layer, with larger values of W leading to a larger variety of transformations that can be carried out [8].

Figure 3.9 shows ShallowConvNet applied to a practical classification problem from Chapter 6. The EEG data has 22 channels and a sampling frequency of 128Hz, leading to a 2s input segment of (1x22x256). The arrows denote EEG layers, and the intermittent diagrams illustrate how the shape of the data changes

Table 3.3: A summary of the structure of ShallowConvNet.

Layer	**Details**
Input identity layer	Shape: (1, Nº Channels, Samples)
Conv2D Layer *(Temporal convolution)*	40 filters, kernel size (1, W)
Conv 2D Layer *(Spatial filtering)*	40 filters, kernel size (Nº Channels, 1)
Batch Normalization + Dropout	Dropout rate: 0.5
Activation Layer	Square activation function
Average Pooling (2D)	Pool-size (1, Q), kernel size (Q/5,1)
Activation layer	Log activation layer
Flatten layer	-
Dense Layer *(Linear classification output layer)*	4 units; Softmax activation

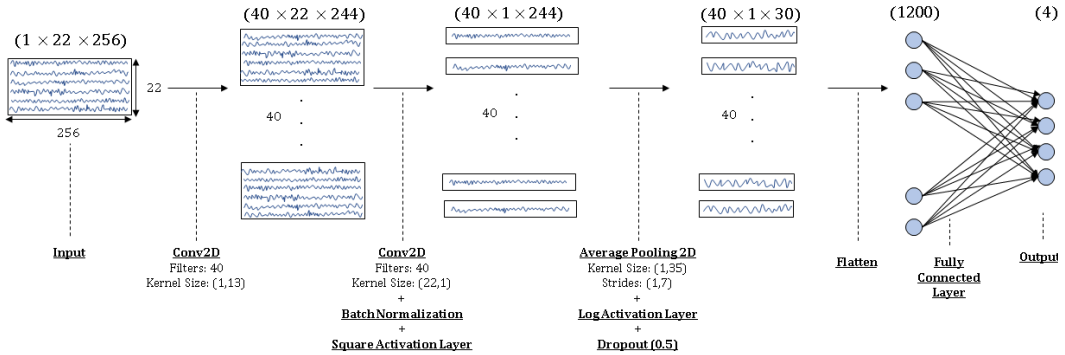

Figure 3.9: ShallowConvNet applied to an EEG segment with 22 channels recorded at 128Hz. The input is a 2s segment of time-series data. The arrows denote CNN layers, which are labelled using the dotted lines. The time-series snippets illustrate the output of each layer, with the sizes of the outputs shown in brackets. Note that batch normalization and dropout layers do not affect the data size.

as it travels through the network. Note that the batch normalization and dropout layers do not affect the data size.

The process of temporal and then spatial filtering was designed to mimic classical CSP feature extraction, whilst allowing the CNN to extract complex features instead of being restricted to the statistical features extracted using CSP [8]. The convolutional layers are followed by batch normalization and dropout layers.

A square activation function is then applied, followed by average pooling. These two layers, together, aim to extract the mean power of the input signals, given that the input signal is zero-mean. Even if the input signal is not zero-mean, the squaring function makes the features extracted sensitive to the power of the signal, which is important in MI EEG classification [8]. Recall that the CSP features in Section 3.1.5 were calculated using the log-variance. In the ShallowConvNet system, the log activation layer aims to mimic this final stage of CSP feature extraction. Finally, a dense layer with four neurons and SoftMax activation is situated at the output. Four neurons are used because ShallowConvNet has been applied to four class problems [8], with the target classes encoded with one-hot encoding.

Numerous EEG CNN classifiers published in recent years have structural similarities to Shallow ConvNet, first performing temporal convolutions followed by spatial convolutions in the initial layers [7], [60], [65].

EEGNET

EEGNet (2018) [7] is a CNN used widely in the literature for benchmarking and as a basis for novel EEG classification techniques [65], [102]. Both EEGNet and ShallowConvNet were included in Chapter 6 because they have different architecture and activation functions but have been shown to have similar performance on an MI EEG dataset [7]. This made them effective for testing the novel channel selection approach presented in Chapter 6.

Like ShallowConvNet, EEGNet was inspired by CSP-based feature extraction techniques, and processes data in the temporal domain to extract feature maps associated with different frequencies, and then processes data in the spatial domain [7]. Table 3.4 shows the general layout of EEGNet. In the original paper [7], the number of filters, F_1, in the first layer were fixed at 4 or 8, the depthwise multiplier D in the depthwise convolution layer was set to 2 and the number of filters in the separable convolution, F_2, was fixed as $F_1 \times D$. EEGNet

Table 3.4: A summary of the structure of EEGNet.

Layer	**Details**
Input identity layer	Shape: (1, N^o Channels, Samples)
Block 1	
Conv2D Layer *(Temporal convolution)*	F_1 filters, kernel size (1, F_s/2)
Batch Normalization	-
Depthwise Conv2D	Kernel Size (N^o Channels, 1), depthwise multiplier D
Batch Normalization	-
Activation Layer	Activation function: Elu
Average Pooling (2D)	Pool size: (1,4)
Dropout Layer	Dropout rate: 0.5
Block 2	
Separable Conv2D	F_2 filters, kernel size (1,16)
Batch Normalization	-
Activation Layer	Activation function: Elu
Average Pooling (2D)	Pool size (1,8)
Dropout Layer	Dropout rate: 0.5
Block 3	
Flatten Layer	-
Dense Layer *(Linear classification output layer)*	4 units; SoftMax activation

is one of the earliest examples of EEG-based CNN classifiers using depthwise and separable convolution layers [7]. EEGNet has some similarities with ShallowConvNet: the input signals are arranged in a similar shape and the 'Samples' variable encapsulates 2s worth of time samples. Note that in its original implementation, EEGNet was tested using MI EEG signals with a sampling frequency of 128Hz, and some of the design parameters discussed in the rest of this sub-section are borne out by this fact [7]. Figure 3.10 shows EEGNet applied to a practical EEG classification problem from Chapter 6. It uses data similar to that in the example for ShallowConvNet in Figure 3.9 and can also be used as reference. The depthwise multiplier, D, is set to 2 as in [7].

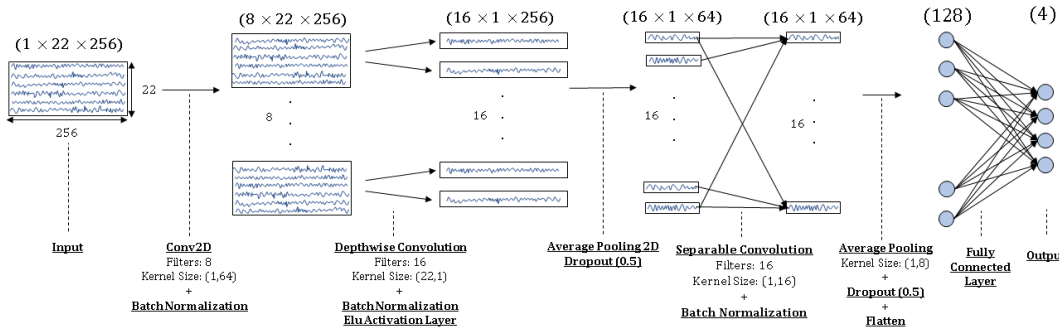

Figure 3.10: EEGNet applied to an EEG segment with 22 channels recorded at 128Hz. The input is a 2s segment of time-series data. The arrows denote CNN layers, which are labelled using the dotted lines. The time-series snippets illustrate the output of each layer, with the sizes of the outputs shown in brackets. Note that batch normalization and dropout layers do not affect the data size.

Consider Block 1 in Table 3.4. The Conv2D layer carries out a temporal convolution which extracts F_1 feature maps. In Figure 3.10 eight feature maps are extracted. These feature maps are intended to extract salient frequency content from the time series. In fact, the kernel size of (1, $F_s/2$) was designed such that frequencies of 2Hz or more would be extracted in the feature maps. In Figure 3.10 this corresponds to a kernel of size (1,64) since the sampling frequency is 128Hz. Batch normalization is carried out followed by depthwise convolution, which aimed to obtain a spatial filter for each feature map. The block closes with batch normalization, 'elu' activation, average pooling, and a dropout layer. The average pooling layer uses a pooling size of (1,4) to effectively downsample the input signal of 128Hz to 32Hz.

The separable convolution layer in Block 2 aims to extract temporal features that summarize the information in each feature map, and then mix this information using the pointwise convolution step. The separable convolution has a kernel size of (1,16) meaning that the depthwise convolution step summarises the activity in each feature map into a 500ms segment. A separable convolution layer was used because it requires fewer parameters to be trained that a standard convolutional layer, leading to shorter computational times. Furthermore, using a separable layer means that an initial depthwise convolution is carried out on each individual feature map, enabling decoupling of the feature maps. This is advantageous because different feature maps may represent activity in different frequency bands. Like Block 1, Block 2 closes off with batch normalization, elu activation, average pooling, and a dropout layer. The final average pooling layer was used for down-sampling of the extracted features.

EEGNets' final block performs classification in the same way as ShallowConvNet, by flattening out the feature maps and inputting the feature vector to a dense layer with SoftMax activation.

ShallowConvNet, EEGNet and Related Works in the Literature

EEGNet and the ConvNet architectures have been widely used for benchmarking and comparison in MI EEG classification research [7], [24], [58], [59], [164], [165], [183]. EEGNet has formed the basis of other DL classifiers such as S-EEGNet [102] and the fusion CNN [65], and has also been used in a neurological study [38]. In the S-EEGNet system [102], the signals are pre-processed using the Hilbert-Huang transform to produce TFD signals that are input to an EEGNet-based architecture with bilinear interpolation. The fusion CNN [65] classifier has three branches, with each branch based on a traditional EEGNet structure. Recall that the stride size used in the first convolutional layer of EEGNet controls the frequency content that can be extracted. In fusion CNN, the branches have stride sizes of (1,64), (1,128) and (1,256), which were designed to capture frequency features in different ranges. Both S-EEGNet [102] and fusion CNN [65] obtained better results than the original EEGNet architecture, as shown in Table 3.2. The ConvNet architectures have also inspired other works, such as the system by Hou

et al. [170] that first locates a region of interest within the motor cortex, then extracts features using wavelets, and classifies signals using a ConvNet-inspired pipeline.

Multi-branch CNNs are becoming more common in the literature, although the way in which the outputs of multiple branches are fused differs [59], [60], [171]. For example, Amin et al. [60] trained four CNNs of varying depths separately. The CNNs were designed and trained to extract frequency content in different bands. The outputs of the CNNs were then concatenated, fused using an autoencoder/decoder, and then a fully connected layer attached to the decoder output assigned the final classification label. The autoencoder and fully connected classification layer were trained separately. The multi-branch configuration outperformed each of the individual CNNs as well as different combinations of the individual CNNs, but at the expense of greater computational complexity. In 2019 Zhao et al. [171] developed a multi-branch CNN based on 3D CNNs. The other CNNs discussed till now were 2D CNNs, that looked at EEG data in a 2D space consisting of channels (rows) and samples (columns). The 3D CNNs enabled the input data to be structured such that the spatial locations of the electrodes could be included in the representation. The three branches were designed to extract data from small, medium, and large receptive fields. The outputs of the three branches are summed within an addition layer, and a final SoftMax layer is used to obtain the classification label. The system was trained end-to-end, unlike the implementation by Amin et al. [60], that required each CNN to be trained separately. Zhao et al. [171] compared the performance of the proposed 3D CNN which takes raw EEG data to that of a 2D CNN taking CSP features as input, and the 3D CNN was found to perform better. However, since they did not compare the performance of the 3D CNN classifier to that of a 2D CNN taking raw EEG data as input, such as EEGNet [102], they did not ascertain whether using a 3D architecture actually leads to significant improvement in classification accuracy when compared to a 2D architecture with similar input. Networks with inception modules for EEG time series classification have also

been developed, but current models need to be tailored for the classification problem, limiting their versatility [9].

In Chapter 6 of this thesis a novel and versatile CNN-based channel selection approach is presented. In order to test the novel CNN channel selection approach, robust and reliable CNN-based classifiers were needed. Since EEGNet[7]and ShallowConvNet [66]have been popular for benchmarking, or as a basis for novel techniques [7], [65], [102], they were adopted for assessing the versatility and performance of the channel selection approach. An attempt was made to implement the multi-branch approach proposed by Roots et al. [65] as an improvement to the original EEGNet classifier, however when testing on the datasets in Chapter 6 it achieved poorer performance than EEGNet despite thorough parameter tuning that covered dropout, convolutional parameters (number of filters and stride length), and the size of the kernel in the pooling layers. Thus, this network was not used in the analysis. Roots et al. [65] did not test on the datasets used in Chapter 6, and it is known that the performance of CNN-based architectures can vary between datasets [9], which may explain its poorer performance. The dataset used by Roots et al. [65] was not used in Chapter 6, since the focus in this chapter was specifically on multiclass classification problems, and Roots et al. [65] used a two-class dataset.

3.4.2 Cross-Subject Classification

Cross-subject classification is a groundbreaking frontier in MI EEG signal processing and has been a topic of investigation for CNN-based classification systems [7], [58]–[60]. In an ideal cross-subject framework, the BCI is trained on data from a group of subjects, known as the source signals, and tested on the data of another subject/s, known as the test signals. Although systems with subject-specific training tend to outperform cross-subject systems [7], [58]–[60], the collection of training data from subjects and the ensuing latency for training the classification system are time-consuming and could limit the widespread use of BCI technology. High performing cross-subject BCIs would solve these issues, paving the way for commercial systems.

The current challenge is finding a cross-subject classification pipeline which performs on-a-par with subject-specifically trained systems. Many papers presenting cross-subject systems also test their pipelines with subject-specific training, and the results for subject-specific training are consistently better [7], [58]–[60]. This could be due to inter-subject variability: MI EEG activity can occur within different frequency bands and on different EEG channels for different subjects [13], [34], [163], [184], [185]. There can also be a lag between EEG channels which exhibit ERD/ERS behaviour [82]. Another issue is the lack of an independent ground-truth within MI EEG datasets: it is universally assumed that, when instructed, the users are generating MI signals. There is no standardized way to identify whether subjects correctly carried out the MI task for every trial, or whether they maintained concentration for the duration of task generation. Some studies attempt to verify that MI activity exists by monitoring the frequency content of the EEG signals [30], identifying ERD/ERS behaviour [82], [124], [186], or using z-score analysis to remove outlier channels which may not exhibit MI activity [131]. However, there is no standard quantitative approach for ground-truth assessment. Furthermore, some subjects seem to have an inherent inability to carry out some BCI-related mental activities, a phenomenon known as BCI illiteracy [89]. Some subjects generate signals which perform poorly across studies [7], [172], so much so that some researchers generate separate results for subjects which generally perform well, and those who generally perform 'poorly' across studies [52], [172].

Many approaches use training data from the target subject in order to calibrate the cross-subject system [12], [29], [58], [60], [65], [138], [164], [172]–[176]. Although these approaches fall short of an ideal, generalized, cross-subject system which requires no calibration data from the target, they aim to reduce the amount of training data required from target subjects [29], [187]. Calibrated approaches can be broken down into 3 categories: i) including target-specific data together with source data within the training dataset, called mixed data transfer learning (MTL) in this thesis [65], [172]; ii) retraining transfer learning

(RTL) [58], [60], [174], [175], and iii) minimizing distribution divergence between the source and target domains [164], [173].

Transfer learning is a classic technique for improving the performance of DL systems. RTL, where the network is pre-trained on source data and then fine-tuned for the target subject, is widely used in the MI EEG classification literature [58], [60], [174], [175]. The most straightforward approach, presented by Li et al. [175], involves pre-training using data from source subjects in the same dataset, and then uses a small amount of target data to fine-tune the networks. Amin et al. [58], [60] have pre-trained their CNN networks using data from another dataset, however they did not report whether this pre-training led to an improvement when compared to using randomly initialized weights. Wu et al. [174] also pre-trained their network on an open dataset, then fine-tuned using subject data. To reduce the amount of target data needed, they mixed the target data with data from another randomly selected dataset. This transfer learning method enabled the target data to be reduced by a factor of 2.8 with minimal deterioration in accuracy. However, there has been no thorough investigation into whether mixing source and target data for training the network (MTL) [65], [172], [174] or pre-training on data from other subjects within the same dataset and fine-tuning using only target data (RTL) [175] leads to better results. In Chapter 6 RTL and MTL are compared for two different CNN architectures on two different datasets. The methods are compared in terms of classification performance as well as computational times.

Pretraining has also been done using non-EEG data. For example, a CNN classifier was pre-trained using electromyogram (EMG) signals which capture muscle activity and then fine-tuned on EEG signals, resulting in improved performance compared to randomly initialized weights [188]. Other approaches have transformed EEG data into images using the STFT [12], [176] or CWT [12] and performed retraining transfer learning on the pretrained image classification CNNs AlexNet [12] and VGG-16 [176] .The CWT may be more effective than the STFT for this application [12].

Other studies use calibration approaches to minimize the distribution shift between the source and target domains [164], [173]. These systems are implemented as Siamese CNN networks, where two CNNs are trained in parallel, sharing weights between them. In this case, one CNN is trained using source data whilst the other is trained on target data [164], [173], and the training process aims to encourage universal features to be learnt. To align the features learnt, X. Zhao et al. [164] added joint distribution matching and marginal distribution alignment. Marginal distribution alignment makes the centroids of the source and target distributions aligned regardless of the classes, whereas joint distribution matching calibrates the output by considering both the features and class labels. H. Zhao et al. [165] used a three-pronged approach to extract more general features: i) centre-loss was added to the feature space to decrease inter-subject non-linearity differences; ii) a domain discriminator uses adversarial learning to decrease the distribution differences between the feature distributions of sources and targets; and iii) the classifier is trained using data from both domains. The approach used by X. Zhao et al. [164] obtained an accuracy of 69.6% for the Graz 2A dataset, which is a four-class problem. The approach by H. Zhao et al. [165] exhibited better performance, giving an accuracy of 74.75% for the same dataset.

These works into cross-subject classification illustrate that CNNs can extract generalizable representations from EEG data. This observation on the generalizability of CNNs helped form the fundamental concept underlying Chapter 6, which investigates how subject-independent channel selection can be carried out using CNNs. In subject-independent channel selection, a subset of EEG channels is selected for a target subject based on data from a group of source subjects. Channel selection, and the contribution in Chapter 6, are discussed in more detail in Section 3.5.2.

3.5 Static EEG Channel Subsets and Automated Channel Selection

This section discusses channel selection for MI EEG classification. The first part covers static EEG channel subsets used in the literature, where 'static' refers to manually selected subsets of channels. It then goes on to discuss automated channel selection techniques, in which channels are selected algorithmically.

3.5.1 Static EEG Channel Subsets Used with Conventional Classifiers

The static EEG channel subsets used for testing conventional MI EEG classifiers varies across the literature. In these approaches no algorithmic channel selection is carried out.

Some studies use all the EEG channels available within the chosen datasets [10], [21], [104]. However, because the number of EEG channels used for recordings varies widely across different datasets, this has led to different studies using different numbers of EEG channels. For example, Baig et al. [104] used 118 channels, Olias et al. [10] used datasets with 22, 60 and 118 channels, and Oikonomou et al. [21] used a dataset with just 3 EEG channels.

Other studies use an arbitrarily chosen subset of EEG. Here the term 'arbitrary' is used to refer to channel subsets that were not selected algorithmically but are chosen manually. Typically, these channel subsets are constructed of channels in the vicinity of the scalp region associated with the motor cortex, which are around the central scalp region. However, the number of channels included in these subsets and the scalp regions from which electrodes are derived varies [21], [22], [31]. Some studies use electrodes from the central-associated scalp regions C, CP and CF, for example Oikonomou et al. [21] selected 18 such channels from a 64-channel dataset, and Ilyas et al. [31] selected 11 such channels from a dataset with 59 channels. Other studies choose even more restrictive EEG subsets, selecting electrodes from only the C region of the scalp, such as Kevric and Subasi [22] who considered only electrodes C3, C4 and Cz from

a dataset containing 118 EEG channels. Although all 118 channels were used during pre-processing for de-noising, for classification only channels from the C region were considered [22].

There is conflicting evidence as to whether having a lower number of electrodes within a static subset improves performance or not [32], [33]. For example, Yang et al. [32] used a 32-channel dataset for left- and right-hand MI classification. From this dataset, they considered two static channel subsets, one consisting of eight EEG channels from the C, CP and P regions, and another consisting of three channels from the central region (C3, C4 and Cz). They found that the three-channel subset gave similar or improved performance when compared to the eight-channel subset. In a different study, Siuly et al. [33] considered data from two datasets, one containing right-hand and right-foot MI, and the other containing data for left-hand and foot MI. Both datasets had 118 channels available. They compared the performance of three different classification pipelines when using all 118 EEG channels and when using a static subset of 18 EEG channels from the C, CP, and P regions. For both datasets each of the three classification pipelines gave a higher classification accuracy when all 118 EEG channels were used, as opposed to the subset of 18 EEG channels.

These results of Yang et. al [32] and Siuly et al. [33] are conflicting, however comparison between the studies is limited by some factors. Firstly, the studies consider different MI classification problems. Left- and right-hand MI, as used in [32], produce desynchronization in the right and left hemispheres, respectively. The channels C3 and C4 are found on opposite hemispheres and may be adequate at capturing this activity. However, in [33] foot imagery was included. Foot imagery is known to produce bi-hemispheric activity, regardless of which foot is moved [189]. This may be a reason why using more channels improved the classification accuracy when the foot class was included [33], because it enabled the spatio-spectral patterns associated with each MI class to be identified by the classifier. Furthermore, Yang et al. [32] did not conduct a test using all 32 EEG channels. Thus, reducing the channels from eight to three may have removed some noisy EEG channels which were hindering classification, but considering a

wider variety of electrodes, such as those in the CF region, may have improved the identification of MI when compared to just three or eight channels [32].

The works of Yang [32] et al. and Siuly et al. [33] are examples of studies that focus on how the number of electrodes used can affect accuracy, without a strong focus on which scalp regions those electrodes derive from. As mentioned in the previous chapter (Section 2.1.1), different EEG scalp regions are associated with different mental processes. There is a lack of investigation into the contribution of each scalp region surrounding the strictly C region (namely regions CP and CF) to classification accuracy when a static EEG subset is used. For example, past research has suggested the parietal-associated regions can contribute actively to movement planning and execution [190]. However, an explicit analysis of the impact on classification performance of these channel groupings has not been carried out to the best of this authors' knowledge. In Chapter 4, a cross-classifier analysis is carried out to assess the contributions of channels from different scalp regions to classification performance.

3.5.2 Automated EEG Channel Selection

Automated channel selection involves using a computational or algorithmic approach to obtain a subset of EEG channels for classification. Channel selection is typically carried out on the training data, and the selected channel subset is then used on the test data to assess its generalizability. There are two main motivations for carrying out channel selection: i) improving classification accuracy through the removal of redundant or noisy channels; and ii) improving the computational time on the test set whilst maintaining the classification performance [44]. Sometimes both aims are the motivation, whereas on other occasions only one of the aims is the motivation [44].

In Chapter 5, the classification accuracy of the SL classifier presented was found to be high, but when using all the EEG channels available in the dataset, the computational time for test-set samples was much higher than that of some of the benchmarking systems, and possibly unacceptably high for a real-time BCI. A

novel channel selection module is placed before the SL classifier to select a subset of channels to use on the test set. The aim of channel selection was to preserve the high classification accuracy whilst, at the same time, reducing the computational complexity on the test dataset. This is a novel contribution because state-of-the-art dictionary-based sparse learning approaches typically use an arbitrarily chosen subset of EEG channels for processing [11], [56], [57], which may not lead to optimal accuracy.

Channel selection techniques are generally classified into two categories: filter or wrapper techniques [44]. Filter techniques involve using some form of statistical or information measure to rank channels in order of importance, then a number of channels are selected for classification [13], [46], [47], [84]. Wrapper techniques optimize the channel subset with respect to a performance measure, usually classification accuracy [48]–[50]. This means that wrapper channel selection involves training the classifier multiple times with different subsets, typically leading to improved results compared to filter techniques, at the cost of greater computational complexity [44]. Some techniques are a hybrid of filter and wrapper techniques [44], for example by selecting a large subset of EEG channels using a filter technique, then refining the subset using a wrapper technique [49]. Table 3.5 summarizes a collection of filter and wrapper channels selection techniques that are discussed in more detail throughout this section [13], [23], [24], [47], [48], [51], [105]. The table also includes CNN-based channel selection methods introduced later in this section.

3.5.2.1 Filter Techniques

Filter techniques based on correlation and covariance are common [13], [46], [47], although implementations can vary [13], [46]. For example, Jin et al. [13] assumed that channels involved in MI would be highly correlated, and removed channels that were correlated with relatively few other channels across trials. Conversely, in the first step of their algorithm, Park et al [46] obtained a correlation coefficient value for each channel that captured the correlation between the signals on that channel for two different MI tasks. Since highly discriminative channels can be expected to have a low correlation with

Table 3.5: A selection of channel selection approaches in this thesis. Note that the paper by Qui et al. was tested on two datasets, and that by Jin et al. was teste on three datasets.

Paper	Dataset	Channel Selection Method	Features	Classifier	Accuracy
He et al. [49] (2013)	BCI Competition III, Dataset IVa	Genetic Algorithm	Rayleigh coefficient features	LDA (named Fishers' linear discriminant classifier in the paper)	88.20%
Qiu et al. [48] (2016)	BCI Competition IV, Dataset 1	Improved SFFS	CSP	SVM	78.00%
	BCI Competition III, Dataset 2A				83.30%
Jin et al. [13] (2019)	BCI Competition IV, Dataset 1	Correlation Coefficient-based	Regularized CSP	SVM	81.60%
	BCI Competition III, Dataset IVa				87.40%
	BCI Competition III, Dataset 3A				91.90%
Gurve et al. [47] (2020)	Proprietary dataset	Non-negative matrix factorization analysis	Riemannian tangent-space features	LDA	96.66%
Mzurikwao et al. [23] (2019)	Proprietary dataset	CNN weight analysis	EEG time-series	CNN	91.50%
Idowu et al. [51] (2021)	Proprietary dataset	MPSO	CSP	MLP neural network	89.54%
Zhang et al. [24] (2021)	Proprietary dataset	CNN-based automatic channel selection module	CWT TFD images	CNN	87.20%

themselves for different classes, Park et al [46] extracted the channels with the lowest coefficients, labelling them 'highly discriminative' channels. For each highly discriminative channel they selected a collection of other channels called a support, with the support channels being highly correlated with the discriminative channel. In this way, they obtained a collection of different candidate subsets. They obtained CSP features using each candidate subset and selected the best group based on the Fisher score, which can be used to assess discriminability. Gurve et al. [47] used covariance instead of correlation whilst Yang et al. [84] used mutual information to rank and select a candidate subset of EEG channels that were most correlated with the class labels. Afterwards, channels within the subset were analysed to remove any that were redundant. Other filter techniques are based on obtaining discriminative CSP features [191] or Fisher score [30].

3.5.2.2 Wrapper Techniques

Sequential searches are a traditional wrapper channel selection technique [44]. They are greedy algorithms that iteratively search for the optimal subset by gradually adding ('forward search') or removing ('backward search') channels from a candidate subset and monitoring the effect of this on classification performance [44]. However, sequential searches are computationally expensive [44], [48]. Qiu et al [48] proposed an improved sequential floating forward search (SFFS) algorithm which considered electrodes that were physically near each other on the scalp as a single channel for search purposes. This reduced the computational time of the SFFS without any detriment to performance.

Heuristic wrapper channel selection techniques have gained popularity in recent years [49]–[51]. Heuristic techniques use general principles to help them reach a suitable solution in a faster time than traditional wrapper techniques [44]. The faster computational times can come at the cost of a less-than-optimal subset of channels, although heuristic techniques have been shown to perform strongly [49]–[51], in one study producing subsets that improved classification accuracy [49].

Metaheuristic techniques, that are inspired by natural processes [49], have been used for channel selection in BCI problems [49]–[51]. For example, in 2013 He et al. [105] used a genetic algorithm to select a subset of EEG channels based on extracted features. Genetic algorithms are inspired by the concept of natural selection, where a population of candidate solutions is created and then optimized such that better characteristics of the subsets are more likely to be promoted to a subsequent generation of candidate solutions. He et al. [105] showed that the GA approach could produce improved accuracy and outperformed sequential channel selection algorithms. In 2020 Moctezuma et al. used a GA for channel selection within a biometric EEG-based identification system [192].

Particle swarm optimization (PSO) is another metaheuristic technique that facilitates exploration of the solution space by mimicking the movement of a

flock of birds searching for food: each bird moves in the general direction of the current best-known food source scouted by the whole flock, but also explores the area around its local best food source [51]. In 2021 Idowu et al. [51] presented a modified particle swarm optimization (MPSO) algorithm that introduces an additional term into the PSO update equation to prevent it from getting stuck in local minima. MPSO channel selection outperformed other heuristic methods, namely: traditional PSO, GA, simulated annealing, and ant bee colony optimization techniques. Although the classification accuracy always decreased as the number of EEG channels was decreased, MPSO experienced a less rapid deterioration in performance.

In Chapter 5 channel selection is applied to a SL classifier with the primary aim of reducing the test-set computational time without affecting the high accuracy of the classifier. Since the SL classifier already had an involved computational time [54], a metaheuristic wrapper technique was used, since it provided a trade-off between the increased computational time associated with wrapper techniques and a better likelihood of accuracy preservation through the selected subset [44]. Thus, a novel GA was implemented for the selection of EEG channels for the dictionary-based sparse-learning EEG classifier. A GA was chosen because GAs have a good track record in EEG channel selection [49], [192], and have even showed potential for improving accuracy [37]. The following section gives a technical overview of GAs.

Genetic Algorithms

GAs are optimization algorithms inspired by evolutionary mechanisms that occur naturally within biological chromosomes and animal populations [193]. GAs are not problem-specific and generally do not converge to the global optimal solution but search for an adequate solution [44], [193]. In fact, they are often applied to combinatorial problems which can have very large solution spaces, such as EEG channel selection [44], [51], [105], [192]. In this work, a GA was used to solve a maximization problem in Chapter 5, but they can also be applied to minimization

problems [193]. In GAs, a candidate solution is represented by a group of values stored in a vector [193]. In GA nomenclature, the vectors are known as chromosomes or individuals, and the values in the vector are known as genes [193].

Figure 3.11 shows a generic flowchart for a GA, which will be discussed in depth throughout the rest of this sub-section. The first step, which occurs once at

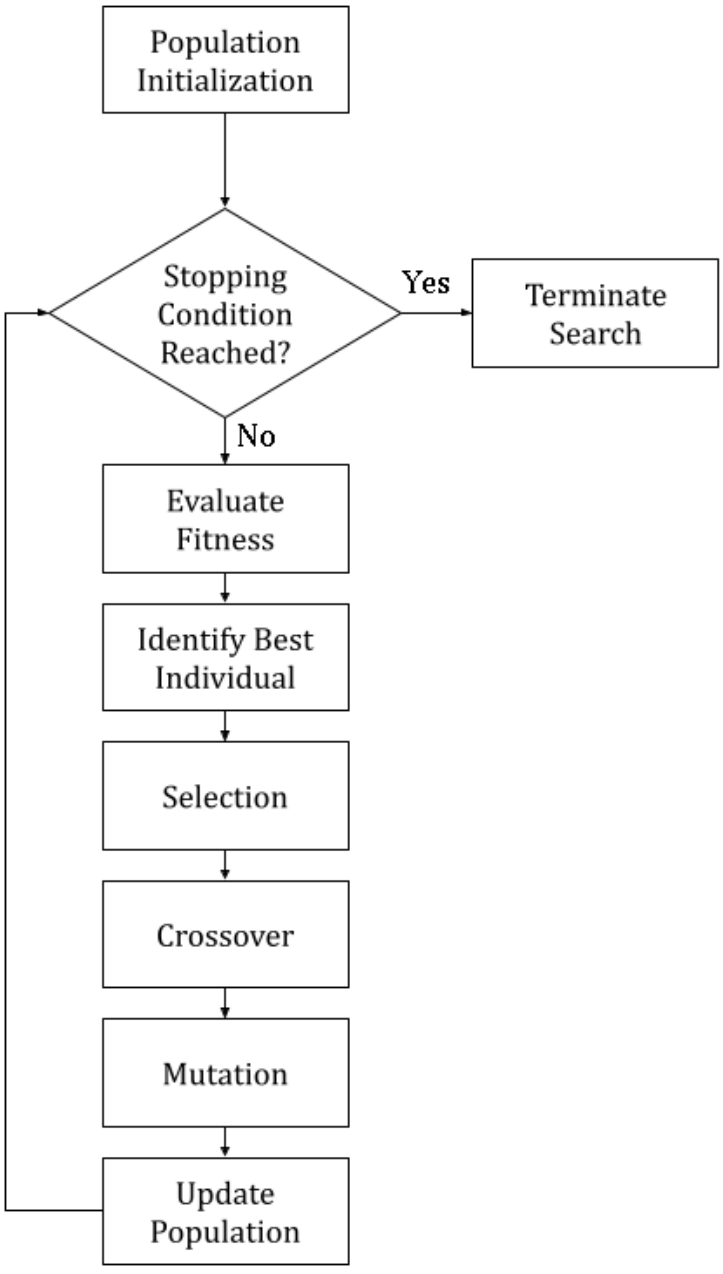

Figure 3.11: A flowchart showing the operation of a genetic algorithm at high-level.

the start of the algorithm, is population initialization [193]. It involves randomly generating a collection of chromosomes which represent candidate solutions. The size of the population (i.e., the number of candidate solutions in the population), is a hyperparameter that can be tuned.

After initialization, the algorithm then enters an iterative loop which searches the solution space. During each iteration, the population is updated, leading to a new generation of the population [193].

In this loop, the fitness of each chromosome in the population is first calculated [193]. The fitness is the metric which is used to compare the candidate solutions, and depends on the application of the GA. In this thesis, a GA is applied to an EEG channel selection problem, where the candidate solutions were subsets of EEG channels, and the fitness is the classification accuracy obtained using each subset. The fitness results are then used to identify the best individual in the population, which is the individual with the greatest fitness [193]. The algorithm keeps a memory of the current best individual and its corresponding fitness, and this is updated each iteration.

The GA searches the solution space using exploitation and exploration techniques [193]. Exploitation means the algorithm favors characteristics of the fittest individuals being promoted to subsequent generations, whereas exploration involves producing random mutations in the population to prevent the search becoming trapped within local solution spaces.

The exploitation aspect is executed through selection and crossover [193]. In selection, several individuals are randomly selected from the population to create the new generation of the population [193]. Usually, a bias is introduced into the selection process, such that fitter individuals are selected [193]. This is an allegory for mate selection during reproduction within a biological population, where healthier individuals tend to be selected as mates. The selected individuals are known as parents, and the number of parents is a tunable parameter. A common selection strategy is the fitness proportionate selection strategy (also known as roulette wheel selection) [193], demonstrated graphically in Figure

Table of Fitnesses

Chromosome Name	*Chromosome 1*	*Chromosome 2*	*Chromosome 3*	*Chromosome 4*
Fitness	82	96	71	55

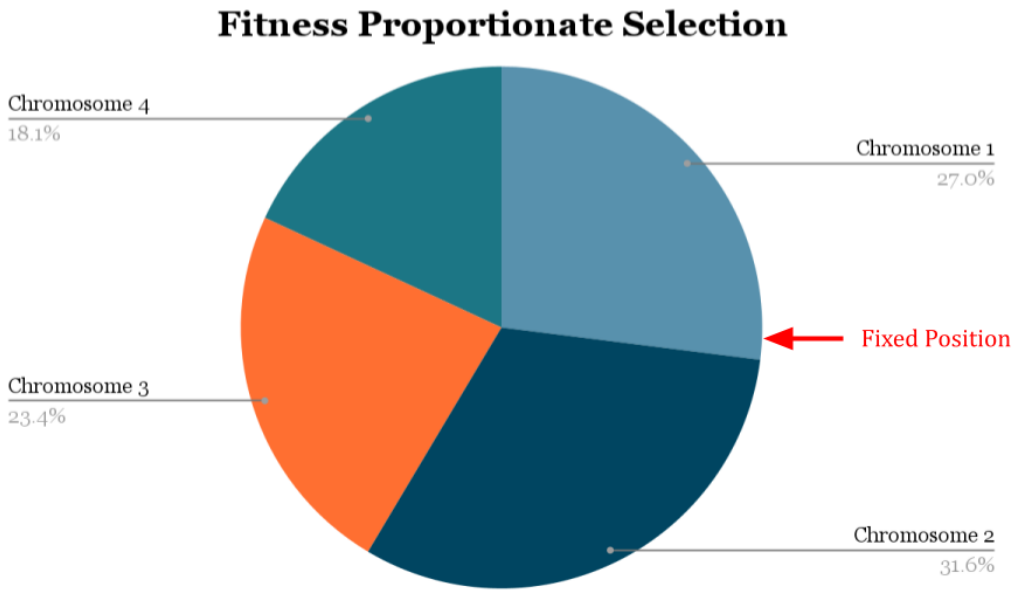

Figure 3.12: Fitness proportionate selection, also known as roulette wheel selection.

3.12. In this example, there are 4 chromosomes in the population, and the table at the top of Figure 3.12 shows the corresponding fitness of each chromosome. The fitness has no units to keep the example generic, since the units depend on the problem. The fitness table is then translated to the roulette wheel at the bottom of the figure, with the percentage of the wheel covered by each chromosome depending on its fitness. For example, chromosome 2 had the greatest fitness of 96, and this translated to it occupying the greatest area (31.6%) on the roulette wheel. The roulette wheel is then 'spun', and once the wheel stops spinning, the red Fixed Position arrow is used to identify the individual selected. This is then repeated multiple times to select the parents. Note that the same individual can be selected multiple times to be a parent.

The area occupied corresponds to the probability, p, that the individual, k, is selected. Given that the fitness of k is $Fitness_k$, p can be calculated as:

$$p = \frac{Fitness_k}{\sum_{j=0}^{z-1} Fitness_j} \times 100\% \quad (3.17)$$

During crossover, the parents are used to generate new individuals, known as children. In this thesis, the two-point crossover method [193], illustrated in Figure 3.13, is the basis of the crossover strategy used. Two parent vectors, Parent 1 (blue) and Parent 2 (green) are selected from the population. Two points within the vectors are randomly selected and these are known as the crossover points, denoted by dotted lines in Figure 3.13. Crossover is then carried out to produce the children and involves swapping the portions of Parent 1 and Parent 2 which are found within the bounds of the crossover points, thus producing Child 1 and Child 2. Child 1 and Child 2 replace Parent 1 and Parent 2 in the next generation of the population.

The next generation can be comprised completely the children produced by selection and crossover [193]. However, in some manifestations of GAs, the

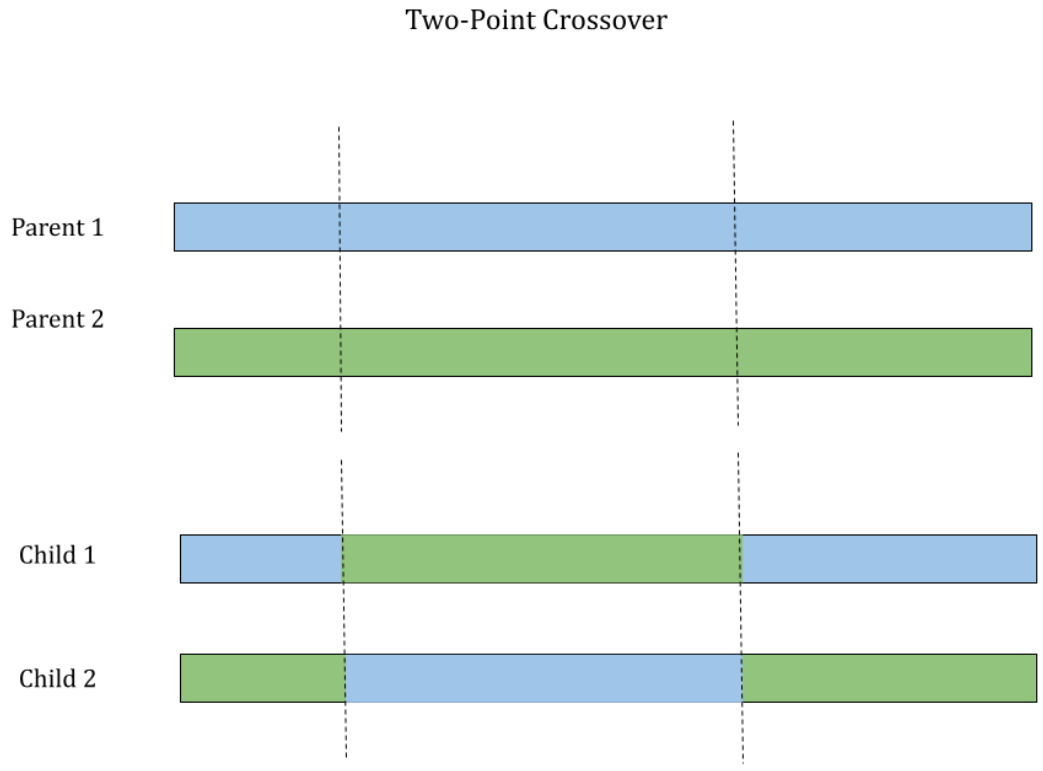

Figure 3.13: An illustration of two-point crossover being carried out on two parents to produce two children. The dotted lines denote the crossover points.

fittest *n* individuals can be automatically promoted to the next generation, and then (*n*-1) individuals are generated during crossover [193].

Once the next generation has been created through selection and crossover, mutation is carried out on the population [193]. This is the exploration step and involves introducing random changes to individuals within the population. Typically, mutations can occur with a certain percentage within the population (for example, there is a 20% chance of mutation happening within an individual in a population) or they can occur at a gene level, with each gene in the population possibly having a mutation introduced, usually at a low mutation rate, such as 0.01%. After mutation, the fitness of the new generation of individuals is calculated.

The GA can exit the iterative search if the stopping criteria are reached [193]. The stopping criteria could consist of a maximum number of iterations, or stagnation within the algorithm, which occurs when the best fitness has remained the same for a predefined number of iterations. Once the GA exits the iterative search, the current best individual is the final solution [193].

3.5.2.3 CNN-Based Channel Selection

Deep learning has brought to light a new area of channel selection research. In 2019 Mzurikwao et al. [23] developed a channel selection method based on the analysis of the weights in a CNN. First the CNN was trained using all the channels available in the EEG dataset, then the weights associated with each channel were summed and sorted. The 20 channels with the largest weights were selected, and the network retrained. In a small analysis of four subjects, they found accuracy tended to decrease by only 1% when reducing the number of input channels from 64 to 20. The benefit of channel selection was improved computational time, and the possibility of fewer electrodes being used in testing, leading to a more comfortable user experience.

In 2021 Zhang et al. [24] incorporated a novel automatic channel selection (ACS) layer as the first layer of their CNN network. The EEG time series

on each channel is first converted to a TFD image using the CWT. These images are then input to the ACS layer, which consists of a squeezing-excitation module. The squeezing phase compresses the information in the scalograms along the frequency and time dimensions to obtain a single description for each channel. The excitation phase learns salient relationships between channels. Finally, the input CWT images are reconstructed at the output of the ACS layer, with a sparsity constraint reducing the number of channels input to the subsequent CNN layers. Using the ACS module led to an improvement of 2.7% when compared to using just the CNN, indicating that their implementation was effective in suppressing the effect of the noisy or redundant channels. However, the implementation by Zhang et al. [24] does not involve a retraining step using the selected channel subset, implying that the whole cohort of electrodes were used for training and testing, with the ACS module suppressing redundant channels. In the wider CNN literature, discussed previously in Section 3.4, many state-of-the-art deep learning classification systems do not include automatic channel selection [7], [9], [64], [65], [102].

In Chapter 6 a novel approach to channel selection in CNNs is presented. It involves a custom layer which is inserted at the start of a CNN network. After training, an analysis is carried out on the weights in the ICS layer, and an importance score is assigned to each EEG channel. The *n* channels with the highest scores are then selected, where *n* is the number of channels in the subset, which is set by the user. The channel selection approaches of Mzurikwao et al. [23] and Zhang et al. [24] were each only tested on one CNN architecture, and the channel selection approaches were integrated into the designs of these CNN networks. The novelty of the ICS layer approach is in its versatility: it can, in theory, be easily applied to any 2D CNN network which takes EEG time-series as input. In Chapter 6, the approach was found to be effective for two different CNN networks when tested using two different datasets. This is the first time, to the authors knowledge, that such a versatile but straightforward approach has been presented for MI EEG channel selection within a CNN.

3.5.2.4 Subject-Independent Channel Selection

Channel selection is often subject-specific [13], [23], [46]–[49]. This means that the training data of the individual subject is used to obtain a subset of EEG channels to be used on the test data. This can lead to improved computational times on the test set, and in a real-time system could provide a more comfortable experience for the subject since some electrodes can be removed. However, the channel selection process can be time-consuming, particularly for wrapper channel selection techniques. In subject-independent channel selection, data from other subjects is used for channel selection, then the selected channel subset is used directly on a target subject. Although some authors have observed that subject-specific training of BCIs leads to more reliable performance [3], other authors have shown that subject-independent channel selection holds promise [52].

Some of the motivations for subject-independent EEG channel selection are as follows:

- EEG electrodes are an added cost to commercial BCIs: Ideally, commercial systems are sold with the minimum number of EEG electrodes possible that still provide an acceptable performance. Having a subset of electrodes that are known to work well for a wide variety of users would increase commercial viability.
- Removing the need for end-user channel selection: Subject-specific channel selection is time-consuming in and of itself and contributes to an overall increased training time. This is investigated in Chapter 5. If the best subset of EEG electrodes could be selected beforehand through a subject-independent EEG channel selection process without any negative impact on the classification performance with the target subject, this would remove the need for time-consuming subject-specific channel selection.
- Improved or more stable performance: Some electrodes may hold redundant or noisy data that either contributes little to the classification process, or even affects it adversely [13]. By removing channels that are redundant across subjects, the classification performance could become

more stable. Furthermore, reducing the number of EEG channels could reduce the training times for the end-user.

- Better look and feel: EEG recording systems with many electrodes may appear cumbersome and aesthetically unappealing to users. Systems with more electrodes may also be less comfortable. Thus, an added benefit of reducing the number of EEG electrodes may be enhanced commercial value based on comfort and aesthetics.

Despite these motivations, many studies still favor subject-specific channel selection [13], [23], [46]–[49]. Handiru et al. [52] showed that using a subject-independent channel subset resulted in a decrease in average classification accuracy of only 2.8% when compared to using a subject-specific channel selection, an observation they used to support subject-specific channel selection. Even recent works into CNN-based channel selection do not present results for subject-independent channel selection [23], [24]. Studies into cross-subject DL classification discussed previously in Section 3.4 also do not delve into the subject of subject-independent channel selection [7], [58]–[60].

In Chapter 6, a novel analysis assessing the capabilities of the ICS layer channel selection method within the context of subject-independent channel selection is presented. Subject-independent channel selection with the ICS layer did not result in a statistically significant decrease in classification performance when compared to selecting subject-specific channels, indicating that it is an effective method for subject-independent channel selection. The benefits of using subject-independent channel selection on training latency for the user are also summarized.

3.6 Windowing Techniques in EEG Classification

EEG time-series data is typically segmented, with features being extracted from the segment and input to the classifier. Approaches for segmenting EEG time-series data for classification vary across studies. Some studies consider whole trials of EEG data, the length of which depends on the dataset, although trials 3.5s and 4s long are common [22], [194]. Other studies pass a sliding window over the

EEG data, creating multiple segments for each EEG trial [5]. This technique can be used with different aims in mind, either as a 'cutting and splicing' technique to increase the training data for a DL system [8], [9], [58] or to mimic within the constraints of an offline BCI the buffering system that might exist within a real-time BCI, where segments of EEG data are classified in almost real-time [19], [25]. The sliding window technique is characterized by the window size (how many samples are in a segment) and the window increment (how many samples the window moves with each step). EEG data around 1.5s in length or less can be considered approximately stationary [91], [120], [195] so segmentation may be used for certain feature extraction techniques that can be adversely affected by non-linearities, such as CSP [120], [196].

Effective BCI classification would require refinement of the window size and window increment size. Some studies use maximally overlapped windows [8], [58], although many studies also use partial overlap, moving the window by increments of milliseconds [25], [26], [197], possibly to improve computational efficiency. Samuel et al. [25] used FFT features and an LDA classifier, and investigated window sizes in the range 100ms to 350ms, and window increment sizes in the range 25ms to 100ms. They obtained a peak classification accuracy of 99.97% when using a window size of 100ms and a window increment of 25ms. However, their study used private data so the results cannot be replicated. Using a simple CSP-LDA pipeline, Asensio-Cubero et al. [26], [135] segmented EEG trials using overlapping windows, and obtained a feature vector from each segment. They then investigated two classification approaches: i) obtaining a classification label for each segment and using majority voting to obtain the final label for the trial; and ii) concatenating the feature vectors from the segments, and then inputting that to the classifier to obtain the label for the trial. They found that (i) provided better classification accuracy, and that using a window size over 2s long gave better accuracy. The core difference in the windowing approaches in the works of Samuel et al. [25] and Asensio-Cubero et al. [135] is that in [25] the segmentation just augmented the dataset, mimicking a basic buffering system, whilst in [135] the information from different segments of a trial were joined to

obtain the final classification of the trial. In 2019 Yang et al. [32] optimized the time window for EEG segmentation in a conventional classification system, and found that the optimization improved classification performance by 5%-15%. However, they did not investigate fusing the decisions made on multiple windows. In 2017, Schirrmeister et al. [8] applied a post-processing technique similar to that used by Asensio Cubero et al. [135] to a CNN classifier. They found that taking the mean of the SoftMax output for different segments from a trial led to improved accuracy when compared to just classifying the whole trial. However, none of these studies have conducted an in-depth investigation into the impact of window size and window increment size for a wide variety of classifiers.

EMG signals are noisy biosignals recorded from muscle and, like EEG signals, are non-stationary and non-linear [28]. In a 2020 study, Wahid et al. [27] found that segmenting EMG signals and then merging classification results across temporal windows led to a significant improvement in classification results for a variety of classifiers. Although present studies into EEG classification indicate that using windowing can improve classification accuracy [8], [135], more investigation is needed to identify how effective windowing can be for different classification pipelines, and any relationship between performance, window size, window increment and classifier.

To address these gaps in the literature, Chapter 4 presents a multi-segment majority-voting decision fusion framework. This framework involves segmenting the EEG data in a trial, obtaining a classification label for each segment, and then performing majority voting on the segments to obtain the final classification label for the trial. The approach was applied to a variety of conventional classifiers, and the effects of window size and window increment size are discussed in depth. Using the multi-segment decision-fusion framework generally led to a significant improvement in classification performance when compared to just assigning a label using the whole EEG trial with no segmentation. The framework is assessed using an open-access dataset [71], such that results could be replicated, as opposed to the work by Samuel et al. [25],

which was carried out on a private dataset and was therefore not easily replicable.

3.7 Conclusion

This chapter provided a broad overview of the literature related to MI EEG classification, highlighting gaps in the literature that are explored in the contribution chapters of this thesis. It also provided a technical background into CSP feature extraction, conventional classifiers, OMP, CNN-based classification and GAs.

Although EEG data segmentation is common [8], [25], [26], [58], [197], the literature has not explored in-depth whether time domain-based decision fusion from multiple EEG segments can significantly improve classification accuracy. Although some studies have suggested potential improvements from TD decision-fusion [8], [26], these have been limited to experiments on just one kind of classifier, or have not explored in-depth the relationship between window design parameters (i.e. window size and window increment size) on classification performance. A study into EMG signals has indicated that a TD decision fusion approach can boost classification performance across different classifiers [27]. In Chapter 4, a majority-voting based multi-segment decision fusion framework is presented. In this approach, EEG trials are segmented in the time domain, and each segment is assigned a label by a classifier. Majority voting is carried out on all the labels assigned to segments to obtain the final label for the trial. The framework was assessed on multiple conventional classifiers, and the effect of window design parameters was investigated in-depth.

Conventional machine learning approaches remain a current topic of research [10], [13], [14], [21], [22], [30], [31], [104], [118]. Although many studies have compared different conventional classifiers [5], [101], [104], to the knowledge of this author, no study has compared the classification performance of a variety of classifiers (namely SVM, LDA, RF, NB and MLP) across different channel subsets. Since the channels included in a subset can have a substantial

impact on classification performance, this kind of analysis would provide a novel perspective for classifier comparisons. In Chapter 4, the performance of several conventional classifiers is compared when different channel subsets are used.

Discussions in the literature on static channel subsets for EEG classification have been focused on whether increasing or decreasing the number of EEG channels used improves performance [32], [33]. However, it is known that electrodes from different scalp regions can be broadly related to different mental processes. Chapter 4 presents a straightforward methodology for developing a static EEG channel subset based on an analysis of the contributions of electrodes from different scalp regions.

Dictionary-based SL approaches have performed with high classification accuracy in the literature [11], [39], [56], [57]. Many approaches have been based on frequency or time-frequency feature extraction techniques and extract multiple features per channel [11], [56], [57]. In Chapter 5, a dictionary-based SL approach based on TD band power is presented. This approach outperformed a state-of-the-art SL approach based on wavelet features [11]. Furthermore, an analysis showed that using just one feature per channel was adequate for high performance when using the band power in the combined alpha and beta frequency range as the feature.

Despite their strong performance, dictionary-based SL approaches are based on encoding algorithms that can be computationally expensive [11], [54], [55], [168]. In Chapter 5, the test-set computational time of a SL classifier using all the EEG channels was found to possibly be unacceptably long for a real-time application such as a graphical user interface. Subject-specific channel selection was applied to reduce the number of EEG channels and improve the computational time. A GA channel selection module was developed to select EEG channels using training and validation datasets. A metaheuristic channel selection method was used since these methods are renowned to provide a trade-off between accuracy and faster convergence when compared to other wrapper techniques [44]. A GA framework was chosen since it had been previously shown to work well for EEG channel selection [49], [192], actually improving the

accuracy in the work of He et al. [49]. Using a subset of channels produced by the GA maintained the classification accuracy and reduced the run-time computational times to be more compatible with a real time application. To the authors' knowledge this is the first time that metaheuristic channel selection has been applied to a dictionary-based SL classifier for MI EEG.

Whilst subject-specific channel selection is popular in the literature, there are many motivations for effective subject-independent channel selection, including a decreased training latency for the end-user and cheaper setups with fewer electrodes which makes them more commercializable. One impediment to subject-independent channel selection is that it can result in deteriorated classification performance when compared to using subject-specific channels [52].

Although CNNs have been effective for MI EEG classification [6]–[8], [60], [65], [102] and research into cross-subject classification has been heavily focused on CNNs [7], [58]–[60], they have not been applied to the subject-independent channel selection problem for MI EEG. Furthermore, recent methods of CNN-based channel selection have been tested only on specific CNN pipelines, and in the case of the work of Mzurikwao et al. [23], the channel selection approach is intrinsically tied to the structure of the CNN used. In Chapter 6, a novel CNN-based channel selection method for subject-independent channel subsets is presented, called the ICS layer method. This method consists of an integrated channel selection layer that can, in theory, be easily added to any 2D CNN network that processes segments of EEG time-series data. The versatility of the method is illustrated by applying it to two different CNN pipelines and testing it on two datasets. Furthermore, there was no statistically significant difference in performance when using the ICS method to extract subject-specific or subject-independent channel subsets, indicating that this method is suitable for subject-independent channel selection.

Transfer learning involves using source data from other subjects to improve the classification performance on a target subject. It is widely applied in DL classifiers for MI EEG classification problems [12], [58], [60], [65], [138],

[172], [174]–[176], however no study has compared the performance of the RTL and MTL approaches. In Chapter 6, transfer learning is used to improve the performance of the CNN classifiers when using a reduced channel subset, and the RTL and MTL methods are compared in terms of classification performance and computational times. This experiment is not only novel because of the comparison between the two techniques, but because of the application of transfer learning to improve the classification performance after CNN-based channel selection, which other studies in the literature have not done.

The next chapter, Chapter 4, presents work in conventional classifiers. The core contribution of this chapter is the multi-segment decision fusion framework. The chapter also presents a comparison of various classifiers based on different channel subsets, and a straightforward approach to selecting a static channel subset using an analysis of the contributions of electrodes from different scalp regions.

Chapter 4 : Multi-Segment Fusion for MI EEG Classification with Static Channel Analysis

4.1 Introduction

As previously discussed in Chapter 3 (Section 3.6) the segmentation of EEG trial data is common in the literature, for reasons of dataset augmentation [25] or for decision fusion [8], [26]. This segmentation also mimics the buffering that can happen in a real-time BCI [19], [25]. A recent study into the classification of EMG signals found that using majority-based decision fusion on segmented time series could lead to a significant improvement in performance [27]. However, to the best of the author's knowledge following extensive literature review, no study has investigated the relationship between segmentation approaches and classification accuracy in the context of various classifiers.

The main contribution of this chapter is a novel investigation into the effect of a majority voting-based multi-segment decision fusion classification approach. This approach segments each EEG trial, labels each segment, and then uses majority voting to obtain the final classification label for a trial. The investigation analyses the effect of different window sizes and increment sizes for segmentation. It also analyses the effect of multi-segment decision fusion on the accuracy of LDA, SVM, NB, RF and MLP classifier. These classifiers were chosen because they have been widely used in the literature [10], [13], [14], [21], [31], [104], [133], [140], [147], [148], [188], and cover a variety of different approaches to constructing decision boundaries, as previously discussed in Chapter 3. This study also summarises the impact windowing can have on execution time. A paper related to multi-segment decision fusion was published in *Cognitive Computation* [198].

The secondary contribution of this chapter is a novel and straightforward methodology for building a static EEG channel subset. This subset was then used in the multi-segment decision fusion classifier. As previously discussed in Chapter 3 (Section 3.5.1), it is common for studies in the literature to use arbitrarily chosen subsets of EEG channels for classification [10], [21], [29], [30], although there is a lack of consensus as to whether decreasing the number of channels leads to a degradation in classification performance or not [32], [33]. The static subsets chosen vary across studies [10], [21], [29], [30], and it is relatively rare for a study to present results for multiple arbitrarily chosen channel subsets [30]. In this work, the static channel subsets are constructed by considering the EEG scalp regions surrounding the central region, which is most associated with motor activity [34]. Although EEG scalp potentials at any one point are produced from mixing of signals from different brain regions [70], there is a relationship between the scalp region and the underlying cerebral functions [34]–[37]. This method for constructing the channel subsets aims to analyse the effect of electrodes from different scalp regions on classification performance.

The analysis of static channel subsets as well as the full cohort of channels in the dataset was conducted for the five different classifiers mentioned previously. Although studies comparing different classifiers are common in the literature [5], [31], [101], as previously discussed in Chapter 3 (Section 3.2) these comparisons are rarely carried out when considering different channel subsets. Thus, the novelty of the approach proposed in this chapter lies in the analysis and comparison of the results from various static channel subsets and the impact that multi-segment decision fusion can have on classification performance.

The rest of this chapter is organized as follows. Section 4.2 describes the proposed static channel subset analysis and the multi-segment fusion approach. Then, Section 4.3 describes the datasets used, hyperparameter tuning, performance metrics and statistical analysis. The chapter then continues with Section 4.4 where the results are presented and discussed before conclusions are drawn in Section 4.5 and the main contributions of this chapter are reiterated.

4.2 Scalp Region-Based Static Channel Analysis and Implementation of the Majority Voting-Based Multi-Segment Decision Fusion Approach

This section covers three areas: the static EEG channel analysis, pre-processing and feature extraction, and the proposed multi-segment fusion classification approach. For both the static channel analysis and the multi-segment classification fusion, five different classifiers were used for analysis, namely: LDA, SVM, RF, NB, and MLP classifiers. SVM classifiers with linear, polynomial and RBF kernels were considered, and are labelled as SVM-Linear, SVM-poly and SVM-RBF, respectively.

4.2.1 Proposed Static EEG Channel Analysis

Figure 4.1 shows a map of electrodes used in the EEG recording montage. The electrode map shows the channels that are available in the extended 10-20 electrode setup [199], and includes the regions most of interest in this chapter. In the original dataset, a special high-density cap of 118 electrodes was used. This cap contains electrodes that are in the extended 10-20 system as well as additional electrodes to make a high-density covering of the scalp. These caps are generally expensive and impractical [199], thus the experiments in this chapter are mostly focused on electrode regions that are in the extended 10-20 montage. The full montage of 118 channels was used just for baseline comparisons. The channels involved in this static analysis are highlighted in red. Four channel subsets are considered in this analysis, alongside the use of all 118 channels in the dataset. The four subsets are:

1. Central Channels (C): C5, C3, C1, Cz, C2, C4, C6, 7 channels in total;
2. Central and Central-Parietal channels (C+CP): C5, C3, C1, Cz, C2, C4, C6, CP5, CP3, CP1, CPz, CP2, CP4, CP6, 14 channels in total;
3. Central and Central-Frontal channels (C+CF): C5, C3, C1, Cz, C2, C4, C6, FC5, FC3, FC1, FCz, FC2, FC4, FC6, 14 channels in total;

4. Central, Central-Parietal and Central-Frontal (C+CP+CF): C5, C3, C1, Cz, C2, C4, C6, CP5, CP3, CP1, CPz, CP2, CP4, CP6, FC5, FC3, FC1, FCz, FC2, FC4, FC6, 21 channels in total.

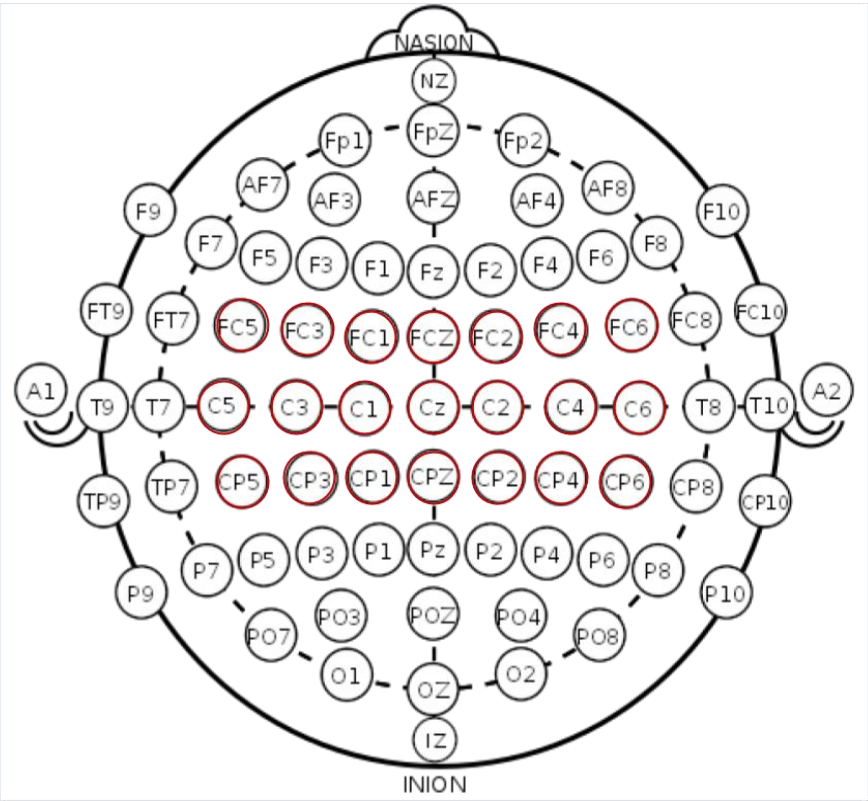

Figure 4.1: A map of electrodes used in the EEG recording montage, with the channels which were used in the static subsets highlighted in red.

Electrodes in the CP and CF regions were considered in the analysis because they border the central scalp region, which is associated with MI activity. Although the central region has been linked to MI, due to volume conduction in the scalp (previously discussed in Section 2.1.1.1), signals from one scalp region can mix with signals in neighboring regions [70], meaning salient information could be obtained from the CP and/or CF regions.

These channel subsets were designed to combine different scalp regions which are associated with particular mental processes. All the subsets contain the C group, which has been fundamentally linked to the sensorimotor region of the brain and to MI [34]. The CP region is at the overlap between the central and

parietal region, which is associated with motivated attention and concentration [34], which are both important mental components of completing MI tasks. The CF region is at the overlap of the central and frontal region, which is associated with planning of voluntary movement [35]. The aim of this brief static analysis is to study the contributions of the CP and CF regions individually, as well as their synergistic contribution when included together in a subset. This analysis is important because, although many studies use static channel subsets [10], [21], [29], [30], the contribution of smaller groups of channels in the same scalp regions bordering the central region has not been studied. This study presents a new and straightforward methodology which could be used in studies which opt for a static EEG subset. Although some papers present results using different static subsets [30], this methodology does not just arbitrarily select the EEG channels, but factors-in the broad relationship between scalp location and underlying mental processes.

In this analysis, features are extracted from the whole trials and passed onto the classifier for classification. For simplicity, the two best static EEG subsets obtained from this analysis are then used to study the multi-segment fusion classification approach. The 'best' static subset is the one that obtained the best classification accuracy in the static analysis.

4.2.2 Pre-Processing and Feature Extraction

EEG data for each trail is pre-processed and then common spatial pattern (CSP) features are extracted for classification. CSP-based features were chosen because of their wide use in benchmarking [5], [22], [29], [163], [197] and in investigations into novel classification [21], [131], [133], [134], [153] or channel selection [48] pipelines, which were previously discussed in Chapter 3 (Section 3.1.5).

Pre-processing involves filtering and mean-centering the EEG channels in the static subset. Filtering is carried out using an elliptic bandpass filter [41], [135] with a passband from 8Hz to 32Hz, which covers the alpha and beta frequency bands important for MI [34]. Afterwards, data is mean-centered by

taking the mean of the data in the trial, and subtracting it from each of the EEG channels [10]. This is a type of EEG re-referencing which aims to reduce the effect of artifacts and noise [200].

After pre-processing, a standard CSP feature vector [10], [139] is extracted from each trial. The technical aspects of CSP feature extraction were discussed previously in Chapter 3 (Section 3.1.5). Based on the results obtained by Olias et al. [10] p, the dimensionality of the subspace, is set to 8. The approach in [10] was followed since the CSP feature extraction approach in this chapter was based on the implementation used in [10] and the same dataset is also used.

4.2.3 Proposed Multi-Segment Decision Fusion Classification

The proposed multi-segment fusion classification approach is shown in Figure 4.2. In the proposed approach, each EEG trial is divided into multiple segments using an overlapping, moving window approach. A CSP feature vector is obtained from each segment, and a classification label is obtained from each feature vector. The classification label is obtained from a classifier. Majority voting based on the classification labels obtained from the segments is used to obtain the final classification label for the trial. Note that in the case of the dataset used in this chapter, each EEG trial is 3.5s long [86]. More information on the dataset used can be found in Section 4.3.1.

The moving window approach consists of two design features: the window length, xs, which controls how many time samples are included in the window, and the window increment, ys, which controls the overlap between the present data window and the following one. To clarify, the first window would extract data from time 0s to time xs, whilst the second window would extract time samples from ys to xs + ys, and so on until the leading edge of the window reaches the end of the trial. It follows that the number of segments obtained per trial depends on the values of both x and y. The effect of different window sizes

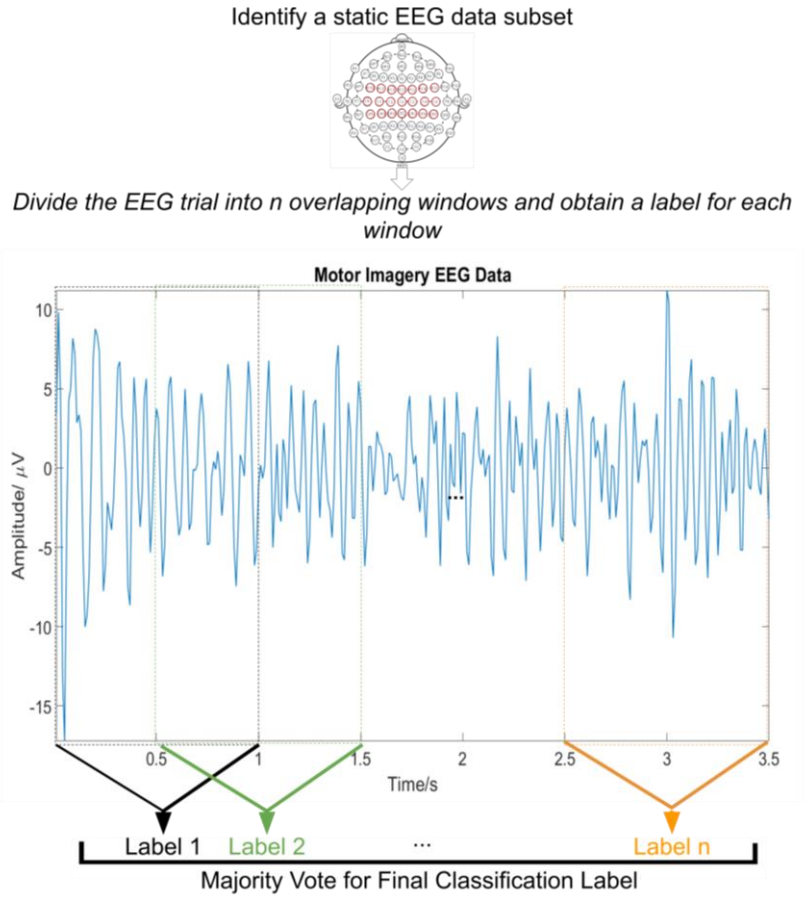

Figure 4.2: The multi-segment fusion classification approach, showing segmentation of the EEG trial and majority voting label assignment. The labels used in majority voting are obtained from a classifier.

Table 4.1: A table of the different window sizes and window increments that were used in this study.

Window Size	Window Increment
2s, 1.75s, 1.5s, 1.25s, 1s, 0.75s, 0.5s	0.5s, 0.25s, 0.1s
0.25s	0.25s, 0.1s

and window increments is studied, with the values documented in Table 4.1. All the possible combinations of window size and increment in each row are considered. Note that for the 0.25s window, increments of 0.25s and 0.1s only are used since an increment size of 0.5s would be larger than the size of the window, resulting in certain trial samples not being processed. Windowing schemes are described in short as (xs, ys), for example (1.25s,0.5s) for a window size of 1.25s and increment of 0.5s.

Training and testing using the proposed multi-segment fusion approach first involves dividing trials into training and test sets, then segmenting them using the moving window. Feature vectors from the segmented training trials are then used to train the classifier, and feature vectors from the test set are used for multi-segment fusion classification, with labels being obtained for each of the segments in a trial and majority voting being used to obtain the final label for each trial. *studies investigating the effect of segment-based majority voting for trial classification have only used one classifier, LDA*

Figure 4.3 and Figure 4.4 show an illustrative example of how majority voting can work with an actual segment of EEG data. Both images show the same 3.5s MI EEG being classified, but Figure 4.3 shows the traditional method of classifying the whole trial, whilst Figure 4.4 shows the multi-segment decision fusion approach applied to the same trial. In both Figures, the EEG time series is made up of the central (C) EEG channels. Class 1 corresponds to an imagined right-hand movement whilst class 2 corresponds to imagined right-foot movement. Figure 4.3 shows an example of a whole trial from MI class 1 being incorrectly classified: features are extracted from the whole 3.5s of the trial and are input to the classifier, which assigns the class label 2 to the trial. Figure 4.4 (overleaf) shows the same trial segmented using a 2s window size and a 0.25s increment size, leading to 7 segments. A feature vector is extracted from each of

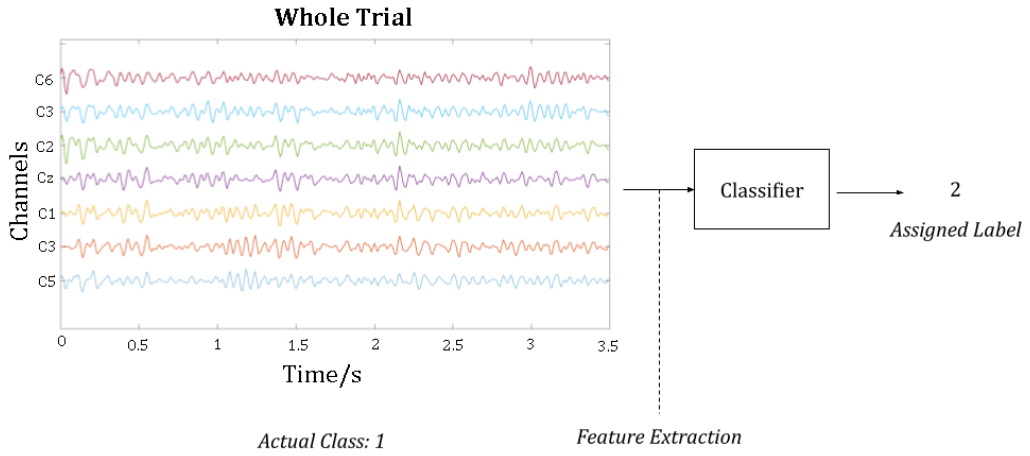

Figure 4.3: An example of a whole trial classification leading to misclassification.

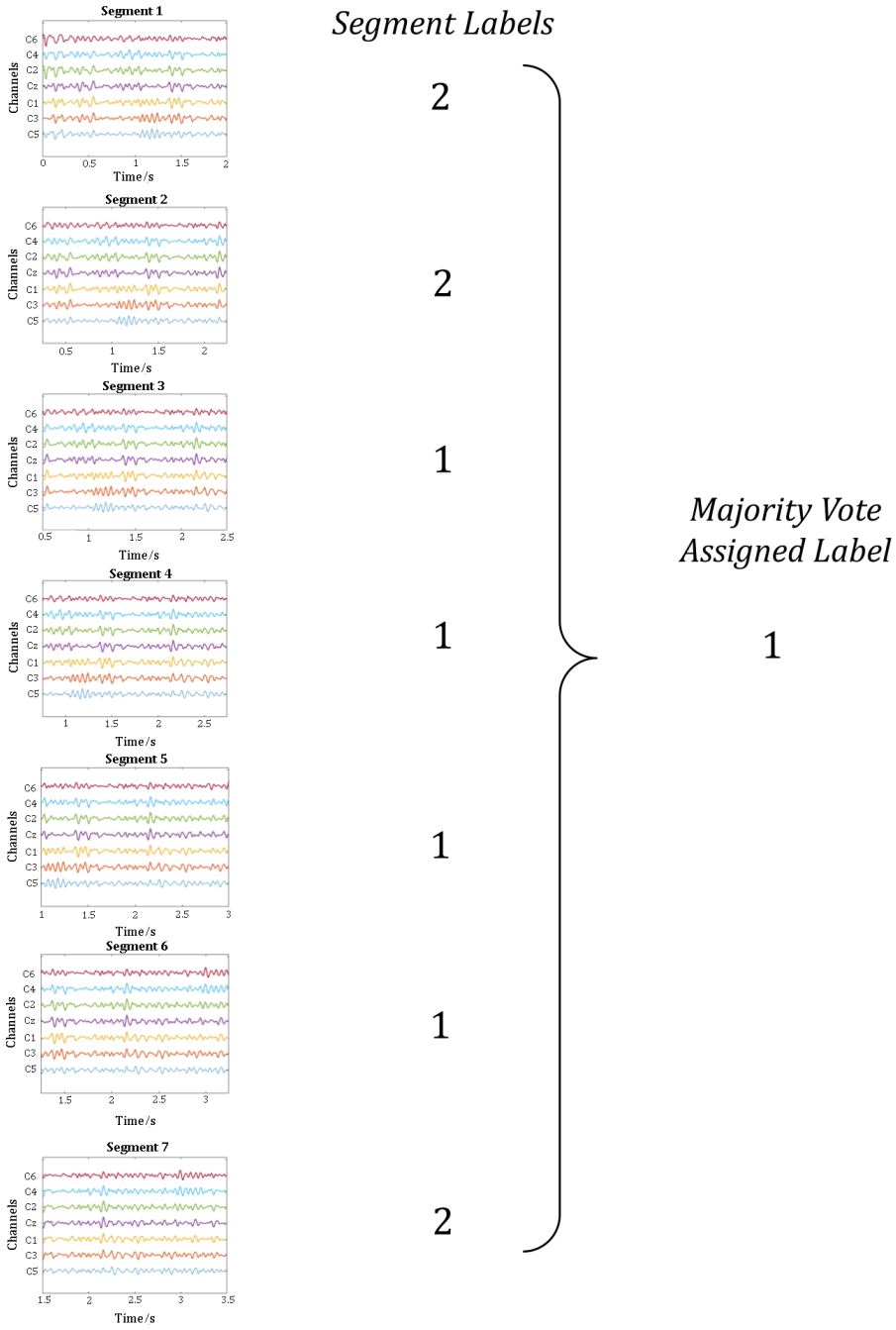

Figure 4.4: An example of segmentation and majority-voting based classification for a trial. The 3.5s long EEG trial has been segmented into seven segments using a 2s long window and a 0.25s window increment.

the segments, and the feature vectors are classified to obtain an assigned label for each individual segment. For simplicity, the feature extraction step and classifier have not been shown graphically in the image. Note that three of the segments were classified as class 2, whereas four were classified as class 1. Thus, when a

majority vote is then conducted over the assigned labels, the final classification label assigned to the trial is 1, which is the correct class.

Although many studies segment EEG data, either to augment the dataset, to perform decision fusion [135], to mimic the buffering in a real-time classification system and/or dataset augmentation [19], [25], [59], studies investigating the effect of segment-based majority voting for trial classification have only used one classifier, LDA [26], [135]. The aim of this study is to fill this conceptual gap in the literature. The novelty in this chapter is in investigating the effect of majority voting-based decision fusion across various classifiers, namely LDA, SVM, NB, RF and MLP classifiers. In particular, the analsysis is focused on the effect of window size and window increment size on performance across these five different classifiers.

4.3 Experimental Methodology

The experiments in this section use subject-specific training, meaning that the classifiers were trained using training trials from each subject and tested using a different, "unseen" set of testing trials from that subject. This is a widespread approach in the literature [3], [11], [12], [173], particularly for CSP-based classification systems [5], [21], [22], which have not performed well in cross-subject classification tasks [133], [197].

4.3.1 Dataset

Experiments were carried out on the BCI Competition III Dataset IVa [86]. This dataset was described in more depth in Section 2.2.1. The training trials are used for hyperparameter tuning and classifier training. The testing trials are used as 'unseen data' for obtaining the results for the cross-classifier analysis and multi-segment classification fusion approach. All the training data was used for hyperparameter tuning. However, since EEG data is artifact-prone and non-stationary, a 10-fold evaluation approach was used, in which the training data of each subject was divided into ten segments, and performance was recorded when training on nine segments of the data. This was repeated for ten times, each time

with a different segment 'left out'. The average results over the ten folds were recorded as the performance for the subject. This general approach has been used in other EEG studies [120], [201], [202].

Only one dataset is used in this chapter, whereas in Chapter 5 and Chapter 6, two datasets are used. To assess the generalizability of a proposed approach, it is good practice to test on additional datasets. However, the work presented in this chapter was the first completed during the course of the PhD research, when the evaluation methodology was still being refined. Furthermore, knowledge about other datasets available was still being built, and many of the studies reviewed at that point had focused on one dataset [104], [106], the BCI Competition III dataset IVa, which is used in this chapter.

4.3.2 Hyperparameter Tuning

Grid-searches were used to tune the hyperparameters of the classifiers. During a search, for each combination of hyperparameters, the 10-fold cross-validation accuracy was evaluated for each subject in the dataset. This involved partitioning the training data into 10 folds and using 9 folds for training and 1 to obtain the validation accuracy. This is repeated 9 more times, using a different fold for validation each time. The average validation accuracy across the folds is then obtained. To select final parameters for each classifier that are generalizable, the parameters which resulted in the highest average classification accuracy across subjects were used. The relevance of each parameter was previously discussed in Chapter 3. The following parameter values were considered in the grid-search:

- LDA classifier: Linear coefficient threshold (Δ) values of 2^n, for values of n from -10 to 10, increasing in steps of two. The value zero was also included in the set. Regularization coefficient (Υ) values of 2^n for values of n from -10 to -1 in increments of two. The values 0, -0.5 and -0.25 for n were also included.
- SVM classifiers: For linear, RBF and polynomial SVMs, the search for the regularization parameter (C) and the kernel scale (g) were in the set 2^n for n

in the range -10 to 10 in increments of two. For the SVM-poly classifier, polynomial factors of 2 and 3 were considered.

- NB classifier: Four different kernel functions were considered: Gaussian, Epanechnikov, box and triangular. Kernel widths of 10^n for n in the range -1 to 1, increasing in increments of $\frac{1}{50}$, were considered.
- RF classifier: The number of trees was increased from 20 to 1000 in increments of 20, the predictions at each node were increased from 1 to 8 in increments of two and the observations per leaf were increased from 4 to 20 in increments of four.
- MLP: The number of neurons in the hidden layer (HN) were from the set {3, 5, 10, 20, 100}, maximum training iterations in the set {200, 400, 800}, the momentum term (β) was from the set {0.6, 0.7, 0.8, 0.9}, the learning rate (η) was from the set $\{10^{-2}, 10^{-3,} 10^{-4}\}$ and the regularization coefficient (α) values were from the set $\{10^{-2}, 10^{-3,} 10^{-4}\}$. The activation functions tanh, relu and logistic were considered.

In this chapter a grid search was used to tune the hyperparameters because it is a standard, reliable approach [203]. However, in Chapter 5, Bayesian optimization [90] was used to tune the classifier hyperparameters. This is because further literature review indicated that Bayesian optimization has the potential to produce similar results to grid searches with lower computational expenditure [204].

All the classifiers were implemented using standard functions in MATLAB, except for the MLP classifier which was implemented in Python 3. The classifiers were tuned during the static EEG analysis, with each classifier being assigned a set of tuned parameters for each of the subsets. The 10-fold cross validation grid-search was carried out for each subject independently. However, to decide on the final parameters, the cross-validation accuracy, averaged across the subjects, was used. The parameters which gave the highest accuracy were used. This meant that individual subjects may have had better results for other hyperparameters, but the overall optimal classifier for each channel subset was selected. These

hyperparameters were used for both the static EEG analysis and the multi-segment fusion classification assessment, so that the effects of multi-segment fusion could be analysed with all other factors controlled. The aim of selecting the best overall parameters was to obtain universal classifiers on which to assess performance. This is a similar concept to finding the best channel subset across subjects, which is also explored in this chapter.

Figure 4.5 shows the hyperparameter tuning results for the different classifiers when 118 channels are used. The axes of the plots denote the parameter values in the grid-search, and the colours of the data points denote the average accuracy value. The red data points denote the data point with the greatest accuracy. The parameter values associated with these data points were the values used for further analysis. Similar results were obtained for the C, C+CP, C+CF and C+CP+CF channel subsets. The hyperparameter results for the MLP classifier are not shown since six hyperparameters were tuned, requiring a 6D plot for visualization. Table 4.2 shows the selected parameter values obtained for each channel set and classifier pairing, and the associated grid-search accuracy.

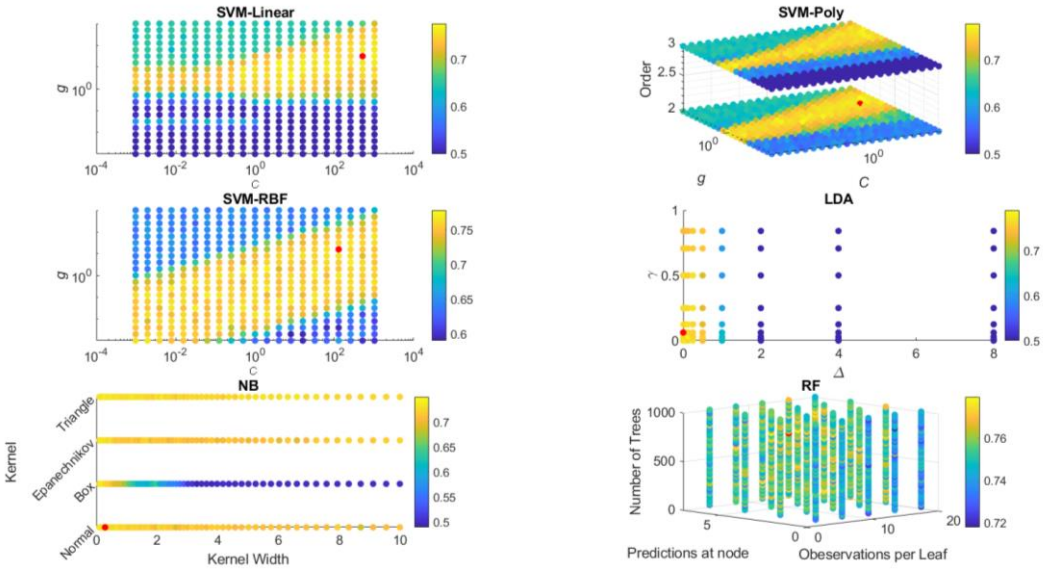

Figure 4.5: Hyperparameter tuning of the classifiers. The axes of the plots denote the parameter values, and the colours of the data points are related to the accuracy according to the colour bar on the right-hand side of each plot. The red data points have the highest accuracies and are associated with the parameters chosen for further analysis.

Table 4.2: The chosen hyperparameters chosen for each classifier and channel subset pairing. The grid-search accuracies obtained for each pairing are also shown.

Classifier	C	C+CP	C+CF	C+CP+CF	118
SVM-Linear	C =0.0039; g =0.0313 (76.29%)	C=0.25; g = 0.0078 (82.00%)	C = 0.0078; g = 0.0313 (79.43%)	C = 128; g = 1 (82.71%)	C = 128; g = 16 (77.79%)
SVM-Poly	C = 16; g = 2; Order = 2 (76.14%)	C = 128 g = 3; Order = 3 (81.93%)	C = 256; g = 16; Order = 2 (79.64%)	C = 8; g = 2; Order =2 (82.00%)	C = 256; g = 16; Order = 2 (77.71%)
SVM-RBF	C = 2; g = 4 (75.86%)	C = 512; g = 32 (81.86%)	C = 32; g = 4 (79.43%)	C = 32; g = 16 (81.57%)	C = 512; g = 32 (77.50%)
LDA	Δ = 0.0313 ϒ = 0.002 (76.07%)	Δ = 0.0078 ϒ = 0.0625 (82.07%)	Δ = 0.002 ϒ = 0.0078 (79.22%)	Δ = 0.0313 ϒ = 0.25 (81.93%)	Δ = 0.0156 ϒ = 0 (82.07%)
NB	Width = 0.2399 Fn.: Box (75.07%)	Width = 0.1259 Fn.: Normal (80.86%)	Width = 0.5495 Fn.: Triangle (77.36%)	Width = 0.1585 Fn.: Epanechnikov (81.07%)	Width = 0.2884 Fn. = Normal (75.00%)
RF	Trees = 260 Preds at node = 3 Obs/leaf = 12 (76.07%)	Trees = 620 Preds at node = 3 Obs/leaf = 4 (81.71%)	Trees = 560 Preds at node = 3 Obs/leaf = 8 (78.79%)	Trees = 660 Preds at node = 7 Obs/leaf = 20 (82.43%)	Trees = 660 Preds at node = 7 Obs/leaf = 16 (77.93%)
MLP	HN = 3 Max. Iter = 400 $\eta = 10^{-2}$ $\alpha = 10^{-4}$ β = 0.6 Act. Fn: relu (75.86%)	HN = 100 Max. Iter = 200 $\eta = 10^{-2}$ $\alpha = 10^{-4}$ β = 0.9 Act. Fn: logistic (82.57%)	HN = 10 Max. Iter = 800 $\eta = 10^{-2}$ $\alpha = 10^{-3}$ β = 0.9 Act. Fn: logistic (78.29%)	HN = 5 Max. Iter = 800 $\eta = 10^{-3}$ $\alpha = 10^{-3}$ β = 0.6 Act. Fn: relu (81.21%)	HN = 100 Max. Iter = 200 $\eta = 10^{-2}$ $\alpha = 10^{-4}$ β = 0.7 Act. Fn: tanh (74.93%)

These parameter values were used for experimentation throughout the rest of this chapter.

4.3.3 Evaluation Methodology

This section discusses the performance metrics used, statistical tests carried out, and explains the methodology used for the execution time analysis.

4.3.3.1 Performance Metrics

The statistics used to evaluate performance were Accuracy and Sensitivity. These metrics are based on the values recorded in a confusion matrix: true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN), and are calculated as follows [205]:

$$Accuracy = \frac{TN+TP}{P+N} \quad (4.1)$$

$$Sensitivity = \frac{TP}{TP+FN} \quad (4.2)$$

where $P = TP + FP$ and $N = TN + FN$. These statistics are used in the literature to assess the classification performance of EEG BCI classifiers [10]–[12], [107], [188], [205]–[207]. The sensitivities to classes 1 (Cl 1) and 2 (Cl 2) were both considered in this analysis. Note that the sensitivity to Cl 2 corresponds to the specificity to Cl 1, with specificity defined as [205]:

$$Specificity = \frac{TN}{TN+FP} \quad (4.3)$$

4.3.3.2 Statistical Analysis

In all statistical analysis, a significance level of 0.05 was used. This means that *p*-values which were less than or equal than 0.05 were considered statistically significant. Statistical analysis in this study was carried using t-tests, Wilcoxon signed-rank tests, analysis of variance (ANOVA), McNemar's test and the Pearson correlation coefficient.

t-tests and an ANOVA compare sets of data to assess if there are any statistically significant differences between the sets. Whilst t-tests can only compare two sets of data, an ANOVA can compare multiple sets of data at once. The ANOVA can be conducted along only one dimension of a table, called a one-way ANOVA, or along the rows and columns of a table, called a two-way ANOVA. Both t-tests and the ANOVA assume that the data is normally distributed, thus an Anderston-Darling test was carried out to test whether the data was normal or non-normal prior to testing. In the case of ANOVA tests in this chapter, the data was always normally distributed, however in the case of t-test analysis, some samples were not normally distributed, and in these cases, the Wilcoxon signed-rank test was used because it is suitable for non-normal data.

McNemar's test [208] can be used to compare the performance of two classification approaches by factoring in the sensitivity and specificity of the classifiers. In this study, the test has been used to compare the classification results when using a channel subset, called *k,* to the case when 118 channels are

used. Table 4.3 shows the contingency table used to calculate the McNemar's test statistic, χ^2. The test statistic is calculated as: $\chi^2 = \frac{(b-c)^2}{b+c}$. It has a chi-squared distribution and using a 0.05 level of significance, the threshold for χ^2 to be significant is 3.84146. Thus, if χ^2 is above this threshold, the difference between the classifiers, measured in terms of b and c, is significant. If χ^2 exceeds the threshold and $c > b$, the reduced channel subset was deemed to have outperformed the case when 118 channels were used.

To identify correlations between two sets of data, the Pearson correlation coefficient was used. The value of this coefficient can range from -1 to 1, which indicate perfect anti-correlation and perfect correlation, respectively. A value of 0 indicates no correlation. In general, positive values of the coefficient indicate a positive correlation and negative values denote a negative correlation. A p-value is generated along with the coefficient and this was used to assess whether any correlations in the data were statistically significant. Pearson correlation uses a linear approximation of the data to derive the coefficient, and the coefficient value is not an implication of a direct relationship between the two sets of data. However, it was utilized in this study as an observational data analysis tool.

4.3.3.3 Execution Time Analysis Methodology

Execution time is an important aspect of functionality for BCI interfaces and is often recorded in studies [44], [65], [124], [134], [209] because it can impact user experience and the hardware required to operate BCIs [44], [124]. In this chapter,

Table 4.3: The contingency for McNemar's test. Values a-d are integers which represent the number of classified results falling into each category.

	Classifier Subset *k;* Correct Classification	Classifier Subset *k*; Incorrect Classification
118 EEG Channels; Correct Classification	*a*	*b*
118 EEG Channels; Incorrect Classification	*c*	*d*

the training and testing execution times of the multi-segment decision fusion classification system were recorded.

For accurate estimation of execution times, all other programs on the computer used were closed and elective background processes were suspended. To obtain an averaged result, 10 runs were carried out per subject. In each run, the total time taken to train the classifier, and the total time taken to classify the test set samples, were recorded using the *tic* and *toc* functions in MATLAB. The total times were then divided by the number of training or testing samples processed. The median time across all 10 runs was saved. The median and not the mean was chosen because it is more robust to outlier values, which could occur in the data if a mandatory background process in the computer biased the results of some of the folds. This process was repeated for each of the subjects, and finally the average training and testing processing times, calculated across the subjects, were used for the analysis. A Lenovo™ ideapad 330 laptop using a 64-bit Windows 10 operating system and an Intel® Core™ i5- 8300H, 2.30GHz CPU was used.

4.4 Results and Discussion

The results and discussion are divided into three sections. The first discusses the static EEG channel analysis, the second compares the performance of the different classifiers used, and the final section discusses the results for the multi-segment decision fusion classification approach.

4.4.1 Static EEG Channel Analysis

This section investigates how the different EEG channel subsets performed when compared to using all 118 EEG channels. The aim was to establish which subsets performed best and would be used for the multi-segment decision fusion approach. This section begins with a statistical analysis and concludes with a discussion of peak performance.

4.4.1.1 Statistical Analysis: Static EEG Channel Subsets

A two-way ANOVA was carried out to establish whether there were any significant differences in performance: i) between different channel subsets for each classifier, and ii) between the classifiers for each channel subset. This analysis was carried out for each subject individually. The *p*-values are recorded in Table 4.4. Considering the results for channel subset, it is evident that there were significant differences in performance across the five channel subsets (p<0.05). There were also significant differences in performance across the classifiers in most instances (p<0.05), except for accuracy and sensitivity to class 2 for subject *aa* and sensitivity to class 1 for subject *ay*. This was a generic comparison across all the channel subsets and classifiers to assess if there was a significant difference in performance across classifiers and channel subsets.

A McNemar's test [208] was then used to directly compare the results obtained for each channel subset to the case when 118 EEG channels were used. This analysis was carried out for each classifier individually, with the results shown in Figure 4.6. In the bar chart there is a separate group of bars for each classifier, with each group consisting of four colour-coded bars representing the four channel subset groupings, namely: C, C+CP, C+CF and C+CP+CF. Note how some of the bars are 'missing' for the LDA and RF classifiers - this is because $b = c$ and therefore $\chi^2 = 0$. The black dotted line denotes the value 3.84146, which is the threshold of significance: bar chart values that exceed the threshold are statistically significant, whilst those below the threshold are not statistically significant.

Table 4.4: p-values obtained from a two-way ANOVA. Results which are not statistically significant are shaded.

	aa		al		av		aw		ay	
	Channel Subset	Classifier	Channel Subset	Classifier	Channel Subset	Classifier	Channel Subset	Classifier	Channel Subset	Classifier
Accuracy	1.5e-13	6e-1	1.4e-05	1.1e-16	2.6e-05	3.4e-3	7.2e-05	3.7e-3	1.6e-05	4e-2
Sensitivity (Cl1)	1.9e-10	4e-2	5.0e-05	4.0e-18	4.5e-5	7.8e-3	4.7e-4	1.4e-2	7e-3	6e-1
Sensitivity (Cl 2)	2.3e-11	7.1e-1	3.5e-05	6.0e-13	3.3e-3	2e-2	1e-2	1.0e-2	9.6e-07	3e-2

In most cases, using subsets of channels instead of 118 EEG channels did not significantly impact classification performance. Exceptions to this trend were the C and C+CP+CF subsets for the SVM-Linear classifier, and the C+CP and the C+CP+CF subsets for the MLP classifier. To determine whether using the channel subset resulted in an improvement or degradation in classification performance in these cases, the values of c and b were compared: if $c > b$ then using the channel subset led to an improvement in performance compared to using all 118 channels, otherwise it led to a degradation in performance. In the cases when the C+CP or C+CP+CF subsets were used, there was an improvement in performance, however when the C subset was used, there was a degradation in performance.

These results are promising since they indicate that the static subsets C+CP and C+CP+CF are suitable across various classifiers, without leading to a significant depletion in performance. Perhaps more significantly, for some classifiers, such as the SVM-Linear and the MLP classifier, these static subsets showed the potential to lead to significant improvements in performance. Considering the results where there was no significant difference in performance, these results illustrate that the proposed channel subsets may be suitable for

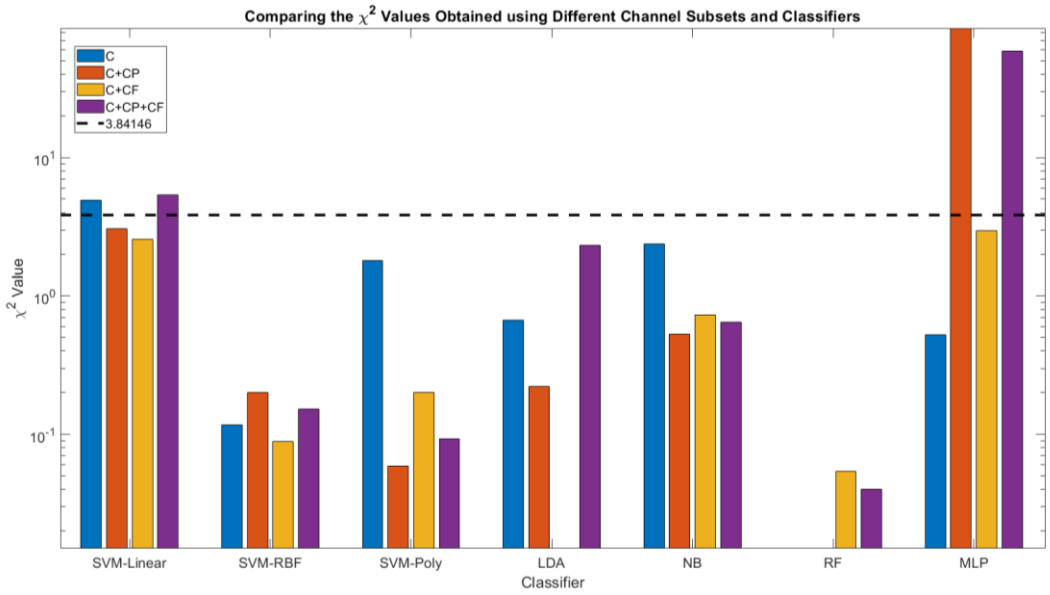

Figure 4.6: Comparing the χ^2 values obtained from a McNemar's test for different channel subsets and classifiers. The black dotted line denotes the threshold of singificance, values below the line are not statistically significant, whilst those above the line are statistically singificant. 'Missing' bars occur when χ^2=0, since b=c.

consideration if a static subset of electrodes is desired. This is an important finding because it is an established concept in the literature that reducing the number of EEG channels leads to lower computational processing demands, less redundant data and lower costs for electrodes [44], [48], [210]. Therefore, having candidate static channel subsets that may be suitable for EEG classification is useful.

4.4.1.2 Channel Subsets and Peak Performance

Table 4.5 shows the average performance for each classifier and channel subset combination, calculated across subjects. The column 'Average$_1$' contains the row averages, which capture the average performance of each channel subset across the classifiers. The rows labelled 'Average$_2$' contain column averages, which capture the performance of each classifier across the channel subsets. Peak average results are in bold.

Considering the Average$_1$ results in the final column of the table, the C+CP configuration had the highest accuracy and sensitivity to class 1, followed by the C+CP+CF configuration. Considering the results for sensitivity to class 2, the C+CP+CF subset gave the best performance, closely followed by the C+CP subset.

Table 4.5: The accuracy and sensitivity results for different channel subset and classifier combinations. The peak average results are in bold. Average$_1$ are results averaged across the classifiers for each channel subset and Average$_2$ are results averaged across the channel subsets for each classifier.

	SVM-Lin	SVM-Poly	SVM-RBF	LDA	NB	RF	MLP	Average$_1$
				Accuracy				
C	74.64	74.36	74.07	74.64	73.43	73.43	67.50	*73.15*
C+CP	80.57	80.93	80.36	80.93	78.64	79.50	78.64	***79.94***
C+CF	78.07	77.86	78.43	76.93	75.50	76.07	70.50	*76.19*
C+CP+CF	81.07	80.64	81.43	81.93	80.00	76.07	76.93	*79.72*
118	72.92	75.86	76.00	75.36	75.00	75.86	70.14	*74.45*
Average$_2$	*77.45*	*77.93*	***78.06***	*77.96*	*76.51*	*76.19*	*72.74*	
				Sensitivity Class 1				
C	76.14	76.57	78.57	76.57	74.29	74.00	69.00	*75.02*
C+CP	82.57	80.00	82.14	82.29	79.28	80.57	78.57	***80.77***
C+CF	78.57	79.57	80.14	77.86	72.00	77.29	71.43	*76.69*
C+CP+CF	80.57	79.86	81.72	83.43	80.14	77.29	76.43	*79.92*
118	74.43	75.86	77.14	77.43	72.43	77.29	70.86	*75.06*
Average$_2$	*78.46*	*78.37*	***79.94***	*79.52*	*75.63*	*77.29*	*73.26*	
				Sensitivity Class 2				
C	73.14	72.14	69.57	72.71	72.57	72.86	66.00	*71.28*
C+CP	79.29	81.14	78.57	79.57	78.00	79.03	79.43	*79.29*
C+CF	77.57	80.15	76.71	76.00	79.00	74.86	69.57	*76.27*
C+CP+CF	80.43	80.57	81.14	81.43	79.86	78.00	77.43	***79.84***
118	71.43	75.86	74.86	73.29	77.57	74.43	69.43	*73.84*
Average$_2$	*76.37*	***77.97***	*76.17*	*76.60*	*77.40*	*75.84*	*72.37*	

When considering the accuracy results obtained for each individual classifier, peak accuracy was achieved using the C+CP+CF channel subset 57% of the time and using the C+CP channel subset 43% of the time. To determine whether there was any significant difference in performance using the C+CP+CF and C+CP subsets, paired t-tests were used to compare the results obtained for each individual subject. The *p*-values obtained were: 0.0134, 0.0977, 0.2462, 0.0852 and 0.9910 for subjects *aa*, *al*, *av*, *aw* and *ay*, respectively. This indicated no significant difference in classification accuracy between these channel subsets except for subject *aa*, which had a *p*-value less than 0.05.

To conclude, these results indicated that the C+CP+CF and C+CP subsets tended to lead to peak performance across a variety of classifiers, and the results they produced were statistically similar for four out of five subjects. Based on the analysis in this sub-section and the previous one, subsets C+CP and C+CP+CF were selected for further processing and analysis using multi-segment decision fusion classification. These results indicated that the central and central-parietal regions provide important information for MI classification, possibly because the parietal region is involved with concentration [34], which is required for MI tasks. Furthermore, the synergistic combination of the C+CP+CF regions provided more discriminative information than just the C+CF regions, indicating again the importance of the contribution of the central-parietal region to classification performance.

4.4.2 Comparison of Classifiers

This section is an extension of the discussion about Table 4.5 in the previous section and is focused on comparing the performance of the classifiers, averaged across the different channel subsets, captured in the Average_2 rows. Although extensive work has been carried out into comparing conventional classifiers [5], [31], [101], [140], it is uncommon for a discussion to be based the performance of conventional classifiers averaged across different channel subsets. The opportunity was taken to analyse these conventional classifiers using this novel perspective.

First, an analysis was carried out to identify whether there was any significant difference in the Average$_2$ results. A Wilcoxon signed-rank test was used on the accuracy results, which were found to be non-normal, and *t*-tests were used on the sensitivity results, which were found to be normal. The *p*-values obtained for the accuracy, sensitivity to class 1 and the sensitivity to class 2 were $1.56e^{-2}$, $1.54e^{-10}$ and $3.47e^{-11}$, indicating that the results varied significantly between classifiers.

Considering the Average$_2$ results, the SVM-based classifiers gave the best performance, with the SVM-RBF classifier giving the best accuracy and sensitivity to class 1, and the SVM-poly classifier giving the best sensitivity to class 2. Considering the individual results for subsets C+CP and C+CP+CF, the peak accuracies were obtained for the SVM-poly classifier and the LDA classifier, respectively, with the accuracy of the latter, 81.93%, being the peak accuracy in the table.

The robustness of the classifiers was then assessed. A comparison was carried out to identify whether there were any classifiers for which there was *no* significant difference in the sensitivity results for classes 1 and 2. Finding no significant difference would indicate that the classifier had a robust performance across classes, which is important in practical BCIs. Since the data was found to be normal, paired *t*-tests were used to compare the sensitivity results obtained for class 1 for all the channel subsets to the corresponding results obtained for class 2, and this test was carried out for each classifier. Table 4.6 shows the *p*-values obtained from this analysis. The *p*-values above 0.05 indicate that the SVM-Poly, SVM-RBF, NB, RF and MLP classifiers all exhibited robust performance when classifying each of the two classes across all five EEG channel montages investigated. Notably, the LDA classifier failed to have uniform performance across the two classes. Similar results of high accuracy and poor specificity for

Table 4.6: p-values obtained for each classifier when using paired t-tests to compare the results the sensitivities to classes 1 and 2 across the different channel subsets. Results which were not statistically significant are shaded.

Classifier	SVM-Linear	SVM-Poly	SVM-RBF	LDA	NB	RF	MLP
p-value	3e-2	7e-1	6e-2	3e-3	4e-1	8e-2	3e-1

LDA have been observed previously in [211]. Another contributing factor to this observation could be that both MI classes were associated with the right-hand side of the body, leading to strong MI patterns on the left-hand side electrodes for both classes [34], [163], possibly making linear discrimination with high specificity less likely.

Based on these results, the SVM-RBF or SVM-poly classifiers, with the channel subsets (C+CP) or (C+CP+CF) would be most strongly recommended for conventional classification problems when whole trials are classified. Since channel subsets affect the noisy data entering a classifier, they can affect the decision boundaries formed, and thus the overall classification performance. Previously, various studies in the literature have compared different classifiers using just one static EEG subset [5], [31], [101], [140]. By comparing different classifiers within the context of multiple static channel subsets, the analysis in this section was a novel contribution to this area of research.

4.4.3 Evaluation of Multi-Segment Fusion Classification Approach

This section analyses the multi-segment fusion classification approach. It begins by assessing the effect of multi-segment fusion classification on performance when compared to whole-trial classification. Afterwards, the relationship between each windowing scheme and performance is discussed, including an execution time analysis. Finally, a comparison to the literature is made.

4.4.3.1 Multi-Segment Fusion Performance Analysis

Recall that channel subsets C+CP and C+CP+CF were selected for assessing the multi-segment fusion classification approach. The same classifier parameters as those obtained through hyperparameter tuning and ten-fold cross-validation were used in this section. Figure 4.7 and Figure 4.8 show the results using the channel subsets C+CP and C+CP+CF, respectively. These figures can be found at the end of this section. The colour coding was used to compare the results using multi-segment fusion to whole-trial classification ('No Windowing'). The marker

type indicates whether accuracy, sensitivity or specificity are denoted. The accuracy and sensitivity values were obtained by averaging across the five subjects. The MLP classifier is not included in these figures because the multi-segment fusion approach did not result in any significant changes in performance for any window size or window increment.

The multi-segment fusion approach has the potential to improve performance when compared no windowing. For example, an overall peak accuracy of 84.51% was obtained using the LDA classifier with the C+CP subset and windowing scheme of (1.75s, 0.25s), which was an improvement when compared to the accuracy when not using windowing, 80.93%, as shown in Figure 4.7. This windowing scheme also improved the sensitivity of the LDA to classifier to classes 1 and 2. Considering the results for the C+CP+CF channel subset in Figure 4.8, a peak accuracy of 84.43% was obtained for the SVM-Linear classifier for scheme (1.25s, 0.1s), compared to 81.07% accuracy when no windowing was used.

There are a number of cases where multi-segment decision fusion improved all three-performance metrics, namely: i) LDA-C+CP for the schemes: (2s,0.25s), (1.75s,0.25s), (1.75s,0.1s) and (1s,0.1s) and ii) NB-C+CP for the window/increment scheme (1.75s,0.25s). In other cases, multi-segment fusion classification improved one or two areas of performance, with no significant change in the other area/s.

There are also instances where multi-segment fusion classification led to a deterioration in one area of performance and a coinciding improvement in another area of performance. One pattern that is evident in the data is an increase in accuracy coinciding with a decrease in sensitivity to class 1. Examples of this with the C+CP subset were the SVM-Linear classifier with scheme (1.5s,0.5s) and LDA classifier with scheme (0.5s,0.5s). Examples with the C+CP+CF subset were SVM-RBF classifier with scheme (1.25, 0.5) and the NB classifier also with scheme (1.25,0.5). Lower sensitivity to class 1 could indicate a reduction in false positives for that class, which led to an improvement in overall accuracy [205].

Negative deterioration in one area of performance, without a complementary increase in another area of performance, was observed only in the SVM-Linear classifier, for subset C+CP and (025s, 0.1s). The impact of multi-segment fusion appears to be dependent on the classifier and channel subset pairing. This result further highlights the importance of researchers using static subsets consider multiple configurations during the design stage.

These results illustrate that multi-segment fusion classification has the potential to significantly improve classification accuracy for a variety of classifiers, with LDA, SVM-Linear and NB classifiers being particularly susceptible to improvement. Data segmentation is widespread in the MI EEG literature, and this analysis showed that majority voting-based decision fusion can be used to exploit the segmented data to boost classification performance. However, the window size and increment must be tuned to prevent any deterioration in performance. This tuning could be carried out on the training set using a grid-search through parameters. The correlation analysis later in this chapter provides further recommendations for window design in practice.

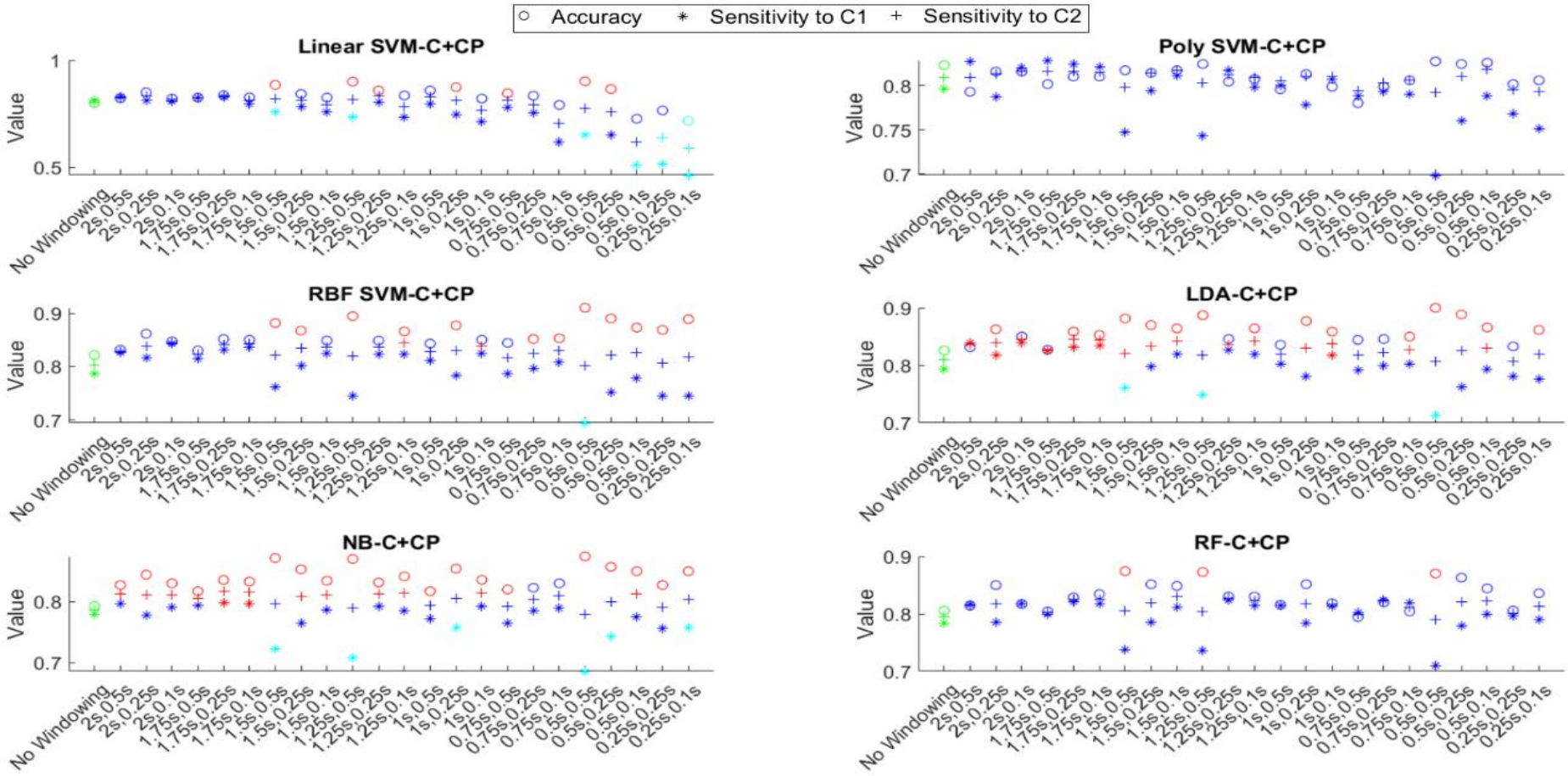

Figure 4.7: Results for subset C+CP showing how accuracy and sensitivities to classes 1 and 2 change with multi-segment fusion with different windowing schemes when compared to no windowing. The x-axes denote the windowing schemes used. The values of each statistic, averaged across the five subjects, are plotted. The colour coding is as follows: green – without windowing, blue – multi-segment fusion had no significant effect, red/cyan – multi-segment fusion had a statistically significant effect, improving (red) or diminishing (cyan) the performance. Statistical significant was tested using an ANOVA test.

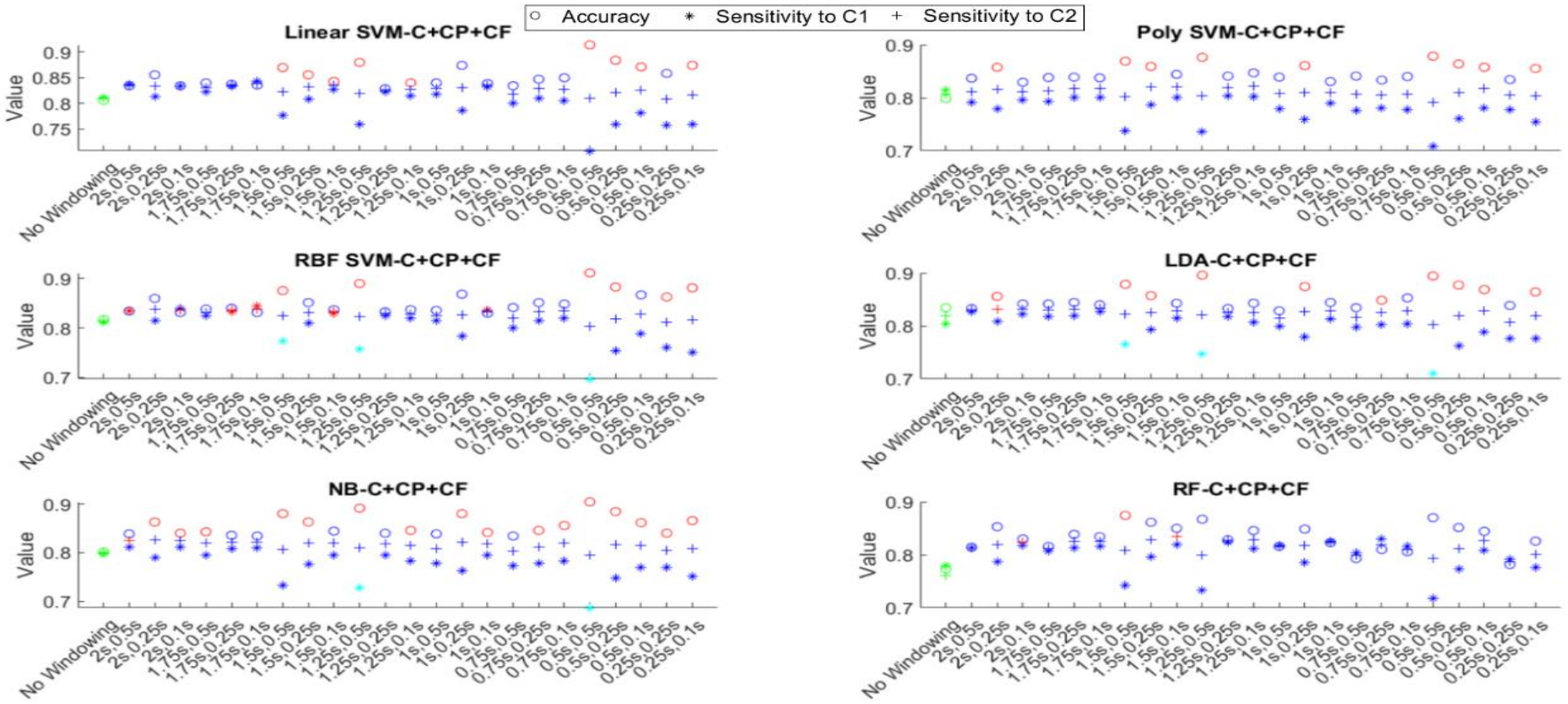

Figure 4.8: Results for subset C+CP+CF showing how accuracy and sensitivities to classes 1 and 2 change with multi-segment fusion with different windowing schemes when compared to no windowing. The x-axes denote the windowing schemes used. The values of each statistic, averaged across the five subjects, are plotted. The colour coding is as follows: green – without windowing, blue – multi-segment fusion had no significant effect, red/cyan – multi-segment fusion had a statistically significant effect, improving (red) or diminishing (cyan) the performance. Statistical significant was tested using an ANOVA test.

4.4.3.2 Execution Time Analysis of Multi-Segment Fusion Classification

The LDA classifier with channel subset C+CP, which provided peak performance for multi-segment fusion, was used for this analysis. Figure 4.9 shows the processing time per trial results obtained for different window sizes and window increments, for the training and the testing phases. Although both the window size and window increment size impact the number of segments obtained from each trial and thus how many feature vectors are extracted and processed, it is evident that the results for different window sizes are grouped closely together, and it is the increment size which has the most substantial impact on processing time. Smaller increment sizes led to an increased processing time because more segments had to be processed for each trial. However, the peak testing time is below 500µs, which is well below the human visual perception time of 13ms [45], indicating the multi-segment fusion approach had an acceptable latency for a BCI.

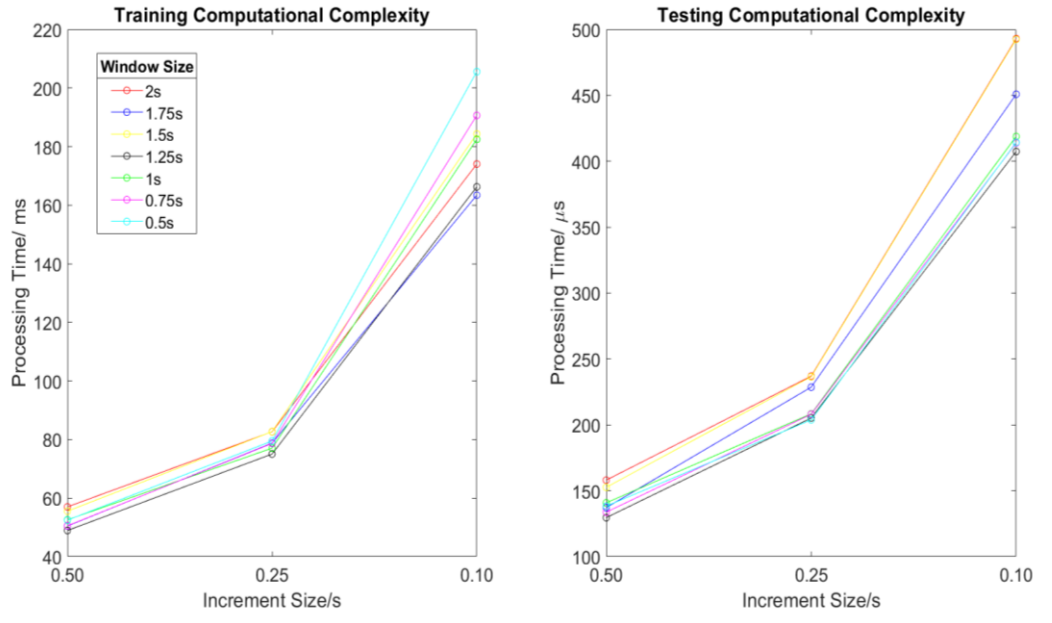

Figure 4.9: Computational complexity analysis for training and testing. The x-labels denote the window increment size, and the colours of the plots denote the window size.

4.4.3.3 Window Scheme Correlation Analysis

The relationship between window scheme and performance was then analysed using the Pearson correlation coefficient. Although the Pearson correlation coefficient provides limited inferential information, it is still a useful tool for observational analysis.

This analysis was conducted separately for the three performance metrics: accuracy, sensitivity to class 1 and sensitivity to class 2. The results, averaged across subjects, were considered. For each windowing scheme, the peak averaged classifier metric was used for the analysis. The metric results were then split into three groups based on the window increment sizes used (0.5s, 0.25s and 0.1s). The Pearson correlation coefficient (ρ) and its corresponding p – value was calculated for each of the groups. The results for subsets C+CP and C+CP+CF are recorded in Table 4.7.

The coefficient results for accuracy are always positive and had p-values less than 0.05, indicating that there was a significant positive correlation between the accuracy and window size, inferring larger windows tended to provide a higher accuracy. This was observed in the previous section, with the peak accuracies being associated with window sizes of 1.75s and 1.25s, some of the larger window sizes considered. For the sensitivity results, there was no significant correlation with window size for increments of 0.5s or 0.25s, but there was a significant negative correlation between the size of the window and sensitivity to class 1, indicating that as the window got larger the sensitivity to class 1 tended to get smaller. This is the inverse of the relationship with accuracy,

Table 4.7: Observing how the Pearson correlation coefficients and corresponding p-values vary with decreasing window size for each window increment. Channel subsets C+CP and C+CP+CF were considered.

C+CP									
Inc Size	0.5			0.25			0.1		
Statistic	Acc.	Sens. C1	Sens. Cl 2	Acc.	Sens. Cl 1	Sens. Cl 2	Acc.	Sens. Cl 1	Sens. Cl 2
ρ	0.84	-0.57	0.61	0.92	-0.38	0.64	0.86	-0.76	0.94
p-value	0.02	0.18	0.15	1.2e-3	0.35	0.09	6.3e-3	0.03	$5.4e^{-4}$
C+CP+CF									
Inc Size	0.5			0.25			0.1		
Statistic	Acc.	Sens. Cl 1	Sens. Cl 2	Acc.	Sens. Cl 1	Sens. Cl 2	Acc.	Sens. Cl 1	Sens. Cl 2
ρ	0.87	-0.45	0.60	0.87	-0.39	0.59	0.76	-0.88	0.87
p-value	0.01	0.31	0.16	3e-3	0.35	0.12	0.03	4.1e-3	4.8e-3

possibly indicating that as the window size got larger, false positives attributed to class 1 also decreased. This would agree with observations made previously in Section 4.4.3.1, in which a significant decrease in sensitivity to class 1 tended to coincide with a significant improvement in accuracy. This also coincided with significant positive correlation between window size and sensitivity to class 2 for an increment of 0.1s in Table 4.7, which further supports this conclusion.

Despite these results, the peak accuracy results were not obtained using the 2s window size, which was the largest window available. EEG data of length 1.5s or less have been considered approximately stationary in previous studies [91], [195]. This was previously mentioned in Chapter 3 (Section 3.1.5). This tendency towards stationarity could enhance CSP feature extraction, which is adversely affected by non-stationarities [196]. However, windows smaller than 1s would be an even closer approximation to stationarity but are not correlated with improved accuracy. This may be because there was not adequate data for the CSP features to be highly discriminative. These correlation results, together with the peak accuracy results obtained for multi-segment decision fusion with window sizes of 1.75s and 1.25s, may indicate that the accuracy of multi-segment decision fusion classification depends on a trade-off of two factors when it comes to window size: i) having small enough window sizes that the EEG data could tend towards stationarity; and ii) having a window size large enough to allow the extraction of discriminative features.

4.4.3.4 Comparison to Current State-of-the-Art EEG MI Classification Methods

Table 4.8 compares the peak multi-segment decision fusion classification results, obtained using subset C+CP, window scheme (1.75s,0.25s) and LDA classifier to results obtained using comparable state-of-the-art approaches that have been presented in the literature. Note that this configuration of channel subset, window scheme and classifier were used across subjects and that the hyperparameters for the LDA classifier were as those tuned in Section 4.3.2. In the table, conventional approaches, like the multi-segment decision fusion

approach, are shaded. Comparisons in this section are centred on the Average Accuracy results in the table (last column).

The studies in Table 4.8 were selected because they had generally similar testing approaches to the 10-fold cross-validation approach used in this chapter. In particular, Baig et al. [104] used 10-fold cross-validation, Kumar et al. [106] used a 10-by-10-fold cross-validation approach, and She et al. [107] used a nine-fold cross-validation approach. He et al. [49] used the same train-test splits in the data but do not mention reporting cross-validated results. Olias et al. [10] used a Monte-Carlo based testing approach which involved splitting all the data available for each subject into training and test sets 40 times, and then averaging the test set accuracies. In each split, 40 trials are used for training and 40 for testing. These differences in testing methodologies may have impacted the fairness of the comparison. In the literature, comparisons like those in Table 4.8, in which the performance of a proposed system is compared to the reported performance of systems in other studies that use the same dataset but which may have been tested with a slightly different methodology, have been made [9], [104]. However, it is important to note that the conclusions made from this kind of discussion can be limited due to the methodological differences in testing.

Overall, the multi-segment decision fusion approach outperformed some other conventional approaches presented in papers in the literature [10], [104],

Table 4.8: Comparing the multi-segment fusion classification approach to conventional [10], [103], [49] *and deep learning* [122], [123] *approaches in the literature. Results for the proposed approach are bold.*

			Classification Accuracy (%)					
Papers	**Features & Classifiers used**	**Channels**	***aa***	***al***	***av***	***aw***	***ay***	***Average***
Olias et al. (2019) [10]	CSP+LDA (classical)	118	67.68	96.81	62.12	85.12	87.68	*79.88*
	Normalised CSP+tangent space logistic regression		70.31	96.31	67.87	87.75	91.62	*82.77*
Baig et al. (2017) [104]	CSP+SVM (classical)	118	82.00	94.00	70.00	87.00	87.00	*84.00*
	CSP with differential evolution feature selection +SVM		95.80	98.80	89.80	99.20	96.50	*96.02*
He et al. (2013) [49]	Rayleigh coefficient+LDA	118	67.90	88.30	59.20	87.60	80.40	*76.68*
	Rayleigh coefficient+LDA with GA optimisation to select channel subset	Subject-specific (13-18)[1]	86.40	98.50	75.10	93.90	87.10	*88.20*
She et al. (2019) [15]	Hierarchical extreme learning machine with deep architecture	118	61.70	100	73.88	88.17	79.64	*79.33*
Kumar et al. (2017) [106]	CSP + encoder-based deep neural network	118	90.00	97.5	73.00	97.00	96.00	*90.70*
Multi-segment decision fusion	**CSP+LDA (Fusion: 1.75s,0.25s)**	**C+CP (14)**	**80.40**	**95.36**	**72.50**	**79.29**	**95.00**	***84.51***

[105] and an extreme learning approach based on deep learning [107]. However, it was outperformed by some systems with automated channel selection [104], [105], and an encoder-based deep learning approach [106]. The rest of this section discusses the comparisons in Table 4.8 in more depth.

The multi-segment decision fusion approach with static channel subset as proposed in this thesis outperformed the classic conventional techniques by Olias et al. [10], Baig et al. [104] and He et al. [49] (highlighted in grey). It also outperformed the implementation of She et al. [15], which involved sparse feature extraction and extreme learning machine classification. These implementations used 118 EEG channels, whilst the implementation presented in this chapter used only 14 channels for the C+CP channel subset. It is important for the development of practical BCIs to identify classification approaches which use fewer electrodes without diminishment in classification performance, thus the proposed approach is competitive. In fact, the multi-segment decision fusion approach outperformed state-of-the-art conventional classification approaches using fewer sensors than used in the related work.

The multi-segment decision fusion classification approach was outperformed by the pipeline with genetic channel selection by He et al. [49] and the differential evolution feature selection approach by Baig et al.[104] (results not highlighted in grey). Although not a pure channel selection approach, the implementation by Baig et al. [104] extracts 236 salient features from the 118 EEG channels, and then uses differential evolution for feature selection. This process can therefore exclude features from certain channels which were deemed redundant.

The deep learning CNN implementation by Kumar et al. [106] also outperformed the multi-segment decision fusion approach. Notwithstanding this, deep learning approaches require significant training times and investment in more expensive graphical processing unit (GPU) technology, which can impact their widespread practical use in some cases.

This comparison to the literature shows the benefits and limitations of the multi-segment decision fusion classification approach presented. The approach

proposed in this chapter, which included a static channel subset selection methodology, outperformed other works in the literature that used static channel subsets [10], [107], and this confirms the validity and relevance of the method presented in this chapter. The multi-segment decision fusion approach is also versatile and can be applied to CPU-based and GPU-based classifiers.

The main contribution of this chapter in relation to the wider literature is the versatility and low execution time of the multi-segment decision fusion approach. This approach was found to be effective in significantly improving the performance of four different classifiers, namely LDA, SVM, RF and NB classifiers. In comparison, He et al. [49] only tested their channel selection approach on one classifier, meaning that its versatility is still open to investigation. Due to its versatility, the multi-segment decision-fusion approach could be applied to various pipelines with the aim of boosting performance, even possibly the approaches of He et al. [49], Baig et al. [104] or Kumar et al. [106]. This is an area where future work could be done.

The multi-segment decision fusion approach introduced less than 500μs execution time overhead per trial during testing. Human visual perception time is 13ms [45], meaning that it is unlikely that this overhead would be perceived by subjects in an online system. Unfortunately, the papers reviewed do not report the execution time overhead of the channel or feature selection approaches proposed [49], [104], [106]. However, the approaches by both He et al. [49] and Baig et al. [104] are both metaheuristic wrapper techniques. These techniques, previously discussed in Chapter 3, are known to notably increase the latency experienced by the subject between training data recording and online testing [44]. This is because the metaheuristic channel or feature selection occurs on the training data [44], [49], [104]. The multi-segment decision fusion approach was found to introduce less than 220ms to the training time, making it a competitive approach for boosting performance at a relatively low computational expense.

The performance of the approach presented in this chapter falls short of the performance of more sophisticated channel/feature selection techniques [104], [105] and a deep learning technique [106]. Motivated by this observation,

Chapter 5 presents a GA-based channel selection algorithm that produces an optimized subset of channels for an individual subject. Although this method is found to be effective for subject-specific channel selection, it is found to be computationally expensive for subject-independent channel selection, which brings a range of benefits when compared to subject-specific channel selection including eradication of channel selection latency for the end user and systems with fewer electrodes. In Chapter 6, a CNN-based channel selection method is presented, which selects a subject-independent subset of channels in a more efficient way than the GA channel selection method.

4.5 Conclusion

The main contributions of this chapter can be summarised as follows:

- From the static channel analysis:
    - Developing channel subsets based on scalp region groupings can be a viable method for developing a static channel subset for MI EEG BCI studies which do not use automated channel selection.
    - A study found that the C+CP and C+CP+CF channel subsets were the most reliable for static channel analysis. In fact, these channel subsets led to no deterioration in performance and, in some cases, improvement in results.
    - A novel comparison of machine learning classifiers was carried out within the context of several channel subsets. In this analysis, the SVM-Poly and SVM-RBF classifiers were found to outperform SVM-Linear, LDA, NB, RF, and MLP classifiers.
- The main contribution of this chapter was the extensive investigation into the effect of window size and window increment size within a majority voting-based decision fusion framework. A correlation analysis indicated that greater accuracy tended to be correlated with larger window sizes, and peak performance was obtained with a window of size 1.75s.

- The majority voting-based multi-segment decision fusion classification framework was demonstrated to significantly improve performance for LDA, NB, RF and SVM classifiers. It was rare for the multi-segment decision fusion approach to lead to a deterioration in all areas of performance.
- 84.51% was the peak accuracy obtained with the multi-segment decision fusion approach. This was achieved using an LDA classifier with the C+CP subset and windowing scheme (1.75s,0.25s). This was on par with or outperformed some other classical conventional classifiers [10], [49], [104] which used more EEG channels. It also outperformed a deep learning approach using extreme learning machines [107].

EEG data segmentation is widely used in the literature. The multi-segment decision fusion framework was demonstrated as a way of exploiting this segmentation approach to improve the classification performance. It was also versatile, leading to significant improvements across different classifiers. The execution time analysis showed that the approach is lightweight, leading to training and testing latencies in the order of milliseconds and microseconds, respectively. This post-processing could be added to pipelines in order to boost performance with low additional computational overhead.

However, multi-segment decision fusion did not perform as well as some systems using algorithmic channel selection [104], [105] or deep learning [106], possibly because it was applied to a conventional machine learning pipeline in the analysis within this chapter. Traditional CSP-based pipelines are still used in the literature for design and investigation [82], [104], [106], [107], [133], [134], [138], and the proposed approach is still an effective tool that could be applied to these and other pipelines for improved performance. The versatility of the multi-segment decision fusion approach means that in future work it could be applied to the works in [104], [105] and [106], possibly improving their classification performance further.

Motivated by the superior performance of algorithmic channel selection methods in [104], [105] the next chapter of this chapter focuses on

algorithmic channel selection. It presents a GA-based channel selection approach for selecting subject-specific channels for a SL classifier.

Chapter 5 : Sparse Learning and Genetic Channel Selection for MI EEG Classification with the Idle State

5.1 Introduction

In this chapter, a new genetic algorithm band power sparse learning classification system is presented for EEG (GABSLEEG). It consists of a dictionary-based SL classifier preceded by a GA module used for channel selection. A paper related to the work in this chapter has been published in *Neurocomputing* [212].

A SL dictionary-based classification module was chosen because of the strong history of this kind of classifier in the literature, discussed previously in Chapter 3 (Section 3.3) [11], [56], [57]. Similar systems in the literature use frequency or time-frequency features [11], [56], [57], however the novel implementation in this chapter uses time-domain band power features. This decision was motivated by the work of Arnin et al. [124], who found that time-domain features can be just as effective as frequency-domain features, and at a lower computational cost. In this chapter, the feature vector is constructed by extracting the band power in the combined alpha and beta frequency bands – so, 7.5Hz-32Hz – for each EEG channel. This is novel because similar works in the literature have typically constructed the feature vector using more than one feature per channel [11], [56], [57].

Furthermore, recent works in SL classification in the literature have not investigated the impact of sparsity level or window segmentation size on accuracy [11], [56], [57]. During system calibration in Section 5.3.5.1 the relation between classification performance and these two design features is investigated.

In Section 5.4.1, the SL classifier was compared to three popular conventional classifiers, namely SVM, k-NN and RF. To the authors' knowledge this kind of comparison between conventional classifiers and a dictionary-based SL system for MI EEG classification has not been carried out before in the literature.

Despite the strong performance of the SL classifier presented in this chapter, the OMP encoding algorithm used is computationally expensive [54], [55], [168]. This kind of encoding algorithm was used due its popularity in similar systems in the literature [11], [56], [57]. The test-set execution time results in Section 5.4.3 confirmed that the proposed SL classifier using all the EEG channels in the dataset may be unsuitable for use in a real-time system. It should be noted that during these tests the code was not specifically optimized to reduce the computational time, so with optimized code the times recorded may be reduced.

The execution time of the OMP algorithm is linked to the number of channels used, with the execution time increasing when a greater number of channels are used [54]. Previous studies focused on SL classification have used hand-picked subsets of EEG channels which may not be optimal [11], [56], [57]. Although these channels are typically chosen to be close to the central scalp region, which is known to be where MI activity manifests, there is no analysis or automation associated with channel selection [11], [56], [57]. In this chapter, a metaheuristic channel selection approach is applied to select a subject-specific subset of channels from training and validation datasets. The selected subset is then used on the test-set, resulting in improved execution times. Channel selection is applied with the core aim of maintaining or enhancing the already good classification performance of the SL classifier, whilst improving the computational speed on the test-set through a reduced channel subset. A metaheuristic approach was chosen because it provides a trade-off between the strong performance of wrapper techniques and faster convergence [44]. A GA

was the metaheuristic method of choice because GAs already have a history of good performance in EEG channel selection [49], [192]. To the authors' knowledge this is the first time that a GA channel selection module has been used in conjunction with a dictionary-based SL classifier for MI EEG. Automated channel selection techniques were previously discussed in Chapter 3 (Section 3.5.2), and a high-level introduction to GAs was provided in Chapter 3 (Section 3.5.2.2). Thus, the proposed GABSLEEG system consists of two modules: the GA channel selection module and the SL classification module. The GABSLEEG system is compared to state-of-the-art systems [56] in Section 5.4.5.2.

An in-depth analysis of the performance of the GABSLEEG system was carried out. Previous works in SL classifiers for MI EEG have only assessed classification performance in problems involving data generated during different MI tasks [11], [56], [57]. However, if SL classifiers are to be applied to practical scenarios, it is important to assess their capability in classifying the idle state, which is when the user is not actively imagining any movement. In all experiments in this section, except for those in Section 5.4.5.1 when comparing to the general literature, the idle state is a class within the classification problem. Another aspect of SL classifiers that has not been assessed in the literature is robustness to changes in training data size [11], [56], [57]. In Section 5.4.2 the performance of the GABSLEEG classifier is assessed as the amount of training data used is reduced. This is an important assessment since the recording of training data from subjects can cause user fatigue and introduces an impractical latency before the BCI can be used. Thus, algorithms that can perform with less training data are more favorable.

The layout of this chapter is as follows. Section 5.2 discusses the proposed GABSLEEG implementation and Section 5.3 explains the experimental methodology and hyperparameter tuning. Results and discussion are covered in Section 5.4. The chapter closes with a conclusion of the contributions in Section 5.5.

5.2 Proposed Sparse Representation and Genetic Channel Selection Approach

A flowchart of the proposed GABSLEEG system is shown in Figure 5.1. Figure 5.2 shows in detail the structure of the SL classifier module, labelled as 'Sparse Learning Classifier' in Figure 5.1. In the pre-processing stage, the EEG time series is first filtered and then segmented. Feature vectors based on the band power of each channel are then extracted from each segment, and are divided into training, validation, and test sets. The training data is used to construct a dictionary for sparse learning, which consists of three sub-dictionaries: one for MI class 1, another for MI class 2 and the final one for idle state data. A GA is used for channel selection, taking the dictionary and validation data as input. The GA selects a

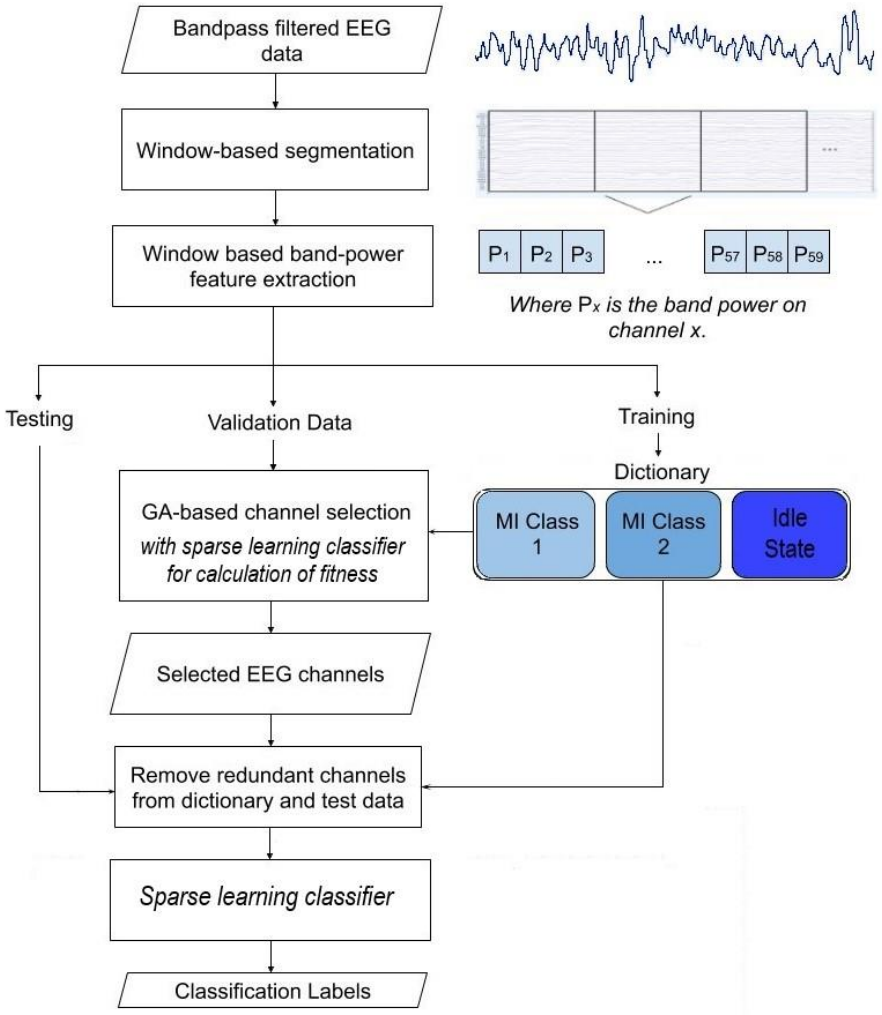

Figure 5.1: The proposed GABSLEEG classification system.

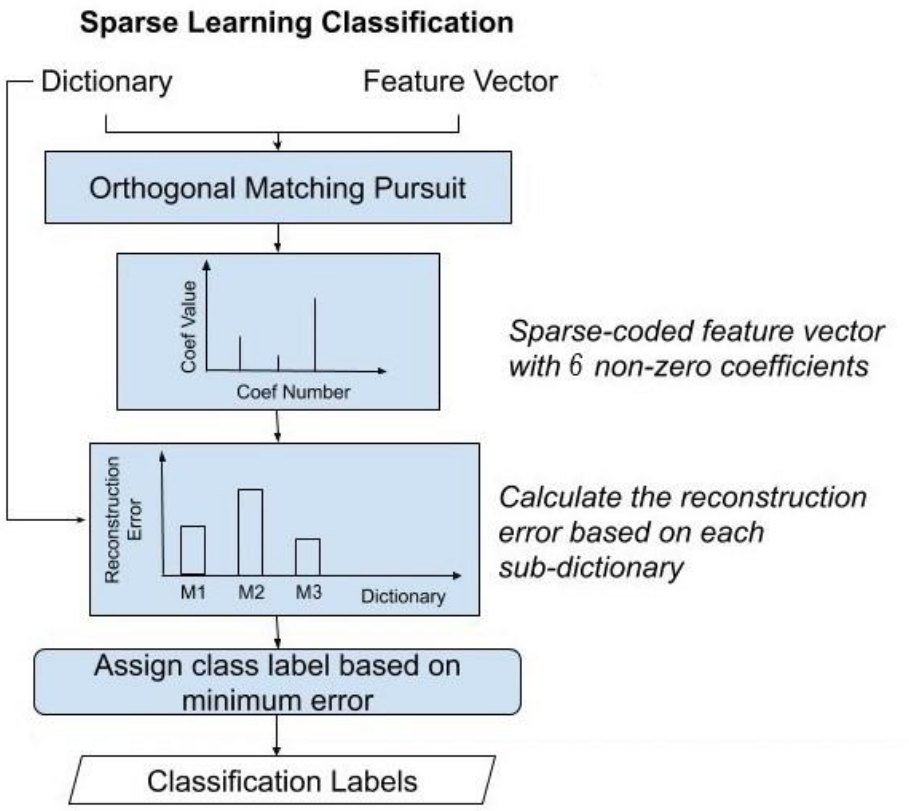

Figure 5.2: The sparse learning (SL) classifier.

suitable subset of EEG data through an iterative process which aims to maximize the accuracy of the SL classifier. Based on the channel subset selected by the GA, channels are removed from the dictionary and test data accordingly. A SL classification module is used to assign the final labels to each test set feature vector. As shown in Figure 5.2, each test feature vector is sparse encoded over the dictionary using OMP, with the sparse encoding having six non-zero coefficients. The reconstruction errors based on each of the three sub-dictionaries are then calculated, and the class label of the sub-dictionary which gives the minimum error is assigned to the test feature vector. The proposed system, as well as the benchmarking classifiers, are implemented in Python 3.

5.2.1 Pre-Processing and Feature Extraction

The EEG data is mean-centered and then filtered using a 10^{th} order Butterworth filtered, with a passband from 7.5Hz to 30Hz, which spans the alpha and beta

frequency bands [34]. Afterwards, the data is segmented using a window size of t seconds. This means that each segment of EEG data is of size $T \times M$, where T is the number of time samples, calculated as: $T = t \times Fs$, with Fs is the sampling frequency, and M is the number of EEG channels. In this work an investigation was carried out to identify the optimal window size, the details of which are discussed in the Section 5.3.5.1.

For each segment, the average power on each EEG channel, p_j, is calculated using: $p_j = \frac{1}{T}\sum_{i=1}^{T} x_{ji}^2$, where x is the value of the EEG data on channel j at time i. These power values are concatenated together to produce a feature vector of size $(M \times 1)$ for the segment.

5.2.2 Sparse Learning

The SL dictionary is constructed of feature vectors obtained from the training dataset. As shown in Figure 5.1, the dictionary has three sub-dictionaries constructed from feature vectors from MI EEG class 1, MI EEG class 2 and the idle state. Each sub-dictionary has a size of $M \times L$, where M is the number of EEG channels in the subset (which corresponds to the length of the feature vectors), and L is the number of feature vectors in the sub-dictionary. The value of L is determined by the class which had the lowest number of training feature vectors. The total length of the dictionary is therefore $3L$.

For classification, feature vectors from the test set are sparse encoded over the dictionary using OMP. Technical details of the OMP algorithm can be found in Chapter 3, Section 3.3.1. If the test feature vector is $\boldsymbol{y}$, the sparse reconstruction $\hat{\boldsymbol{y}}$ is calculated as: $\hat{\boldsymbol{y}} = \boldsymbol{D}\boldsymbol{x}$, where $\boldsymbol{D}$ is the dictionary and $\boldsymbol{x}$ is the row vector of sparse coefficients of length $3L$. The aim of sparse learning is to calculate the coefficients vector, $\boldsymbol{x}$, which has r non-zero coefficients. The value of r is six for this system, and the tuning process is described in Section 5.3.5.1 of this chapter. Figure 5.3 shows an illustrative example of the dictionary, which has three sub-dictionaries and 59 rows, one for each channel. The figure also shows an example of a sparse coefficient vector with orange boxes representing non-zero values and black boxes representing zero values.

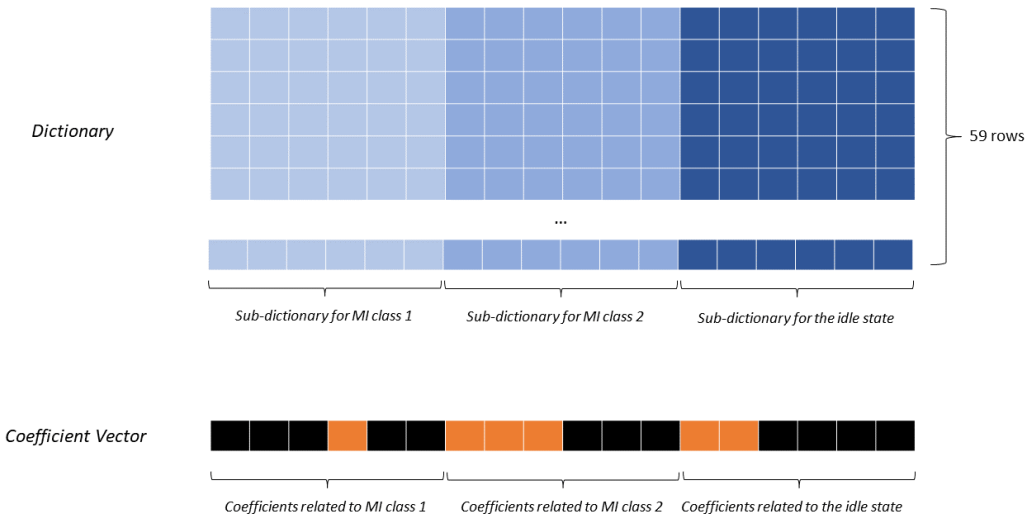

Figure 5.3: An illustrative example of a dictionary and a vector of sparse coefficient values. In the coefficient vector, orange boxes represent non-zero values and black boxes represent zero values.

In the first step of calculating $\boldsymbol{x}$, OMP creates a support vector, $\boldsymbol{S}$, which contains the indices of the dictionary entries which are most representative of the test feature vector, $\boldsymbol{y}$. The length of the support vector is r. The support indices are based on the residual error, e, calculated as:

$$e = \|\boldsymbol{y} - \sum_{i \epsilon \boldsymbol{S}} \boldsymbol{x}_i \boldsymbol{d}_i\|_2 \tag{5.1}$$

where $\boldsymbol{d}_i$ is the dictionary atom, $\boldsymbol{x}_i$ is the coefficient associated with that dictionary atom, and $\|.\|_2$ is the Euclidean norm. OMP uses a greedy search approach for constructing the support vector, adding the index of the next atom in the dictionary which correlates with the residual error.

The values of the non-zero coefficients in $\boldsymbol{x}$ are obtained by minimizing the reconstruction error, $\mathbf{r_e}$:

$$\boldsymbol{r_e} = \min_{\boldsymbol{x_S}} \|\boldsymbol{y} - \boldsymbol{D_S x_S}\|_2 \tag{5.2}$$

where $\boldsymbol{D}_S$ is constructed from dictionary atoms selected through the support vector and $\boldsymbol{x}_S$ are the non-zero coefficients corresponding to the dictionary atoms selected in the support. This optimization problem is solved using the optimization process described previously in Section 3.3.1.

The sparse coefficients in $\boldsymbol{x}$ span the entire length of the dictionary. Breaking $\boldsymbol{x}$ into three parts of length L gives the coefficient vectors associated

with each sub-dictionary. For classification, the reconstruction error based on each of the sub-dictionaries is calculated, and the class of the sub-dictionary which results in the lowest error is the class assigned to the test feature vector. The reconstruction error associated with the j^{th} sub-dictionary, $\boldsymbol{D}_j$, is given by:

$$e_j = \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_2^2 = \left\|\boldsymbol{y} - \boldsymbol{D}_j \boldsymbol{c}_j\right\|_2^2 \tag{5.3}$$

where $\boldsymbol{c}_j$ is the vector of coefficients associated with $\boldsymbol{D}_j$.

The approach to dictionary construction and SL presented in this section has previously been used for MI EEG classification[11], [56]. It was chosen due to its strong performance, making it a good candidate for investigating the use of genetic channel selection in conjunction with a SL system. The sparse learning module presented also has some technical differences when compared to similar modules in the literature. Firstly, previous implementations had dictionaries which used multiple features to represent each channel within the feature vectors [11], [56]. For example, one implementation [11] represented each channel with two features, namely the energies in the wavelet detail and approximation coefficients obtained from a DWT decomposition of each channel, and another [56] represented each channel with three features, namely wavelet energy, the alpha and the beta band powers. The difference between the approach presented in this chapter and what has gone before is that each channel is represented using only one feature: the band-power within the joint alpha and beta frequency bandwidth. The alpha and beta bands exhibit salient activity during MI actions [68]. However, the exact frequency sub-bands in which MI-associated activity occurs can vary between individuals [82], as well as between trials for a particular individual [53]. Thus, in the proposed system, the combined alpha and beta bands were used. The work in this chapter also investigates the effect of window size on classification performance, whereas other works used window segmentation sizes of 0.5s [11] or 2s [56] without any mention of parameter tuning. Furthermore, previous works selected arbitrarily (manually) chosen subsets of EEG channels [11], [56], whereas the approach presented in this chapter uses algorithmic channel selection.

5.2.3 Genetic Channel Selection

Genetic channel selection is used to obtain a subset of EEG channels which maximizes the classification accuracy of the SL module. The algorithm is driven to find a subset of EEG channels which provides the greatest classification accuracy, with the overall aim of maintaining or improving the accuracy when compared to the case when the original, full montage of EEG channels is used. Although the accuracy of the selected subset may not exceed that of the original montage [213], the aim is to drive the algorithm to search for the best possible solution.

An exhaustive search through all possible channel combinations is practically impossible; as an example, the BCI Competition IV dataset I [108] used for evaluation in this chapter has 59 EEG channels, and if a subset of 30 EEG channels were to be selected, there would be 5.91×10^{16} possible combinations to consider. A GA was chosen for channel selection because it is especially suitable for dealing with combinatorial problems [193] and has previously been applied with success to other MI EEG channel selection problems [49], [192]. However, GAs are not guaranteed to find the global maximum and can get stuck at local maxima/minima [44]. Despite this shortcoming, they have a track-record of finding acceptable solutions [49], [125], [192], making them a suitable candidate for channel selection in the proposed design. Since this is a subject-specifically trained system, channel subsets are selected for each subject individually.

A high-level overview of GAs was provided in Chapter 3, Section 3.5.2.2. In this chapter, a main contribution lies in the *application* of a GA to the channel selection problem for a SL classifier. Although the general framework of GAs are well established in theory [193], each implementation of a GA has its own particularities which tailor it to the application [49], [51], [192]. Therefore, although the aim of this chapter is not to delve into GA algorithm design or optimization, the GA implementation in this chapter has its own original design aspects. The rest of this section discusses the GA implementation used in this

thesis and is mainly focused on the design details tailoring the GA to the application.

Algorithm 2 summarizes the operation of the GA. The GA is a wrapper method [44], meaning that it optimizes the choice of channel subset based on the classification accuracies obtained for each candidate channel subset with the SL classifier previously described in Section 5.2.2. The dictionary is constructed using the training dataset and the classification accuracies are calculated on the validation set. The GA optimizes the candidate channel subsets with respect to the validation set classification accuracy, and the candidate that had the greatest accuracy is the one applied to the test-set (lines 24- 26 in Algorithm 2). Therefore, the GA channel selection process is only applied at the training stage, with the selected channel subset is immediately deployed at the test stage.

The GA encodes the candidate subsets as a row vector of numbers, with each number being uniquely associated with a particular EEG channel. The length of the vector, n, denotes how many channels will be selected, and is a global variable fixed by the user. These vectors are the chromosomes. On initialization, the GA randomly generates a population of z chromosomes. The size of the population remains constant throughout the running of the algorithm.

Algorithm 2: Genetic Algorithm Based Channel Selection

1: *Inputs*: n, the size of the EEG subset, z the size of the population, $\boldsymbol{D}$ the dictionary based on the training dataset $\boldsymbol{X}_{train}$, $\boldsymbol{X}_{val}$ and $\boldsymbol{X}_{test}$
2: *Initialization*: ***stagnation*** = 0, ***tolerance*** = 0.000009, ***bestFitness*** = 0
3: ***population*** = a set of z randomly generated chromosomes of length n
4: **while** ***stagnation*** < 3: *# Monitor if the **bestFitness** has remained the same for three consecutive iterations*
5: **for** each chromosome in the ***population***: *# Calculate the population fitness*
6: SL classification with ***D***[***population***{current chromosome},:] and $\boldsymbol{X}_{val}$ [***population***{current chromosome},:]
7: Calculate the accuracy and store in ***population_fitness***
8: **end**
9: ***lastBestFitness*** = ***bestFitness*** *# Update the best fitness and the best chromosome*
10: **if** max(***population_fitness***) > ***bestFitness***:
11: ***bestFitness*** = max(***population_fitness***)
12: ***bestChromosome*** = ***population***[argmax(***population_fitness***)]
13: **end**
14: **if** abs(***lastBestFitness*** - ***bestFitness***) < ***tolerance***: *# Monitor for convergence*
15: ***stagnation*** = ***stagnation*** +1
16: **end**
17: ***cumulative_fitness*** = sum(***population_fitness***) *# Carry out selection*
18: Select (z-5) chromosomes to be used in crossover. Probability of selection, ***p*** is calculated in (4).Store in ***parents***.
19: From the ***parent*** chromosomes generate the ***children*** as described in (5) and (6).
20: In the population replace the (z-5) chromosomes with the lowest fitness with the ***children***.
21: Mutation step: 20% probability that a random mutation in one chromosome will occur.
22: **end**
23: Calculate the ***test_set_accuracy*** for the final ***bestChromosome***:
24: Carry out classification using (3) and (5) with ***D***[population{***bestChromosome***},:] and $\mathbf{X}_{test}$ [population{***bestChromosome***},:] and calculate the
25: accuracy
26: Output: ***bestChromosome*** and ***test_set_accuracy***
27: **END**

At the start of each iteration of the GA, the fitness of all the chromosomes in the population is calculated. In this implementation, the fitness is the classification accuracy obtained on the validation set. The GA monitors the current best fitness in the population and the corresponding chromosome. As previously explained in Chapter 3, Section 3.5.2.2, during each iteration the GA updates the population. In this implementation, the 5 individuals in the current population with the greatest fitness are directly promoted to the next generation, leaving $(z-5)$ new individuals to be constructed through selection and crossover. Since a type of two-point crossover (previously explained in 3.5.2.2) is used, $(z-5)$ parents must be selected from the population. Selection is carried out using a fitness proportional roulette wheel approach [193], previously described Section 3.5.2.2.

Crossover is carried out using a kind of two-point crossover approach [193]. In this approach, two chromosomes from the selected group are randomly paired and called $\boldsymbol{parent}_1$and $\boldsymbol{parent}_2$. A random position within each parent chromosome is chosen, and the gene at that position and the one right-adjacent to it are selected for crossover. Crossover merely involves exchanging the gene pairs between parents 1 and 2. This process produces two new chromosomes, named $\boldsymbol{child}_1$ and $\boldsymbol{child}_2$, and can be summarised by (5.4) and (5.5):

$$\boldsymbol{child}_1 = [\boldsymbol{parent}_1[0,\dots,p_1], \boldsymbol{parent}_2[p_2, p_2+1], \ \boldsymbol{parent}_1[p_1+2:end]] \quad (5.4)$$

$$\boldsymbol{child}_2 = [\boldsymbol{parent}_2[0,\dots,p_2], \boldsymbol{parent}_1[p_1, p_1+1], \ \boldsymbol{parent}_2[p_2+2:end]\ ] \quad (5.5)$$

where the parent and child variables are row vectors of length n, p_1 and p_2 represent a random location in $parent_1$ and $parent_2$ respectively, and ($p_1, p_2 \neq n$). Crossover only occurs if it will not result in any gene duplication within the children. This is because a duplicate gene would mean that a particular channel was selected twice within a subset. If gene duplication will occur, crossover is not carried out, and other random locations within the parent chromosomes are selected as crossover points. In this way, crossover is attempted up to 4 times for a particular pairing before the parent chromosomes are deemed too similar for crossover and are returned to the population. In total, crossover must produce

$(z-5)$ child chromosomes. The new population consists of the $(z-5)$ child chromosomes produced during crossover and the five fittest individuals from the previous population.

Mutation, an exploration step in the GA, is then carried out on the new population. In the GA design presented, one chromosome can experience mutation per iteration and there is a 20% chance that this mutation will occur. If the algorithm decides that mutation will occur in the iteration, a chromosome is selected from the new population using a uniform probability distribution. Then, one gene is selected at random within the chromosome and this gene is replaced with a randomly generated channel number which does not result in any gene duplication within the chromosome.

The GA concludes its search when the current best accuracy has stagnated for three consecutive iterations. Stagnation is considered to have occurred in an iteration when the improvement in accuracy is less than 0.0009% when compared to the previous iteration.

The classification accuracy on the test set, which is the classification accuracy of the GABSLEEG algorithm, can then be obtained. The best chromosome found by the GA is used to select the EEG channels from the dictionary and the test-set, and then these are input to the SL classification module to obtain the test-set classification accuracy.

5.3 Experimental Methodology and Hyperparameter Tuning

This section first introduces the datasets used, and then goes on to summarize the evaluation methodology. It then discusses hyperparameter tuning and design decisions made for the GABSLEEG system and benchmarking systems.

5.3.1 Datasets

Two datasets were used in this study, the BCI Competition IV, dataset I [108] and the BCI Competition III dataset IVa [86], which were previously described in Chapter 2, Section 2.2.

The BCI Competition IV dataset I was used because it specifically included idle state data. The calibration dataset was used for hyperparameter tuning in Section 5.3.5, and then the evaluation dataset was used to obtain the results in Section 5.4.

The BCI Competition III, dataset IVa was used to assess the performance of the GABSLEEG system on another dataset, and for comparing the GABSLEEG system to other implementations due to the popularity of this dataset in the literature [11], [12], [15], [40], [129], [214].

Since both datasets had 2 MI classes and the idle state, the work in this chapter is focused on three-class classification problems.

5.3.2 Evaluation Methodology

This section discusses the general evaluation approach, then summarizes the systems used for benchmarking or comparison, and the performance metrics used.

5.3.2.1 General Evaluation Approach

Subject-specific training was carried out, meaning that the GABSLEEG system was trained from scratch for each individual subject in a dataset [3]. The data for each subject was divided into training, validation, and test sets, comprising of 80%, 10% and 10% of the total data, respectively. This data division was carried out once for each subject, with a single set of performance metrics being obtained for each experiment involving the GABSLEEG system. The performance metrics were then averaged across subjects to obtain a general overview of the system's performance. The same data divisions were used for the benchmarking and comparison systems. This method of data division has been used in other studies,

for example in the work of Chaudhary et al. [12] which involved a CNN-based classifier which could have long training times. In fact, this approach was adopted due to the long execution time of the GA module.

5.3.2.2 Systems used for Benchmarking and Comparison

This section discusses the benchmarking classifiers and comparison systems used to assess the effectiveness of the SL classifier and the GABSLEEG system. First the benchmarking classifiers are introduced, then the state-of-the-art comparison systems are discussed.

Benchmarking Classifiers

The benchmarking classifiers are the k-NN, RF and SVM-RBF classifiers. These classifiers were chosen due to their widespread use in the literature [13], [14], [21], [22], [31], [101], [104], [140], [148], and because they have been used as part of novel classification systems [13], [14], [21], [22], [31], [101], [104], [118], [140], [148], [187], [215]. In Section 5.4.1 the classification performance of the SL classifier was first compared to the performance of the three benchmarking classifiers. After, the performance of the full GABSLEEG classification system was compared to GA-kNN, GA-RF and GA-SVM classifiers. Results for the benchmarking systems were generated by replacing the SL classifier in the GABSLEEG framework with a benchmarking classifier. In Section 5.4.3, the execution time of the GABSLEEG system is compared to those of the benchmarking classification systems.

State-of-the-Art Comparison Systems

The whole GABSLEEG system was compared to two state-of-the-art classification approaches: the SL classification pipeline presented by Sreeja et al. [11] and a deep learning classifier named EEGNet [7]. The work of Sreeja et al. was discussed at length in Chapter 3 (Section 3.3.1), and EEGNet was introduced in Chapter 3 (Section 3.4.1.3). Code scripts for these implementations were obtained from [216] and [217], respectively. The effectiveness of the GA channel selection module was also compared to Fisher score channel selection, which is a commonly used channel selection method in the literature, particularly for

comparison to new channel selection approaches [30], [52]. These comparisons to the literature can be found in Section 5.4.5.2. The rest of this section discusses the experimental settings for each comparison system.

SL Classifier Comparison System: The SL approach by Sreeja et al. [11] is a state-of-the art sparse learning system which is similar to the approach used in this chapter, with a dictionary constructed of features derived from EEG data. However, Sreeja et al. [11] used 30 arbitrarily chosen EEG channels, the features extracted were wavelet features, and the authors used an arbitrary window size of 0.5s for dictionary construction. Later in this section the window size of the GABSLEEG system is tuned to be 50ms. To ensure a thorough comparison, results for the implementation by Sreeja et al. [11] and the GABSLEEG approach were generated for window sizes of 0.5s and 50ms. This approach, published in 2020, is the best-performing sparse learning system in the literature reviewed

EEGNet Comparison System: EEGNet [7] is a benchmark CNN-based MI EEG classification system, which has been used for comparison to the state-of-the-art [59], [164], [218]. It uses all the EEG channels within the dataset. In the original study, EEG data was broken down into 2s segments, and due to the structure of the CNN used, this could not be reduced to 50ms as in this study. Thus, when comparing to EEGNet, the data was segmented using 2s windows.

Fisher Score Channel Selection: For comparison, the GA channel selection module in the proposed pipeline was replaced with a Fisher score channel selection algorithm. This approach has been used in the state-of-the-art, in the works of Park et al. [46] and Sadiq et al. [30]. In the Fisher score channel selection approach, each channel, h, is assigned a Fisher score, F_h, using the equation [219]:

$$F_h=\frac{\sum_{k=1}^{3}\left(m_{k,h}-m_{h_{total}}\right)^2}{\sum_{k=1}^{3} v_{k,h}} \tag{5.6}$$

where $m_{k,h}$ and $v_{k,h}$ are the mean and variance of the features extracted from channel h for class k *(k=1,2,3)*. $m_{h_{total}}$ is the mean of all the features extracted from channel h. The channels are then ranked based on this score, with higher

scores associated with greater importance. In this analysis, the 30 channels with the highest scores were selected. Since the Fisher score does not require a validation set, the training and validation datasets were combined for the calculation of the Fisher score.

5.3.2.3 Performance Measures

Accuracy, sensitivity, and specificity were used to assess the performance of the systems. The equations for these measures were previously included in Chapter 4, Section 4.3.3.1. As in Chapter 4, the performance metrics were calculated for each individual subject. However, the classification problem in this chapter is a multi-class problem, whereas the problem in Chapter 4 was a binary problem. In this chapter, the accuracy was calculated as the ratio of the number of correctly classified samples to the number of in correctly classified samples. Sensitivity and specificity, however, are class specific. Therefore, the confusion matrix values TP, FP, TN, and FN were calculated separately for each individual class, using a one-vs-rest approach to the calculation.

5.3.2.4 Training Data Size Analysis Methodology

The performances of the GA-based systems, namely the GABSLEEG, GA-kNN, GA-SVM and GA-RF classifiers, with reduced training data size were analyzed. This experiment involved reducing the training data from 100% to 20% in steps of 20% and calculating the accuracy, sensitivity, and specificity at each stage. This analysis is important because the recording of training data is time consuming and can decrease the practicality of BCIs using subject-specific training since the recording of training data increases latency.

5.3.4 Execution Time Analysis Methodology

Execution time is an important aspect of performance to assess because it can have a significant effect on user experience of a BCI. The training times in subject-

specifically trained systems such as those in this chapter represent the waiting time between recording the training data and the user being able to use the BCI. Testing times represent the latency a user would experience during 'real-time' use of the BCI algorithm. Therefore, it is important to identify systems with shorter training and test times, or to identify pipelines with high accuracies but which have long execution times to highlight areas for improvement.

An execution time analysis was carried out in which the training and testing times of the GABSLEEG, GA-kNN, GA-SVM and GA-RF systems were compared. All tests were carried out on a Lenovo™ ideapad 330 laptop using a 64-bit Windows 10 operating system and an Intel® Core™ i5- 8300H, 2.30GHz CPU. Prior to the experiments, all non-essential background processes were suspended to ensure more accurate results.

The total training time involved the GA channel selection process, and the training of the classifier in the k-NN, SVM and RF pipelines, or structuring of the dictionary for the SL classifier. In this work, the total testing time was the time taken for the classifier to assign all the classification labels to the test-set feature vectors. To obtain the average execution times pre-segment for each subject, the total training time and total test time were recorded, and then divided by the number of segments included in the training and test sets, respectively. In this way, the training and testing times per segment were obtained. The execution times were recorded for each subject individually, and then the average times, calculated across the subjects, were considered for discussion.

5.3.5 Hyperparameter Tuning for the GABSLEEG and Benchmarking Classifiers

As previously mentioned, hyperparameter tuning was carried out using the calibration sub-dataset in the BCI competition IV Dataset I [108]. This section first discusses the tuning of the GABSLEEG system and then that of the benchmarking

classifiers. For the GABSLEEG system, a discussion of the effects of different parameters on classification accuracy is also included.

5.3.5.1 Hyperparameter Tuning and Analysis for the GABSLEEG System

The GABSLEEG system had two distinct modules which require tuning: the SL module and the GA module. The SL module had two tunable parameters: the window size used for segmentation and the number of non-zero coefficients in the representation. The GA also had two tunable parameters: the initial population size and the number of channels in the subset. The average accuracy across subjects was used to decide which parameters to use, which is the same approach used in Chapter 4 for the classifiers used for multi-segment decision fusion. Since the GA is a wrapper-based channel selection method designed with the aim of preserving the accuracy of the SL module, the SL classifier was tuned, followed by the GA module.

The SL module shown in Figure 5.2 was tuned using a grid-search which spanned window sizes of {50ms, 60ms, 70ms, 80ms, 90ms, 100ms, 150ms, 200ms} and non-zero coefficients in the set {3, 4, 5, 6, 7}. The 10-fold cross-validation accuracy for each subject and parameter pairing was calculated, then the results were averaged across subjects. Figure 5.4 shows the grid-search results for the SL classifier. A peak accuracy of 99.07% was obtained on the calibration dataset using a window size of 50ms and six non-zero coefficients, highlighted with a red marker in Figure 5.4. These parameters were then used in the calibration of the GA module.

To confirm that using the combined alpha-beta band power was the best design choice, the SL classifier grid-search was carried out again, but this time characterizing each channel in the feature vector using the separate alpha and beta band powers. Peak average accuracy was 98.97%, obtained for a window size of 50ms and five non-zero coefficients. This was slightly lower than the accuracy obtained when using the combined alpha-beta band power features and confirmed that this was a suitable design choice.

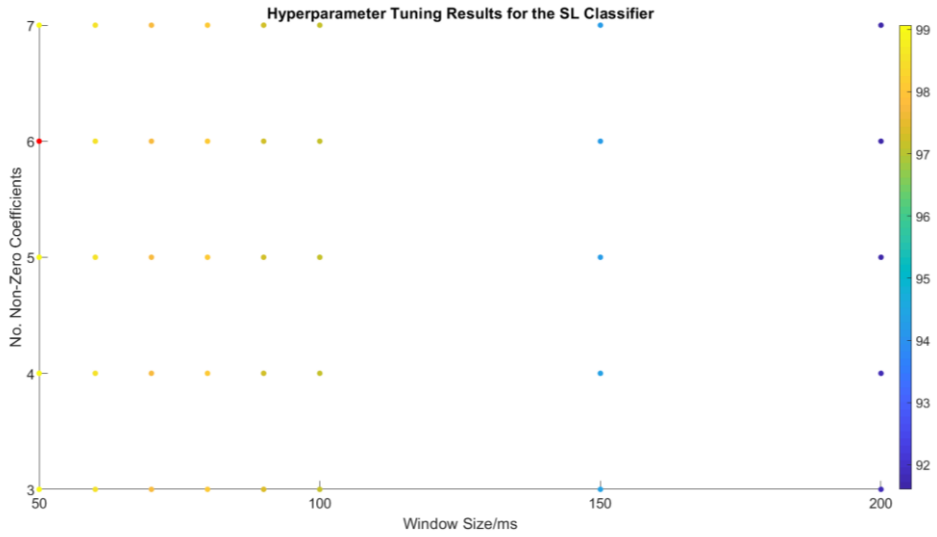

Figure 5.4: Results for grid-search hyperparameter tuning for the SL classifier.

A grid-search was also used to tune the GA module, covering initial population sizes in the set {10, 15, 20, 25, 30, 35} and the channel subset sizes in the set {5, 10, 15, 20, 25, 30}. A maximum of 30 channels was chosen to observe whether, using the GABSLEEG approach, better performance could be obtained when compared to the approach of Sreeja et al. [11], which used 30 arbitrarily chosen channels. This comparison is included in Section 5.4.5.2. Due to the relatively long execution time taken for the GA search, the calibration data for each subject was divided into training, validation, and test sets just once, the test-set accuracy for each parameter pairing was calculated, then the results were averaged across subjects. 80% of the data was used for training, 10% for validation, and 10% for testing.

Table 5.1: Average calibration test-set accuracy of the GABSLEEG system with different sizes of channel subsets. Channel subset size refers to the number of channels in each subset.

Channel Subset Size	Average Accuracy
5	$61.42 \pm (9.6\times10^{-3})$
10	$95.29 \pm (5.5\times10^{-3})$
15	$97.28 \pm (2.4\times10^{-3})$
20	$97.93 \pm (1.3\times10^{-3})$
25	$98.28 \pm (5.5\times10^{-4})$
30	$\mathbf{98.44 \pm (4.3\times10^{-4})}$

Table 5.1 shows the average accuracy results for each of the channel subset sizes. The average accuracy for each channel subset size was calculated by averaging the grid-search results across all the initial population sizes (10-35) for that subset size. The results are recorded as *average accuracy* $\pm$ *standard deviation.* The standard deviation therefore captures the variability in accuracy between the different population sizes for a particular channel subset size. The standard deviation is always less than 1%, which is relatively low compared to the accuracy values. It is evident that increasing the channel subset size increased the average accuracy, although the greatest improvement was when increasing the number of channels in the subset from 5 to 10, with the effect plateauing as the subset size is increased linearly in steps of five. Peak accuracy was obtained when a channel subset size of 30 was used. As the subset size increased, the standard deviation tended to decrease, indicating that a larger number of channels in the subset reduced the impact of the initial population size.

An overall peak test-set accuracy of 98.49% was obtained for an initial population size of 35 and 30 channels in the subset. Using the same train and test splits and only the SL module with all 59 EEG channels, a test-set accuracy of 98.62% was obtained. Thus, reducing the number of EEG channels from 59 to 30 resulted in a decrease in accuracy of just 0.13%. In further analysis involving the GABSLEEG system, an initial population size of 35 and a subset of 30 EEG channels were used. However, considering the results in Table 5.1, reducing the number of EEG channels in the subset from 30 to 10 resulted in just a 3.15% decrease in average accuracy. Due to this relatively small decrease in accuracy, and because a core aim of this chapter was to reduce the number of EEG channels used to improve execution times on the test-set, in the execution time analysis in Section 5.4.3, the case when a subset of 10 EEG channels is used is also analyzed, to investigate the trade-off between execution time and accuracy.

Finally, the effect of sparsity level on test set fitness (i.e. classification accuracy) of the GA was observed. Sparsity level is determined by the number of non-zero coefficients used in the OMP encoding of the test set feature vector, with fewer non-zero coefficients indicating increased sparsity. In this analysis, the

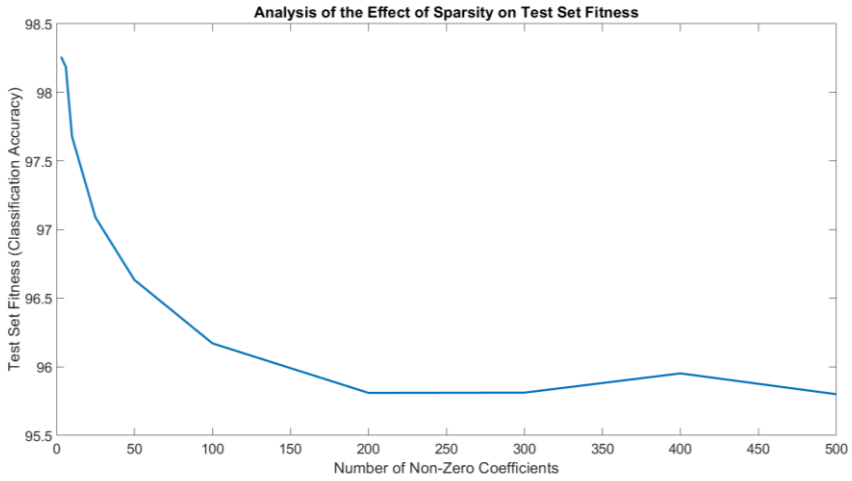

Figure 5.5: Analysing the changes in test set fitness for changes in the number of non-zero coefficients used in the OMP reconstruction. A smaller number of non-zero coefficients indicate increased sparsity.

number of non-zero coefficients used in the encoding was varied from 6, which gave the maximum average accuracy during tuning, to 500, and the test set accuracy was recorded for each case and plotted in Figure 5.5. Increased sparsity was related to higher accuracy, with the accuracy being over 98% when 25 or fewer non-zero coefficients were used. When 200 or more non-zero coefficients were used, the classification accuracy tended to fluctuate around 95.75%. In all further analysis in this paper, 6 non-zero coefficients were used.

5.3.3.2 Hyperparameter Tuning for the Benchmarking Classifiers

The k-NN, SVM and RF classifiers were tuned using Bayesian optimization [90], which facilitated exploration of the vast parameter spaces of these classifiers. Like the methodology for the SL classifier, the average 10-fold cross-validation classification accuracy calculated over the subjects was used to choose the best hyperparameters. To carry out Bayesian optimization the Python function *skopt.BayesSearchCV* from the *scikit-optimize* toolbox was used.

The k-NN classifier was tuned for the *k* parameter, which determines the number of nearest neighbours considered when delineating decision boundaries.

Values of k in the range 3-20 were considered, and a peak accuracy of 93.32% was obtained for a k of 3. For the SVM classifier, the C and Υ parameters were tuned considering values in the ranges 0.1 to 100 and 0.01 to 10, respectively. A peak accuracy of 84.99% was obtained for a C value of 100 and a Υ value of 10. For the RF classifier, the number of trees was tuned for the range 10-500, the predictions at each node were tuned in the range 2-20 and the number of observations per leaf were tuned for values in the range 1-10. A peak classification accuracy of 86.27% was obtained for 500 trees, minimum predictions at each node of 2 and 1 observation per leaf. For completeness, the hyperparameter tuning results for the k-NN, SVM and RF classifiers are shown in Table 5.2. The parameters that gave peak accuracy are in bold.

Table 5.2: The hyperparameter tuning results for the k-NN, SVM and RF classifiers.

Classifier	*Hyperparameter Tuning Parameters*	*Accuracy (%)*
k-NN	***k***	
	12	81.65
	7	87.53
	12	81.64
	6	87.80
	7	87.53
	17	77.64
	19	76.23
	12	81.64
	5	90.22
	9	85.24
	3	**93.00**
	3	93.00
	3	93.00
SVM	***C*/** ϒ	
	100/10	**84.99**
	100/1	82.22
	100/0.1	71.53
	100/0.01	55.35
	10/10	83.85
	10/1	78.10
	10/0.1	63.43
	10/0.01	46.67
	1/10	80.02
	1/1	71.80
	1/0.1	55.41
	1/0.01	42.10
	0.1/10	65.70
RF	**Number of trees /observations per leaf/predictions at each node**	
	270/7/13	82.24
	137/10/18	80.48
	269/2/8	85.34
	82/8/10	81.40
	124/3/15	83.50
	421/10/15	80.39
	484/8/10	81.70
	260/6/15	82.38
	55/9/16	80.67
	191/5/7	83.66
	500/1/2	86.72
	19/1/2	85.44
	500/1/2	**86.77**
	500/1/2	86.66

5.4 Results and Discussion

This section first analyses the impact of the GA channel selection module on classification accuracy. Afterwards, the performance of the GABSLEEG system was compared to that of the benchmarking classifiers. This comparison is then expanded by comparing the performance of the GABSLEEG system and benchmarking classifiers with reduced training data size and by conducting a execution time analysis. The channels most frequently selected by the GA are then analysed. Finally, the GABSLEEG system is compared to other implementations in the literature, including Fisher channel selection [30], [52], the SL classifier by Sreeja et al. [11] and EEGNet [7].

5.4.1 Classification of Motor Imagery and the Idle State

The effectiveness of the SL and the GA modules were assessed separately using the evaluation sub-dataset of the BCI competition IV Dataset I [108]. First, the performance of the SL classification module ('SLEEG') was compared to the benchmarking classifiers, with results recorded in Table 5.3. The results in this table are called the 'control' test set accuracies since these results were obtained without the GA channel selection module and when using all 59 channels in the EEG dataset. The highest accuracies are highlighted in bold and indicate that the SL classifier outperformed the benchmarking classifiers. The k-NN classifier was the best performing benchmarking classifier whilst the RF classifier was the poorest.

The effectiveness of the GA module was then assessed in Table 5.4, which contains the results for the GABSLEEG system and the benchmarking classifiers with the GA channel selection module. Again, the GABSLEEG system had the strongest performance, whilst the GA-kNN classifier was the best performing of the benchmarking methods and the GA-RF method had lowest accuracy.

The effect of the GA module on classification accuracy can be observed by comparing the results in Table 5.3 and Table 5.4. GA channel selection resulted in a depreciation in average accuracy for the SL, k-NN and RF classifiers, which amounts to decreases of 0.19%, 0.50% and 0.91%, respectively. However, the GA module increased the accuracy of the SVM classifier by 1.38%. These conflicting results agree with the literature, with some studies reporting a slight decrease in accuracy with the introduction of channel selection [213], and others reporting an increase in classification accuracy [48], [220]. These results may indicate that the effect of the GA channel selection module on accuracy depends on the classifier used. Furthermore, the decrease in accuracy was always less than 1%, which was deemed to meet the aim of maintenance of accuracy. Of the classifiers which experienced a decrease in accuracy, the GABSLEEG classifier experienced the lowest decrease in accuracy.

Recall that in this chapter the main aim of channel selection is to produce a subset of EEG channels that maintains the classification accuracy so that the test-set execution time can be reduced. The results in Table 5.3 and Table 5.4 confirm that the GA is capable of selecting a subset of channels that preserves the accuracy performance for the SL classifier, as well as for three conventional classifiers.

Table 5.3: The control test accuracies (%) for the proposed SL classifier and three benchmarking classifiers. The results of the best performing system are in bold.

Systems	1a	1b	1f	1g	Average
SLEEG	**99.89**	**99.70**	**99.86**	**99.92**	**99.84**
SVM	94.49	93.80	96.80	93.66	94.69
k-NN	98.05	94.89	99.54	97.55	97.51
RF	89.94	85.39	95.57	90.74	90.40

Table 5.4: The test accuracy (%) for the proposed GABSLEEG system and three benchmarking systems. The results of the best performing system are in bold.

Systems	1a	1b	1f	1g	Average
GABSLEEG	**99.52**	**99.62**	**99.77**	**99.69**	**99.65**
GA-SVM	96.89	94.73	98.54	94.13	96.07
GA-kNN	97.63	94.76	99.03	96.63	97.01
GA-RF	89.60	85.78	96.25	89.09	89.49

Table 5.5 contains results for sensitivity and specificity for each class. For all classifiers the performance was relatively stable across classes, particularly for the GABSLEEG system. Again, the GABSLEEG system had the highest results, followed by the GA-kNN classifier. As expected from the accuracy results in Table 5.4, the GA-RF classifier had the lowest performance.

These results indicated the GABSLEEG system had the strongest performance in terms of accuracy, sensitivity and specificity when compared to the three benchmarking systems. Furthermore, the GA module was found to be effective at maintaining the classification accuracy of various classifiers. Later, in the execution time analysis in Section 5.4.3, the computational benefits of using the channel subsets is confirmed.

5.4.2 Performance with Reduced Training Data Size

Figure 5.6 shows how the accuracy, sensitivity, and specificity of the GA-based systems vary with decreasing training data size. The sensitivity and specificity results shown were obtained by averaging across the classes. For all systems, the performance tended to decrease with decreased training data, and the accuracy plots had a noticeable knee when training data was reduced from 40% to 20%. The GABSLEEG system had the best performance across the metrics for all training data sizes. The GA-RF pipeline consistently had the poorest performance, whilst the GA-kNN and GA-SVM systems had mid-range performance. Most notably, when only 20% of the training data was used, only the GABSLEEG system exhibited over 90% accuracy, sensitivity, or specificity.

Table 5.5: Comparison of the test-set sensitivity and specificity associated with each class (%), averaged across subjects, for the proposed GABSLEEG System and three benchmarking systems. The results of the best performing system are in bold.

	Sensitivity			*Specificity*		
Systems	*MI Class 1*	*MI Class 2*	*Idle Class*	*MI Class 1*	*MI Class 2*	*Idle Class*
GABSLEEG	**99.92**	**99.86**	**99.44**	**99.81**	**99.77**	**99.96**
GA-SVM	94.31	92.21	99.01	99.56	99.70	93.39
GA-kNN	99.13	98.49	95.14	98.13	98.39	99.23
GA-RF	93.58	91.83	86.20	94.32	94.32	96.00

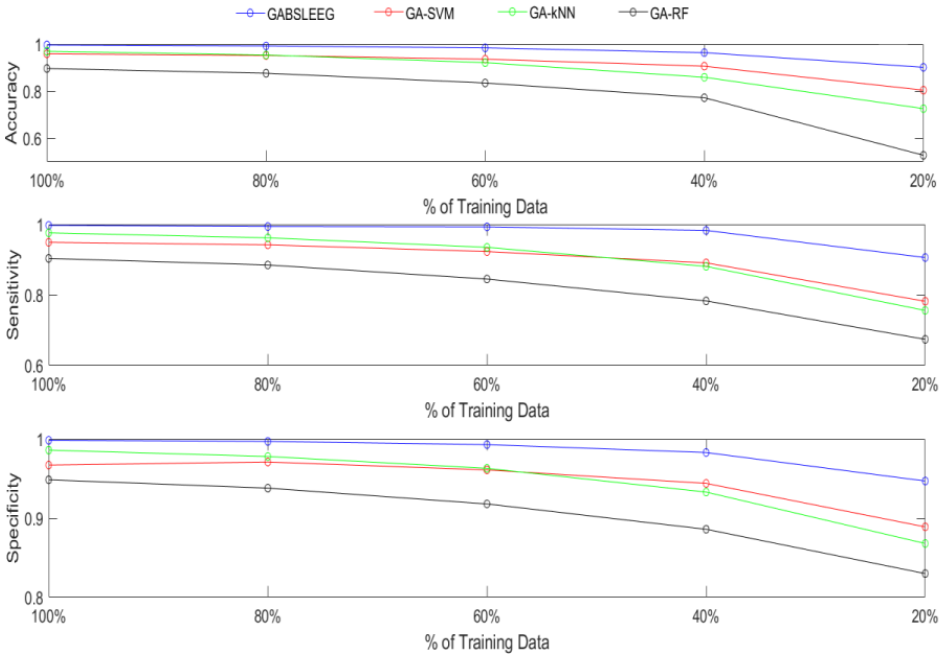

Figure 5.6: Comparing decrease the in accuracy, sensitivity and specificity for reduced training data size for the GABSLEEG (blue), GA-SVM (red), GA-kNN (green) and GA-RF (black) classifiers.

When using 100% of the data, there is 24.20±3.35 minutes of training data per subject, on average. When using 20% of the data, an average of 4.98±0.83 minutes of training data were used per subject. The less training data that is required, the lower the latency experienced by the user, and the more practical the system is. The performance of the GABSLEEG system is robust to changes in training data, indicating its practicality. When comparing to practical EEG-based BCIs for device control in the literature, the amount of training data used throughout this section could be considered reasonable: for the deep-learning system in [221], an average of 24 minutes of training data per subject was used, for the SVM-based classifier in [222] 10 minutes of training data were used, and for the single hidden layer RBF network in [223], 4 minutes were used. These results from the literature indicate that, in terms of training data requirements, the GABSLEEG system is comparable.

These results indicated that the GABSLEEG system was robust to decreases in training data size and could be a suitable candidate for practical MI EEG classification systems with subject-specific training since, ideally, such systems use the minimum possible amount of training data.

5.4.3 Execution Time Analysis

The results for training and testing times per segment are shown in Table 5.6. Both the average times and the worst case (longest) times were recorded. When considering the training times, the GA-kNN was the fastest, followed by the GA-RF classifier and then the GABSLEEG system. The GA-SVM classifier was the slowest to train. When considering the testing execution times, which are of particular interest, the GA-kNN and GA-RF classifiers are the fastest, followed by the GABSLEEG system and then the GA-SVM.

When using a graphical user interface, humans can only perceive visual latencies of approximately 13ms [45]. Since the test latency of the GABSLEEG system was, on average, 13.1ms, it could possibly be suitable for seamless control of a BCI, particularly if a computer with a faster CPU or alternative streamlined hardware implementation were used, such as graphical processing unit (GPU) or field-programmable gate array (FPGA). The GA-SVM pipeline had a test latency of 36ms, which was over twice the acceptable latency for a visual interface, possibly making it unsuitable for a real-time BCI unless more powerful hardware is used, or if the SVM classifier were adapted to run on a GPU.

The execution time analysis justifies the use of the GA channel selection module. The SL EEG classifier using all 59 EEG channels has an average testing time per segment of 32.39ms, which is substantially greater than 13ms. Using the subset of 30 EEG channels selected by the GA module, this testing time was reduced to 13.1ms, a decrease of 60%. This result illustrates the value of using a reduced channel subset for classification. The GA module is particularly effective

Table 5.6: Comparing the averaged and worst case training and testing times per segment for the GABSLEEG and benchmarking systems. The results of the best performing system are bold.

Systems	Average Training Time/ms	Average Testing Time/ms	Worst case Training Time/ms	Worst Case Testing Time/ms
GABSLEEG	199.0±10.7	13.1±0.7	208.5	13.9
GA-SVM	557.0±49.1	36.2±2.0	601.8	39.2
GA-kNN	**4.0±0.3**	**0.3±0.0**	**4.0**	**0.3**
GA-RF	5.2±1.8	**0.3±0.0**	7.8	**0.3**

because it managed to find channel subsets that maintain the classification accuracy within a 1% tolerance, meaning that the substantial computational benefits of using a reduced channel subset do not come at the cost of a substantial decrease in accuracy. The improvement observed in the testing time, however, comes at the expense of the training time of the GA channel selection module.

5.4.3.1 Further Decreasing the Channel Subset Size: The Accuracy-Execution Time Trade-Off

In the GABSLEEG system, there is a trade-off between classification accuracy and execution time, with the number of channels in the EEG subset playing a key role in controlling this trade-off. In all experiments so far in this results and discussion section, the GA has selected a subset of 30 EEG channels. However, in the hyperparameter tuning process in Section 5.3.5.1 (Table 5.1) it was observed that a subset of just 10 EEG channels gave an accuracy over 95.29% for the GABSLEEG system on the calibration dataset. This was a relatively small decrease when compared to using 30 channels, which gave an accuracy of 98.44% with the calibration set (see Table 5.1).

If a subset of 10 EEG channels were to be selected in the GABSLEEG pipeline on evaluatio2n dataset, an average classification accuracy of 96.12% was obtained, which is a 3.65% decrease in accuracy compared to when 30 channels were used (99.65%, as per Table 5.4). However, when using 10 EEG channels, the average training time was decreased to 70.3ms and the average testing time was decreased to 6.6ms, which is a substantial improvement on the 199ms and 13.1ms training and testing times in Table 5.6, when 30 channels were used. These results encapsulate the trade-off between classification accuracy and execution time, in which reducing the number of channels can substantially improve execution time but at the cost of a decrease in accuracy. The exploitation of this trade-off depends on the application, which will determine the required speed of response of the BCI, the number of electrodes desired, and the minimum acceptable classification accuracy.

The results from this execution time analysis indicate that the GABSLEEG system has some potential for being used in a real-time scenario. Using a more powerful machine or programming the BCI on an FPGA or GPU [224] could possibly narrow the computational performance gap between the GA-kNN and GABSLEEG system to provide a means of deploying high performance EEG classification in real time.

5.4.4 Discussion about the Selected EEG Channels

The aim of this analysis is to observe which channels were most selected across all the subjects using GA-based systems, and to discuss how these could be related to underlying mental processes. This analysis therefore covered 16 channel subsets, obtained from the four subjects for the GABSLEEG, GA-kNN, GA-SVM, and GA-RF systems. Figure 5.7 is a graphical summary of how often each channel was selected. Note that there are 11 scalp regions in the recording montage, namely the: central (C), central-frontal-central (CFC), central-central-parietal (CCP), frontal-central (FC), central-parietal (CP), frontal (F), parietal (P), parietal-occipital (PO), temporal (T), occipital (O), and anterior-frontal (AF) regions. The discussion is focused on channels which were selected at least 50%

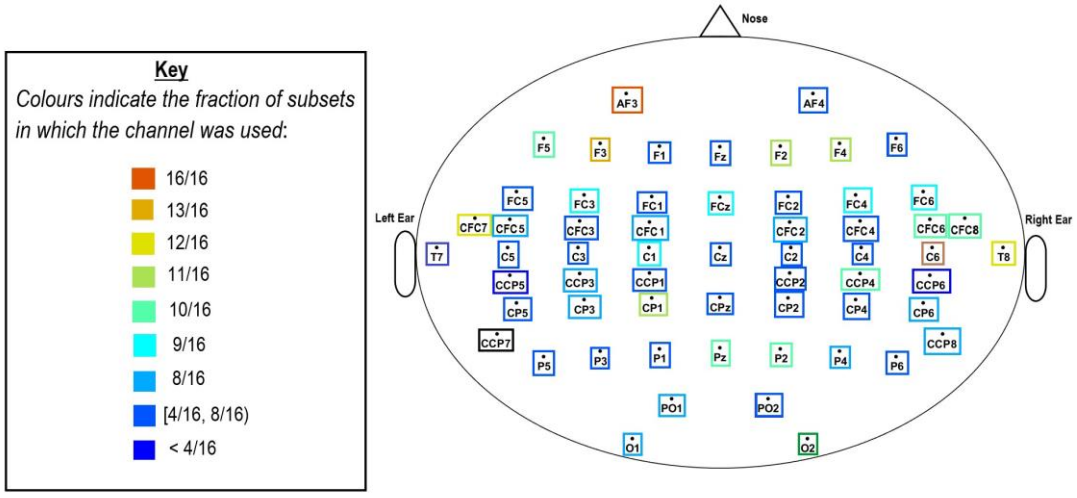

Figure 5.7: Electrode map of the 59 EEG channels in the montage, with the frequency of selections highlighted. The fractions denoting the frequency of selection were calculated across 16 channel selection events – one for each of the four subjects and for the four GA-based classifiers: GABSLEEG, GA-kNN, GA-SVM and GA-RF.

of the time, so in 8 or more of the 16 subsets, since these were less likely to be due to random choice.

The central-associated regions C, CFC and CCP accounted for 58.33% of the selected channels. This makes sense since the central region is strongly associated with MI EEG activity, and these regions tend to be prominent in algorithmically selected channel subsets [34], [48], [225]. This result also loosely links back to the work in Chapter 4 which indicated the synergistic importance of the CFC and CCP regions.

However, nearly half of the most selected channels were outside of the central and central-associated regions. This coincided with various central-associated channels such as CCP6, CCP5, C6, and CCP2 being infrequently selected. Although channels outside the central region do tend to be included within channel subsets[48], [105], [225], central-associated channels are known to dominate in algorithmically selected channel subsets[48], [225]. Nearly half of the most selected EEG channels are not central-associated, and since this is not due to a lack of central-associated channels left to select, these results suggest that non-central channels could be important for classification. For example, the occipital-associated channels PO1 and O1 both featured in 8/16 channel subsets. The occipital region has been associated with MI activity, with ERD being observed during MI, particularly when subjects visualize during the imagined movement [226], [227]. Purely parietal (P) and frontal (F) electrodes each made up 19% of the most selected electrodes. The parietal region is associated with concentration [228] and the frontal region is associated with planning motor movements [35], thus electrodes in these regions could, plausibly, have been associated with distinction between the idle state and MI. The T8 electrode also features in 12/16 subsets. The temporal regions are associated with responses to audial stimuli [37], and this channel may have been selected because subjects were instructed though an audio cue which MI movement to carry out [2].

The channel subsets selected can be related to possible underlying neurological processes, however the relevance of the channels in the subsets is limited by the nature of the GA. Firstly, a classic limitation of GAs is their risk of

getting trapped in local minima or maxima, meaning that the final subsets may not be globally optimal [193]. Furthermore, the GA introduces exploration to the algorithm through randomization in two ways: i) the subsets in the initial population are randomly generated, and ii) the mutation step also introduces random changes to the subsets. Although this exploration feature can help the GA get closer to the global maximum accuracy, it could also introduce randomly selected channels into the subsets, which have little or no contribution to the final classification label. To mitigate for this effect the discussion was limited to channels which were selected in most subsets, however the limitations of the discussion are still acknowledged.

5.4.5 Comparison to the Literature

This comparison to the literature has two distinct parts: 'General Comparison' and 'Comparison using Implemented Systems'. The first part compares the results of the GABSLEEG system to those reported in the literature for state-of-the-art. The BCI Competition III, dataset IVa [86] was used for this comparison due to its popularity in the literature. Although other studies have carried out similar comparisons by simply reporting results previously published in other studies and comparing them to their own [9], due to the methodological differences between studies in the literature, this comparison is limited in value. Thus, the second part of this section compares the performance of the GABSLEEG system to some state-of-the-art implementations from the literature, namely the SL classifier by Sreeja et al. [11], a SL classifier with Fisher score channel selection, and the CNN classifier EEGNet [7]. These were previously introduced in Section 5.3.2.2. To assess the generalizability of the GABSLEEG system, comparisons in this section were carried out using both the BCI Competition IV dataset I [108] which has been used in the earlier results sections, as well the BCI Competition III, dataset IVa [86].

5.4.5.1 General Comparison

For this analysis, two-class classification was carried out using only the MI EEG data, since leading works in the literature tend to use only the MI classes for evaluation [11], [12], [15], [22], [40], [46], [48], [105], [106], [129], [214], [229]. To assess the generalizability of the results previously tuned on the BCI Competition IV, dataset I [108], the same hyperparameters were used in this section. Thus, from the 118 EEG channels available in the BCI Competition III dataset IVa [86], the GA module chose a subset of 30 channels. This analysis was also beneficial because it showed how the GABSLEEG classifier performs with a two-class problem.

The results are recorded in Table 5.7, which summarizes the category of the approach, the features, classifiers, and numbers of channels used in each study. A variety of approaches were chosen, including implementations using all the available EEG channels [12], [15], [106], [229], those using arbitrarily chosen subsets [11], [40] and those with automated channel selection [46], [48], [105].

Table 5.7: Comparing the results obtained within this paper and state-of-the-art SL, conventional, channel selection and deep learning methods. The results of the best performing system are in bold.

Papers, Year	Category	Features	Classifiers	Channels	Average Accuracy
GABSLEEG		Band power	Dictionary-based	30	98.74
[11], 2020	Sparse Learning	Discrete wavelet transform	Dictionary-based	30[1]	97.98
[40], 2019		TQWT features	Least Squares - SVM	5[1]	96.89
[15], 2018		CSP, Fisher discriminant structured dictionary	ELM	118	80.68
[229], 2015	Conventional Approaches	Spatial and spectral features with maximized mutual information	SVM-RBF	118	90.70
[214] , 2017		Analytic intrinsic mode function features	Least Squares-SVM	10[1]	97.56
[22], 2017		Multiscale principal component analysis de-noising, wavelet packet decomposition features	*k*-NN	3[1]	94.50
[129], 2019	Deep Learning	Blind source separation, continuous wavelet features	CNN	18[1]	94.66
[106], 2017		CSP features and LDA feature scoring	Autoencoders & softmax	118	90.70
[12], 2019		Time-frequency representation	AlexNet with transfer learning	118	**99.35**
[49], 2013	Automated Channel Selection	CSP with Rayleigh coefficient maximization based genetic algorithm channel selection	Fisher's LDA	15.5[2]	88.20
[48], 2016		CSP and selected floating forward channel selection	SVM-RBF	30.8[2]	83.30
[46], 2020		Filter-bank CSP and correlation coefficient channel selection	SVM	8.2[2]	88.62

[1]Channels selected arbitrarily, not with channel selection.

[2] Subject-specific channel subset sizes were used, results show the average subset length.

Comparison systems based on SL [11], [15], [40], conventional machine learning approaches [22], [214], [229], and deep learning [12], [106], [129] are included. The GABSLEEG system performed better than all the state-of-the-art SL and conventional classification methods, as well as two of the deep learning systems. The only approach which outperformed the GABSLEEG approach [12] did not perform channel selection, and in theory had all 118 EEG channels available for classification. It also may have benefitted from using the pre-trained AlexNet DL classifier.

Consider the results for systems with automated channel selection [46], [48], [105]. These approaches adapted the number of channels within the EEG channel subset according to the subject and Table 5.7 records the average number of electrodes selected, calculated across the subjects. Although the channel selection approaches in [105] and [46] performed worse than the GABSLEEG system, on average they used substantially fewer channels than the GABSLEEG system and this may have been a contributing factor. The GABSLEEG approach fixed the number of EEG channels in the EEG subset at 30 for all subjects, and its strong performance indicates that this was an effective approach which may improve classification accuracy compared to systems with variable numbers of EEG channels in the subsets.

5.4.5.2 Comparison to the Implemented Systems

This comparison was carried out using both datasets, and the idle state was always included as a class. The inclusion of the idle state is justified for these particular systems because Sreeja et al [11] have recommended their sparse learning system for application in practical BCIs, which include the idle state, the Fisher score channel selection method is a generic method used for comparison in the literature [30], [52] and EEGNet [7] has been designed for practical BCI use. EEGNet has also been found to perform on-a-par with or better than other CNNs which have been applied to classification of the idle state[7]. Furthermore, in Chapter 6 of this thesis, EEGNet is applied to a dataset with the idle state included as a class and is found to perform similar to ShallowConvNet, which was originally designed for a classification problem that included the idle state [8].

The results are recorded in Table 5.8, with the best results highlighted in bold. The GABSLEEG system outperformed the SL approach by Sreeja et al. [11] for both datasets and window sizes, and the difference in performance was greater for Dataset IVa. Although Sreeja et al. [11] used a similar structure for the SL dictionaries, they construct the dictionary by obtaining the DWT of the signal on each channel, and then calculate the average energy of the detail and approximation coefficients, before concatenating the results for all the channels into a single feature vector. Since Sreeja et al. [11] use raw EEG data, the detail coefficients may have captured high-frequency noise within the EEG data, leading to poorer performance when compared to the GABSLEEG system, which used EEG data bandpass filtered in the alpha-beta bandwidth. Furthermore, the work of Sreeja et al. [11] used an arbitrarily chosen subset of EEG channels, whereas the channel subsets used by the GABSLEEG system were optimized through the GA.

The GABSLEEG classification system performed on-a-par with the Fisher score channel selection approach. In the case of Dataset I, the GABSLEEG system performed better, whereas with Dataset IVa, the Fisher score method performed better. Since the SL classifiers are the same for both systems, this result confirmed that the GA module performed useful channel selection.

EEGNet [7], a state-of-the-art CNN classifier for EEG, was outperformed by the GABSLEEG approach on both datasets, with a larger margin exhibited for Dataset IVa. Furthermore, note that for larger window sizes (2s for comparison to EEGNet), the GABSLEEG system performed more poorly as opposed to when smaller window sizes were used (e.g. 50 ms – 500 ms for other comparisons

Table 5.8: Comparing the performance of the GABSLEEG system to contemporary implementations. The results of the best performing systems are in bold.

		Dataset I		**Dataset IVa**	
Comparison System	Window Size	GABSLEEG Accuracy	Comparison System Accuracy	GABSLEEG Accuracy	Comparison System Accuracy
Sreeja et al. [11]	0.5s	**99.69%**	97.95%	**98.11%**	92.11%
Sreeja et al. [11]	50ms	**99.65%**	98.01%	**96.08%**	90.14%
Fisher Score Channel Selection + SL [219]	50ms	**99.68%**	99.54%	96.08%	**96.17%**
EEGNet [7]	2s	**78.33%**	77.81%	**87.43%**	83.75%

shown in Table 5.8). This may be because larger segments of EEG are not approximately stationary or approximately linear [91], [195]. Since the OMP algorithm sparse encodes the feature vectors over the dictionary using a linear approximation approach [54], the nonlinearities introduced by the larger window size may negatively impact performance.

It was noted that the GABSLEEG system outperformed the implementation of Sreeja et al. [11] and EEGNet [7] to a notably larger margin for Dataset IVa. It should be noted that this dataset was intended to be used for MI EEG classification only, with the idle state not included. This may mean that during breaks between MI EEG stimuli, which were the idle state for this analysis, may have been times when the subjects could blink or change position, leading to higher artifact and noise content in the data. Since both the implementation by Sreeja et al. [11] and EEGNet [7] use raw data without filtering, this additional noise in the data may have negatively impacted performance.

5.5 Conclusion

The main contribution of this chapter is the design and implementation of the proposed novel GABSLEEG system: a MI EEG classifier which merges GA channel selection with a dictionary-based SL classifier. This work showed that the GA and SL module work effectively for classification of MI EEG and the idle state.

The SL classifier had a better classification performance than the benchmarking classifiers (k-NN, SVM and RF). However, its execution time on the test-set suggested that it may be unacceptably slow for a real-time BCI.

The aim of the GA channel selection module was to produce a subset of channels to be used with the test-set that preserved the high accuracy of the SL classifier but led to computational improvement. The GA channel selection module was effective in preserving the classification accuracy of the SL classifier as well as the benchmarking classifiers (k-NN, RF and SVM). In fact, the GA channel selection module improved the accuracy of the SVM classifier and resulted in less than 1% decrease in accuracy for the SL, k-NN and RF classifiers. Using a reduced channel subset substantially improved test-set execution time

and the testing time of the GABSLEEG system indicated that it has the potential to be used in a real-time BCI. It was faster than the GA-SVM classifier in terms of training and testing times; however, it was notably slower than the GA-kNN system. This indicates an area for improvement in future i.e. reducing the GABSLEEG testing time, which is contingent on the SL classifier.

The GABSLEEG system outperformed conventional benchmarking approaches, namely GA-kNN, GA-SVM and GA-RF pipelines in terms of accuracy, sensitivity, and specificity. The GABSLEEG system had the most robust performance when training data size was reduced. The GABSLEEG system also performed on-a-par with many systems in the literature, and outperformed state-of-the-art SL [11] and CNN-based [7] classifiers which were published as recently as 2020 [11] and 2018 [7] . The GA module was just as effective as a state-of-the-art Fisher channel selection algorithm [30], [52], [219].

Although the GABSLEEG classifier performed robustly, it has a noticeable drawback. The dictionary was constructed by segmenting trials, extracting a feature vector with a length equal to the number of channels in the subset, and then storing the entire dictionary of feature vectors in memory. Since a single dataset can have hundreds of training trials, this could lead to a large amount of occupied memory. During experimentation for this Chapter, typically 6.4GB or more of the 8GB of memory on the laptop used was occupied during use of the SL classifier. Furthermore, the OMP algorithm is also memory intensive [54]. Although the GABSLEEG system could be run for subject-specifically trained systems where training data from just one subject is required, the memory demands of such a system for cross-subject EEG data, where hundreds of training trials from multiple subjects are used, may be great. Furthermore, the time taken for the OMP algorithm to encode test samples depends, in part, on the dictionary size, possibly leading to even slower test times. Although using less data would be a possible solution, due to the great inter-subject variability of EEG data [34], arbitrarily reducing the number of training samples in cross-subject EEG classification may lead to a less optimal performance. Dedicated algorithms for dictionary reduction could be an option for future work [230]. Deep learning

systems, particularly CNNs, have already been found to be very effective for cross-subject MI EEG classification [7], [58]–[60], [164] They can be trained to extract abstract features based on large training datasets without needing the whole training set to be loaded into memory as is the case for a SL dictionary-based classifier.

Another issue impeding the expansion of the GABSLEEG system to subject-independent channel selection is the involved execution times for GA channel selection. This issue is discussed in greater depth in Chapter 6, where the execution time of the GA channel selection module for subject-independent channel selection is compared to other channel selection methods, including the novel ICS layer channel selection method, which is the focus of Chapter 6.

Deep learning systems are ideal candidates for subject-independent channel selection due to their track record in cross-subject EEG classification [7], [58]–[60], [164]. The next chapter, Chapter 6, is focused on subject-independent channel selection using CNNs and presents a versatile method that was applied to two different architectures. This channel selection approach is found to be more computationally efficient than the GA channel selection approach.

Chapter 6 : An Integrated Channel Selection Layer for Subject-Independent Channel Selection in CNN Networks

6.1 Introduction

This chapter presents a novel, integrated channel selection (ICS) layer which can be used for subject-independent channel selection in different CNN networks. CNN-based channel selection for MI EEG channel selection has been gaining popularity in recent years [23], [24], as discussed in Chapter 3 (Sections 3.5.2.3 and 3.5.2.4). Two main gaps in the literature were identified: i) a lack of focus on subject-independent channel selection, and ii) CNN-based channel selection methods are typically tested on just one architecture, meaning their versatility is undocumented. The wider literature in MI EEG channel selection also tends to focus on subject-specific channel selection [13], [23], [46]–[49], and a study by Handiru et al. [52] has found that subject-independent channel selection may not perform on-a-par with subject-specific channel selection. Notwithstanding this, subject-independent channel selection is desirable because it removes the channel selection latency time associated with subject-specific training. It can also lead to lower hardware costs because effective subject-independent channel selection means that systems can be sold with fewer electrodes, making for a more practical BCI set up. See Chapter 3 (Section 3.5.2.4) for a more in-depth discussion on the benefits of subject-independent channel selection.

The ICS layer method presented in this chapter addresses these gaps in the literature. Firstly, it was found to be effective for subject-independent channel selection; in fact, there was no statistically significant difference between classification accuracy results when channels were selected in a subject-specific

and subject-independent manner (Section 6.4.1). The ICS layer channel selection technique also outperformed two other state-of-the-art channel selection techniques (Section 6.4.2).

The new ICS layer that is presented in this chapter was found to be versatile: it was effective for MI EEG channel selection when applied to two different CNN architectures and tested on two different datasets. In all cases, it outperformed benchmarking systems which select channels based on a ranking scheme like that used in the ICS layer method (Section 6.4.2).

The method in this chapter can be compared with the GA channel selection method from Chapter 5. In Chapter 5, channel selection was applied to improve the execution times in the testing phase by selecting subject-specific channels during training. This approach could also improve the practicality of the system by reducing the number of electrodes used in the test phase, which represents practical use of the system. Subject-specific channels are selected using the training data of the individual subject and the selection process increases the training latency of the BCI system. The work in this chapter goes a step further and is focused on subject-independent channel selection, where channels are selected using data from a pool of source subjects, and the subset is applied directly to a target subject. An ICS layer, which can be added to the start of a 2D CNN network for channel selection, was designed for this purpose. The ICS layer has the same size as the input data segment and consists of trainable weights with sparse L1 activity regularization to suppress channels that are redundant. The proposed, new ICS-CNN system is trained end-to-end, then the weights of the ICS layer are extracted. An average weight value for each channel is obtained, and the channels are ranked, with larger average weights being associated with more important channels. In this chapter, the GA channel selection method and the ICS layer method are compared for the subject-independent channel selection problem in Section 6.4.3.

Reducing the number of EEG channels using the ICS layer did, however, result in a statistically significant decrease in classification accuracy when compared to using the full EEG montage. Transfer learning was applied to try and

improve classification performance. Two kinds of transfer learning approaches were compared: RTL and MTL, which were previously introduced in Chapter 3 (Section 3.4.2). RTL involves pre-training the CNN network using source data and then fine-tuning using the targets' data, whilst MTL involves merging the source and target data and training the CNN models. In three out of the four experiments carried out in this chapter, MTL performed worse than RTL in terms of classification accuracy. Furthermore, MTL consistently increased the training latency experienced by the target subject by a notable amount. Applying RTL to the CNN classifier was an effective way of improving the classification accuracy when the number of channels was reduced. In fact, when using RTL, there was no statistically significant difference in accuracy when comparing results obtained using the full EEG channel montage and when using half the channels. These results (in Section 6.4.4) are a novel contribution because, to the best of the authors' knowledge, there has been no explicit investigation focused on comparing RTL and MTL in this way for 2D CNNs for MI EEG classification, particularly within the context of a channel selection framework. This highlights the significance of the results in this chapter, in which a channel selection approach has been developed for CNN classifiers, and transfer learning used to improve performance.

A comprehensive execution time analysis (Section 6.4.5) investigates the computational benefits of subject-independent channel selection, the effect of RTL and MTL on training times, and compares the times taken for channel selection when using the ICS layer, the GA channel selection module, and the benchmarking approaches.

The rest of this chapter is structured as follows. The proposed ICS layer method is introduced in Section 6.2, and then the experimental methodology is discussed in detail in Section 6.3. In Section 6.4 the results are presented together with a discussion of their relevance. Finally, Section 6.5 discusses the concluding remarks.

6.2 Proposed CNN-Based Integrated Channel Selection Layer Method

Figure 6.1 shows how the proposed ICS layer can be applied to a CNN network to carry out channel selection. The CNN classification module can be based on any CNN classifier that takes as input EEG time segments of size $(N \times T)$, where N is the number of EEG channels in the dataset and T is the number of samples in the time segment. These segments are structured as $(1, N, T)$ for input to the CNN to create an extra dimension for the CNN filter signals to be stored. As discussed previously in Chapter 3, CNN classifiers typically have an identity layer which takes the EEG segments as input, which in Figure 6.1 is called the 'Input Layer'. As shown in Figure 6.1, the ICS layer is placed between this Input Layer and the 'CNN Classification Module', which consists of the CNN feature extraction and classification layers. Note that prior to being passed into the ICS layer, the input data is reshaped to size $(N \times T)$, and after exiting the ICS layer the data is reshaped to size $(1, N, T)$. The reshape layers were not shown in Figure 6.1 for simplicity.

The ICS layer has size $(N \times T)$ and consists of trainable weights. In the ICS layer there is a trainable weight for every time sample on every channel in the input segment. When a segment of data is passed to the ICS layer, each sample is

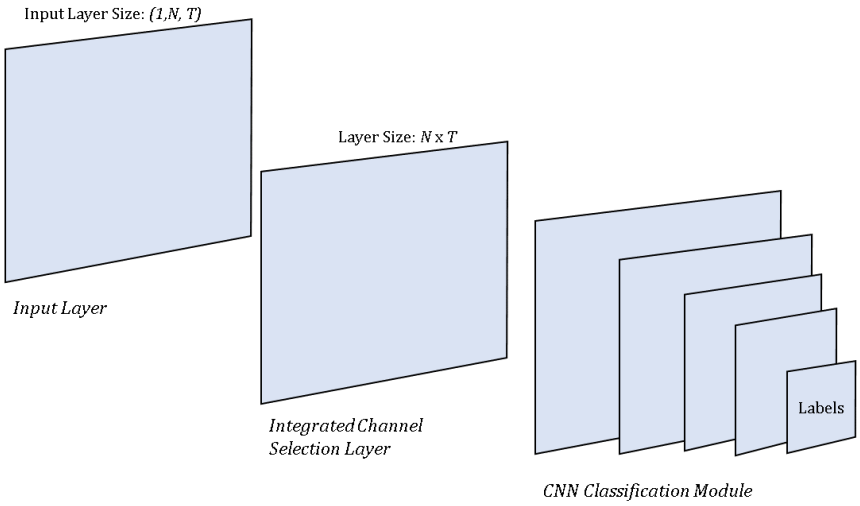

Figure 6.1: A high level diagram of how the ICS layer can be applied to a CNN classifier.

multiplied with its corresponding weight in the layer. Thus, the ICS layer can produce a 'gain' or 'attenuation' in each sample in the input through the multiplicative weights. At the start of training, the weights in this layer are all initialized to a value of one. Therefore, at the start of training, the ICS layer behaves as an identity layer, passing the time series signals directly to the CNN Classification Module. Note that the weights in the CNN Classification Module are randomly initialized in keeping with the literature [7], [8]. The weights in the ICS layer are subject to an L1 regularization penalty, governed by a regularization factor of 0.01. The aim of this sparsity constraint is to force channels that have a minimal contribution to the classification process to be assigned smaller weights during training. The ICS layer was implemented as a custom layer in TensorFlow with Python 3.

The first step in the channel selection process involves training the system in Figure 6.1 in an end-to-end manner. Information about the training methodologies used can be found in Section 6.3.3.1. After training, the weights of the ICS layer are extracted as an array of size $(N \times T)$ for post-processing. The core hypothesis of this channel selection approach is that larger weights in the ICS layer correspond to greater importance within the CNN network. The ICS layer weights are averaged such that a vector of length N is obtained, with a single average weight value for each EEG channel. The average weight values are the 'scores' of the EEG channels, and the channels are ordered in descending order. The channel associated with the highest score is considered the most important and the channel with the lowest score is considered the least important. The M channels with the highest scores are selected for the EEG channel subset, where the value of M is set by the user. The original CNN classifier (i.e. without the ICS layer) is then trained using the selected EEG channel subset.

Figure 6.2 shows an example of the ICS layer weight analysis for subject 1A from the Graz 2A dataset [72], [111]. More information on the dataset can be found in Section 6.3.1.1. In this example, the ICS layer was applied to the EEGNet architecture [7]. The first image on the left shows the ICS layer weights obtained through subject-independent training. The ICS layer directly maps onto the input

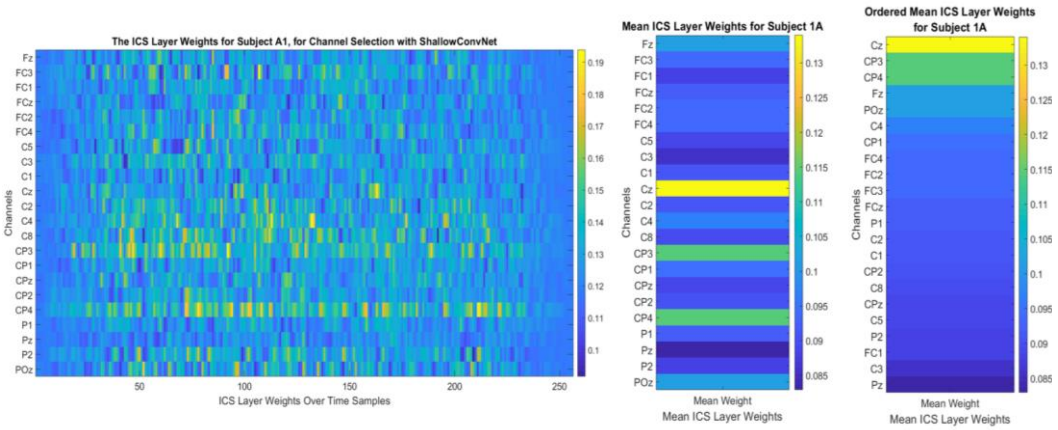

Figure 6.2: Analysis of the ICS layer weights. The first image on the left shows an example of the ICS layer weights obtained for subject 1A in the Graz 2A dataset when cross-subject training is used. The x-axis corresponds to the time samples and the y-axis corresponds to the channels. The colormap shows the value of each weight. The second image shows the average weight value for each channel and the third image shows the channels sorted in descending order (top to bottom) according to the mean weight value.

data segment, so the y-axis is associated with the different EEG channels in the dataset, whilst the x-axis is associated with time samples. There are 256 time samples at the input since two-second segments of EEG data recorded at 128Hz were used. The colours denote the size of the weights, and the colourbar on the right-hand side gives an indication of the weight values. Note how the larger weights are found towards the centre of the image, which corresponds to the centre of the trial, where ERD activity can be expected [68]. The second image shows the average weight value associated with each channel, and indicates that in this case channels Cz, CP3 and CP4 are of greatest importance. The final image shows the channels ordered in descending order according to mean weight, with channels Cz, CP3 and CP4 at the top of the list being the most important channels, and channel Pz with the lowest weight value being considered the least important at the bottom of the list.

6.2.1 Applying the ICS Layer to State-of-the-Art CNNs

The ICS layer is applied to two different state-of-the-art CNN networks, namely ShallowConvNet [8] and EEGNet [7], which were previously introduced in Chapter 3 (Section 3.4.1.3). For channel selection, the ICS layer is added after the

input layers of the CNNs. Complete layer-wise descriptions of the ShallowConvNet and EEGNet networks with the ICS layer included are shown in Table 6.1 and Table 6.2, respectively. The layers added for channel selection are highlighted in red, and the descriptions include the reshape blocks as well as the ICS layer. The hyperparameters were set as in the Tables, which were recommended by the literature [7], [8]. The values of other parameters were set as follows: Samples is 256 since the segments are always 2s long and recorded at

Table 6.1: A summary of the structure of ShallowConvNet with the ICS layer added.

Layer	**Details**
Input identity layer	Shape: (1, Nº Channels, Samples)
Reshape Layer	Output Shape: (Nº Channels, Samples)
ICS Layer	Shape: (Nº Channels, Samples)
Reshape Layer	Output Shape: (1, Nº Channels, Samples)
Conv2D Layer *(Temporal convolution)*	40 filters, kernel size (1, W)
Conv 2D Layer *(Spatial filtering)*	40 filters, kernel size (Nº Channels, 1)
Batch Normalization + Dropout	Dropout rate: 0.5
Activation Layer	Square activation function
Average Pooling (2D)	Pool-size (1, 35), stride size (1,7)
Activation layer	Log activation layer
Flatten layer	-
Dense Layer *(Linear classification output layer)*	4 units; Softmax activation

Table 6.2: A summary of the structure of EEGNet with the ICS layer.

Layer	**Details**
Input identity layer	Shape: (1, Nº Channels, Samples)
Reshape Layer	Output Shape: (Nº Channels, Samples)
ICS Layer	Shape: (Nº Channels, Samples)
Reshape Layer	Output Shape: (1, Nº Channels, Samples)
Block 1	
Conv2D Layer *(Temporal convolution)*	F_1 filters, kernel size (1, F_s/2)
Batch Normalization	-
Depthwise Conv2D	Kernel Size (Nº Channels, 1), depthwise multiplier D
Batch Normalization	-
Activation Layer	Activation function: Elu
Average Pooling (2D)	Pool size: (1,4)
Dropout Layer	Dropout rate: 0.5
Block 2	
Separable Conv2D	F_2 filters, kernel size (1,16)
Batch Normalization	-
Activation Layer	Activation function: Elu
Average Pooling (2D)	Pool size (1,8)
Dropout Layer	Dropout rate: 0.5
Block 3	
Flatten Layer	-
Dense Layer *(Linear classification output layer)*	4 units; SoftMax activation

a sampling frequency (Fs) of128Hz, W is 13, and D is 2, in accordance with the literature [7], [8]. Nº Channels depends on the dataset.

For better visualization, Figure 6.3 shows the ICS layer added to the ShallowConvNet architecture with 22-channel EEG data as input. Note how it has been added immediately after the input layer and before the first Conv2D layer. The batch normalization and droupout layers for ShallowConvNet and the reshape layers associated with the ICS layer have been omitted from the illustration for simplicity. The ICS layer does not in any way alter the characteristic classification layers of the original CNN but simply weights the input.

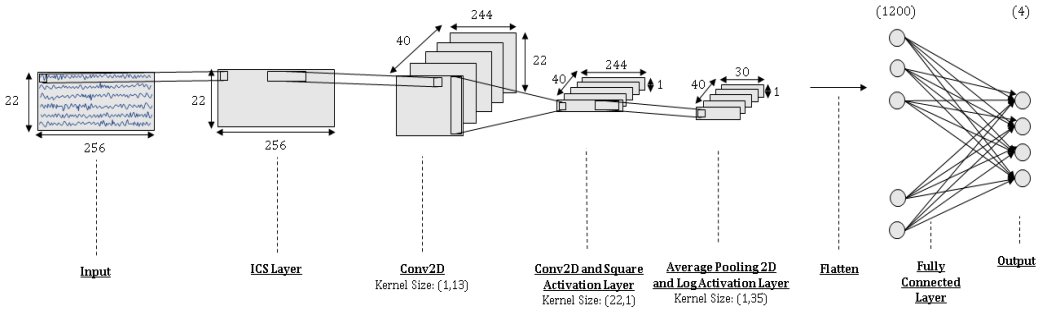

Figure 6.3: An example of the ICS layer added to ShallowConvNet. The ICS layer was inserted between the input layer and the first convolutional layer. For simplicity, the reshape layers before and after the ICS layer have been omitted.

6.3 Experimental Methodology

This section opens with a discussion of the datasets and pre-processing methods used, and then describes the performance measure used. It then explains the training methodologies for channel selection and transfer learning. This section concludes with a summary of the benchmarking systems used and the execution time analysis methodology.

6.3.1 Datasets, Pre-Processing and Data Augmentation

Two datasets were used for evaluation: the Graz 2A dataset [72], [111] and the high gamma (HG) dataset [8], [112]. Both datasets present a four class MI EEG

classification problem and were chosen because they have been widely used in the literature for CNN-based classification [6]–[9], [59], [60], [64]. They have also been used in previous studies [7], [8] with the CNN architectures to which the ICS layer is applied in this chapter. Furthermore, these datasets have more subjects than the BCI datasets used in Chapter 4 and Chapter 5, making them more suitable for investigating transfer learning.

The Graz 2A and HG datasets were pre-processed according to recommendations in the literature [7], [8], [112]. Regardless of the dataset used, for testing, data in the interval 0.5s - 2.5s in the EEG trials was used for classification, and this was in-keeping with the literature [7]. Data between the 0.5s and 2.5s time points of trials is commonly used because this is where the ERD activity that characterizes MI is typically captured [7], [68].

The data in the Graz 2A dataset was pre-processed according to the recommendations by Lawhern et al. [7]. The steps were as follows: each EEG trial is pre-processed using a two-step method. First, to remove low frequency drifts and artifacts, the data was high pass filtered with a passband frequency of 4Hz. A minimum-order filter that automatically compensates for any filter delays was used. Afterwards, the signals were down sampled to 128Hz. This reduces the folding frequency of the signals to 64Hz, thus eliminating higher frequencies which are more often associated with noise, whilst still encapsulating the alpha band, beta band, and part of the gamma band. No additional pre-processing was carried out on the HG dataset, since it had already experienced some pre-processing, described in Section 2.2.

As previously mentioned in Chapter 3 (Section 3.6) some systems segment EEG data to augment the training dataset. This is particularly common for DL classification systems [6], [8], [58], [60] since they need large datasets to prevent overfitting [6]. Thus, the training data from both datasets was augmented through a segmentation method to increase its size by a factor of three. This was achieved by segmenting each training trial into three using an overlapped moving window approach that obtains segments from the time intervals: 0.5s-2.5s, 1.0s-

3.0s and 1.5s-3.5s within trials. Cropping techniques such as this are widely used to augment EEG datasets for training CNN classifiers [6], [8], [58], [60].

6.3.2 Performance Measure

Since multi-class classification problems are tackled in this chapter, categorical accuracy was used to assess the classification performance of the systems. This is a measure used for EEG-based CNNs in the literature [7], [59], [60], [231], and comprises of the percentage of correctly classified trials to the total number of trials classified, as shown in (6.1):

$$Categorical\ Accuracy = \frac{Number\ of\ correct\ classifications}{total\ number\ of\ classifications} \tag{6.1}$$

6.3.3 Training Methodologies

This section summarizes the training methodologies used for channel selection and transfer learning.

6.3.3.1 Channel Selection

The training process for ICS-based channel selection involves two steps:

A) Train the CNN classifier with the ICS layer added and carry out the post-training weight analysis previously described in Section 6.2.
B) Train the CNN classifier using the reduced channel subset. The ICS layer is not included in this step.

For both steps A and B, categorical cross-entropy loss was used for CNN training because both datasets in this chapter are multiclass. The Adam optimizer was used, and training was carried out for 100 epochs, which was in-keeping with trends in the literature [188], [232], [233]. To prevent overfitting, the early stopping approach was used [7]. In this approach, the training data is split into a training subset and a validation subset, with the training subset being used for learning but the weights that gave the best results on the validation subset being loaded at the end of training. This is an established training method for the CNNs

used [7]. At the end of this section, a brief analysis verifies that using 100 epochs did not lead to underfitting.

In step A, the training data was randomly divided into two groups, a training subset, and a validation subset. The ratio used for splitting data into the training and validation subsets was 80:20. The data division was stratified, ensuring that the training and validation subsets were comprised of the same percentages of trials for each class. When carrying out subject-specific channel selection, the training data of the subject was used, whereas when subject-independent channel selection was carried out, the training data of all the other subjects was combined into a single training dataset. In the case of the Graz 2A dataset, data from 8 other subjects was merged, whereas for the HG dataset, data from 12 other subjects was merged. The CNN-ICS network was trained for 100 epochs, and at the end of training the network weights that gave the best categorical classification accuracy on the validation subset were loaded to the model. The ICS layer weights were then extracted for further processing. At the end of step A, a set of M channels was selected.

Step B uses subject-specific data from the target. The performance of the channel subset was evaluated using five-fold cross-validation, which was carried out as follows:

1) Randomly divide the training data into five groups. This data division resulted in an 80:20 split between training and validation data, like in step A. Stratified k-fold cross-validation was used, meaning that the percentage of samples from each class is the same for every fold, ensuring balanced training.
2) Randomly initialize the weights in the CNN model.
3) Train the CNN for 100 epochs using four groups of data for learning and one for validation. At the end of the 100 epochs, load the best weights and evaluate the performance of the model on the test set.
4) Repeat steps 2) and 3) for every fold, and then calculate the average performance on the test set.

Different channel subset sizes were considered in investigations. For the Graz 2A dataset, subset sizes of 11, 6 and 3 were considered, whilst for the HG dataset subset sizes of 22, 11, 6 and 3 were considered. Note that at each step, the channel subset size is decreased by half or approximately half.

To verify that 100 epochs provided adequate training and did not lead to underfitting, an analysis of the validation training curves for 500 epochs was carried out. In this analysis ShallowConvNet and EEGNet were trained in a subject-independent manner on the Graz 2A and the HG datasets for 500 epochs. In this analysis all the channels in the dataset were input to the classifiers. A five-fold cross validation approach was used, such that five validation data training curves were obtained per subject. These were averaged to obtain a single training curve per subject. The resulting curves for individual subjects were then averaged again for each dataset-classifier pairing to obtain the plots in Figure 6.4. The vertical red lines mark 100 training epochs. In every case, by 100 epochs the bulk of the learning has been achieved, with the curves already flattening out notably by the time 100 epochs has been reached. Beyond 100 training epochs there is limited change in validation accuracy. These results suggest that training for 100 epochs was acceptable for the CNNs considered.

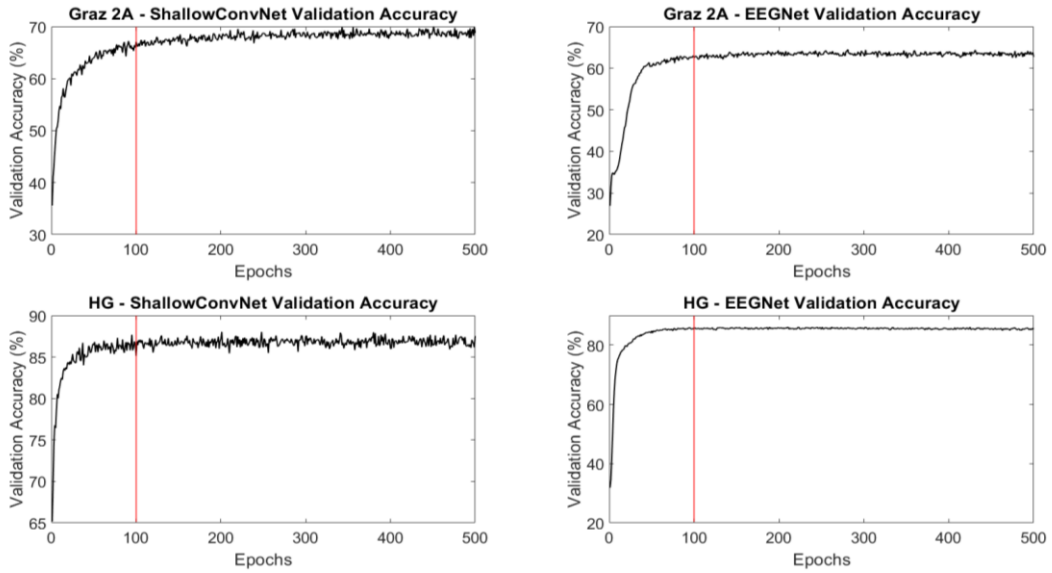

Figure 6.4: Plots of the average validation training curves for ShallowConvNet and EEGNet when using the Graz 2A and HG datasets. The y-axes show the validation set accuracy and the x-axes show the epochs.

6.3.3.2 Transfer Learning

Transfer learning was applied to the CNNs with the aim of improving the classification performing when using a reduced channel subset. Results obtained using two different methodologies, namely RTL and MTL, were compared. Regardless of the transfer learning strategy used, it only affects the outcome of step B in Section 6.3.3.1 and thus has no effect on the channels selected. These methodologies are discussed in more depth in the rest of this section.

RTL is a two-step process: first a base model is trained using source data from other subjects, then the model is fine-tuned using target data. Consider the base model training process. The Graz 2A dataset has a total of 9 subjects, meaning that for each target subject there is source data available from 8 other subjects. The training trials from the source subjects were combined and randomly divided into a training subset and validation subset using an 80:20 ratio. Again, stratified data division was used. In the case of the HG dataset, 12 source subjects are available for each target subject, and the training and validation subsets were obtained using the same ratio. The base model was trained using 100 epochs and the Adam optimizer. After training, the weights that gave the best categorical classification accuracy with the validation set were loaded to the base model.

The base model was then fine-tuned using the target data. This was achieved by carrying out step B in Section 6.3.3.1, but instead of randomly initializing the weights in the CNN model, the base weights were used.

Note that base model training can be carried out well in advance of the fine-tuning step and, in a well-planned practical implementation, the time taken to train the base model would not contribute to the training latency time experienced by the target subject.

In the case of MTL, the source data was combined with the target training data, and the CNN in step B was trained from scratch using this data.

In this analysis, the aim was to use transfer learning to improve performance so that there is no degradation in accuracy when the number of

channels was reduced by half through ICS channel selection. Thus, for the Graz 2A dataset 11 channels were selected and for the HG dataset, 22 channels were selected. Regardless of the transfer learning method used, the channels were selected in a subject-independent way.

6.3.4 Systems used for Benchmarking

The channel selection methods used for benchmarking the novel ICS layer method were based on two state-of the art methods from the literature [13], [47]. Both approaches are filter-based and use a statistical analysis on the raw EEG data to rank the channels. The first approach is a correlation-based method presented by Jin et al. [13] and the second is a covariance-based method presented by Gurve et al [47]. These methods were chosen since they resemble the ranking approach used by the ICS layer method.

6.3.4.1 Correlation Coefficient Based Channel Selection

Correlation coefficient channel selection (CCS) [13] is based on the hypothesis that channels related to MI activity will be strongly correlated across trials. Based on a Pearson correlation analysis, channels which are correlated with comparatively few other channels across trials are deemed to be redundant and are excluded from the channel subset. The approach is as follows:

1. The signals in all EEG training trials are Z-score normalized so they have a mean of zero and a standard deviation of 1.
2. For each trial in the training dataset:
    a. Obtain the correlation matrix based on the correlation coefficient. Since the directionality of correlation is not of interest, only absolute values are included in the matrix.
    b. The mean value of each row is calculated. This gives a score that is related to the strength of the correlation between a given channel and the other channels in the dataset. Channels with a higher score are considered more important.
    c. Rank the channels according to the score and choose the M channels with the highest scores to form a candidate subset.

To obtain the final subset of channels, the M channels that are most frequently included in the candidate subsets are used.

6.3.4.2 Covariance and Non-Negative Matrix Factorization Channel Selection

The second benchmarking approach is based on the covariance and non-negative matrix factorization (CNMF) method presented by Gurve et al [47]. CNMF-based channel selection consists of the following steps:

1. For each trial in the training dataset:
    a. Compute the covariance matrix, C_N. If the EEG signal consists of N channels and the data on channel k is denoted by $x_k(t)$, where t is the time sample, then the covariance matrix is [47]:
    $\boldsymbol{C_N} = \begin{pmatrix} Var(x_1) & \cdots & Cov(x_N, x_1) \\ \vdots & \ddots & \vdots \\ Cov(x_1, x_N) & \cdots & Var(x_N) \end{pmatrix}$, where $Var(.)$ is the variance and $Cov(a, b)$ is the covariance between signals a and b. Since directionality is not important, the absolute value of the covariance matrix is considered.
    b. The covariance matrix is decomposed using non-negative matrix factorization (NMF). The NMF decomposition can be summarized as [47]: $\boldsymbol{C}_N \approx \boldsymbol{WH}$, where $\boldsymbol{W}$ is called the template or basis matrix and $\boldsymbol{H}$ is called the activation matrix. These matrices have a size of $N \times r$ and $r \times N$, respectively, where r is the rank for matrix factorization. The authors recommend a rank of 3 [47]. Details on how NMF is carried out can be found in [47].
    c. The rows of the activation matrix are then normalized using: $H_{jN} = \frac{H_j - \min(H_j)}{\max(H_j) - \min(H_j)}$, where H_j is the j^{th} row in $\boldsymbol{H}$ and H_{jN} is the normalized row.
    d. The root-mean-square-deviation (RMSD) of each row from H_{ref}, which is a straight line of value 0.5, is then calculated. The RMSD is calculated as:

$$RMSD = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left\|H_{jk} - H_{ref}\right\|^2} \quad (6.2)$$

e. The values in the row of the activation matrix which has the maximum RMSD score are used as the weights for the channels.
f. Guave et al. [47] consider larger weights to be associated with more important channels. Thus, the *M* channels with the highest weights are selected as a candidate subset.

Like the approach used for the CCS method, the *M* channels which feature most frequently within the candidate subsets are selected to form the final subset.

6.3.5 Comparison to the GA Channel Selection Module

The ICS layer method was also compared to the GA channel selection module presented in Chapter 5. The GA was applied to the CNN channel selection process by replacing the SL classifier with ShallowConvNet or EEGNet in Algorithm 2, which was presented in Section 5.2.3. A comparison between the GA channel selection module and the ICS layer method for subject-independent channel selection is presented in Sections 6.4.3 and 6.4.5.4. In this analysis, channel selection was carried out for both ShallowConvNet and EEGNet with the Graz 2A dataset. Only the Graz 2A dataset was used for this comparison because the GA tuning and channel selection process took significant execution time, and results based on the Graz 2A dataset were deemed sufficient for comparison in terms of classification accuracy and execution times.

The GA had to first be tuned for the CNN channel selection problem. Previously, in Chapter 5, the number of channels in the subset and the size of the population were both tuned (Section 5.3.5.1). Since the tuning process is involved, the number of channels in the subset was set to 11. This corresponds to half the channels in the Graz 2A dataset and is also the size of the channel subset used in other analysis in this chapter (namely in Section 6.4.4). Thus, only the population size of the GA was tuned in this chapter. Population sizes of 5, 10, 20 and 30 were considered during the grid-search tuning process. In this tuning process, the source data associated with a target subject was divided into an 80:20 ratio, with 80% being used for learning and 20% being used for validation.

Recall that the source data consists of the training data of all the other subjects in the dataset, so if subject A1 is the target, the source data is the training data from subjects A2-A9. The GA channel selection approach was then run for each of the different population sizes. The validation accuracy obtained with the best individual (i.e. the chosen candidate subset), and the execution time taken to execute the GA channel selection process were recorded. This process was repeated for all subjects in the Graz 2A dataset. Note that this whole search process was carried out twice, once when using ShallowConvNet as the classifier and then when using EEGNet as the classifier.

The execution time of the tuning process was also monitored. For each population size, the time taken for the GA to run on each subject was recorded. The execution times for each population size were then averaged across subjects. The total time taken to accumulate the results for all subjects and population sizes was 21.89 hours for ShallowConvNet and 24.1 hours for EEGNet. Whilst recording these computational results, all optional background processes on the laptop used were halted. This analysis was carried out using a Lenovo™ ideapad 330 laptop using a 64-bit Windows 10 operating system and an Intel® Core™ i5-8300H, 2.30GHz CPU.

Figure 6.5 shows the validation accuracy results, averaged across subjects, and the execution times obtained for each population size. The blue plots are for EEGNet and the red plots are for ShallowConvNet. The individual results for each subject can be found in Table A 1- Table A 4 in the Appendix. Considering the plot for validation accuracy (on the left) it is clear that, in general, the average accuracy tended to improve as the population size increased. Considering the plot on the right, the execution time always increased substantially as the population size was increased. Previously, when tuning the GA module in Chapter 5, the best

hyperparameter was the one which gave the greatest accuracy during the grid-search. However, it is evident that population size can have a substantial impact on the computation times of the GA module. Thus, in this chapter the tuning process of the GA was refined to include a statistical analysis that factors in both validation accuracy and execution time.

To assess whether there was any significant difference in classification accuracy for the different population sizes, a one-way analysis of variance (ANOVA) was carried out on Table A 1 and Table A 3 in the Appendix, which contain the individual subject results for classification accuracy. An ANOVA was used because the data in the tables was previously found to be normal using an Anderston- Darling test. For ShallowConvNet, the ANOVA gave a p value of 0.9916 and for EEGNet it gave a p value of 0.9863. Since, in both cases, $p > 0.05$, there was no significant difference in classification accuracy across the different population sizes. Similarly, a statistical test was used to assess whether there was a significant difference in execution times when different population sizes were used (data in Table A 2 and Table A 4 in the Appendix was involved in this analysis). An ANOVA test on the results of ShallowConvNet gave a p value of $1.29e^{-9}$ and a Kruskal-Wallis test on the results of EEGNet gave a p value of $7.24e^{-}$

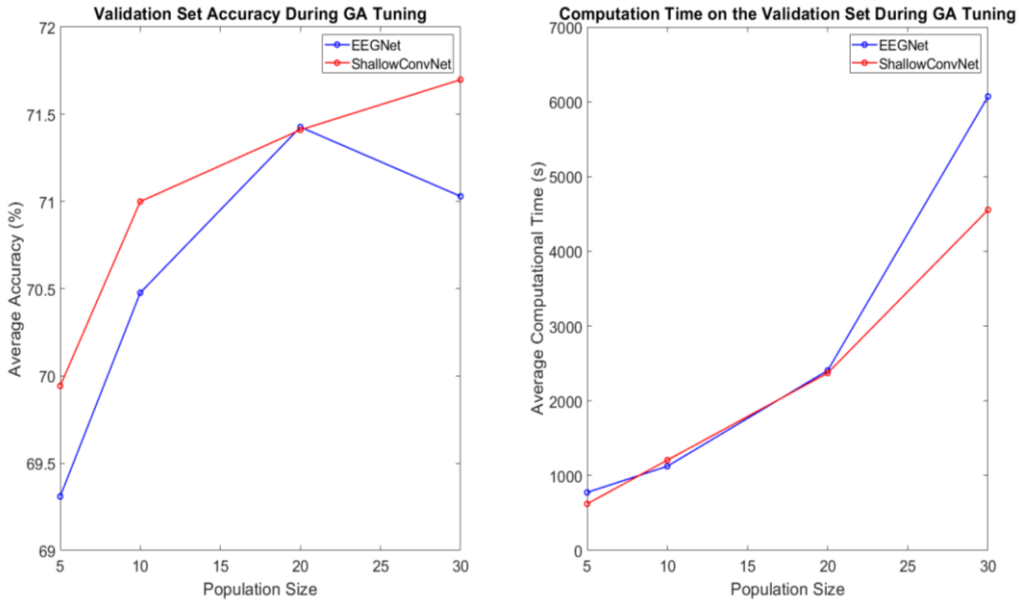

Figure 6.5: Plots of how the average validation set accuracy (left) and the average computational times during GA tuning (right), vary with population size for ShallowConvNet and EEGNet.

[7]. Since $p < 0.05$, these results indicate that the execution time varied significantly with population size. A Kruskal-Wallis test was used for the EEGNet results because an Anderston-Darling test found that the result set was non-normal, and thus a test for non-normal data was used.

Based on this hyperparameter tuning analysis, a population size of 5 was chosen because it provided similar accuracy results to larger population sizes, but with a substantially lower execution time.

6.3.6 Execution Time Analysis Methodology

A four-part execution time analysis was carried out. Figure 6.6 and Figure 6.7 capture the training and testing latencies when subject-specific and subject-independent channel selection is carried out, respectively. In both diagrams, the 'Target Training Phase' captures the latency experienced by the target subject before they can use the BCI. In the case of subject-specific channel selection (Figure 6.6) this latency consists of the recording of the target subjects training data, channel selection, and training of the classifier using the selected channels. In the case of subject-independent channel selection (Figure 6.7) the 'Target Training Phase' consists of just the training data recording and the training of the classifier. This is because subject-independent channel selection (yellow block in

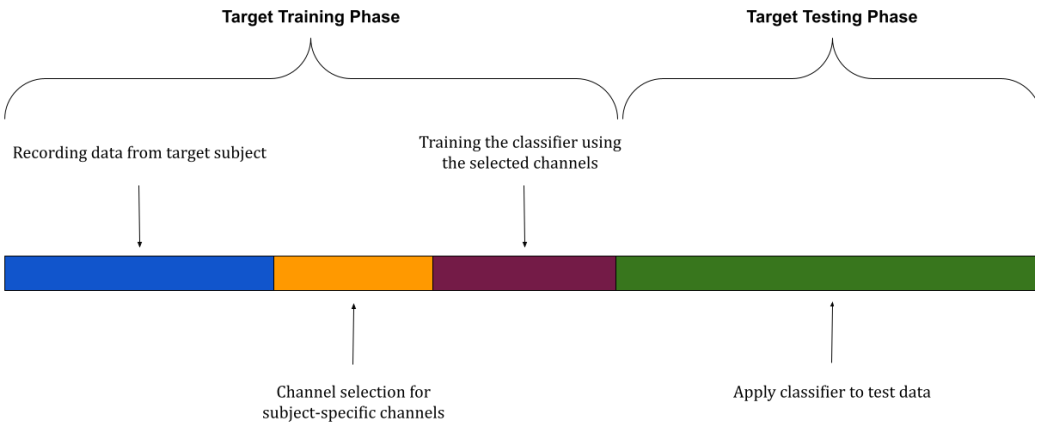

Figure 6.6: A diagram showing a breakdown of the latencies involved in the training and testing phases when subject-specific channel selection is carried out.

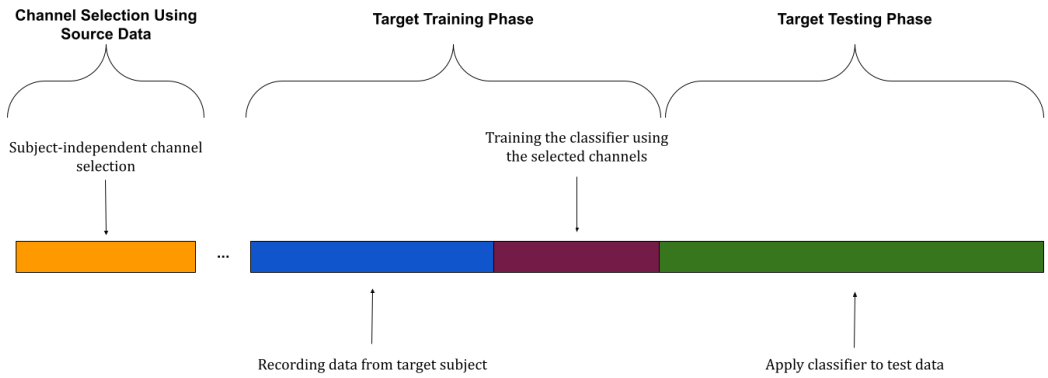

Figure 6.7: A diagram showing a breakdown of the latencies involved in the channel selection, training, and testing phases when subject-independent channel selection is carried out.

Figure 6.7) uses source data from other subjects and can be carried out in advance, so it does not create a latency experienced by the target subject.

The first part of the execution time analysis investigated the time saved during the 'Target Training Phase' when subject-independent channel selection was carried out. This corresponds to the time taken for subject-specific channel selection, denoted by the yellow block in Figure 6.6.

The second and third parts of the analysis are focused on the training times experienced by the user when subject-independent channel selection is used. The second part assessed the impact of using a reduced channel subset on the target classifier training time, denoted by the purple block in Figure 6.7. Results were recorded when using the full EEG montage of the dataset, and when half the electrodes were used. In the third part, the classifier training latency times attributed to the RTL and MLT methods were compared. For the RTL method, this involves the fine-tuning step carried out using target data, whilst for the MTL method, this involves the training of the CNN using merged source and target data. Since transfer learning involves the training of the classifier, this part of the analysis is also focused on the purple block in Figure 6.7.

The fourth and final part of this analysis compares the execution times for subject-independent channel selection when using the ICS layer method, CCS, CNMF, and GA methods. This latency corresponds to the yellow block in Figure

6.7. This is the only latency in this analysis that is not experienced by the target subject, since subject-independent channel selection can be carried out ahead of time.

The first and fourth parts of this analysis involved recording channel selection latencies and use the same methodology for recording execution times. The execution time for channel selection was recorded for each subject, then the average time across subjects was calculated. In the case of the first part, the times for subject-specific channel selection using the target training data were recorded, whereas for the fourth part the times for subject-independent channel selection using the source data were recorded.

The second and third parts of this analysis are concerned with the time taken to train the classifier and used the same methodology for recording execution times. This involved recording the training times of the CNN during step B of the channel selection process in Section 6.3.3.1. For each subject, the CNN training times for each of the 5 folds were recorded, then the median time was saved. Finally, the median times were averaged across the subjects to obtain the mean training time, and the maximum of the median times was recoded as the worst-case time.

For this computational analysis a Lenovo™ ideapad 330 laptop with a 64-bit Windows 10 operating system and an Intel® Core™ i5- 8300H, 2.30GHz CPU was used. All optional background processes on the laptop were halted whilst these execution time results were recorded.

6.4 Results and Discussion

This section opens with a comparison of the results obtained when applying the ICS layer method to subject-specific and subject-independent channel selection. It then compares the performance of the ICS layer method to the two benchmarking methods. After, it presents the results obtained when applying transfer learning to improve performance with a reduced channel subset. Finally,

this section closes with an execution time analysis and a discussion of the most frequently selected channels. In all statistical analysis, a 0.05 level of significance was used. This means that *p*-values below 0.05 were considered to indicate statistical significance.

6.4.1 Comparing Subject-Specific Channel Selection and Subject-Independent Channel Selection when using the ICS Layer Method

In this section, the results obtained when applying the ICS layer method to subject-specific channel selection and subject-independent channel selection are compared. Table 6.3 and Table 6.4 present the average categorical accuracy results for the Graz 2A and HG datasets, respectively. Both tables contain the results for subject-specific and subject-independent channel selection for different channel subset sizes. Results for both the ShallowConvNet and EEGNet architectures are shown.

The subject-specific and subject-independent results were compared using statistical tests, with the *p*-values for each comparison recorded in brackets

Table 6.3: Comparing classification accuracy results on the Graz 2A dataset for subject-specific and subject-independent channel selection when applying ICS to ShallowConvNet and EEGNet, for different channel subset sizes.

Subset Size	ShallowConvNet		EEGNET	
	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
3	50.40%	49.66% (*p*= 0.78)	51.34%	50.17% (*p* =0.49)
6	57.17%	59.17% (*p* =0.17)	57.40%	57.84% (*p* =0.77)
11	62.12%	63.47% (*p* =0.10*)	59.84%	59.70% (*p* =1.00*)

* A Wilcoxon signed-rank test was used.

Table 6.4: Comparing classification accuracy results on the HG dataset for subject-specific and subject-independent channel selection when applying ICS to ShallowConvNet and EEGNet, for different channel subset sizes.

Subset Size	ShallowConvNet		EEGNET	
	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
3	70.73%	68.29% (*p*= 0.1046)	64.32%	70.63% (*p*=0.0464)
6	79.62%	78.56% (*p*=0.3596)	71.48%	75.90% (*p*=0.0114)
11	83.12%	83.07% (*p*=0.2076)	76.30%	79.91% (*p*=0.0151)
22	87.86%	86.95% (*p*=0.4071)	79.56%	82.09% (*p*=0.0233)

in the Tables as (p=…). The categorical accuracy results in Table 6.3 and Table 6.4 are the result of averaging across subjects. The individual results for subject A1 with ShallowConvNet is 53.61% for subject-specific channel selection and 66.88% for subject-independent channel selection, and the rest of the results for individual subjects can be found in the Appendix, Table A 5 to Table A 11. The statistical tests were carried out using paired subject data from these tables. So, for example, when assessing whether there was any significant difference in performance when using subject-specific and subject-independent channels for the Graz 2A dataset and the ShallowConvNet architecture with a subset size of 3, the paired accuracy results for subjects A1-A9 were compared in the statistical test. Prior to carrying out the tests, each set of accuracy results were tested for normality using an Anderston-Darling test. If both the subject-specific and subject-independent result sets were normal, then a paired t-test was used for comparison, otherwise a Wilcoxon signed-rank test was used. In the tables, p-values marked with a superscript Asterix were obtained using a Wilcoxon signed-rank test, otherwise t-tests were used. The focus in this section is on comparing subject-independent channel selection to subject-specific channel selection. A comparison between results with channel selection and the full cohort of channels is included in Section 6.4.4.

Considering the results for the Graz 2A dataset in Table 6.3, no significant difference was found between the subject-specific and subject-independent results for any subset size (p>0.05). Considering the results for the HG dataset in Table 6.4, no significant difference was found in results when using ShallowConvNet (p>0.05). However, when using EEGNet, there was always a significant difference in performance (p<0.05). In fact, the results indicate that using subject-independent channel selection always resulted in a significant improvement in accuracy when compared to using subject-specific channels. This indicates that the ICS layer channel selection method can select EEG channels that generalize well to new subjects. It also indicates that this CNN-based channel selection method may work better with more data, even if that data is from other subjects. Effective subject-independent channel selection like this is desirable

since it can save the target subject from the latency introduced when the channel selection algorithm is run on their training data to select a subject-specific channel subset. This computational saving is analysed in Section 6.4.5.1.

These results indicate that the ICS channel selection method was a viable approach for subject-independent channel selection for both ShallowConvNet and EEGNet, since the results for subject-independent channel selection were either not significantly different from those for subject-specific channel selection, or they were significantly better. Using the ICS channel selection layer, the EEG channels could be pre-selected using data from a pool of source subjects, then applied to a target subject with no significant deterioration in performance when compared to selecting channels using the subjects' own training data. This means that fewer electrodes can be used for the target subjects, resulting in lower cost and improved practicality. It also means that the training times for subjects can be improved since the channel selection latency during the individual subjects' training could be axed when using subject-independent ICS channel selection.

6.4.2 Comparing the ICS Layer Method to Other State-of-the-Art Channel Selection Techniques

The performance of the ICS layer for subject-independent channel selection was compared to that of the CCS and CNMF methods previously described in Section 6.3.4. The metric for performance comparisons was the average categorical accuracy. Table 6.5 shows the results obtained with the Graz 2A dataset, whilst Table 6.6 shows the results obtained with the HG dataset. The tables present the results obtained with different channel subset sizes and the peak accuracy values are in bold. Subject-wise results obtained with the CCS and CNMF methods can be found in Table A 12 to Table A 15 in the Appendix.

A pairwise statistical comparison was carried out between the paired subject results of the ICS method and the CCS or CNMF method in a similar way to the analysis in Section 6.4.1, with *t*-tests being used if all the data was normal and a Wilcoxon signed-rank test if it was not. Therefore, for example, the *p*-value of 0.948 in Table 6.5 is based on the comparison of the subject-wise results

obtained with the CCS method and those obtained with the ICS layer method, when using ShallowConvNet and a channel subset size of 3.

Across both datasets, the ICS layer method always obtained peak accuracy when compared to the other methods, regardless of the channel subset size or CNN classifier used.

Considering the statistical results for the Graz 2A dataset in Table 6.5**,** when using ShallowConvNet, the ICS method performed on-a-par with the state of the art. It also significantly improved the performance when compared to CNMF when 6 channels were used ($p<0.05$). Considering the results obtained for EEGNet, the ICS method gave significantly improved performance compared to the other methods when 6 channels were used ($p<0.05$). It also significantly improved performance compared to the CCS method when 3 channels were used($p<0.05$). Considering the statistical results for the HG dataset in Table 6.6, using the ICS method almost always resulted in significantly improved performance, for both ShallowConvNet and EEGNet ($p<0.05$). The only exception was when using EEGNet and the CCS method when 22 EEG channels were selected ($p>0.05$).

These results indicate that the ICS layer method had a strong performance when compared to the state-of-the-art. It always exhibited a greater categorical accuracy than the comparison methods, and in 68% of the instances considered in Table 6.5 and Table 6.6, the *p*-value results indicated that the ICS method produced significantly improved results compared to the comparison systems. In the rest of the instances, there was no significant difference in performance between the ICS layer method and the comparison methods. Thus, these results indicate that the ICS layer method exhibited strong potential to outperform the state-of-the-art, and in the worst-case-scenario, performed on-a-par.

Table 6.5: Comparing the results of ICS, CCS, and CNMF channel selection techniques with ShallowConvNet and EEGNet for different numbers of channels in the subset. Results for Graz 2A dataset.

	ShallowConvNet			EEGNET		
Subset Size	ICS	CCS	CNMF	ICS	CCS	CNMF
3	**49.66%**	49.51% (p=0.948)	45.93% (p=0.195)	**50.17%**	42.14% (p = 0.012)	48.58% (p=0.327)
6	**59.17%**	55.88% (p=0.546)	51.71% (p= 9.3e-3)	**57.84%**	48.83% (p= 0.008)	52.04% (p=0.030)
11	**63.47%**	61.32% (p=0.546)	59.64% (p=0.387)	**59.70%**	57.20% (p = 0.652*)	57.76% (p=0.164*)

* A Wilcoxon signed-rank test was used.

Table 6.6: Comparing the results of ICS, CCS, and CNMF channel selection techniques with ShallowConvNet and EEGNet for different numbers of channels in the subset. Results for the HG dataset.

	ShallowConvNet			EEGNET		
Subset Size	ICS	CCS	CNMF	ICS	CCS	CNMF
3	**68.29%**	60.03 (p=1.45e-05)	55.75% (p=1.0e-04)	**70.63%**	59.85% (p= 6.7e-05)	55.90% (p=4.1e-05)
6	**78.56%**	70.20% (p=2.7e-06)	65.51% (p=2.9e-07)	**75.90%**	70.01% (p=0.008*)	65.98% (p=9.7e-05)
11	**83.07%**	77.29% (p=0.001)	75.95% (p=8.0e-05)	**79.91%**	75.56% (p=8.5e-04)	74.03% (p=0.013*)
22	**86.95%**	88.72% (p=0.029)	82.89% (p=0.002)	**82.09%**	81.25% (p=0.274)	79.71% (p=0.036)

* A Wilcoxon signed-rank test was used.

6.4.3 Comparing the ICS Layer Method to GA Channel Selection

As explained previously in Section 6.3.5, the comparison was conducted for subject-independent channel selection when using the Graz 2A dataset, and when a subset of 11 channels was selected. Table 6.7 compares the classification performance of the ICS layer method to the performance of the GA channel selection method for ShallowConvNet and EEGNet. The peak results for ShallowConvNet and for EEGNet are highlighted in bold.

Considering the average results (bottom row), the ICS layer method performs slightly better than GA channel selection in terms of classification accuracy. Considering the results for individual subjects, the best channel selection method tended to vary with the subject-classifier pairing. For example, the ICS layer method gave a higher accuracy for subject A3 when using ShallowConvNet, whilst the GA gave a better performance for the same subject when using EEGNet.

A statistical analysis of the results was also carried out. Since an Anderston-Darling test indicated that the results for the ICS layer were non-normal, a Wilcoxon sign-rank test was used for comparisons. When comparing the results of the ICS layer and GA methods for subjects A1-A9 for the ShallowConvNet classifier, a p-value of 0.0391 was obtained. Since $p < 0.05$, this result indicates a significant difference in performance between the two methods.

Table 6.7: Comparing the classification accuracy results obtained with subject-independent channel subsets selected using the ICS layer method and the GA channel selection method from Chapter 5.

	ShallowConvNet		EEGNet	
Subject	ICS Layer Method	GA	ICS Layer Method	GA
A1	**77.71%**	75.76	**76.39%**	70.28%
A2	**45.14%**	44.44	40.90%	**43.06%**
A3	**80.14%**	80.00	78.96%	**84.44%**
A4	**56.77%**	53.75	48.96%	**51.25%**
A5	**42.57%**	35.83	**28.47%**	26.18%
A6	**42.29%**	39.65	**40.28%**	39.86%
A7	77.22%	**77.50**	**70.97%**	65.69%
A8	77.74%	**77.98**	**75.96%**	75.81%
A9	**71.67%**	69.24	76.39%	**77.64%**
Average	***63.47%***	*61.57%*	***59.70%***	*59.36%*

When comparing the results for EEGNet, a *p*-value of 0.8438 was obtained, indicating there was no significant difference in the performance.

These results indicate that channel subsets obtained by the ICS layer channel selection method and the GA channel selection method performed similarly in terms of classification accuracy. However, on average, the ICS layer method tended to outperform the GA channel selection method by a small margin for ShallowConvNet.

6.4.4 Transfer Learning for Improved Performance

Ideally, reducing the number of EEG channels through channel selection should not result in a significant decrease in performance when compared to using all the EEG channels in the dataset.

Consider the results in Table 6.8 and Table 6.9 for the Graz 2A dataset when using the ICS layer method. Table 6.8 shows the results for ShallowConvNet and Table 6.9 shows the results for EEGNet. The tables compare the categorical accuracies obtained when using the full EEG montage of 22 channels ('All Channels') to the case when a subset of 11 channels were used with ShallowConvNet and EEGNet. Results for random weight initialization (w/o TL), for RTL (w. RTL) and for MTL (w. MTL) were compared to the results obtained using all channels. Statistical comparisons were carried out using Wilcoxon signed-rank tests since the data was found to be non-normal, and the *p*-values for

*Table 6.8: For ShallowConvNet- Comparing the categorical classification accuracy obtained when using the full cohort of 22 channels in the **Graz 2A dataset** to using 11 channels selected via the ICS layer method, when using randomly initiated weights in the CNN classifier and when using transfer learning.*

Subject	All Channels	11 Channels w/o TL	11 Channels w. RTL.	11 Channels w. MTL
A1	79.50%	77.71%	81.18%	**83.89%**
A2	48.79%	45.14%	46.94%	**56.32%**
A3	80.21%	80.14%	80.07%	**84.72%**
A4	63.37%	56.77%	63.44%	**65.10%**
A5	43.21%	42.57%	52.78%	**57.01%**
A6	44.86%	42.29%	46.94%	**54.38%**
A7	81.29%	77.22%	76.88%	**82.50%**
A8	79.84%	77.74%	**78.98%**	78.44%
A9	75.86%	71.67%	**72.08%**	67.92%
Average	*66.33%*	*63.47%* *(p=0.004*)*	*66.59%* *(p=0.8203*)*	***70.03%*** *(p=0.129*)*

* Calculated using a Wilcoxon sign-rank test

*Table 6.9: For EEGNet- Comparing the categorical classification accuracy obtained when using the full cohort of 22 channels in the **Graz 2A dataset** to using 11 channels selected via the ICS layer method, when using randomly initiated weights in the CNN classifier and when using transfer learning.*

Subject	All Channels	11 Channels w/o TL	11 Channels w. RTL.	11 Channels w. MTL
A1	73.64%	76.39%	**77.71%**	76.88%
A2	44.07%	40.90%	45.28%	**51.81%**
A3	81.86%	78.96%	81.74%	**84.03%**
A4	52.61%	48.96%	**52.40%**	50.42%
A5	36.29%	28.47%	**46.88%**	43.54%
A6	44.00%	40.28%	49.31%	**52.57%**
A7	77.21%	70.97%	72.29%	**75.90%**
A8	78.04%	75.96%	**78.65%**	78.24%
A9	79.43%	76.39%	**77.08%**	67.64%
Average	*63.02%*	*59.70%*	***64.59%***	*64.56%*
		(p=0.012)*	*(p=0.426*)*	*(p=0.426*)*

*Calculated using a Wilcoxon sign-rank test

each comparison are recorded in the final row of the tables. The highest accuracies are highlighted using bold font.

For both ShallowConvNet and EEGNet, there was a significant decrease in classification performance when using the reduced channel subset without transfer learning (p<0.05). Using either transfer learning strategy led to an improvement in average accuracy, and there was no statistically significant difference (p>0.05) between the results using the channel subsets with transfer learning and the full montage for either transfer learning approaches. Using MTL provided noticeably improved performance when compared to using RTL for ShallowConvNet, although the improvement due to MTL, when compared to using all channels, was not significant (p>0.05).

Consider now the results for the HG dataset, presented in Table 6.10 and Table 6.11. RTL did not have as noticeable an impact on the average accuracy as it did with the Graz 2A dataset, with a change of less than 1% when compared to not using transfer learning. However, like the Graz 2A dataset, when no transfer learning was used, the results were significantly different to the case when all channels were used (p<0.05), whereas when RTL was used, there was no significant difference (p>0.05). Considering the results for MTL, there was always a significant decrease in classification performance when compared to using the full montage (p<0.05).

*Table 6.10: For ShallowConvNet- Comparing the categorical classification accuracy obtained when using the full cohort of 44 channels in the **HG dataset** to using 22 channels selected via the ICS layer method, when using randomly initiated weights in the CNN classifier and when using transfer learning.*

Subject	All Channels	11 Channels w/o TL	11 Channels w. RTL.	11 Channels w. MTL
H2	83.62%	**87.25%**	83.13%	80.13%
H3	95.50%	**96.75%**	96.63%	92.00%
H4	94.25%	**97.25%**	96.00%	93.88%
H5	93.00%	90.63%	**94.25%**	84.63%
H6	95.75%	91.88%	**93.00%**	87.63%
H7	84.15%	**83.27%**	81.01%	79.37%
H8	95.00%	91.50%	**94.25%**	91.25%
H9	82.38%	**83.25%**	75.88%	68.00%
H10	87.25%	84.00%	**86.88%**	79.75%
H11	78.63%	75.88%	**78.13%**	74.25%
H12	96.88%	**96.88%**	96.25%	91.38%
H13	94.72%	**92.83%**	89.43%	83.27%
H14	67.50%	59.00%	**63.38%**	62.63%
Average	*88.36%*	***86.95%***	*86.78%*	*82.16%*
		(p=2.4e-14)*	*(p=0.091*)*	*(p=2.44e-4*)*

*Calculated using a Wilcoxon sign-rank test

*Table 6.11: For EEGNet- Comparing the categorical classification accuracy obtained when using the full cohort of 44 channels in the **HG dataset** to using 22 channels selected via the ICS layer method, when using randomly initiated weights in the CNN classifier and when using transfer learning.*

Subject	All Channels	22 Channels w/o TL	22 Channels w. RTL.	22 Channels w. MTL
H2	84.50	**83.13%**	79.00%	76.13
H3	93.88	**92.63%**	92.50%	90.63
H4	97.50	**96.88%**	96.63%	91.13
H5	90.63	**88.88%**	86.00%	79.38
H6	88.13	89.63%	**90.50%**	77.75
H7	77.48	71.45%	**81.89%**	73.21
H8	86.63	85.25%	**91.88%**	81.63
H9	78.00	**72.88%**	68.50%	65.38
H10	87.50	86.25%	**86.88%**	83.38
H11	69.13	68.75%	**70.50%**	68.00
H12	92.37	92.88%	**93.50%**	86.63
H13	87.55	**79.25%**	78.49%	72.20
H14	64.00	59.38%	**60.63%**	58.88
Average	*84.41%*	*82.09%*	***82.84%***	*77.25%*
		(p=4.8e-12)	*(p=0.091)*	*(p=4.44e-4)*

The Graz 2A dataset may have benefitted more from RTL because it has less training data per subject than the HG dataset. For the Graz 2A dataset, transfer learning may have enabled the classifiers to learn generalizable information that improved accuracy. In the case of the HG dataset, there may not have been much additional information that could be extracted from the source data that could already be obtained from the target data. Furthermore, the data in the HG dataset was recorded within a Faraday cabin with electromagnetic shielding [8]. This is an idealistic experimental condition that reduces the impact

of environmental noise [8]. In the case of the Graz 2A dataset, results were recorded under more 'practical' conditions without electromagnetic shielding [72], [111], meaning that the data may contain more noise. Thus, RTL may have been more effective with the Graz 2A dataset because it helps the CNN to learn more meaningful representations from the data by reducing overfitting to noise.

These results indicate that RTL had the potential to ensure that EEG classifiers operating with reduced channel subsets performed on-a-par with classifiers trained using the full montage. In fact, when using RTL, the number of EEG channels used could be halved without any statistically significant change in performance when compared to using all channels. Furthermore, RTL could improve the categorical classification accuracy when using the reduced channel subset, but this effect depends on the dataset used. When compared to MTL, RTL had a more stable performance across datasets and classifiers. The results suggest MTL may run the risk of leading to a significant deterioration in performance, thus RTL is recommended.

6.4.5 Execution Time Analysis

6.4.5.1 The Impact of Subject-Independent Channel Selection on Target Training Times

Table 6.12 records the average training latencies introduced when subject-specific channel selection was carried out on ShallowConvNet and EEGNet, using the Graz 2A and HG datasets. The times are longer for the HG dataset because it has more training trials per subject than the Graz 2A dataset, as previously discussed in Section 6.3.1. The times in Table 6.12 capture the average time saved per subject when using subject-independent channel selection, since channel

Table 6.12: A table showing the average latency introduced by subject-specific channel selection for ShallowConvNet and EEGNet, when using the Graz 2A and HG datasets.

	Subject-Specific Channel Selection Latency Times/s	
	ShallowConvNet	EEGNet
Graz 2A Dataset	44.44	35.89
HG Dataset	208.12	142.77

selection does not introduce a latency experienced by the subject. In the case of the Graz 2A dataset, the saving is of less than a minute, whereas for the HG dataset, the time saving was of approximately 3.5 minutes for ShallowConvNet and 2.4 minutes for EEGNet. The time saving is linked to the size of the dataset used, with larger datasets experiencing a greater saving since the CNN takes longer to train when there is more data. These results confirm that performing subject-independent channel selection can lead to a notable decrease in training phase time, particularly for larger datasets.

6.4.5.2 The Impact of Channel Selection on Classifier Training Times

The impact of using a reduced channel subset on target classifier training times was analysed. In the previous section, it was shown that the number of EEG channels used could be halved without any significant change in categorical accuracy if RTL is used. This sub-section is focused on comparing the training execution times when half the EEG channels are used to when the full cohort was used. In RTL, the pre-training on source data can be carried out prior to the target using the system, and therefore is not factored in the training times. This sub-section is interested in the latency a target subject would experience between recording of their own training data and being able to use the system, which would encompass just the retraining step.

Table 6.13 and Table 6.14 show the average training times and worst-case training times recorded on the Graz 2A and HG datasets when all channels were used, and when they were halved. Reducing the number of channels led to an improvement in the average and worst-case testing times for both datasets. The benefit appeared to depend on the dataset used: when the Graz 2A dataset was used, the improvement in the average training time for ShallowConvNet was of 7.1s and 3.27s for EEGNet, which would be negligible in a practical scenario. However, for the larger HG dataset, the improvement in average timings was of 68s and 44s for ShallowConvNet and EEGNet, respectively. The difference in

Table 6.13: Comparing the average training times and worst-case training times for the ShallowConvNet and EEGNet classifiers for the full EEG montage of the Graz 2A dataset (22 channels) and half the montage.

	Average Training Times/s		Worst Case Training Times/s	
	22 Channels	11 Channels	22 Channels	11 Channels
ShallowConvNet	40.95	**33.87**	45.30	**37.75**
EEGNet	33.55	**29.83**	37.39	**35.30**

Table 6.14: Comparing the average training times and worst-case training times for the ShallowConvNet and EEGNet classifiers with the full EEG montage of the HG dataset (44 channels) and half the montage.

	Average Training Times/s		Worst Case Training Times/s	
	44 Channels	22 Channels	44 Channels	22 Channels
ShallowConvNet	180.94	**112.74**	199.44	**122.32**
EEGNet	132.61	**88.72**	146.71	**100.21**

results between the datasets is attributable to the difference in their training data sizes. In both cases, however, the improvement in training times was modest from a practical perspective.

6.4.5.3 Comparing the Classifier Training Latencies attributed to RTL and MTL

Table 6.15 and Table 6.16 show the classifier training latencies attributed to RTL and MTL for the Graz 2A and HG datasets, respectively. It is immediately evident that MTL resulted in a notably larger latency than RTL. The best results for each classifier are shown in bold. For the Graz 2A dataset, averaging the mean results for both CNNs, the training latency using RTL was under half a minute, whilst the latency attributed to MTL was 2.7 minutes. Since the HG dataset had a larger

Table 6.15: Comparing the average and worst-case latency times (in s) contributed by RTL and MTL for ShallowConvNet and EEGNet classifiers when using the Graz 2A dataset with 11 channels selected.

	Average Latency Times/s		Worst Case Latency Times/s	
	RTL	MTL	RTL	MTL
ShallowConvNet	**27.03**	164.79	**27.50**	176.65
EEGNet	**24.53**	159.22	**25.09**	290.69

Table 6.16: Comparing the average and worst-case latency times (in s) contributed by RTL and MTL for ShallowConvNet and EEGNet classifiers when using the HG dataset with 22 channels selected.

	Average Latency Times/s		Worst Case Latency Times/s	
	RTL	MTL	RTL	MTL
ShallowConvNet	**114.29**	529.00	**117.07**	528.71
EEGNet	**85.31**	398.61	**86.48**	390.46

amount of training data and more subjects than the Graz 2A dataset, the discrepancies between the RTL and MTL latencies are even more notable. Averaging the mean results for both CNNs, MTL contributed 7.73 minutes of training latency, whilst RTL contributed just 1.7 minutes. Papers in the literature use MTL [65], [172], [174], and this analysis has highlighted the computational cost of MTL when compared to RTL. To conclude, RTL has a more stable performance in terms of accuracy across classifiers and datasets than MTL, and the target-user training latency was approximately 4.5 times shorter.

6.4.5.4 Comparing the Subject-Independent Channel Selection Times for the ICS layer Method and Comparison Systems

Table 6.17 shows the average subject-independent channel selection latencies for the ICS layer method and the CCS, CNMF, and GA methods. The results for both ShallowConvNet and EEGNet, obtained using the Graz 2A dataset, are shown. The CNMF method has the lowest latency and its results have been emphasized in bold. Although the ICS layer method has a longer latency than the CCS and CNMF methods, it still takes under a minute to select the channels. The GA has a notably longer training time than the other methods, taking approximately 10 minutes and 11 minutes to select channels for ShallowConvNet and EEGNet, respectively. Although, in practice, an 11-minute training time might be acceptable, this analysis highlights the computational efficacy of the proposed ICS layer method, which produces similar classification performance to GA channel selection (as previously discussed in Section 6.4.3), but selects channels approximately 12 times faster for ShallowConvNet and 17 times faster for EEGNet.

Table 6.17: Comparing the average subject-independent channel selection latencies of the ICS layer method and comparison methods, namely the CCS, CNMF and GA methods. Results for both ShallowConvNet and EEGNet, recorded on the Graz 2A dataset are shown.

	Channel Selection Latency/s	
Method	ShallowConvNet	EEGNet
ICS	49.05	38.55
CCS	28.49	24.55
CNMF	**28.18**	**21.81**
GA	622.16	683.82

The ICS approach was not only faster than the GA channel selection method, it also produced richer information during the processing time. Although the GA channel selection method aimed to select the best subset of 11 channels, it does this through the exploration and exploitation of candidate subsets and does not give a ranking of the importance of each individual channel. Thus, if the subset size were to be changed from 11 channels to some other size, say 5 channels, the GA channel selection process would have to be run again from the start. In contract, the ICS layer method can rank all the channels in the dataset as part of the channel selection process. This means that if the required subset size changes, the rankings just need to be consulted to decide which channels should be used, without needing to re-run the channel selection process. Thus, the ICS layer method is more computationally efficient than the GA channel selection method.

GA channel selection, being a wrapper channel selection method, is expected to take longer than the CCS and CNMF methods, which are filter channel selection methods [44]. The ICS layer method forms part of a newer subset of channel selection methods which are based on CNNs. However, since CNN learning involves backpropagation based on the classification error, this means CNN-based channel selection methods are affected by the classification accuracy. This makes them conceptually similar to wrapper channel selection techniques, which optimize the channel subset based on the classification accuracy. This contrasts with filter-based methods which rank the channels independently of the classifier. Thus, the ICS layer method is a channel selection method that has a sensitivity to the classification error like wrapper methods, which performed on-a-par with the GA wrapper technique in terms of classification accuracy but was found to operate much faster than the GA wrapper method. This analysis verifies that the ICS layer method is a computationally efficient tool for channel selection in CNNs.

6.4.6 Analysing the Selected Channels

This section discusses the subject-independent channels selected using the ICS layer method. These are the same channel subsets that gave the results for the ICS layer in Table 6.5 and Table 6.6, previously. In the case of the Graz2A dataset, this analysis is focused on the case when 11 channels were selected, and in the case of the HG dataset, it is focused on the case when 22 channels were selected. The scenario when channel selection is used to approximately halve the number of channels was also considered for the channel analysis in Chapter 5 (Section 5.4.4).

Since there are 9 subjects in the Graz 2A dataset, 9 subject-independent channel subsets were obtained for both ShallowConvNet and EEGNet. Similarly, 13 subject-independent channel subsets were obtained for both ShallowConvNet and EEGNet when using the HG dataset.

Figure 6.8 shows the frequencies that each of the channels in the Graz 2A dataset were selected in the channel subsets. The top image shows the results for ShallowConvNet and the bottom image shows the results for EEGNet. 'Frequency of Selection' means the number of times that a channel was included in a selected

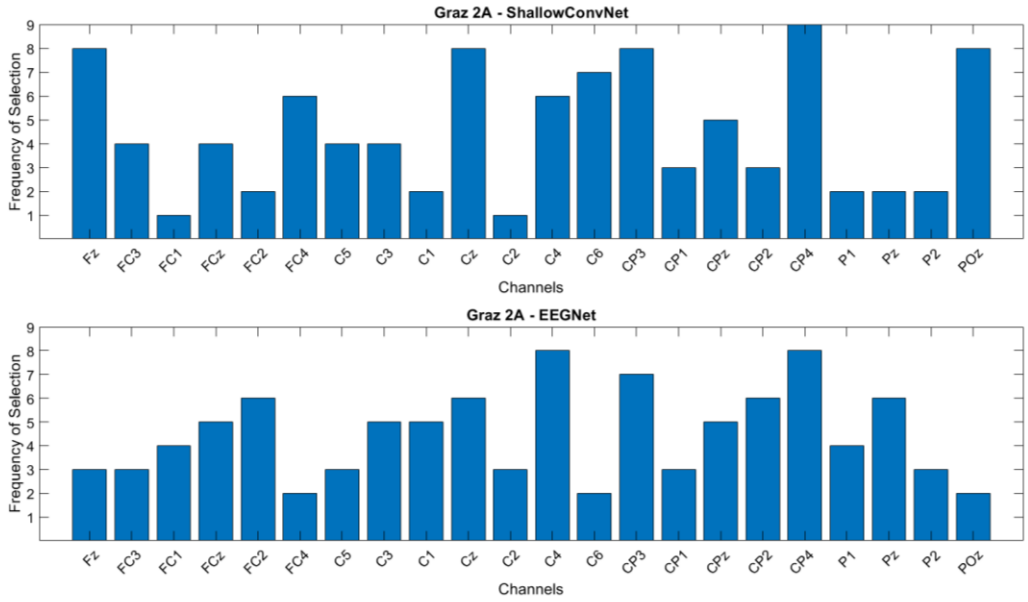

Figure 6.8: Bar plots showing the frequency of selection of different channels from the Graz 2A dataset.

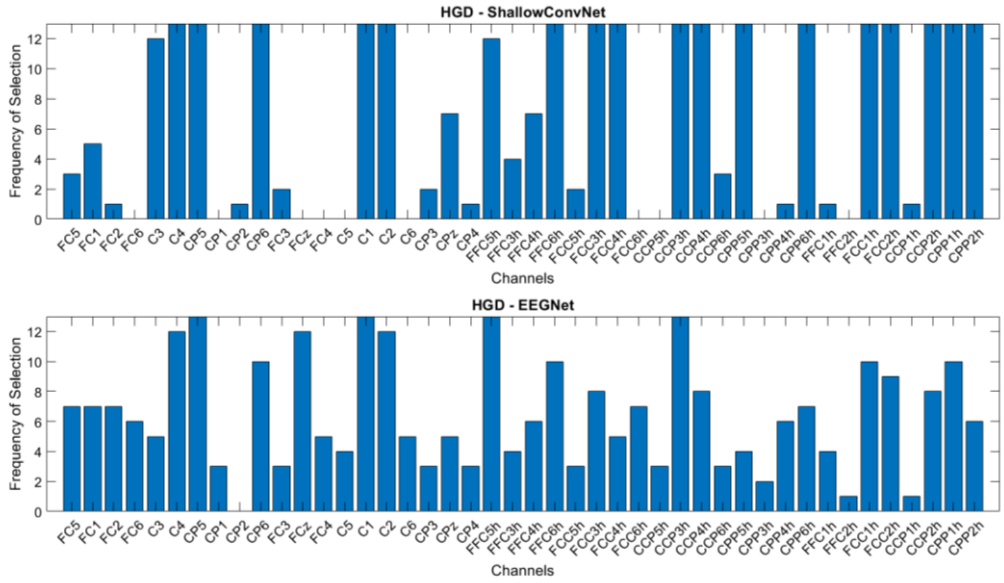

Figure 6.9: Bar plots showing the frequency of selection of different channels from the HG dataset.

channel subset. So, for example, channel Fz was selected in the channel subsets of 8 out of the 9 subjects when ShallowConvNet was used. Figure 6.9 shows similar results for the HG dataset.

Figure 6.10 and Figure 6.11 show the EEG recording montages used in the Graz 2A dataset, with the most popular electrodes selected by the ICS layer circled in red. Figure 6.10 shows the most popular channels selected for ShallowConvNet and Figure 6.11 shows the most popular channels for EEGNet. Figure 6.12 and Figure 6.13 show similar results for the HG dataset. A channel is 'popular' if it was selected in more than half the subsets, which corresponds to 5 or more subsets in the case of the Graz 2A dataset and 7 or more in the case of the HG dataset. Note that the frontal (F) electrodes are at the front of the scalp, above the face.

From the bar charts in Figure 6.8 and Figure 6.9, it is evident that the channels detected by the ICS layer depend, to some extent, on the classifier used. For example, considering the results of the Graz 2A dataset in Figure 6.8, channel Fz was selected in 8 out of the 9 subsets for ShallowConvNet, but was selected in only 3 out of the 9 for EEGNet. Conversely, channel Pz was only selected in 2 out

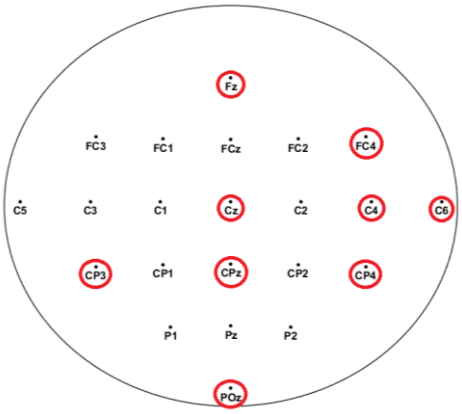

Figure 6.10: The most popular channels selected from the Graz 2A dataset when using the ICS layer method with ShallowConvNet.

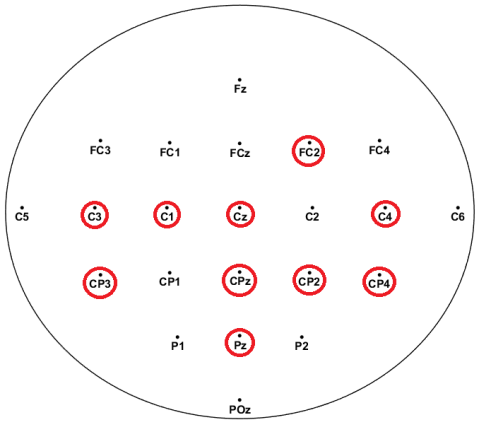

Figure 6.11: The most popular channels selected from the Graz 2A dataset when using the ICS layer method and EEGNet.

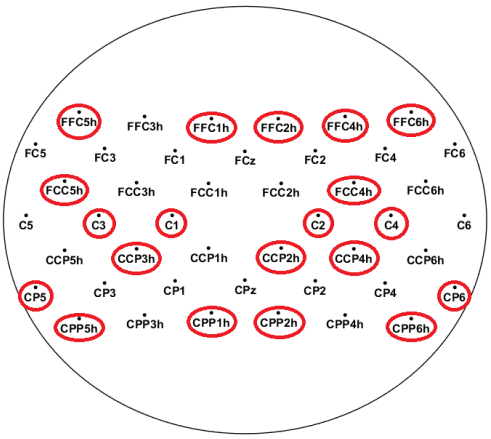

Figure 6.12: The most popular channels selected from the HG dataset when using the ICS layer method with ShallowConvNet.

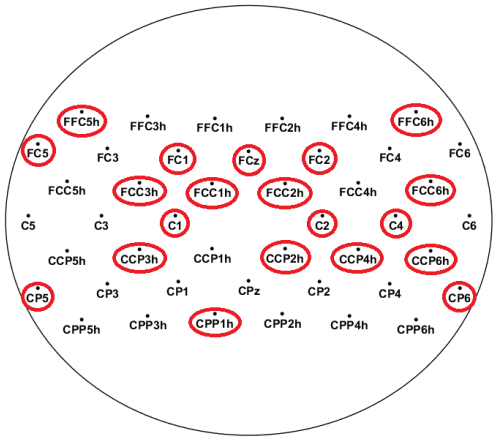

Figure 6.13: The most popular channels selected from the HG dataset when using the ICS layer method with EEGNet.

of the 9 subsets with ShallowConvNet but was selected in 6 out of 9 when EEGNet was used. Considering the results for the HG dataset in Figure 6.9, channel CPP5h was selected in 13 out of 13 of the channel subsets for ShallowConvNet, but in only 4 out of 13 when EEGNet was used. Conversely, channel FCz was not used in any subset for ShallowConvNet but in 12 with EEGNet.

Notwithstanding this, the ICS layer determined some channels to be important for both classifiers. Examples of these for the Graz 2A dataset are CP4 and CP3, whereas examples for the HG dataset are C4, CP5, CP6, C1, C2, FFC5h and CCP3h.

Consider the scalp electrode results for the Graz 2A dataset in Figure 6.10 and Figure 6.11. Comparing the results obtained for ShallowConvNet and EEGNet, it is evident that the most popular electrodes for ShallowConvNet are skewed towards the right-hand side, whereas those for EEGNet are more balanced across both hemispheres. Considering the results for the HG dataset in Figure 6.12 and Figure 6.13, ShallowConvNet had more channels in the FFC and CPP regions, whereas EEGNet had more channels in the FC and FCC regions. For both classifiers, the C and CCPh electrodes were popular.

Whilst this discussion was based on a purely observational analysis, it is possible to draw some limited preliminary conclusions. These results indicate that the ICS layer method could have the potential to tailor the channel subsets to the classifier. This makes sense since the weights in the ICS layers are optimized by backpropagation through the classifier network itself. This means it could possibly be used with different CNN networks with similar success. The ICS channel selection layer also identified some universally important channels across classifiers, possibly indicating that it was able to identify channels important for MI activity, regardless of the classifier used.

6.5 Conclusion

This chapter presented a novel ICS layer method for subject-independent channel selection in CNN classifiers. The proposed method was shown the be versatile, achieving good performance with two different CNN classifiers on two different datasets.

The proposed, novel, ICS layer was found to be effective for subject-independent channel selection and did not result in any statistically significant deterioration when compared to subject-specific channel selection. Effective methods for subject-independent channel selection are important, since subject independent channel selection can lead to faster training times, lower hardware costs and improved user comfort, as previously discussed in Chapter 3. In fact, using the ICS layer for subject-independent channel selection resulted in notable

decreases in the target training phase when compared to carrying out subject-specific channel selection. The magnitude of the decrease depends on the size of the target training dataset, with larger datasets leading to longer training times for subject-specific channel selection, and thus the positive impact of subject-independent channel selection is more notable.

For subject-independent channel selection the ICS layer method gave a higher categorical classification accuracy when compared to two other state-of-the-art channel selection methods [13], [47]. In 68% of the experiments, this improvement was statistically significant. The ICS layer channel selection method also had a similar classification performance to the GA channel selection method presented in Chapter 5 when applied to the subject-independent channel selection problem. The average accuracy of the ICS layer method was consistently greater than that of the GA channel selection method, although the difference between the two methods were not found to be statistically significant.

The ICS layer method was found to be more computationally efficient than the GA channel selection method and completed the subject-independent channel selection process a factor of 12-13 times faster. In fact, the ICS layer method completed the channel selection process in under a minute, exhibiting a execution time similar to the state-of-the-art filter-based methods used for comparison [13], [47]. Furthermore, whilst the GA channel selection method produces a candidate subset of channels based on a subset size, z, fixed *a priori* by the user, the ICS channel selection layer produces a ranking for all channels, and the z channels with the highest scores are chosen for the subset. If, later, a larger or smaller subset of channels is needed, the GA channel selection method would have to re-run from the start, whereas using the ICS layer method, the rankings would just need to be consulted to know which channels to add or remove in order of importance. These results indicate that the ICS layer method is superior to the GA method for channel selection in a CNN, since the classification accuracies obtained with the two methods are similar, but the ICS layer method provides richer information in a shorter time.

RTL was found to be an effective method for improving CNN performance when channel selection is used. In fact, RTL enabled the number of EEG channels to be reduced by half through the ICS layer without any significant decrease in classification accuracy compared to the full cohort of channels. RTL was also found to have a more stable performance across datasets when compared to MTL. Furthermore, the training latency when using RTL is approximately 4.5 times shorter than that of MTL.

Due to its versatility, the ICS channel layer could be applied to other 2D CNN-based classifiers, or to other EEG paradigms. Future work could also look into applying the ICS layer to other paradigms where ShallowConvNet and EEGNet, or architectures based on them, have already shown success, such as emotion recognition [102] or event-related potential detection [7]. In theory, the ICS layer could also be applied to any 2D-input CNN problems outside of EEG classification which involve suppression of irrelevant or less informative data that is known to be present in the input, but which is not obviously identifiable.

Subject-independent channel selection is important because it enables fewer electrodes to be used by the target subject, making for a more practical and easier-to-use BCI experience. This chapter presented a new, viable method for subject-independent channel selection in CNNs and demonstrated a transfer learning approach that preserves performance despite a reduced subset of channels.

Chapter 7 : Conclusions and Further Work

This chapter opens with a summary of the main achievements in this thesis, then discusses the limitations of the contributions made. The future work section explains how the contributions could be applied to a practical, online system. It also discusses possible avenues for new research directly related to the contributions in this thesis, as well as further afield in MI EEG classification.

7.1 Main Achievements

The majority voting-based decision fusion approach in Chapter 4 was found to have the potential to significantly improve the classification performance of LDA, SVM, RF and NB classifiers. An extensive analysis of the effect of the window size and window increment parameters was carried out, and larger windows were found to be correlated with better performance. However, the results suggest that there may be a trade-off between improved performance from more information being included in the feature extraction window, and increased non-stationarity present in EEG segments longer than 1.75s. A peak accuracy of 84.51% was obtained using an LDA classifier, a window size of 1.75s and window increment size of 0.25s. The channel-classifier analysis identified the channel subsets C+CP and C+CP+CF to perform better than electrodes just from the C region or the full montage of 118 channels. The majority-voting based decision fusion approach was computationally lightweight, introducing latencies in the order of milliseconds or microseconds on the training and testing phases, respectively. This means that it can be easily applied to almost any BCI classifier without significant execution time overhead.

The GABSLEEG system in Chapter 5 successfully merged the strong classification capabilities of a dictionary-based SL classifier with a metaheuristic

GA channel selection module. Using the channel selection module to halve the number of channels used led to a 60% improvement in classification times on the test set whilst maintaining classification accuracy. Furthermore, the GA channel selection module was effective for kNN, RF and SVM classifiers. The GABSLEEG system outperformed GA-kNN, GA-RF and GA-SVM systems in terms of accuracy, sensitivity and specificity, and was the most robust to changes in training data size. The GABSLEEG system was also more accurate than a variety of other systems in the literature, including deep learning approaches and SL classifiers.

The ICS layer method presented in Chapter 6 was an effective channel selection approach for CNNs. It was a versatile approach, found to be effective on two different datasets and classifiers. The ICS approach was appropriate for subject-independent channel selection, with the subject-independent channel subsets performing better than or on a par with subject-specific channel subsets. Furthermore, the ICS layer method generally outperformed similar channel selection approaches in the literature. Transfer learning was found to be effective for maintaining the classification performance of a CNN classifier when using a reduced subset of EEG channels. The RTL approach was 4.5 times faster than the MTL approach, and performed more reliably across datasets and classifiers. The ICS channel selection approach was 12-17 times faster than the GA channel selection approach presented in Chapter 5.

7.2 Limitations of the Contributions

A general limitation across all the work in this thesis was the fact that all the proposed approaches were only tested through offline analysis on datasets. Although the focus of the research was to produce more effective and efficient approaches for practical BCIs, the full value of the contributions made can only be determined through assessment in an online, real-time BCI. Online testing would provide information about the robustness of these approaches to real-time EEG data and the latencies that are experienced during use. This kind of investigation would highlight areas where the proposed approaches could be refined and improved. In light of the fact that such testing was not carried out, the

approaches presented may not be quite appropriate for immediate deployment in an online system.

The datasets used contained data recorded from between 4 and 13 different subjects. Although the datasets were chosen due to their popularity in the literature, their limited size limits the assessment of generalizability of the proposed approaches to large populations of BCI users.

The open-access datasets used only contained data from healthy subjects. Although healthy subjects may use some practical BCIs, such as those related to gaming or interactive design [18], practical BCIs may also be used as part of enabling technologies for subjects that are disabled or have impaired mobility due to disease [20]. BCI technologies may also be used as part of stroke rehabilitation[234]. Since the approaches in this thesis were only assessed using data from healthy subjects, it is unclear whether they would be appropriate for BCIs aimed at subjects who have a disease or disability. Future work could focus on testing the proposed approaches with data recorded from stroke patients, subjects with motor neuron diseases such as ALS, or with data from people with spinal cord injuries.

The approaches in this thesis were only tested on MI data. Since testing was not carried out using data recorded using other EEG paradigms, it is unclear whether the proposed approaches could be effective outside of the domain of MI EEG classification.

Throughout the project, execution times have been reported. Although multiple runs were always carried out in order to obtain an average measure, under 100 runs were always completed. These results provide a record of how long computations took, however for a rigorous analysis of execution times over 1000 runs should have been carried out. Moreover, although effort was made to use good programming practices throughout the project, the code was not optimized before the execution time tests were carried out. Future work could involve optimizing the code before recording the execution times. Finally, although all background processes were suspended whilst the programs were

run, essential operating system operations could still take place whilst the programs ran. Together, all these issues limit the reliability and robustness of the execution time results recorded.

The work in Chapter 4 was only assessed on one dataset. Although it is not uncommon in the literature to assess proposed approaches on a single dataset [104], this is a limitation as it remains unclear whether the multi-segment decision fusion approach could be used to improve performance outside of the dataset used. Furthermore, the effects of multi-segment decision fusion were only assessed in an observational way on the test-set data. To assess generalization capabilities of the approach to a practical system, it would have been appropriate to tune the window size and window increment size on the training dataset and then assess the performance of the selected parameters on the test set.

The GABSLEEG system in Chapter 5 was found to be relatively slow for test set classification when compared to the kNN and RF classifiers. Furthermore, the channel selection latencies for subject-specific channel selection were also found to be significant. This could limit its applicability to a practical system, although with more powerful hardware these issues could be overcome.

7.3 Future Work

This section first discusses how the contributions in this chapter could be applied to a practical system. It then goes on to explain future work that could be carried out directly in relation to the contributions made in this thesis. It presents generic suggestions related to additional testing and hyperparameter tuning, as well as specific technical suggestions related to each contribution.

This section then concludes with a general discussion of future work that could be carried out within the wider domain of MI EEG classification. The suggestions in this part are loosely related to the work in this thesis but go beyond the specific contributions made.

7.3.1 Applying the Contributions to an Online System

The work in the contribution chapters could be applied to new, online systems. This sub-section gives a brief overview of how this could be carried out. Estimations of training data size are based on the datasets used for the offline analysis.

A practical system using the contribution presented in Chapter 4 would use channels in the C+CP or C+CP+CF regions. Training data would be recorded from the subject, and then the window size and increment size could be tuned using the training data. Based on the offline analysis, six minutes of training data per subject should be recorded. This value was calculated by finding the average amount of training data across subjects in the dataset. The window size and increment size that give peak cross-validation classification accuracy on the training data are chosen. The training data could be used to tune the hyperparameters of an LDA classifier for the particular subject. In the chapter, there are LDA parameters that may be a suitable starting point for the search, which gave peak performance over five subjects. During online classification, the incoming data would be windowed using the tuned window size and window increment size, and then majority voting would be used to output a label for each trial.

Based on the work in Chapter 5, a practical system would require training data to be recorded from the subject. Depending on time constraints, 25 minutes of training data would be recommended, but as little as five could be adequate, based on the results. The training data is partitioned into a training and validation set, which are used by the GA to select the best subset of channels. In the contribution chapter the number of channels is reduced from 59 channels to 30, and a similar setup could be used in a practical system. The subject must then wait whilst channel selection is carried out. After the channels are selected, the unselected channels are removed from the SL dictionary, and the EEG cap. Online use of the system can begin, in which incoming EEG data is buffered and labels assigned to 50ms segments of data.

A practical system based on the work in Chapter 6 would involve recording EEG data from a number of source subjects, preferably at least eight subjects. Based on the Graz 2A dataset, which had less training data per subject than the HG dataset, at least 20 minutes of data per source subject should be recorded. The ICS layer method is then used to select a subject-independent subset of channels for either EEGNet [7] or ShallowConvNet [66]. Unselected channels are removed from the training data, and the CNN classifier is trained using the source data. This data recording from source subjects, channel selection and pre-training of the CNN on source data can be carried out ahead of time. Training data is then recorded from the target subject using only the selected channels. Again, twenty minutes of training data is recommended. This data can then be used to fine-tune the CNN classifier, which can be used for online classification on 2s-long windows of data.

The discussion in this section was carried out at a high-level and is hypothetical. Since the techniques have only been tested on offline classification problems, it is likely that further refinement would be needed before they can be used in online systems, since not all issues that can arise in online systems can be foreseen in an offline analysis. For example, the classifiers may need to be periodically updated in an online system due to drift in EEG signal characteristics due to their non-stationarity [235]. Also, it is known that EEG classifiers in online systems can spuriously misclassify samples [19]. Some researchers apply smoothing on the output of the classifier to improve performance [19], and this kind of post-processing may also be needed if the approaches presented were applied to online systems. These kinds of adjustments would be carried out as part of refinement of the online system.

7.3.2 Furthering the Contributions

7.3.2.1 Additional Testing for Algorithm Evaluation

In this thesis, a variety of standard, open-access datasets were used [2], [8], [71], [72]. These datasets have been widely used to present new machine learning and deep learning approaches, often without any proprietary data recorded by the

researchers themselves being used to further validate the approaches presented [6]–[8], [10], [13], [48], [60]. However, if new data were recorded, it could be used to validate the techniques presented in this thesis, particularly the channel selection methods. Although the literature often uses open-access datasets to present results for channel selection [13], [46], [48], [49], [225], there could be electrical interference between electrodes [79], [236]. Since the open-access datasets are recorded using a full montage of electrodes, removing electrodes at the signal processing stage may not remove the effects of other electrodes on those left in the dataset. In an ideal scenario, only the chosen subset of electrodes would be used at the testing stage for subject- specific channel selection, and in the recording of target training and testing data in the case of subject-independent channel selection.

The work in this thesis was focused on offline MI EEG classification, where data is recorded and then processed at a later stage. Conversely, when using a real-time system, training data is recorded from the subject, algorithm training is carried out, and then the subject uses the BCI in real time. Although offline analysis is widespread on the literature [7], [8], [10], [11], [13], [46], [48], [49], [60], [225] and can be a necessary step in developing new techniques that can be applied to real-time systems, it still does not replace the validation of novel techniques within the context of a real-time system. Only when applying novel techniques to a real-time system can practicality and effectiveness truly be assessed, because it provides insight into user tolerance for training data recording and algorithm training times, it can highlight unacceptable latencies during real-time classification, and it can provide an idea of the robustness of the algorithm to the concentration spans and signal noise present in a practical scenario. In this thesis, there was a strong focus on algorithm practicality, with key aims involving improving execution times and reducing the number of electrodes used. The next step in validating these techniques would be applying these techniques to a real-time system.

Finally, the techniques proposed in this thesis could be applied to other kinds of EEG classification problems, such as emotion recognition or concentration tasks.

7.3.2.2 Hyperparameter Tuning with Bayesian Optimization

Grid-searches were mostly favoured in this thesis for hyperparameter tuning because they have been widely used as a reliable method in the literature. Notwithstanding this, Bayesian optimization has been shown to outperform the traditional grid-search method for different applications, both in terms of speed and quality of parameters [90]. In Chapter 5, Bayesian optimization was used to tune the conventional classifiers. Future work could investigate whether Bayesian optimization is effective for tuning SL classifiers for MI EEG classification. Also, the tuning of the GA channel selection approach in Chapter 5 was a computationally involved process and applying a Bayesian optimization framework to this tuning problem may lead to substantially reduced execution times.

7.4.2.3 Furthering the Specific Contributions

Future work into the multi-segment decision fusion technique presented in Chapter 4 could investigate applying the decision fusion approach to a pipeline which uses algorithmic channel selection. Different automated channel selection methods could be used in this analysis. In particular, the effectiveness of the decision fusion method in boosting accuracy as the number of EEG channels is reduced could be investigated. Furthermore, instead of using majority voting, other decision fusion techniques [237] such as decision templates [238], Borda count [237], behaviour knowledge spaces [237] or summed or averaged probabilities in the case of classifiers that can output the probability of class membership [239], could be investigated as ways of merging the time-domain classification results.

One of the challenges identified for using the GABSLEEG system (Chapter 5) in practice is the execution times for GA channel selection and test-set execution times for the SL classification module. Developing GPU or FPGA-based

implementations of the OMP encoding algorithm used in the SL classification module could substantially improve the speed of both parts of the GABSLEEG algorithm [168], [224], [240]. Also, other metaheuristic algorithms could be considered for channel selection, such as differential evolution [104] or the firefly algorithm [125], which have been effective for feature selection in MI EEG classification [104], [125].

Another limitation of the GABSLEEG system which could be solved in future work is its high memory demands, both due to the storage of the dictionary in memory and the memory required for OMP encoding. The dictionary size could be reduced by selecting the most salient entries using techniques from band selection in sparse learning systems in hyperspectral imaging [230], [241]. In hyperspectral sensing, multiple images taken at different frequency bands are available. In recent years, sparse representation has been used to select the most salient spectral bands [230], [241]. The hyperspectral matrix is broken down into a dictionary and sparse encoding using techniques based on k- single value decomposition (kSVD). Within the dictionary, each column is related to a different frequency band, and bands can be selected by analysing the sparse encoding: bands that have larger coefficients in the encoding are more important [230]. There are refinements of this technique for sparsity-based band selection in hyperspectral images [241]–[243]. In the work in Chapter 5, each column of the dictionary is related to a training data sample. Techniques like those used for hyperspectral band selection could therefore be applied to the dictionaries used in MI EEG classification to identify which samples provide the most information. This selection process could be done at the training stage, by partitioning the training data into a subset for dictionary construction and a validation set which is encoded over the dictionary and used for dictionary minimization. If a method of developing smaller dictionaries for MI EEG classification can be found, this could also pave the way for the development of cross-subject dictionaries containing data from multiple subjects, that could be used for subject-independent channel selection.

The CNNs to which the ICS layer was applied in Chapter 6, and networks based on them such as S-EEGNet [102], have been used for classifying other kinds of EEG data, not just MI. Examples include P300 visually evoked potentials [7], error-related responses [7] and emotion recognition [102]. Future work could involve applying the ICS layer method to channel selection within the context of different EEG classification problems.

The transfer learning step in the proposed framework could also benefit from source data selection. Since EEG data has high inter-subject variability [81], [82] and some BCI source subjects could suffer from BCI illiteracy [89], it may be beneficial to rank source subjects and select those which are most likely to positively contribute to the transfer learning process. This selection process could be done using a validation set of source subjects, and would aim to reduce the presence of noisy or illiterate subjects in the source data. Alternatively, it could be carried out using a sample of the target training data. Examples of source selection in the literature have been based on measuring the additional information a source can provide [244] or the classification accuracy associated with the source when using a small amount of target data [29]. These approaches could be applied to source selection for EEG transfer learning.

The ICS layer method was only applied to 2D CNNs, which are widely used in the literature [6], [9], [58], [60], [65], [102]. However, it could easily be applied to 3D CNNs, which are gaining popularity in the MI EEG classification literature [6]. Future work could apply the ICS layer method to 3D CNNs for MI EEG classification.

7.3.3 Future Work in the Wider Sphere of MI EEG Classification

An ultimate goal in MI EEG classification research is a subject-independent classifier where no subject-specific data is used for any part of training. Due to the high inter-subject variability in EEG data and the requirement for strong performance in commercial BCIs, a transfer learning approach which uses minimal training data from a target subject is a more practical goal for BCI classification. To improve transfer learning performance or to create better

subject-independent systems, decomposition techniques could be explored as pre-processing techniques for extracting universal signal components from EEG signals. Within the BCI literature, the Hilbert-Huang transform [102], [209], and empirical mode decomposition [22], [214] are frequently used, however other decomposition techniques such as singular spectrum analysis (SSA) [245], [246], which has recently shown promise for EEG signal processing [247], [248], could be explored in more depth. Furthermore, 2D CNNs have been widely researched for subject-independent EEG classification [7], [9], [58]–[60], and 3D CNNs may be a new frontier for development [6]. CNNs with inception layers have excelled at MI EEG classification based on subject-specific training and may also hold promise for improved subject-independent classifiers [9]. However, these CNNs have been heavily tuned to the datasets used, and so inception-based architectures with stronger generalization capabilities need to be developed [9].

Signal decomposition techniques could also be applied to the problem of data augmentation. A recent data augmentation technique has explored extracting noise from EEG signals using filtering and then adding the noise to filtered training samples to produce an augmented training dataset [9]. Instead of using filtering to extract the noise, decomposition techniques could be used to extract noise components and mix these with the main MI EEG signal components to augment the dataset. Decomposition techniques such as SSA, wavelets and empirical mode decomposition may be effective for this.

The development of novel machine learning and deep learning techniques is fundamental to BCI research; however, this development is limited by the data available. A practical and commercial BCI should, ideally, be like other electronic devices such as phones and laptops: an initial setup is carried out, and then fine-tuning of settings over time is quick and involves minimal effort from the user. In a practical BCI, extensive training by the user should ideally be carried out once, and then the device can be used at different times or on different days with minimal training. These systems may require adaptive algorithms that use few-shot or zero-shot learning and may need to be able to compensate for slightly different positions of electrodes. Many algorithms in the literature are not trained

on data recorded on one day and tested on data recorded on another day [6]–[10], [12], [49], [60]. Reasons for this dearth of research may be a lack of awareness, but also a lack of data, since there are few datasets available with significant gaps in time between the training data recording and test data recording [71], [249]. New, larger datasets with BCI data recorded on multiple days from the same pool of subjects would be important for the development of BCIs that would require minimal retraining during different sessions.

Many EEG databases also depend on data recorded using laboratory-grade equipment [8], [71], [72], [111]. However, the domain of portable biotech is growing [250], [251], and open-access datasets that provide data recorded from consumer-level EEG devices would help aid the development of algorithms that can deal with data that may be closer to a practical scenario.

Single-limb MI EEG classification, in which the MI classes are derived from movements imagined within the same limb, for example left hand open and left hand closed, is generally more challenging than multi-limb MI EEG classification, such as imagined left-hand and right-hand movement [6]. Single-limb MI may be useful for some MI applications such as controlling an avatar in a game and in neurorehabilitation. It is acknowledged that more research and development is required in single-limb EEG data [6]. Future work could therefore also involve the publication of a large open-source dataset with single-limb MI data.

The work in this thesis contributed to the area of offline signal processing for the classification of MI EEG. Offline signal processing research is often a necessary preliminary step which precedes developments in practical, online BCIs [3]. Future work could focus on bridging the gap between classification approaches that work well on EEG datasets, such as those presented in this thesis, and their application in online BCIs.

References

[1] M. Van Steen and G. Kristo, “Contribution to Roadmap,” 2015. pdfs.semanticscholar. org/5cb4/11de3db4941d5c7ecfc19de8af9243fb63d5.pdf (accessed Jan. 28, 2019).

[2] M. Tangermann *et al.*, “Review of the BCI competition IV,” *Frontiers in Neuroscience*, vol. 6, no. JULY, pp. 1–31, 2012, doi: 10.3389/fnins.2012.00055.

[3] F. Lotte *et al.*, “A review of classification algorithms for EEG-based brain-computer interfaces: A 10 year update,” *Journal of Neural Engineering*, vol. 15, no. 3, 2018, doi: 10.1088/1741-2552/aab2f2.

[4] K. S. Hong, M. J. Khan, and M. J. Hong, “Feature Extraction and Classification Methods for Hybrid fNIRS-EEG Brain-Computer Interfaces,” *Feature extraction and classification methods for hybrid fNIRS-EEG brain-computer interfaces.*, vol. 12, no. 246, 2018.

[5] V. P. Oikonomou, K. Georgiadis, G. Liaros, S. Nikolopoulos, and I. Kompatsiaris, “A Comparison Study on EEG Signal Processing Techniques Using Motor Imagery EEG Data,” *Proceedings - IEEE Symposium on Computer-Based Medical Systems*, vol. 2017-June, no. 1, pp. 781–786, 2017, doi: 10.1109/CBMS.2017.113.

[6] A. Al-Saegh, S. A. Dawwd, and J. M. Abdul-Jabbar, “Deep learning for motor imagery EEG-based classification: A review,” *Biomedical Signal Processing and Control*, vol. 63, no. October 2020, p. 102172, 2021, doi: 10.1016/j.bspc.2020.102172.

[7] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, “EEGNet: A compact convolutional neural network for EEG-based brain-computer interfaces,” *Journal of Neural Engineering*, vol. 15, no. 5, pp. 1–30, 2018, doi: 10.1088/1741-2552/aace8c.

[8] R. T. Schirrmeister *et al.*, “Deep learning with convolutional neural networks for EEG decoding and visualization,” *Human Brain Mapping*, vol. 38, no. 11, pp. 5391–5420, 2017, doi: 10.1002/hbm.23730.

[9] C. Zhang, Y.-K. Kim, and A. Eskandarian, “EEG-inception: an accurate and robust end-to-end neural network for EEG-based motor imagery classification,” *Journal of Neural Engineering*, 2021, doi: 10.1088/1741-2552/abed81.

[10] J. Olias, R. Martin-Clemente, M. A. Sarmiento-Vega, and S. Cruces, “EEG signal processing in mi-bci applications with improved covariance matrix

estimators," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 5, pp. 895–904, 2019, doi: 10.1109/TNSRE.2019.2905894.

[11] S. R. Sreeja, Himanshu, and D. Samanta, "Distance-based weighted sparse representation to classify motor imagery EEG signals for BCI applications," *Multimedia Tools and Applications*, 2020, doi: 10.1007/s11042-019-08602-0.

[12] S. Chaudhary, S. Taran, V. Bajaj, and A. Sengur, "Convolutional Neural Network Based Approach Towards Motor Imagery Tasks EEG Signals Classification," *IEEE Sensors Journal*, vol. 19, no. 12, pp. 4494–4500, 2019, doi: 10.1109/JSEN.2019.2899645.

[13] J. Jin, Y. Miao, I. Daly, C. Zuo, D. Hu, and A. Cichocki, "Correlation-based channel selection and regularized feature optimization for MI-based BCI," *Neural Networks*, vol. 118, pp. 262–270, 2019, doi: 10.1016/j.neunet.2019.07.008.

[14] A. Hekmatmanesh, H. Wu, F. Jamaloo, M. Li, and H. Handroos, "A combination of CSP-based method with soft margin SVM classifier and generalized RBF kernel for imagery-based brain computer interface applications," *Multimedia Tools and Applications*, 2020, doi: 10.1007/s11042-020-08675-2.

[15] Q. She, K. Chen, Y. Ma, T. Nguyen, and Y. Zhang, "Sparse representation-based extreme learning machine for motor imagery EEG classification," *Computational Intelligence and Neuroscience*, vol. 2018, 2018, doi: 10.1155/2018/9593682.

[16] A. Nijholt, "The Future of Brain-Computer Interfacing (Keynote Paper)," 2016.

[17] B. Kerous, F. Skola, and F. Liarokapis, "EEG-based BCI and video games: A progress report.," *Virtual Real.*, vol. 22, pp. 119–135, 2017.

[18] N. Padfield, J. Zabalza, H. Zhao, V. Masero, and J. Ren, "EEG-based brain-computer interfaces using motor-imagery: Techniques and challenges," *Sensors (Switzerland)*, vol. 19, no. 6, pp. 1–34, 2019, doi: 10.3390/s19061423.

[19] L. Tonin, F. C. Bauer, and J. del R. Millán, "The role of the control framework for continuous teleoperation of a brain–machine interface-driven mobile robot," *IEEE Transactions on Robotics*, vol. 36, no. 1, pp. 78–91, 2019, doi: 10.1109/TRO.2019.2943072.

[20] R. Leeb, L. Tonin, M. Rohm, L. Desideri, T. Carlson, and J. D. R. Millán, "Towards independence: A BCI telepresence robot for people with severe

motor disabilities," *Proceedings of the IEEE*, vol. 103, no. 6, pp. 969–982, 2015, doi: 10.1109/JPROC.2015.2419736.

[21] V. P. Oikonomou, S. Nikolopoulos, and I. Kompatsiaris, "Motor imagery classification via clustered-group sparse representation," *Proceedings - 2019 IEEE 19th International Conference on Bioinformatics and Bioengineering, BIBE 2019*, pp. 321–325, 2019, doi: 10.1109/BIBE.2019.00064.

[22] J. Kevric and A. Subasi, "Comparison of signal decomposition methods in classification of EEG signals for motor-imagery BCI system," *Biomedical Signal Processing and Control*, vol. 31, pp. 398–406, 2017, doi: 10.1016/j.bspc.2016.09.007.

[23] D. Mzurikwao *et al.*, "A channel selection approach based on convolutional neural network for multi-channel EEG motor imagery decoding," *Proceedings - IEEE 2nd International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2019*, pp. 195–202, 2019, doi: 10.1109/AIKE.2019.00042.

[24] H. Zhang, X. Zhao, Z. Wu, B. Sun, and T. Li, "Motor imagery recognition with automatic EEG channel selection and deep learning," *Journal of Neural Engineering*, vol. 18, no. 1, 2021, doi: 10.1088/1741-2552/abca16.

[25] O. W. Samuel *et al.*, "Determining the Optimal Window Parameters for Accurate and Reliable Decoding of Multiple Classes of Upper Limb Motor Imagery Tasks," *2018 IEEE International Conference on Cyborg and Bionic Systems, CBS 2018*, pp. 422–425, 2019, doi: 10.1109/CBS.2018.8612159.

[26] J. Asensio-Cubero, J. Q. Gan, and R. Palaniappan, "A study on temporal segmentation strategies for extracting common spatial patterns for brain computer interfacing," *UKCI 2011 - Proceedings of the 11th UK Workshop on Computational Intelligence*, pp. 98–102, 2011.

[27] M. F. Wahid, R. Tafreshi, and R. Langari, "A Multi-Window Majority Voting Strategy to Improve Hand Gesture Recognition Accuracies Using Electromyography Signal," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, no. 2, pp. 427–436, 2020, doi: 10.1109/TNSRE.2019.2961706.

[28] M. B. I. Reaz, M. S. Hussain, and F. Mohd-Yasin, "Techniques of EMG signal analysis: detection, processing, classification and applications," *Biological Procedures Online*, vol. 8, no. 1, pp. 11–35, 2006.

[29] Y. Liang and Y. Ma, "Calibrating EEG features in motor imagery classification tasks with a small amount of current data using multisource fusion transfer learning," *Biomedical Signal Processing and Control*, vol. 62, no. 220, 2020, doi: 10.1016/j.bspc.2020.102101.

[30] M. T. Sadiq *et al.*, "Motor Imagery EEG Signals Decoding by Multivariate Empirical Wavelet Transform-Based Framework for Robust Brain – Computer Interfaces," *IEEE Access*, vol. 7, no. Mi, pp. 171431–171451, 2019, doi: 10.1109/ACCESS.2019.2956018.

[31] M. Z. Ilyas, P. Saad, M. I. Ahmad, and A. R. I. Ghani, "Classification of EEG signals for brain-computer interface applications: Performance comparison," *Proceedings of 2016 International Conference on Robotics, Automation and Sciences, ICORAS 2016*, pp. 27–30, 2017, doi: 10.1109/ICORAS.2016.7872610.

[32] B. Yang, C. Hu, J. Wang, B. Li, and W. Wang, "Research on Motor Imagery EEG Modeling Based on Window Optimization and a Few Channels PSD," *International Conference on Digital Signal Processing, DSP*, vol. 2018-Novem, 2019, doi: 10.1109/ICDSP.2018.8631698.

[33] S. Siuly, Y. Li, and Y. Zhang, "Comparative Study: Motor Area EEG and All-Channels EEG," in *EEG Signal Analysis and Classification*, 2017, pp. 211–225.

[34] B. Graimann, B. Allison, and G. Pfurtscheller, "Brain–Computer Interfaces: A Gentle Introduction," in *Brain-Computer Interfaces*, Springer, 2009.

[35] S. D. Armstrong, M. V. Sale, and R. Cunnington, "Neural oscillations and the initiation of voluntary movement," *Frontiers in Psychology*, vol. 9, no. DEC, pp. 1–16, 2018, doi: 10.3389/fpsyg.2018.02509.

[36] J. W. Britton *et al.*, "The Posterior Dominant Rhythm," in *Electroencephalography (EEG): An Introductory Text and Atlas of Normal and Abnormal Findings in Adults, Children, and Infants [Internet]*, Chicago: American Epilepsy Society, 2016.

[37] P. Kidmose, D. Looney, M. Ungstrup, M. L. Rank, and D. P. Mandic, "A study of evoked potentials from ear-EEG," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 10, pp. 2824–2830, 2013, doi: 10.1109/TBME.2013.2264956.

[38] A. Vahid, M. Mückschel, S. Stober, A. K. Stock, and C. Beste, "Applying deep learning to single-trial EEG data provides evidence for complementary theories on action control," *Communications Biology*, vol. 3, no. 1, 2020, doi: 10.1038/s42003-020-0846-z.

[39] Y. Shin, S. Lee, M. Ahn, H. Cho, S. C. Jun, and H. N. Lee, "Simple adaptive sparse representation based classification schemes for EEG based brain-computer interface applications," *Computers in Biology and Medicine*, vol. 66, pp. 29–38, 2015, doi: 10.1016/j.compbiomed.2015.08.017.

[40] S. Taran and V. Bajaj, "Motor imagery tasks-based EEG signals classification using tunable-Q wavelet transform," *Neural Computing and*

Applications, vol. 31, no. 11, pp. 6925–6932, 2019, doi: 10.1007/s00521-018-3531-0.

[41] M. Arvaneh, C. Guan, K. K. Ang, and C. Quek, "Optimizing the channel selection and classification accuracy in EEG-based BCI," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 6, pp. 1865–1873, 2011, doi: 10.1109/TBME.2011.2131142.

[42] Y. Zhang, G. Zhou, J. Jin, X. Wang, and A. Cichocki, "Optimizing spatial patterns with sparse filter bands for motor-imagery based brain-computer interface," *Journal of Neuroscience Methods*, vol. 255, pp. 85–91, 2015, doi: 10.1016/j.jneumeth.2015.08.004.

[43] G. Qiao, Q. Song, L. Ma, and L. Wan, "A low-complexity orthogonal matching pursuit based channel estimation method for time-varying underwater acoustic OFDM systems," *Applied Acoustics*, vol. 148, pp. 246–250.

[44] M. Z. Baig, N. Aslam, and H. P. H. Shum, "Filtering techniques for channel selection in motor imagery EEG applications: a survey," *Artificial Intelligence Review*, 2019, doi: 10.1007/s10462-019-09694-8.

[45] M. C. Potter, B. Wyble, C. E. Hagmann, and E. S. McCourt, "Detecting meaning in RSVP at 13 ms per picture," *Attention, Perception, and Psychophysics*, vol. 76, no. 2, pp. 270–279, 2014, doi: 10.3758/s13414-013-0605-z.

[46] Y. Park and W. Chung, "Optimal Channel Selection Using Correlation Coefficient for CSP Based EEG Classification," *IEEE Access*, vol. 8, pp. 111514–111521, 2020.

[47] D. Gurve *et al.*, "Subject-specific EEG channel selection using non-negative matrix factorization for lower-limb motor imagery recognition," *Journal of Neural Engineering*, vol. 17, no. 2, 2020, doi: 10.1088/1741-2552/ab4dba.

[48] Z. Qiu, J. Jin, H. K. Lam, Y. Zhang, X. Wang, and A. Cichocki, "Improved SFFS method for channel selection in motor imagery based BCI," *Neurocomputing*, vol. 207, pp. 519–527, 2016, doi: 10.1016/j.neucom.2016.05.035.

[49] L. He, Y. Hu, Y. Li, and D. Li, "Channel selection by Rayleigh coefficient maximization based genetic algorithm for classifying single-trial motor imagery EEG," *Neurocomputing*, vol. 121, pp. 423–433, 2013, doi: 10.1016/j.neucom.2013.05.005.

[50] Z. A. A. Alyasseri, A. T. Khader, M. A. Al-Betar, and O. A. Alomari, "Person identification using EEG channel selection with hybrid flower pollination algorithm," *Pattern Recognition*, vol. 105, p. 107393, 2020, doi: 10.1016/j.patcog.2020.107393.

[51] O. P. Idowu *et al.*, “Neuro-evolutionary approach for optimal selection of EEG channels in motor imagery based BCI application,” *Biomedical Signal Processing and Control*, vol. 68, no. January, p. 102621, 2021, doi: 10.1016/j.bspc.2021.102621.

[52] V. S. Handiru and V. A. Prasad, “Optimized Bi-Objective EEG Channel Selection and Cross-Subject Generalization with Brain-Computer Interfaces,” *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 6, pp. 777–786, 2016, doi: 10.1109/THMS.2016.2573827.

[53] P. Croce, A. Quercia, S. Costa, and F. Zappasodi, “EEG microstates associated with intra- and inter-subject alpha variability,” *Scientific Reports*, vol. 10, no. 1, pp. 1–11, 2020, doi: 10.1038/s41598-020-58787-w.

[54] R. Rubinstein, M. Zibulevsky, and M. Elad, “Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit,” 2008.

[55] X. Zhang, Z. Liu, L. Wang, J. Zhang, and W. Han, “Bearing fault diagnosis based on sparse representations using an improved OMP with adaptive Gabor sub-dictionaries,” *ISA Transactions*, vol. 106, pp. 355–366, 2020, doi: 10.1016/j.isatra.2020.07.004.

[56] S. R. Sreeja and D. Samanta, “Classification of multiclass motor imagery EEG signal using sparsity approach,” in *Neurocomputing*, 2019, vol. 368, pp. 133–145. doi: 10.1016/j.neucom.2019.08.037.

[57] S. R. Sreeja, J. Rabha, D. Samanta, P. Mitra, and M. Sarma, “Classification of motor imagery based EEG signals using sparsity approach,” in *International Conference on Intelligent Human Computer Interaction*, 2017, pp. 47–59.

[58] S. U. Amin, M. Alsulaiman, G. Muhammad, M. A. Bencherif, and M. S. Hossain, “Multilevel Weighted Feature Fusion Using Convolutional Neural Networks for EEG Motor Imagery Classification,” *IEEE Access*, vol. 7, pp. 18940–18950, 2019, doi: 10.1109/ACCESS.2019.2895688.

[59] S. U. Amin, G. Muhammad, W. Abdul, M. Bencherif, and S. Arabia, “Multi-CNN Feature Fusion for Efficient EEG Classification. The authors are with the Department of Computer Engineering , College of Computer and Information Sciences (CCIS), King Saud University , Riyadh 11543 , Saudi Arabia . They are also with the Cen,” no. Mi, 2020.

[60] S. U. Amin, M. Alsulaiman, G. Muhammad, M. A. Mekhtiche, and M. Shamim Hossain, “Deep Learning for EEG motor imagery classification based on multi-layer CNNs feature fusion,” *Future Generation Computer Systems*, vol. 101, pp. 542–554, 2019, doi: 10.1016/j.future.2019.06.027.

[61] S. Sen, S. Saha, S. Chatterjee, S. Mirjalili, and R. Sarkar, “A bi-stage feature selection approach for COVID-19 prediction using chest CT images,” *Applied Intelligence*, 2021, doi: 10.1007/s10489-021-02292-8.

[62] M. Imani and H. Ghassemian, “An overview on spectral and spatial information fusion for hyperspectral image classification: Current trends and challenges,” *Information Fusion*, vol. 59, no. October 2019, pp. 59–83, 2020, doi: 10.1016/j.inffus.2020.01.007.

[63] W. Zhang, Y. Yu, Y. Qi, F. Shu, and Y. Wang, “Short-term traffic flow prediction based on spatio-temporal analysis and CNN deep learning,” *Transportmetrica A: Transport Science*, vol. 15, no. 2, pp. 1688–1711, 2019, doi: 10.1080/23249935.2019.1637966.

[64] A. S.U., A. M., M. G., H. M.S., and G. M., “Deep Learning for EEG Motor Imagery-Based Cognitive Healthcare,” in *Connected Health in Smart Cities*, 2019.

[65] K. Roots, Y. Muhammad, and N. Muhammad, “Fusion convolutional neural network for cross-subject eeg motor imagery classification,” *Computers*, vol. 9, no. 3, pp. 1–9, 2020, doi: 10.3390/computers9030072.

[66] R. T. Schirrmeister *et al.*, “Deep learning with convolutional neural networks for EEG decoding and visualization,” *Hum. Brain Mapping*, vol. 38, no. 11, pp. 5391–5420, 2017.

[67] Springer, *Computational EEG Analysis, Methods and Applications*. Singapore: Springer Nature Singapore Pte Ltd., 2018.

[68] G. Pfurtscheller and F. H. Lopes Da Silva, “Event-related EEG/MEG synchronization and desynchronization: Basic principles,” *Clinical Neurophysiology*, vol. 110, no. 11, pp. 1842–1857, 1999, doi: 10.1016/S1388-2457(99)00141-8.

[69] M. O’Shea, “Chapter 1: Thinking about the brain,” in *The Brain: A Very Short Introduction*, 2005, pp. 1–11.

[70] H. Hallez *et al.*, “Review on solving the forward problem in EEG source analysis,” *Journal of NeuroEngineering and Rehabilitation*, vol. 4, 2007, doi: 10.1186/1743-0003-4-46.

[71] G. Dornhege, B. Blankertz, G. Curio, and K. R. Müller, “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, doi: 10.1109/TBME.2004.827088.

[72] C. Brunner, R. Leeb, G. R. Leeb, A. Schogl, and G. Pfurtscheller, “BCI Competition 2008 – Graz data set A,” 2008. www.bbci.de/competition/iv/desc_2a.pdf (accessed Aug. 05, 2021).

[73] B. Blankertz, “Data set IVa ‹motor imagery, small training sets›,” *BCI Competition III*, 2003. www.bbci.de/competition/iii/desc_IVa.html

[74] R. Leeb, C. Brunner, G. R. Muller-Pultz, A. Schlogl, and G. Pfurtscheller, “BCI Competition 2008 – Graz data set B,” *BCI Competition 2008*, 2008. www.bbci.de/competition/iv/desc_2b.pdf

[75] A. Bach Justesen *et al.*, “Added clinical value of the inferior temporal EEG electrode chain,” *Clinical Neurophysiology*, vol. 129, no. 1, pp. 291–295, 2018, doi: 10.1016/j.clinph.2017.09.113.

[76] American Epilepsy Society, “The Normal EEG,” in *Electroencephalography (EEG): An Introductory Text and Atlas of Normal and Abnormal Findings in Adults, Children, and Infants*, 2016.

[77] V. De Pascalis, G. Cirillo, and A. Vecchio, “Resting eeg asymmetry markers of multiple facets of the behavioral approach system: A loreta analysis,” *Symmetry (Basel)*, vol. 12, no. 11, pp. 1–20, 2020, doi: 10.3390/sym12111794.

[78] M. Palmiero and L. Piccardi, “Frontal EEG asymmetry of mood: A mini-review,” *Frontiers in Behavioral Neuroscience*, vol. 11, no. November, pp. 1–8, 2017, doi: 10.3389/fnbeh.2017.00224.

[79] M. M. N. Mannan, M. A. Kamran, and M. Y. Jeong, “Identification and removal of physiological artifacts from electroencephalogram signals: A review,” *IEEE Access*, vol. 6, pp. 30630–30652, 2018, doi: 10.1109/ACCESS.2018.2842082.

[80] L. Tonin, T. Carlson, R. Leeb, and J. Del R. Millan, “Brain-controlled telepresence robot by motor-disabled people,” *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pp. 4227–4230, 2011, doi: 10.1109/IEMBS.2011.6091049.

[81] X. Yong and C. Menon, “EEG classification of different imaginary movements within the same limb,” *PLoS ONE*, vol. 10, no. 4, pp. 1–24, 2015, doi: 10.1371/journal.pone.0121896.

[82] Z. Tang, C. Li, and S. Sun, “Single-trial EEG classification of motor imagery using deep convolutional neural networks,” *Optik (Stuttg)*, vol. 130, pp. 11–18, 2017, doi: 10.1016/j.ijleo.2016.10.117.

[83] S. D. Muthukumaraswamy, “High-frequency brain activity and muscle artifacts in MEG/EEG: A review and recommendations,” *Frontiers in Human Neuroscience*, vol. 7, no. MAR, pp. 1–11, 2013, doi: 10.3389/fnhum.2013.00138.

[84] H. Yang, C. Guan, C. C. Wang, and K. K. Ang, “Maximum dependency and minimum redundancy-based channel selection for motor imagery of

walking EEG signal detection," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 1187–1191, 2013, doi: 10.1109/ICASSP.2013.6637838.

[85] Y. Sun, N. Ye, and J. Yang, "An asynchronous MI-BCI system based on masterslave features," in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2016, pp. 1456–1461.

[86] G. Dornhege, B. Blankertz, G. Curio, and K. R. Müller, "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, doi: 10.1109/TBME.2004.827088.

[87] Technische Universität Berlin, "BCI Competition III." http://www.bbci.de/competition/iii/ (accessed Dec. 18, 2019).

[88] C. Brunner, R. Leeb, G. Muller-Putz, A. Schlogl, and G. Pfurtscheller, "BCI Competition 2008–Graz data set A and B," Graz, 2008.

[89] C. Vidaurre and B. Blankertz, "Towards a cure for BCI illiteracy," *Brain Topogr.*, vol. 23, pp. 194–8, 2010.

[90] J. Wu, X. Y. Chen, H. Zhang, L. D. Xiong, H. Lei, and S. H. Deng, "Hyperparameter optimization for machine learning models based on Bayesian optimization," *Journal of Electronic Science and Technology*, vol. 17, no. 1, pp. 26–40, 2019, doi: 10.11989/JEST.1674-862X.80904120.

[91] S. D. Cranstoun, H. C. Ombao, R. Von Sachs, W. Guo, and B. Litt, "Time-frequency spectral estimation of multichannel EEG using the auto-SLEX method," *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 9, pp. 988–996, 2002, doi: 10.1109/TBME.2002.802015.

[92] D. Qin, M. E. Parten, B. Houchins, N. Parker, R. Homan, and A. Petrosian, "Comparison of techniques for the prediction of epileptic seizures," *Proceedings of the IEEE Symposium on Computer-Based Medical Systems*, pp. 151–158, 1995, doi: 10.1109/cbms.1995.465436.

[93] D. J. McFarland, L. M. McCane, S. V. David, and J. R. Wolpaw, "Spatial filter selection for EEG-based communication," *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 3, pp. 386–394, 1997, doi: 10.1016/S0013-4694(97)00022-2.

[94] M. Corsi-Cabrera, J. Ramos, C. Arce, M. A. Guevara, M. Ponce-de León, and I. Lorenzo, "Changes in the Waking EEG as a Consequence of Sleep and Sleep Deprivation," *Sleep*, vol. 15, no. 6, pp. 550–555, 1992.

[95] J. A. Caldwell, B. Prazinko, and J. L. Caldwell, "Body posture affects electroencephalographic activity and psychomotor vigilance task

performance in sleep-deprived subjects," *Clinical Neurophysiology*, vol. 114, no. 1, pp. 23–31, 2003, doi: 10.1016/S1388-2457(02)00283-3.

[96] T. C. Ferree, P. Luu, G. S. Russell, and D. M. Tucker, "Scalp Electrode Impedance and EEG Data Quality," *Clinical Neurophysiology*, vol. 112, pp. 1–9, 2001.

[97] N. Seth, R. C. D. Freitas, M. Chaulk, C. O'Connell, K. Englehart, and E. Scheme, "EMG pattern recognition for persons with cervical spinal cord injury," *IEEE International Conference on Rehabilitation Robotics*, vol. 2019-June, pp. 1055–1060, 2019, doi: 10.1109/ICORR.2019.8779450.

[98] V. Miskovic, K. J. MacDonald, L. J. Rhodes, and K. A. Cote, "Changes in EEG multiscale entropy and power-law frequency scaling during the human sleep cycle," *Human Brain Mapping*, vol. 40, no. 2, pp. 538–551, 2019, doi: 10.1002/hbm.24393.

[99] S. Siuly, H. Wang, and Y. Zhang, "Detection of motor imagery EEG signals employing Naïve Bayes based learning process," *Measurement*, vol. 86, pp. 148–158, 2016.

[100] H. Gao *et al.*, "EEG-Based Volitional Control of Prosthetic Legs for Walking in Different Terrains," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 2, pp. 530–540, 2021, doi: 10.1109/TASE.2019.2956110.

[101] P. Batres-Mendoza *et al.*, "Quaternion-based signal analysis for motor imagery classification from electroencephalographic signals," *Sensors (Switzerland)*, vol. 16, no. 3, 2016, doi: 10.3390/s16030336.

[102] W. Huang, Y. Xue, L. Hu, and H. Liuli, "S-EEGNet: Electroencephalogram Signal Classification Based on a Separable Convolution Neural Network with Bilinear Interpolation," *IEEE Access*, vol. 8, pp. 131636–131646, 2020, doi: 10.1109/ACCESS.2020.3009665.

[103] Q. Wang, N. de Prisco, J. Tang, and V. A. Gennarino, "Protocol for recording epileptiform discharges of EEG and behavioral seizures in freely moving mice," *STAR Protocols*, vol. 3, no. 2, p. 101245, Jun. 2022, doi: 10.1016/j.xpro.2022.101245.

[104] M. Z. Baig, N. Aslam, H. P. H. Shum, and L. Zhang, "Differential evolution algorithm as a tool for optimal feature subset selection in motor imagery EEG," *Expert Systems with Applications*, vol. 90, pp. 184–195, 2017, doi: 10.1016/j.eswa.2017.07.033.

[105] L. He, Y. Hu, Y. Li, and D. Li, "Channel selection by Rayleigh coefficient maximization based genetic algorithm for classifying single-trial motor imagery EEG," *Neurocomputing*, vol. 121, pp. 423–433, 2013, doi: 10.1016/j.neucom.2013.05.005.

[106] S. Kumar, A. Sharma, K. Mamun, and T. Tsunoda, “A Deep Learning Approach for Motor Imagery EEG Signal Classification,” *Proceedings - Asia-Pacific World Congress on Computer Science and Engineering 2016 and Asia-Pacific World Congress on Engineering 2016, APWC on CSE/APWCE 2016*, pp. 34–39, 2017, doi: 10.1109/APWC-on-CSE.2016.017.

[107] Q. She, B. Hu, Z. Luo, T. Nguyen, and Y. Zhang, “A hierarchical semi-supervised extreme learning machine method for EEG recognition,” *Medical and Biological Engineering and Computing*, vol. 57, no. 1, pp. 147–157, 2019, doi: 10.1007/s11517-018-1875-3.

[108] B. Blankertz, G. Dornhege, M. Krauledat, K. R. Müller, and G. Curio, “The non-invasive Berlin Brain-Computer Interface: Fast acquisition of effective performance in untrained subjects,” *Neuroimage*, vol. 37, no. 2, pp. 539–550, 2007, doi: 10.1016/j.neuroimage.2007.01.051.

[109] B. Blankertz, “Data sets 1 ‹motor imagery, uncued classifier application›,” *BCI Compeition IV, https://www.bbci.de/competition/iv/desc_1.html*, Apr. 18, 2022.

[110] M. P. Paulraj, K. Subramaniam, S. bin Yaccob, A. H. bin Adom, and C. R. Hema, “Auditory Evoked Potential Response and Hearing Loss: A Review,” *The Open Biomedical Engineering Journal*, vol. 9, no. 1, pp. 17–24, 2015, doi: 10.2174/1874120701509010017.

[111] B. Blankertz, “BCI Competition IV.” www.bbci.de/competition/iv/#dataset2a (accessed Aug. 05, 2021).

[112] Robintibor, “high-gamma-dataset,” *Github*, 2018. github.com/robintibor/high-gamma-dataset (accessed Aug. 05, 2021).

[113] R. T. Schirrmeister *et al.*, “Supplementary for: Deep learning with convolutional neural networks for EEG decoding and visualization Short title: Convolutional neural networks in EEG analysis-to-end learning, brain-machine interface (BCI), brain-computer interface (BMI), model interpretability, brain mapping,” 2017.

[114] A. Schloegl, K. Lugger, and G. Pfurtscheller, “Using Adaptive Autogregressive Parameters for a Brain-Computer-Interface Experiment,” in *Proceedings - 19th International Conference - IEEE/EMBS*, 1997, pp. 1533–1535.

[115] G. Pfurtscheller, C. Neuper, D. Flotzinger, and M. Pregenzer, “EEG-based discrimination between imagination of right and left hand movement,” *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 6, pp. 642–651, 1997, doi: 10.1016/S0013-4694(97)00080-1.

[116] A. Schloegl, “The Electroencephalogram and the Adaptive Autoregressive Model: Theory,” 2000.

[117] D. J. Krusienski, D. J. McFarland, and J. R. Wolpaw, “An evaluation of autoregressive spectral estimation model order for brain-computer interface applications,” in *Annual International Conference of the IEEE Engineering in Medicine and Biology - Proceedings*, 2006, pp. 1323–1326. doi: 10.1109/IEMBS.2006.259822.

[118] M. T. Sadiq *et al.*, “Motor Imagery EEG Signals Classification Based on Mode Amplitude and Frequency Components Using Empirical Wavelet Transform,” vol. 7, no. Mi, 2019.

[119] B. Hjorth, “EEG analysis based on time domain properties,” *Electroencephalography and Clinical Neurophysiology*, vol. 29, no. 3, pp. 306–310, 1970.

[120] G. Rodŕiguez-Beŕmudez and P. J. Gárcia-Laencina, “Automatic and adaptive classification of electroencephalographic signals for brain computer interfaces,” *Journal of Medical Systems*, vol. 36, no. SUPPL.1, 2012, doi: 10.1007/s10916-012-9893-4.

[121] R. Roy, D. Sikdar, M. Mahadevappa, and C. S. Kumar, “EEG based motor imagery study of time domain features for classification of power and precision hand grasps,” *International IEEE/EMBS Conference on Neural Engineering, NER*, pp. 440–443, 2017, doi: 10.1109/NER.2017.8008384.

[122] M. Mumtaz, M. Afzal, and A. Mushtaq, “Sensorimotor Cortex EEG signal classification using Hidden Markov Models and Wavelet Decomposition,” *2018 IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2018*, pp. 375–381, 2019, doi: 10.1109/ISSPIT.2018.8642672.

[123] M. Hamedi, S. H. Salleh, A. M. Noor, and I. Mohammad-Rezazadeh, “Neural network-based three-class motor imagery classification using time-domain features for BCI applications,” *IEEE TENSYMP 2014 - 2014 IEEE Region 10 Symposium*, pp. 204–207, 2014, doi: 10.1109/tenconspring.2014.6863026.

[124] J. Arnin, D. Kahani, H. Lakany, and B. A. Conway, “Evaluation of Different Signal Processing Methods in Time and Frequency Domain for Brain-Computer Interface Applications,” *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, vol. 2018-July, pp. 235–238, 2018, doi: 10.1109/EMBC.2018.8512193.

[125] A. Liu, K. Chen, Q. Liu, Q. Ai, Y. Xie, and A. Chen, “Feature selection for motor imagery EEG classification based on firefly algorithm and learning

automata," *Sensors (Switzerland)*, vol. 17, no. 11, 2017, doi: 10.3390/s17112576.

[126] K. W. Ha and J. W. Jeong, "Decoding Two-Class Motor Imagery EEG with Capsule Networks," *2019 IEEE International Conference on Big Data and Smart Computing, BigComp 2019 - Proceedings*, pp. 1–4, 2019, doi: 10.1109/BIGCOMP.2019.8678917.

[127] Z. Tayeb *et al.*, "Validating deep neural networks for online decoding of motor imagery movements from eeg signals," *Sensors (Switzerland)*, vol. 19, no. 1, Jan. 2019, doi: 10.3390/s19010210.

[128] X. Mao *et al.*, "Progress in EEG-based brain robot interaction systems," *Computational Intelligence and Neuroscience*, vol. 2017, 2017, doi: 10.1155/2017/1742862.

[129] C. J. Ortiz-Echeverri, S. Salazar-Colores, J. Rodríguez-Reséndiz, and R. A. Gómez-Loenzo, "A New Approach for Motor Imagery Classification Based on Sorted Blind Source Separation, Continuous Wavelet Transform, and Convolutional Neural Network," *Sensors (Switzerland)*, vol. 19, no. 20, p. 4541, 2019, doi: 10.3390/s19204541.

[130] M. Sandsten, "Time-Frequency Analysis of Time-Varying Signals and Non-Stationary Processes," 2020.

[131] G. Lisi, D. Rivela, A. Takai, and J. Morimoto, "Markov switching model for quick detection of event related desynchronization in EEG," *Frontiers in Neuroscience*, vol. 12, no. FEB, 2018, doi: 10.3389/fnins.2018.00024.

[132] B. Orset, K. Lee, R. Chavarriaga, and J. D. R. Millán, "Reliable decoding of motor state transitions during imagined movement," *International IEEE/EMBS Conference on Neural Engineering, NER*, vol. 2019-March, pp. 263–266, 2019, doi: 10.1109/NER.2019.8717171.

[133] L. Zhu, C. Su, G. Cui, C. Zhou, J. Zhang, and W. Kong, "Idle-state detection in multi-user motor imagery brain computer interface with cross-brain CSP and hyper-brain-network," *Proceedings - 2019 International Conference on Cyberworlds, CW 2019*, pp. 225–230, 2019, doi: 10.1109/CW.2019.00045.

[134] Q. Ai *et al.*, "Feature extraction of four-class motor imagery EEG signals based on functional brain network," *Journal of Neural Engineering*, vol. 16, no. 2, 2019, doi: 10.1088/1741-2552/ab0328.

[135] J. Asensio-Cubero, J. Q. Gan, and R. Palaniappan, "Extracting optimal tempo-spatial features using local discriminant bases and common spatial patterns for brain computer interfacing," *Biomedical Signal Processing and Control*, vol. 8, no. 6, pp. 772–778, 2013, doi: 10.1016/j.bspc.2013.07.004.

[136] C. D. Virgilio Gonzalez, J. H. Sossa Azuela, E. Rubio Espino, and V. H. Ponce Ponce, "Classification of motor imagery EEG signals with CSP filtering through neural networks models," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11288 LNAI, 2018, pp. 123–135. doi: 10.1007/978-3-030-04491-6_10.

[137] F. Lotte and C. Guan, "Regularizing common spatial patterns to improve BCI designs: Unified theory and new algorithms," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 2, pp. 355–362, 2011, doi: 10.1109/TBME.2010.2082539.

[138] X. Zhu, P. Li, C. Li, D. Yao, R. Zhang, and P. Xu, "Separated channel convolutional neural network to realize the training free motor imagery BCI systems," *Biomedical Signal Processing and Control*, vol. 49, pp. 396–403, 2019, doi: 10.1016/j.bspc.2018.12.027.

[139] R. Herbert, M.-G. Johannes, and P. Gert, "Optimal spatial filtering of single trial EEG during imagined hand movement," *IEEE Transactions on Rehabilitation Engineering*, vol. 99, no. 5, pp. 441–446, 2000, doi: 10.1103/PhysRevLett.99.055003.

[140] S. Kumar, A. Sharma, and T. Tsunoda, "An improved discriminative filter bank selection approach for motor imagery EEG signal classification using mutual information," *BMC Bioinformatics*, vol. 18, no. Suppl 16, 2017, doi: 10.1186/s12859-017-1964-6.

[141] S. Lemm, B. Blankertz, G. Curio, and K. Muller, "Spatio-spectral filters for improving the classification of single trial EEG," *IEEE Trans. Biomed. Eng.*, vol. 52, pp. 1541–1548, 2005.

[142] G. Dornhege, B. Blankertz, M. Krauledat, F. Losch, G. Curio, and K. R. Müller, "Combined optimization of spatial and temporal filters for improving brain-computer interfacing," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 11, pp. 2274–2281, 2006, doi: 10.1109/TBME.2006.883649.

[143] Q. Novi, G. Cuntai, T. H. Dat, and P. Xue, "Sub-Band Common Spatial Pattern (SBCSP) for Brain-Computer Interface," 2007.

[144] H. Raza, H. Cecotti, and G. Prasad, "Optimising frequency band selection with forward-addition and backward-elimination algorithms in EEG-based brain-computer interfaces," *Proceedings of the International Joint Conference on Neural Networks*, vol. 2015-Septe, pp. 1–7, 2015, doi: 10.1109/IJCNN.2015.7280737.

[145] H. Lu, H. L. Eng, C. Guan, K. N. Plataniotis, and A. N. Venetsanopoulos, "Regularized common spatial pattern with aggregation for EEG

Classification in small-sample setting," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 12, pp. 2936–2946, 2010, doi: 10.1109/TBME.2010.2082540.

[146] J. Harmouche, D. Fourer, F. Auger, P. Borgnat, and P. Flandrin, "The Sliding Singular Spectrum Analysis: A Data-Driven Nonstationary Signal Decomposition Tool," *IEEE Transactions on Signal Processing*, vol. 66, no. 1, pp. 251–263, 2018, doi: 10.1109/TSP.2017.2752720.

[147] K. Giannakaki, G. Giannakakis, P. Vorgia, M. Klados, and M. Zervakis, "Automatic absence seizure detection evaluating matching pursuit features of EEG signals," *Proceedings - 2019 IEEE 19th International Conference on Bioinformatics and Bioengineering, BIBE 2019*, pp. 886–889, 2019, doi: 10.1109/BIBE.2019.00165.

[148] F. Akram, H. S. Han, and T. S. Kim, "A P300-Based Word Typing Brain Computer Interface System Using a Smart Dictionary and Random Forest Classifier," in *The Eighth International Multi- Conference on Computing in the Global Information Technology*, 2013, pp. 106–109.

[149] G. James, D. Witten, T. Hastie, and R. Tibshirani, "Support Vector Machines," in *An Introduction to Statistical Learning. Springer Texts in Statistics, vol 103*, New York: Springer, 2013.

[150] C. M. Bishop, *Pattern Regonition and Machine Learning*. Springer, 2009.

[151] S. Liu, W. Wang, Y. Sheng, L. Zhang, M. Xu, and D. Ming, "Improving the Cross-Subject Performance of the ERP-Based Brain–Computer Interface Using Rapid Serial Visual Presentation and Correlation Analysis Rank," *Frontiers in Human Neuroscience*, vol. 14, no. July, pp. 1–10, 2020, doi: 10.3389/fnhum.2020.00296.

[152] A. O'Shea, G. Lightbody, G. Boylan, and A. Temko, "Neonatal seizure detection from raw multi-channel EEG using a fully convolutional architecture," *Neural Networks*, vol. 123, pp. 12–25, 2020, doi: 10.1016/j.neunet.2019.11.023.

[153] S. Kumar, R. Sharma, A. Sharma, and T. Tsunoda, "Decimation filter with Common Spatial Pattern and Fishers Discriminant Analysis for motor imagery classification," *Proceedings of the International Joint Conference on Neural Networks*, vol. 2016-Octob, pp. 2090–2095, 2016, doi: 10.1109/IJCNN.2016.7727457.

[154] MathWorks, "fitcsvm (Algorithms)," *Matlab Documentation*, 2021. https://uk.mathworks.com/help/stats/fitcsvm.html#bt7oo83-5 (accessed Jun. 07, 2021).

[155] C. Cortes and V. Vapnik, "Support Vector Machines," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1109/64.163674.

[156] MathWorks, “fitcdiscr,” *MATLAB Documentation*, 2021. https://uk.mathworks.com/help/stats/fitcdiscr.html?searchHighlight=fitcdiscr&s_tid=srchtitle (accessed Jun. 07, 2021).

[157] R. Bose, K. Samanta, S. Chatterjee, S. Bhattacharyya, and A. Khasnobish, *Motor imagery classification enhancement with concurrent implementation of spatial filtration and modified stockwell transform*. Elsevier Ltd, 2019. doi: 10.1016/B978-0-08-102420-1.00038-8.

[158] Christoph Reinders, H. Ackermann, M. YingYang, and R. Bodo, “4.2.3 Random Forests,” in *Multimodal Scene Understanding*, 2019, pp. 65–100.

[159] MathWorks, “Tune Random Forest Using Quantile Error and Bayesian Optimization,” *Matlab Documentation*, 2019.

[160] scikit-learn developers, “1.11. Ensemble methods,” *Scikit Learn Documentation*. www.scikit-learn.org/stable/modules/ensemble.html#forest (accessed Sep. 18, 2021).

[161] Mathworks, “Naive Bayes’,” *fitcnb*, 2021. https://uk.mathworks.com/help/stats/fitcnb.html#budugq6-12 (accessed May 13, 2021).

[162] MathWorks, “ClassificationNaiveBayes,” *Mathworks Documentation*, 2021. https://uk.mathworks.com/help/stats/classificationnaivebayes.html#mw_3bfd3ab2-099a-45c7-b1dd-7551acb8e6d0 (accessed Oct. 31, 2021).

[163] Z. Tang, C. Li, and S. Sun, “Single-trial EEG classification of motor imagery using deep convolutional neural networks,” *Optik (Stuttg)*, vol. 130, pp. 11–18, 2017, doi: 10.1016/j.ijleo.2016.10.117.

[164] X. Zhao, J. Zhao, C. Liu, and W. Cai, “Deep Neural Network with Joint Distribution Matching for Cross-Subject Motor Imagery Brain-Computer Interfaces,” *BioMed Research International*, vol. 2020, 2020, doi: 10.1155/2020/7285057.

[165] H. Zhao, Q. Zheng, K. Ma, H. Li, and Y. Zheng, “Deep Representation-Based Domain Adaptation for Nonstationary EEG Classification,” *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2020, doi: 10.1109/tnnls.2020.3010780.

[166] J. McCaffrey, “Neural Network Momentum Using Python,” *Visual Studio Magazine*, 2017.

[167] M. Miao, A. Wang, and F. Liu, “A spatial-frequency-temporal optimized feature sparse representation-based classification method for motor imagery EEG pattern recognition,” *Medical and Biological Engineering and*

Computing, vol. 55, no. 9, pp. 1589–1603, 2017, doi: 10.1007/s11517-017-1622-1.

[168] A. Kulkarni and T. Mohsenin, "Accelerating compressive sensing reconstruction OMP algorithm with CPU, GPU, FPGA and domain specific many-core," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 970–973.

[169] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[170] Y. Hou, L. Zhou, S. Jia, and X. Lun, "A novel approach of decoding EEG four-class motor imagery tasks via scout ESI and CNN," *Journal of Neural Engineering*, vol. 17, no. 1, 2020, doi: 10.1088/1741-2552/ab4af6.

[171] X. Zhao, H. Zhang, G. Zhu, Y. Fengxiang, S. Kuang, and L. Sun, "A Multi-Branch 3D Convolutional Neural Network for EEG-Based Motor Imagery Classificatio," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 10, pp. 2164–2177, 2019, doi: 10.23919/ChiCC.2019.8865152.

[172] P. K. Mishra, B. Jagadish, M. P. R. S. Kiran, P. Rajalakshmi, and D. S. Reddy, "A novel classification for EEG based four class motor imagery using kullback-leibler regularized riemannian manifold," *2018 IEEE 20th International Conference on e-Health Networking, Applications and Services, Healthcom 2018*, 2018, doi: 10.1109/HealthCom.2018.8531086.

[173] W. Hang *et al.*, "Cross-Subject EEG Signal Recognition Using Deep Domain Adaptation Network," *IEEE Access*, vol. 7, pp. 128273–128282, 2019, doi: 10.1109/ACCESS.2019.2939288.

[174] H. Wu *et al.*, "A Parallel Multiscale Filter Bank Convolutional Neural Networks for Motor Imagery EEG Classification," *Frontiers in Neuroscience*, vol. 13, no. November, pp. 1–9, 2019, doi: 10.3389/fnins.2019.01275.

[175] D. Li, J. Wang, J. Xu, and X. Fang, "Densely Feature Fusion Based on Convolutional Neural Networks for Motor Imagery EEG Classification," *IEEE Access*, vol. 7, pp. 132720–132730, 2019, doi: 10.1109/ACCESS.2019.2941867.

[176] G. Xu *et al.*, "A Deep Transfer Convolutional Neural Network Framework for EEG Signal Classification," *IEEE Access*, vol. 7, pp. 112767–112776, 2019, doi: 10.1109/ACCESS.2019.2930958.

[177] Keras, "DepthwiseConv2D layer," *Keras API reference*, 2021. https://keras.io/api/layers/convolution_layers/depthwise_convolution2d/ (accessed May 18, 2021).

[178] Keras, “SeparableConv2D layer,” *Keras API reference*, 2021.

[179] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” in *International conference on machine learning*, 2015, pp. 448–456.

[180] Nitish, Srivastava, Geoffrey, Hinton, Alex, Krizhevsky, Ilya, Sutskever, and Ruslan, Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[181] Keras, “BatchNormalization layer,” *Keras API reference*. https://keras.io/api/layers/normalization_layers/batch_normalization/ (accessed May 14, 2021).

[182] Keras, “Dropout layer,” *Keras API reference*. https://keras.io/api/layers/regularization_layers/dropout/

[183] B. Abibullaev, I. Dolzhikova, and A. Zollanvari, “A Brute-Force CNN Model Selection for Accurate Classification of Sensorimotor Rhythms in BCIs,” *IEEE Access*, vol. 8, pp. 101014–101023, 2020, doi: 10.1109/ACCESS.2020.2997681.

[184] S. Saha and M. Baumert, “Intra- and Inter-subject Variability in EEG-Based Sensorimotor Brain Computer Interface: A Review,” *Frontiers in Computational Neuroscience*, vol. 13, no. January, pp. 1–8, 2020, doi: 10.3389/fncom.2019.00087.

[185] K. Samanta, S. Chatterjee, and R. Bose, “Cross-Subject Motor Imagery Tasks EEG Signal Classification Employing Multiplex Weighted Visibility Graph and Deep Feature Extraction,” *IEEE Sensors Letters*, vol. 4, no. 1, 2020, doi: 10.1109/LSENS.2019.2960279.

[186] B. Xu *et al.*, “Wavelet Transform Time-Frequency Image and Convolutional Network-Based Motor Imagery EEG Classification,” *IEEE Access*, vol. 7, no. Mi, pp. 6084–6093, 2019, doi: 10.1109/ACCESS.2018.2889093.

[187] A. M. Azab, H. Ahmadi, L. Mihaylova, and M. Arvaneh, “Dynamic time warping-based transfer learning for improving common spatial patterns in brain-computer interface,” *Journal of Neural Engineering*, vol. 17, no. 1, 2020, doi: 10.1088/1741-2552/ab64a0.

[188] J. J. Bird, J. Kobylarz, D. R. Faria, A. Ekart, and E. P. Ribeiro, “Cross-Domain MLP and CNN Transfer Learning for Biological Signal Processing: EEG and EMG,” *IEEE Access*, vol. 8, pp. 54789–54801, 2020, doi: 10.1109/ACCESS.2020.2979074.

[189] M. Tariq, P. Trivailo, and M. Simic, “Mu-Beta event-related (de)synchronization and EEG classification of left-right foot dorsiflexion kinaesthetic motor imagery for BCI,” *PLOS ONE*, vol. 15, no. 3, 2020.

[190] L. Fogassi and G. Luppino, “Motor functions of the parietal lobe,” *Current Opinion in Neurobiology*, vol. 15, no. 6, pp. 626–631, 2005, doi: 10.1016/j.conb.2005.10.015.

[191] Y. Wang, S. Gao, and X. Gao, “Common spatial pattern method for channel selection in motor imagery based brain-computer interface,” *Annual International Conference of the IEEE Engineering in Medicine and Biology - Proceedings*, vol. 7 VOLS, pp. 5392–5395, 2005, doi: 10.1109/iembs.2005.1615701.

[192] L. A. Moctezuma and M. Molinas, “Towards a minimal EEG channel array for a biometric system using resting-state and a genetic algorithm for channel selection,” *Scientific Reports*, vol. 10, no. 1, pp. 1–14, 2020, doi: 10.1038/s41598-020-72051-1.

[193] O. Kramer, “Genetic Algorithms,” in *Genetic Algorithm Essentials*, Springer, 2017, pp. 11–19.

[194] L. W. Ko *et al.*, “Multimodal fuzzy fusion for enhancing the motor-imagery-based brain computer interface,” *IEEE Computational Intelligence Magazine*, vol. 14, no. 1, pp. 96–106, 2019, doi: 10.1109/MCI.2018.2881647.

[195] D. Qin, B. Houchins, M. E. Parten, N. Parker, R. Homan, and A. Petrosian, “A comparison of techniques for the prediction of epileptic seizures.,” 1995.

[196] W. Wojcikiewicz, C. Vidaurre, and M. Kawana, “Stationary common spatial patterns for non-stationary EEG data.”

[197] X. Ma, S. Qiu, C. Du, J. Xing, and H. He, “Improving EEG-Based Motor Imagery Classification via Spatial and Temporal Recurrent Neural Networks,” *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, vol. 2018-July, no. Mi, pp. 1903–1906, 2018, doi: 10.1109/EMBC.2018.8512590.

[198] N. Padfield, J. Ren, C. Qing, P. Murray, H. Zhao, and J. Zheng, “Multi-segment Majority Voting Decision Fusion for MI EEG Brain-Computer Interfacing,” *Cognitive Computation*, no. 0123456789, 2021, doi: 10.1007/s12559-021-09953-3.

[199] R. Zerafa, T. Camilleri, O. Falzon, and K. P. Camilleri, “A comparison of a broad range of EEG acquisition devices–is there any difference for SSVEP BCIs?,” *Brain-Computer Interfaces*, vol. 5, no. 4, pp. 121–131, 2018, doi: 10.1080/2326263X.2018.1550710.

[200] X. Lei and K. Liao, “Understanding the influences of EEG reference: A large-scale brain network perspective,” *Frontiers in Neuroscience*, vol. 11, no. APR, pp. 1–11, 2017, doi: 10.3389/fnins.2017.00205.

[201] Y. Zhang, Y. Guo, P. Yang, W. Chen, and B. Lo, “Epilepsy Seizure Prediction on EEG Using Common Spatial Pattern and Convolutional Neural Network,” *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 2, pp. 465–474, 2020, doi: 10.1109/JBHI.2019.2933046.

[202] Y. Li, M.-Y. Lei, W. Cui, Y. Guo, and H.-L. Wei, “A Parametric Time Frequency-Conditional Granger Causality Method Using Ultra-regularized Orthogonal Least Squares and Multiwavelets for Dynamic Connectivity Analysis in EEGs,” *IEEE Transactions on Biomedical Engineering*, no. c, pp. 1–1, 2019, doi: 10.1109/tbme.2019.2906688.

[203] P. Lameski, E. Zdravevski, R. Mingov, and A. Kulakov, “SVM Parameter Tuning with Grid Search and Its Impact on Reduction of Model Over-fitting,” 2015, pp. 464–474. doi: 10.1007/978-3-319-25783-9_41.

[204] J. Wu, X. Chen, H. Zhang, L. Xiong, H. Lei, and S. Deng, “Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization,” *Journal of Electronic Science and Technology*, vol. 7, no. 1, pp. 26–40, 2019.

[205] C. O’Reilly and T. Nielsen, “Automatic sleep spindle detection: Benchmarking with fine temporal resolution using open science tools,” *Frontiers in Human Neuroscience*, vol. 9, no. JUNE, pp. 1–19, 2015, doi: 10.3389/fnhum.2015.00353.

[206] Y. Jiang, J. He, D. Li, J. Jin, and Y. Shen, “Signal classification algorithm in motor imagery based on asynchronous brain-computer interface,” *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, vol. 2019-May, pp. 1–5, 2019, doi: 10.1109/I2MTC.2019.8826883.

[207] L. Duan *et al.*, “Zero-Shot Learning for EEG Classification in Motor Imagery-Based BCI System,” *IEEE Trans Neural Syst Rehabil Eng*, vol. 28, no. 11, pp. 2411–2419, 2020, doi: 10.1109/TNSRE.2020.3027004.

[208] M. Eliasziw and A. Donner, “Application of the McNemar test to non-independent matched pair data,” *Stat. Med.*, vol. 10, no. 12, pp. 1981–91, 1991.

[209] Y. H. Liu, C. A. Cheng, and H. P. Huang, “Novel feature of the EEG based motor imagery BCI system: Degree of imagery,” *Proceedings 2011 International Conference on System Science and Engineering, ICSSE 2011*, no. June, pp. 515–520, 2011, doi: 10.1109/ICSSE.2011.5961957.

[210] J.-R. Su, J.-G. Wang, Z.-T. Xie, Y. Yao, and J. Liu, “A Method for EEG Contributory Channel Selection Based on Deep Belief Network,” in *2019 IEEE 8th Data Driven Control and Learning Systems Conference*, 2019, pp. 1247–1252.

[211] J. Maroco, D. Silva, A. Rodrigues, M. Guerreiro, I. Santana, and A. de Mendonca, “Data mining methods in the prediction of Dementia: A real-data comparison of the accuracy, sensitivity and specificity of linear discriminant analysis, logistic,” *Bmc Res Notes*, vol. 4, p. 299, 2011.

[212] N. Padfield, J. Ren, P. Murray, and H. Zhao, “Sparse learning of band power features with genetic channel selection for effective classification of EEG signals,” *Neurocomputing*, vol. 463, pp. 566–579, 2021, doi: 10.1016/j.neucom.2021.08.067.

[213] K. Ansari-Asl, G. Chanel, and T. Pun, “A channel selection method for EEG classification in emotion assessment based on synchronization likelihood,” in *2007 15th European Signal Processing Conference*, 2007, pp. 1241–1245.

[214] S. Taran, V. Bajaj, D. Sharma, S. Siuly, and A. Sengur, “Features based on analytic IMF for classifying motor imagery EEG signals in BCI applications,” *Measurement: Journal of the International Measurement Confederation*, vol. 116, no. October 2017, pp. 68–76, 2018, doi: 10.1016/j.measurement.2017.10.067.

[215] B. H. Yilmaz, C. M. Yilmaz, and C. Kose, “Diversity in a signal-to-image transformation approach for EEG-based motor imagery task classification,” *Medical and Biological Engineering and Computing*, vol. 58, no. 2, pp. 443–459, 2020, doi: 10.1007/s11517-019-02075-x.

[216] S. R. Sreeja, “MI-classification-using-weighted sparsity.” https://github.com/BCI-HCI-IITKGP/Weighted-Sparse-classification

[217] V. J. Lawhern, “arl-eegmodels.” https://github.com/vlawhern/arl-eegmodels

[218] D. Zhang, K. Chen, D. Jian, and L. Yao, “Motor Imagery Classification via TemporalAttention Cues of Graph Embedded EEG Signals,” *IEEE Journal of Biomedical and Health Informatics*, vol. XX, no. XX, pp. 1–1, 2020, doi: 10.1109/jbhi.2020.2967128.

[219] C. Li and J. Xu, “Feature selection with Fisher score followed by the Maximal Clique Centrality algorithm can accurately identify the hub genes of hepatocellular carcinoma,” *Scientific Reports*, vol. 9, 2019.

[220] W. K. Tam, Z. Ke, and K. Y. Tong, “Performance of common spatial pattern under a smaller set of EEG electrodes in brain-computer interface on chronic stroke patients: A multi-session dataset study,” *Proceedings of the*

Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS, pp. 6344–6347, 2011, doi: 10.1109/IEMBS.2011.6091566.

[221] X. Tang, W. Li, X. Li, W. Ma, and X. Dang, "Motor imagery EEG recognition based on conditional optimization empirical mode decomposition and multi-scale convolutional neural network," *Expert Systems with Applications*, vol. 149, Jul. 2020, doi: 10.1016/j.eswa.2020.113285.

[222] C. Wang, X. Wu, Z. Wang, and Y. Ma, "Implementation of a Brain-Computer Interface on a Lower-Limb Exoskeleton," *IEEE Access*, vol. 6, pp. 38524–38534, Jul. 2018, doi: 10.1109/ACCESS.2018.2853628.

[223] L. Junwei *et al.*, "Brain Computer Interface for Neurodegenerative Person Using Electroencephalogram," *IEEE Access*, vol. 7, pp. 2439–2452, 2019, doi: 10.1109/ACCESS.2018.2886708.

[224] J. Arnin, D. Kahani, H. Lakany, and B. A. Conway, "Heterogeneous Real-Time Multi-Channel Time-Domain Feature Extraction Using Paraellel Sum Reduction on GPU," pp. 6–10, 2019, doi: 10.3217/978-3-85125-682-6-33.

[225] M. Li, J. Ma, and S. Jia, "Optimal combination of channels selection based on common spatial pattern algorithm," *2011 IEEE International Conference on Mechatronics and Automation, ICMA 2011*, pp. 295–300, 2011, doi: 10.1109/ICMA.2011.5985673.

[226] D. Zapała, P. Iwanowicz, P. Francuz, and P. Augustynowicz, "Handedness effects on motor imagery during kinesthetic and visual-motor conditions," *Scientific Reports*, vol. 11, no. 1, pp. 1–12, 2021, doi: 10.1038/s41598-021-92467-7.

[227] P. Chholak *et al.*, "Visual and kinesthetic modes affect motor imagery classification in untrained subjects," *Scientific Reports*, vol. 9, no. 1, pp. 1–12, 2019, doi: 10.1038/s41598-019-46310-9.

[228] D. J. L. G. Schutter, P. Putman, E. Hermans, and J. Van Honk, "Parietal electroencephalogram beta asymmetry and selective attention to angry facial expressions in healthy human subjects," *Neuroscience Letters*, vol. 314, no. 1–2, pp. 13–16, 2001, doi: 10.1016/S0304-3940(01)02246-7.

[229] J. Meng, L. Yao, X. Sheng, D. Zhang, and X. Zhu, "Simultaneously optimizing spatial spectral features based on mutual information for EEG classification," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 1, pp. 227–240, 2015, doi: 10.1109/TBME.2014.2345458.

[230] H. Zhai, H. Zhang, L. Zhang, and P. Li, "Laplacian-Regularized Low-Rank Subspace Clustering for Hyperspectral Image Band Selection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 3, pp. 1723–1740, 2019, doi: 10.1109/TGRS.2018.2868796.

[231] N. Cudlenco, N. Popescu, and M. Leordeanu, “Reading into the mind’s eye: Boosting automatic visual recognition with EEG signals,” *Neurocomputing*, no. xxxx, 2020, doi: 10.1016/j.neucom.2019.12.076.

[232] X. Wang, M. Hersche, B. Tömekce, B. Kaya, M. Magno, and L. Benini, “An accurate EEGNet-based motor-imagery brain–computer interface for low-power Edge computing,” *arXiv*, pp. 0–5, 2020.

[233] J. Yang, S. Yao, and J. Wang, “Deep Fusion Feature Learning Network for MI-EEG Classification,” *IEEE Access*, vol. 6, pp. 79050–79059, 2018, doi: 10.1109/ACCESS.2018.2877452.

[234] V. Quiles, E. Ianez, M. Ortiz, N. Medina, A. Serrano, and J. M. Azorin, “Lessons Learned from Clinical Trials of a Neurorehabilitation Therapy Based on tDCS, BMI, and Pedaling Systems,” *IEEE Systems Journal*, vol. 15, no. 2, pp. 1873–1880, 2021, doi: 10.1109/JSYST.2020.3026242.

[235] J. Andreu-Perez, F. Cao, H. Hagras, and G. Z. Yang, “A Self-Adaptive Online Brain-Machine Interface of a Humanoid Robot Through a General Type-2 Fuzzy Inference System,” *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 1, pp. 101–116, 2018, doi: 10.1109/TFUZZ.2016.2637403.

[236] A. B. Usakli, “Improvement of EEG Signal Acquisition: An Electrical Aspect for State of the Art of Front End,” *Computational Intelligence and Neuroscience*, 2010.

[237] R. Polikar, “Ensemble based systems in decision making,” *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, pp. 21–44, 2006, doi: 10.1109/MCAS.2006.1688199.

[238] M. Aizhong, L. Wang, and Q. Junyan, “A multiple classifier fusion algorithm using weighted decision templates,” *Scientific Programming*, 2016.

[239] A. Hernández, A. Panzio, and D. Camacho, “An Ensemble Algorithm Based on Deep Learning for Tuberculosis Classification,” 2019.

[240] C. O. Quero, D. Durini, R. Ramos-Garcia, J. Rangel-Magdaleno, and J. Martinez-Carranza, “Hardware parallel architecture proposed to accelerate the orthogonal matching pursuit compressive sensing reconstruction,” *Computational Imaging V, International Society for Optics and Photonics*, vol. 11396, p. 113960N, 2020.

[241] S. Li and H. Qi, “Sparse representation based band selection for hyperspectral images,” *Proceedings - International Conference on Image Processing, ICIP*, pp. 2693–2696, 2011, doi: 10.1109/ICIP.2011.6116223.

[242] W. Sun, L. Zhang, L. Zhang, and Y. M. Lai, “A dissimilarity-weighted sparse self-representation method for band selection in hyperspectral imagery

classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 9, pp. 4374–4388, 2016.

[243] E. Elhamifar, G. Sapiro, and R. Vidal, "See all by looking at a few: Sparse modeling for finding representative objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2012, pp. 1600–1607.

[244] M. J. Afridi, A. Ross, and E. M. Shapiro, "On automated source selection for transfer learning in convolutional neural networks," *Pattern Recognition*, vol. 73, pp. 65–75, 2018, doi: 10.1016/j.patcog.2017.07.019.

[245] A. M. Tomé, D. Malafaia, A. R. Teixeira, and E. W. Lang, "On the use of Singular Spectrum Analysis," pp. 1–23, 2018.

[246] A. K. Maddirala and R. A. Shaik, "Separation of sources from single-channel EEG signals using independent component analysis," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 2, pp. 382–393, 2018, doi: 10.1109/TIM.2017.2775358.

[247] S. Ferdowsi and V. Abolghasemi, "Multi layer spectral decomposition technique for ERD estimation in EEG μ rhythms: An EEG–fMRI study," *Neurocomputing*, vol. 275, pp. 1836–1845, 2018, doi: 10.1016/j.neucom.2017.10.016.

[248] J. Cheng *et al.*, "Remove diverse artifacts simultaneously from a single-channel EEG based on ssa and ica: A semi-simulated study," *IEEE Access*, vol. 7, pp. 60276–60289, 2019, doi: 10.1109/ACCESS.2019.2915564.

[249] B. Blankertz, "Data set IVc ‹motor imagery, time-invariance problem›," *BCI Competition III*.

[250] H. S. Bıçakcı, M. Santopietro, M. Boakes, and R. Guest, "Evaluation of Electrocardiogram Biometric Verification Models Based on Short Enrollment Time on Medical and Wearable Recorders," 2021.

[251] M. Boakes, R. Guest, and F. Deravi, "Adapting to Movement Patterns for Face Recognition on Mobile Devices," in *InInternational Conference on Pattern Recognition*, 2021, pp. 209–228.

Appendix

This Appendix contains results relevant to Chapter 6.

A.1 Result Tables Tuning of GA Population Size

Table A 1: Grid-search validation accuracy results for GA hyperparameter tuning with ShallowConvNet. The results for different population sizes are shown. The channel subset size was fixed at 11.

	Population Size			
Subjects	5	10	20	30
A1	72.94%	70.58%	70.58%	71.76%
A2	52.17%	50.93%	52.80%	56.52%
A3	74.19%	75.48%	75.48%	75.48%
A4	86.59%	89.02%	90.24%	89.63%
A5	71.15%	70.51%	71.79%	69.87%
A6	61.04%	63.64%	63.64%	65.58%
A7	68.82%	71.76%	72.35%	70.59%
A8	89.02%	89.63%	89.02%	88.41%
A9	53.55%	57.42%	56.77%	57.42%

Table A 2: The computational times for each population size recorded during the grid-search parameter tuning for the GA on the Graz 2A dataset, with ShallowConvNet. The results for different population sizes are shown.

	Population Size			
Subjects	5	10	20	30
A1	722s	1014s	2017s	3015s
A2	632s	1611s	1943s	6978s
A3	712s	1610s	1874s	4521s
A4	547s	995s	2034s	3120s
A5	518s	949s	3015s	2931s
A6	518s	940s	1976s	3067s
A7	532s	1043s	3060s	6141s
A8	549s	1741s	2603s	4032s
A9	869s	954s	2831s	7188s

Table A 3: The grid-search validation accuracies obtained when tuning the GA channel selection method with EEGNet on the Graz 2A dataset. The results for different population sizes are shown. The channel subset size was fixed at 11.

	Population Size			
Subjects	5	10	20	30
A1	69.41%	71.18%	70.00%	71.18%
A2	52.17%	50.93%	49.69%	54.04%
A3	76.13%	77.42%	76.77%	78.06%
A4	87.80%	87.20%	89.02%	89.02%
A5	64.10%	68.59%	67.31%	64.74%
A6	67.53%	66.23%	69.48%	68.83%
A7	65.88%	68.24%	74.12%	68.24%
A8	87.20%	89.02%	89.02%	89.02%
A9	53.55%	55.48%	57.42%	56.13%

Table A 4: The grid-search computational times obtained when tuning the GA channel selection method with EEGNet on the Graz 2A dataset. The results for different population sizes are shown.

	Population Size			
Subjects	5	10	20	30
A1	769s	910s	1818s	6033s
A2	872s	1213s	2018s	3718s
A3	947s	1070s	2495s	2841s
A4	543s	1395s	2295s	6239s
A5	830s	1731s	1743s	4256s
A6	729s	846s	1788s	6735s
A7	665s	880s	3470s	8995s
A8	934s	1022s	3046s	9044s
A9	683s	1056s	2980s	6774s

A. 2 Result Tables Comparing Subject Specific and Subject Independent Channel Selection Methods

A.2.1 Graz 2A Dataset

Table A 5: The results for individual subjects in the Graz 2A dataset when using ICS for channel selection with ShallowConvNet and EEGNet with 3 channels in the subset chosen with ICS layer selection.

	ShallowConvNet		EEGNET	
Subject	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
A1	53.61%	66.88%	61.88%	63.33%
A2	40.42%	27.64%	40.63%	34.65%
A3	64.72%	65.90%	66.53%	55.35%
A4	44.27%	49.79%	42.29%	47.50%
A5	28.47%	29.24%	25.83%	26.18%
A6	36.81%	35.63%	38.06%	38.89%
A7	55.42%	48.06%	53.68%	56.18%
A8	69.34%	70.43%	67.92%	65.25%
A9	60.56%	53.40%	65.28%	64.17%

Table A 6: The results for individual subjects in the Graz 2A dataset when using ICS for channel selection with ShallowConvNet and EEGNet with 6 channels in the subset.

	ShallowConvNet		EEGNET	
Subject	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
A1	69.58%	76.39%	70.28%	72.64%
A2	42.01%	42.36%	43.06%	43.75%
A3	72.85%	76.74%	78.54%	75.42%
A4	52.50%	56.88%	46.77%	47.19%
A5	29.51%	30.97%	28.61%	28.96%
A6	42.29%	40.42%	36.74%	42.29%
A7	63.33%	70.42%	61.18%	68.26%
A8	74.00%	75.22%	77.95%	75.57%
A9	68.40%	63.13%	73.47%	66.46%

Table A 7: The results for individual subjects in the Graz 2A dataset when using ICS for channel selection with ShallowConvNet and EEGNet with 11 channels in the subset.

	ShallowConvNet		EEGNET	
Subject	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
A1	75.07%	77.71%	73.40%	76.39%
A2	44.51%	45.14%	43.33%	40.90%
A3	79.31%	80.14%	80.83%	78.96%
A4	56.67%	56.77%	48.54%	48.96%
A5	33.26%	42.57%	32.99%	28.47%
A6	41.81%	42.29%	38.33%	40.28%
A7	76.25%	77.22%	69.86%	70.97%
A8	77.51%	77.74%	78.31%	75.96%
A9	74.65%	71.67%	72.92%	76.39%

A.2.2 HG Dataset

Table A 8: The results for individual subjects in the HG dataset when using ICS for channel selection with ShallowConvNet and EEGNet with 3 channels in the subset chosen with ICS layer selection.

	ShallowConvNet		EEGNET	
Subject	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
H2	75.86%	73.25%	67.63%	69.50%
H3	79.75%	77.00%	64.63%	81.38%
H4	84.63%	81.63%	61.75%	84.63%
H5	72.63%	74.25%	64.38%	70.38%
H6	64.63%	70.63%	60.13%	76.38%
H7	63.27%	64.28%	65.16%	63.52%
H8	80.50%	76.75%	68.88%	73.88%
H9	54.25%	53.50%	57.63%	58.00%
H10	70.88%	54.75%	72.38%	69.63%
H11	57.88%	55.25%	53.75%	63.75%
H12	86.13%	84.13%	74.75%	81.20%
H13	74.47%	69.06%	62.26%	68.05%
H14	54.75%	53.38%	62.88%	50.50%

Table A 9: The results for individual subjects in the HG dataset when using ICS for channel selection with ShallowConvNet and EEGNet with 6 channels in the subset chosen with ICS layer selection.

	ShallowConvNet		EEGNET	
Subject	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
H2	82.63%	78.13%	72.25%	75.00%
H3	87.63%	90.38%	83.75%	87.75%
H4	90.88%	93.25%	82.38%	92.38%
H5	80.50%	79.75%	72.63%	75.25%
H6	81.00%	76.38%	72.25%	84.63%
H7	76.10%	74.97%	68.43%	68.18%
H8	88.13%	84.38%	72.25%	80.13%
H9	64.88%	64.38%	64.25%	61.38%
H10	83.88%	76.13%	78.88%	75.75%
H11	68.00%	72.63%	58.13%	67.00%
H12	92.13%	87.63%	75.75%	87.88%
H13	83.77%	81.76%	73.21%	75.35%
H14	55.63%	61.50%	72.25%	56.00%

Table A 10: The results for individual subjects in the HG dataset when using ICS for channel selection with ShallowConvNet and EEGNet with 11 channels in the subset chosen with ICS layer selection.

	ShallowConvNet		EEGNET	
Subject	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
H2	81.88%	83.50%	74.25%	77.00%
H3	95.25%	92.88%	89.00%	91.75%
H4	93.63%	95.13%	90.88%	93.75%
H5	89.38%	86.75%	74.88%	85.38%
H6	91.00%	86.13%	78.75%	87.50%
H7	74.21%	79.87%	74.59%	73.58%
H8	92.63%	90.38%	74.63%	86.50%
H9	72.50%	73.63%	71.75%	68.63%
H10	86.25%	84.38%	80.50%	82.38%
H11	76.63%	72.63%	67.38%	68.25%
H12	93.13%	92.88%	86.88%	87.88%
H13	89.31%	85.53%	70.06%	77.48%
H14	58.13%	56.25%	74.25%	58.75%

Table A 11: The results for individual subjects in the HG dataset when using ICS for channel selection with ShallowConvNet and EEGNet with 22 channels in the subset chosen with ICS layer selection.

	ShallowConvNet		EEGNET	
Subject	Subject-Specific	Subject-Independent	Subject-Specific	Subject-Independent
H2	84.50%	87.25%	78.75%	83.13%
H3	97.50%	96.75%	91.00%	92.63%
H4	93.25%	97.25%	94.38%	96.88%
H5	90.50%	90.63%	80.88%	88.88%
H6	93.00%	91.88%	84.88%	89.63%
H7	81.26%	83.27%	74.97%	71.45%
H8	95.13%	91.50%	76.88%	85.25%
H9	81.13%	83.25%	74.25%	72.88%
H10	88.00%	84.00%	83.63%	86.25%
H11	80.50%	75.88%	64.25%	68.75%
H12	96.63%	96.88%	90.63%	92.88%
H13	92.45%	92.83%	80.25%	79.25%
H14	67.25%	59.00%	59.50%	59.38%

A.3 Result Tables for the CCS Method

Graz 2A Dataset

Table A 12: Results for ShallowConvNet and EEGNet when using the CCS method for subject-independent channel selection on the Graz 2A dataset, with subset sizes of 3, 6 and 11.

	ShallowConvNet			EEGNet		
Subject	3	6	11	3	6	11
A1	58.54%	70.00%	73.06%	46.11%	60.21%	67.99%
A2	33.19%	34.79%	38.40%	27.57%	27.08%	32.85%
A3	63.96%	70.69%	76.11%	57.57%	71.67%	73.96%
A4	45.31%	50.42%	57.81%	43.33%	45.83%	50.73%
A5	29.72%	32.57%	36.25%	27.15%	26.04%	29.24%
A6	35.90%	37.08%	41.39%	31.53%	32.92%	40.28%
A7	56.81%	68.06%	78.47%	37.50%	44.31%	70.42%
A8	61.22%	71.97%	78.41%	50.60%	65.91%	76.05%
A9	60.97%	67.36%	71.94%	57.92%	65.49%	73.26%

HG Dataset

Table A 13: Results for ShallowConvNet and EEGNet when using the CCS method for subject-independent channel selection on HG dataset, with subset sizes of 3, 6, 11 and 22.

	ShallowConvNet				EEGNET			
Subject	3	6	11	22	3	6	11	22
H2	62.75%	68.38%	77.63%	78.25%	62.38%	65.50%	70.25%	79.25%
H3	68.00%	76.13%	84.00%	92.88%	62.25%	76.63%	84.88%	91.75%
H4	71.50%	78.75%	91.88%	78.25%	76.88%	84.13%	90.63%	95.75%
H5	62.00%	73.13%	77.75%	88.13%	62.25%	74.13%	76.50%	85.63%
H6	60.63%	71.13%	82.88%	88.75%	56.38%	68.00%	80.75%	87.88%
H7	50.82%	68.43%	69.31%	81.51%	58.11%	68.81%	72.20%	73.96%
H8	71.25%	81.00%	88.50%	89.13%	72.00%	80.13%	83.38%	86.13%
H9	46.00%	55.13%	56.50%	73.50%	44.13%	56.63%	58.63%	69.25%
H10	55.88%	72.63%	77.25%	84.38%	60.38%	67.50%	77.38%	81.38%
H11	45.75%	60.13%	68.00%	77.75%	48.13%	60.25%	63.63%	70.25%
H12	70.88%	81.00%	89.13%	95.25%	68.25%	83.00%	90.13%	91.50%
H13	65.28%	73.21%	84.03%	87.67%	59.12%	66.92%	77.86%	82.89%
H14	49.63%	53.63%	57.88%	62.63%	50.38%	54.00%	56.13%	60.63%

A.4 Result Tables for CNMF Method

Graz 2A Dataset

Table A 14: Results for ShallowConvNet and EEGNet when using the CNMF method for subject-independent channel selection on the Graz 2A dataset, with subset sizes of 3, 6 and 11.

	ShallowConvNet			EEGNet		
Subject	3	6	11	3	6	11
A1	53.13%	57.99%	73.33%	58.33%	61.53%	70.90%
A2	37.78%	39.51%	45.28%	41.53%	42.36%	44.24%
A3	53.96%	72.22%	76.46%	58.82%	74.93%	78.75%
A4	41.67%	44.58%	53.75%	40.73%	41.15%	45.42%
A5	25.63%	29.93%	26.60%	24.86%	27.43%	26.94%
A6	35.83%	38.61%	42.22%	40.07%	40.35%	42.99%
A7	48.96%	56.39%	74.44%	51.32%	49.65%	63.13%
A8	60.27%	63.97%	76.66%	60.90%	63.98%	74.26%
A9	56.18%	62.15%	68.06%	60.69%	66.94%	73.19%

HG Dataset

Table A 15: Results for ShallowConvNet and EEGNet when using the CNMF method for subject-independent channel selection on the HG dataset, with subset sizes of 3, 6, 11 and 22.

	ShallowConvNet				EEGNET			
Subject	3	6	11	22	3	6	11	22
H2	56.75%	66.00%	76.25%	80.75%	53.63%	63.13%	69.50%	76.63%
H3	50.50%	70.50%	82.13%	92.13%	51.75%	74.00%	82.38%	87.63%
H4	66.88%	71.00%	85.75%	93.88%	72.38%	75.00%	91.00%	95.88%
H5	69.63%	72.75%	81.63%	87.38%	68.75%	71.75%	79.38%	83.75%
H6	58.38%	65.38%	78.00%	87.38%	56.50%	66.88%	76.13%	81.00%
H7	48.68%	64.40%	75.35%	82.64%	50.82%	66.42%	69.06%	73.21%
H8	62.25%	73.63%	77.50%	88.00%	59.88%	73.63%	73.63%	82.63%
H9	43.13%	54.63%	60.00%	70.13%	43.00%	56.75%	60.50%	67.00%
H10	51.00%	58.25%	74.38%	84.13%	49.88%	57.00%	77.25%	85.88%
H11	57.50%	61.88%	66.13%	70.75%	59.50%	61.25%	64.75%	67.00%
H12	59.88%	70.63%	92.75%	95.75%	57.00%	72.00%	87.255	92.38%
H13	54.21%	70.70%	80.25%	85.41%	56.23%	66.42%	75.22%	83.02%
H14	46.00%	51.88%	76.25%	59.25%	47.38%	53.50%	69.50%	76.63%