République Algérienne démocratique et populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
École Nationale Polytechnique
École Militaire Polytechnique

Control Engineering Department
Process Control Laboratory
Complex System Control and Simulators Laboratory

A Thesis Submitted In Fulfillment of The Requirements
For The Degree of State Engineer

Mobile Robot Control via Brain Computer Interface and Fatigue Detection based on EEG Signals

---

Presented on 08/07/2021 By :
Moussaoui Younes
Latreche Mahdi

Jury Members :

President	O.Stihi	MAA	ENP
Examiner	D.Boukhetala	Professor	ENP
Supervisor	M.Tadjine	Professor	ENP
Supervisor	M.Chakir	MCA	ENP
Supervisor	M.Guiatni	Professor	EMP

ENP 2021

République Algérienne démocratique et populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
École Nationale Polytechnique
École Militaire Polytechnique

Control Engineering Department
Process Control Laboratory
Complex System Control and Simulators Laboratory

A Thesis Submitted In Fulfillment of The Requirements
For The Degree of State Engineer

Mobile Robot Control via Brain Computer Interface and Fatigue Detection based on EEG Signals

---

Presented on 08/07/2021 By :
Moussaoui Younes
Latreche Mahdi

Jury Members :

President	O.Stihi	MAA	ENP
Examiner	D.Boukhetala	Professor	ENP
Supervisor	M.Tadjine	Professor	ENP
Supervisor	M.Chakir	MCA	ENP
Supervisor	M.Guiatni	Professor	EMP

ENP 2021

République Algérienne démocratique et populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
École Nationale Polytechnique
École Militaire Polytechnique

Département D'Automatique
Laboratoire de Commande des Processus
Laboratoire de Commande des systèmes Complexes et Simulateurs

Mémoire du projet de fin d'études pour l'obtention du diplome d'ingenieur d'etat en Autimatique

---

Commande d'un Robot Mobile via une Interface Cerveau-Machine et Détection de la Fatigue en utilisant les Signaux EEG

Presenté le 08/07/2021 par :
Moussaoui Younes
Latreche Mahdi

Membres du jury :

President	O.Stihi	MAA	ENP
Examinateur	D.Boukhetala	Prof	ENP
Promoteur	M.Tadjine	Prof	ENP
Promoteur	M.Chakir	MCA	ENP
Promoteur	M.Guiatni	Prof	EMP

ENP 2021

ملخص

في السنوات العشر الماضية، أدى التطور السريع للطرق المعقدة لتسجيل إشارات الدماغ، الارتفاع المتسارع لقوة الحوسبة المتاحة، بالإضافة إلى زيادة الوعي بخلل وظائف الدماغ و إستغلال وضائف الدماغ من أجل التحكم، بالباحثين إلى استخدام تسجيلات فسيولوجية عصبية واسعة النطاق. مخطط كهربية الدماغ (EEG) هو مقياس شائع جدًا لديناميات الدماغ بسبب طبيعته غير الجراحية و المجموعة الكبيرة من التطبيقات الممكنة التي تتضمنه. في هذا المشروع، تم تنفيذ تطبيقين، الأول هو بنية جديدة مقترحة لواجهة الكمبيوتر الدماغي (BCI) على أساس الصور الحركية (MI) للتحكم في روبوت متحرك في الوقت الحقيقي في ثلاثة اتجاهات، مبنية على حوسبة الطاقة الطيفية، النمط المكاني المشترك متعدد الفئات(CSP) ، والتعلم الآلي(ML) . يتضمن التطبيق الثاني تطوير برنامج لتشخيص الإرهاق بالاعتماد على مجموعة بيانات مكتسبة، و بإستخدام التعلم الآلي (ML)، التعلم العميق(DL) ، والخوارزميات الجينية(GA).

الكلمات المفتاحية: مخطط كهربية الدماغ، واجهة الكمبيوتر الدماغي، الصور الحركية ، التعب ، استخراج الميزات ، التعلم الآلي ، التعلم العميق ، الخوارزمية الجينية.

Resumé

Au cours de la dernière décennie, le développement rapide de méthodes d'enregistrement des signaux cérébraux, l'augmentation exponentielle de la puissance de calcul disponible ainsi que la sensibilisation accrue aux dysfonctionnements du cerveau et aux troubles mentaux ont conduit les chercheurs à utiliser les enregistrements neurophysiologiques à grande échelle pour le diagnostic et le control de ces derniers. Les électroencéphalogrammes (EEG) sont une mesure très populaire de la dynamique du cerveau en raison de leur nature non invasive et de leur large gamme d'applications possibles. Dans ce projet, deux applications ont été mises en œuvre, la première est une nouvelle architecture d'interface cerveau-ordinateur (ICO) basée sur l'imagerie motrice (IM) pour contrôler un robot mobile en temps réel, basée sur le calcul de la puissance spectrale, le modèle spatial commun multi-classe (MSC) et l'apprentissage automatique (AA). La seconde consiste à développer une approche de détection de la fatigue en s'appuyant sur un ensemble de données acquises et en utilisant l'apprentissage automatique (AA), l'apprentissage profond (DL) et les algorithmes génétiques (GA).

Mots-clés : EEG, ICO, Imagerie Motrice, Fatigue, Extraction de Caractéristiques, Apprentissage Automatique, Apprentissage Profond, Algorithme Génétique.

Abstract

In the last decade, the rapid development of complex methods for recording brain signals and the exponential rise of available computing power as well as the increased awareness of brain dysfunctions and mental disorders, have led researchers to use large-scale neurophysiological recordings for abnormal behaviours detection, diseases diagnosis, and motor control. Electroencephalograms (EEG) are a very popular measurement for brain activities because of their non-invasive nature and their wide spectrum of possible applications. In this context, two applications have been developed in this project, the first aims to design a novel Brain Computer Interface (BCI) architecture based on Motor Imagery (MI) for real time control of a mobile robot. Spectral power computing, multi-class Common Spatial Pattern (CSP), and Machine Learning (ML) have been used to reach this aim. The second involves the proposal of an approach for fatigue detection using Machine Learning (ML), Deep Learning (DL), and Genetic Algorithms (GA).

Keywords: EEG, BCI, Motor Imagery, Fatigue, Feature Extraction, Machine Learning, Deep Learning, Genetic Algorithm.

Acknowledgments

We would first like to express our gratitude to Pr. **Mohamed TADJINE** and Dr. **Messaoud CHAKIR** for their help and understanding throughout this project as well as for their guidance, advice and mentorship over the past three years.

We thank the President of jury Mr Omar STIHI, Professor at ENP, and the examiner: Mr Djamel BOUKHETALA, Professor at the ENP, for accepting to be members of the reading committee and for their constructive analysis of the present work.

We would like to address a very special thanks to Pr. **Mohamed GUIATNI** for welcoming us into the $L(CS)^2$ where all the needed material and more was at our disposal, for giving us the opportunity to work on this interesting topic, and for being there to answer all our questions.

We would also like to thank Dr. **Fazia Sbargoud** for her kind help and precious advice during this project, her knowledge and feedback were integral to achieving this work.

We thank Mrs. **Hafsa OSMANI**,Ms. **Malika AZZOUZ** Dr. **Amine ALLOUANE**, Dr. **Yasser BOUZID** and Mr. **Ahmed ZERMANE** for their kind help and support during our stay at the LCS^2.

Special thanks go to: **Kheireddine MESKINE**, **Ilyes DEBIH**, **Salah Eddine MAHDJOUB ARAIBI**, **Abdelhak Saadoune**, **Omar CHIKHI** and all who participated in the lengthy data acquisition process which provided us with valuable data and helpful hints.

We would also like to thank the teachers and professors of the National Polytechnic School and especially of the Control Engineering Department who provided us with the necessary tools to succeed in our engineering studies.

Finally, we would like to express our gratitude to all those who have contributed in any way, directly or indirectly, to the realization of this project.

Dedication

Thanks to Noble Allah the Almighty who gave me the courage and the patience to realize this work.

To my parents whom I am very grateful; my dear mother, who never stopped praying for me and supporting every action and decision i make.

To my father for being the example of commitment and discipline, this humble work is dedicated to you.

To my sisters, your encouragement and moral support made me strong to adventure ahead.

With genuine gratitude and warm regard that I dedicate this work to Pr. Mohamed Guiatni for being such an inspiration and mentor during these two years.

I would also like to express my sincere thanks to all my family members who have always supported and encouraged me

To my friends in EMP : Omar, Salah, Kheireddine, Loukman, Oussama and Abdelhak who made my journey in EMP joyful and unforgettable and didn't hesitate for helping me in this project.

To my beloved friends and my second family :my partner Mahdi, Amine, Abdou, Omar, Katia, Dhayaeddine, Soheib, Ramzy, Anis, Rahim, Oussama, Bouche, Yacine, Meriem, Walid. I will always appreciate our friendship.

Younes MOUSSAOUI

Dedication

Thanks to Allah the Almighty for giving me the courage and the patience achieve this project.

I dedicate this thesis to my dear parents, who never stopped praying for me and supporting every action and decision i make.

To my sister and brothers who have always supported and encouraged me, and to my niece Céline.

May Allah protect you all and keep you safe each and every day.

To my partner Younes, and to all other twenty four of my classmates in the Control and Systems Engineering class, past and present.

To my mentor and friend Noureddine, as well as my National Polytechnic School friends, Abderrahmene, Belhadj, Hicham, Mohamed, Omar, and Saber.

To my new friends at the National Military School, Abdelhak, Ikbal, Ilyes, Kheireddine, Omar, Romaissa, and Salaheddine, who made my four month stay in their school enjoyable with their company and helpful guidance.

To my hometown friends, Abdelmounaim, Badie Ezzamene, Nedjmeddine, and Salaheddine.

May you all find joy and happiness wherever life takes you.

Mahdi LATRECHE

Contents

List of Figures

List of Tables

List of Acronyms

AB	Attribute Bagging/Random Subspace Method
AI	Artificial Intelligence
ANN	Artificial Neural Networks
BCI	Brain Computer Interface
CAR	Common Average Reference
CNN	Convolutional Neural Networks
CSP	Common Spacial Pattern
DL	Deep Learning
EEG	Electroencephalogram
EMG	Electromyography
EOG	Electrooculography
ERD	Event RelatedDesynchronization
ERS	Event Related Synchronization
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
GA	Genetic Algorithm
GUI	Graphical User Interface
ICA	Independent Component Analysis
KNN	K-Nearest Neighbors
KSS	Karolinska Sleepiness Scale
LDA	Linear Discriminant Analysis
LSTM	Long Short Term Memory
ML	Machine Learning

N/A	Not Applicable/Not Available
NN	Neural Network
OVR	One-Versus-The-Rest
PCA	Principal Component Analysis
PSD	Power Spectral Density
RNN	Recurrent Neural Networks
SGD	Stochastic Gradient Descent
SNR	Signal To Noise Ratio
SVM	Support Vector Machine
TN	True Negatives
TP	True Positives

Introduction

After several decades of research and many advances in recent years, there has been an increasing interest in exploring brain activity regardless of the complexity of the human brain and the lack of an accurate model of the dynamics of our cerebrum [39]. Thus, many applications exploring and processing the electrical brain activity measurement (also called Electroencephalography (EEG)) [23][61], were successfully achieved by researchers.

EEG is a non-invasive and inexpensive technique that records the electrical activity produced by the brain with good temporal and frequency resolution. However, it remains a challenge to be used effectively due to its complexity and difficulties that arise during the acquisition process [117].

In our work, two applications were developed using EEG signals. In the first application, a Brain Computer Interface (BCI) was realized in order to control an external device. BCIs are recently developed as alternative technologies for human computer interaction. These interfaces aim to interpret the brain's activity as user intentions in active BCI systems or cognitive/ emotional state in passive BCI systems with a unique and attractive property of not requiring muscle movements. Thus, it makes BCIs beneficial to people who lost the control over all muscles such as paraplegic or hemiplegic peoples [58][34].

Even so , we are still far from having BCI applications used for daily tasks. Mainly, they are applied for medical purposes and research works [119][114]. In our work , we aim to design a BCI based application for mobile robot control using advanced EEG signals processing techniques.

In a second application, EEG signals are processed for fatigue detection. According to the World Health Organization, 1.25 million people die on the roads due to accidents each year across the globe [1]. Drowsy driving is one of the leading causes of car crashes around the world. According to the US National Highway Traffic Safety Administration (NHTSA), 679 deaths are registered from drowsy driving related crashes in 2019 and a total of 100,000 crashes related to driver fatigue resulted in an estimated 1,550 deaths, 71,000 injuries [2]. In fact, a large number of existing works in the literature have attempted to deal with the problem of fatigue detection and have propounded various drowsiness and fatigue monitoring systems [50][85]. These studies are mainly focused on pilot fatigue and driving drowsiness detection and evaluation [77].

[1] These statistics are extracted from :"Global Status Report on Road Safety 2015,"

[2] Source: https://www.nhtsa.gov/risky-driving/drowsy-driving

In addition, laboratories prototypes could not make their way into the real world owing to their expensiveness and poor detection performance [24]. Even though various commercialized driver alertness or fatigue monitoring systems have been developed by automobile companies, these systems are only being used in the vehicles of the respective companies [29]. Most of the existing systems use behavioral measure-based drowsiness detection algorithms that involve analyzing facial video or eye tracking data[24].

With the recent advances in artificial intelligence, classification and analysis of EEG signals have become more accessible. From machine learning classifiers to deep learning algorithms, the identification of discriminate characteristics has been automated where the pre-trained model provided the appropriate commands and response.

Thesis Organization

Chapter1: titled "Introduction to Electroencephalography", it is a general introduction on the measurement of the brain activity and EEG signals characteristics. We will also give an overview of the existing systems and applications given in the literature.

Chapter2: titled "State of the art on EEG signals processing ", in this chapter we will derive the different methods applied for EEG processing and classification, from preprocessing, feature extraction methods to machine and deep learning algorithms used for the classification. We will also detail the genetics algorithm used for feature selection and dimensional reduction.

Chapter3: titled " Brain Computer Interface Implementation for Mobile Robot Control" where we derive the implementation process for the realisation of BCI used to control an epuck mobile robot. The chapter includes the obtained performance of our private dataset as well as a selected public dataset.

Chapter4: titled "EEG Signals Classification for Fatigue Detection" where we cover the methods used and the computer implementation of fatigue detection approach based on EEG. In this chapter, we will give some notions in deep learning and genetics algorithms, we will also present the proposed architecture as well as the method used for EEG signals processing.

1 Introduction to Electroencephalography

Abstract

The aim of this chapter is to provide fundamental definitions and concepts of brain signal measurements that are related to and used in this project. The chapter starts with a small description of the Cortex's components in the section 1.1. Then, we define the Electroencephalogram (EEG) signals; the characteristics of these signals are also mentioned in Section 1.2, we also detail the international 10-20 system for electrode placement. In Section 1.3 and 1.4,we discuss the acquisition of EEG signals and the different artifacts and noises associated with them. Furthermore, we will enumerate in Section 1.5, 1.6 and 1.7 the application of EEG signal in Brain Computer Interfaces and drowsiness study and detection. We finally end with a conclusion of the chapter.

1.1 Brain Anatomy

The brain is a well sophisticated organ that allows full control of the functionality of the human body, interprets information from the outside world and reflects the essence of thought and soul. The brain also has full governance over the intelligence, creativity, emotion and memory.

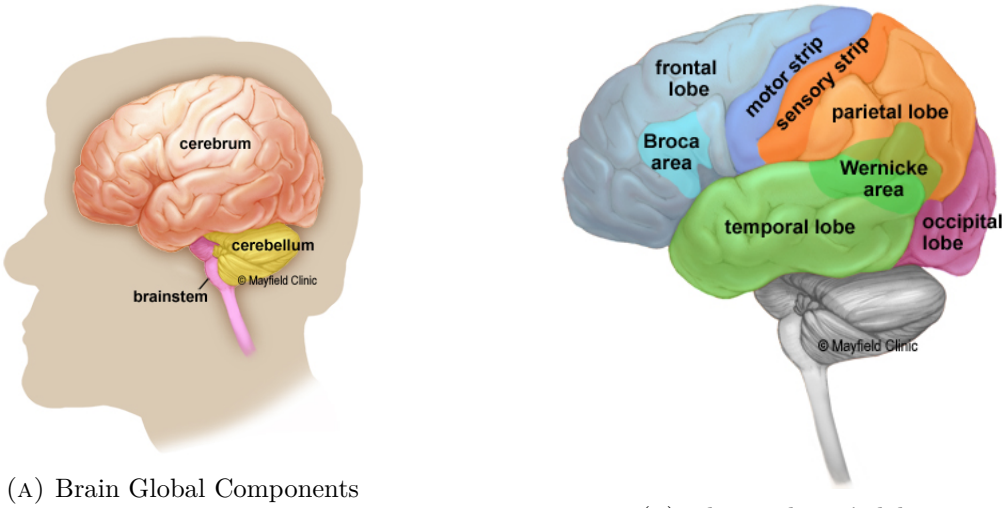
1.1.1 Brain Composition

The brain is composed of:

- **Cerebrum**
  It is the largest component of the brain and composed of right and left hemispheres. Higher functionality such as vision, reasoning, learning and movement control are performed in this part of the brain.
- **Cerebellum**
  Located under the cerebrum with a functionality of coordination of the muscle movements, maintaining posture and balance.

- **Brainstem**
  It is considered the relay center connecting the cerebrum and cerebellum as it can be seen in the figure below. Many automatic functions are performed in this center such as breathing, wake and sleep cycles and heart rate. Figure 1.1a[1] demonstrates the brain components as describe above.

The cerebral hemisphere is also characterized by cracks, dividing the brain into multiple lobes. Each hemisphere has 4 lobes: frontal, temporal, parietal and occipital, and the motor cortex is located between the frontal and the parietal lobes as can be seen in Figure 1.1b[2] below.



(A) Brain Global Components

(B) The cerebrum's lobes

FIGURE 1.1: Brain Anatomy.

1.1.2 Brain Functionalities

The most important functionalities of the frontal lobe are intelligence, concentration, and body movement (motor strip).

The parietal lobe interprets vision, hearing, and sensory signals as well as memory.

The occipital lobe interprets vision (color, light) as well as the movement .

The surface of the brain is called the cortex. It has a folded appearance, with hills and valleys. The cortex contains approximately 16 billion neurons , which are arranged in specific layers. The nerve cell body stains the cortex grayish brown, hence the name-gray matter, this is illustrated in Frigure 1.2[3]. Below the cortex are long nerve fibers (axons) that connect areas of the brain-called white matter.

[1]Source: https://mayfieldclinic.com/pe-anatbrain.html
[2]Source: https://mayfieldclinic.com/pe-anatbrain.html
[3]Source: https://mayfieldclinic.com/pe-anatbrain.html

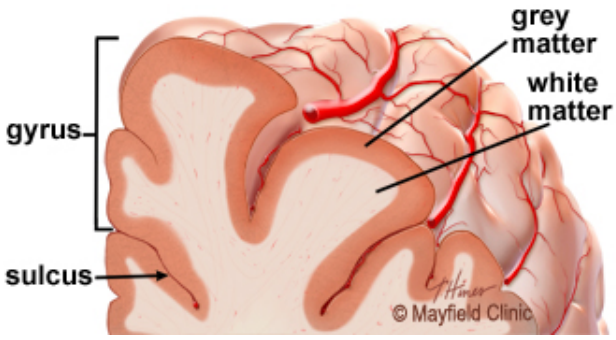


FIGURE 1.2: The cortex components

1.2 Electroencephalography

Electroencephalography (EEG) is a non-invasive measurement of electrical potential that reflects the electrical activity of the human brain. EEG is considered as a ready-made test that provides evidence of how the brain functions over time. EEG is widely used by physicians and scientists to study brain function and diagnose neurological diseases, such as epilepsy [26], brain tumors [68], head injuries [82], sleep disorders [96], dementia and monitoring depth Anesthesia during surgery [97]. It also helps diagnose abnormalities, behavior disorders (such as autism) [1],and depression [16].

In 1929, Hans Berger introduced the world's first EEG recorder. He was a neuropsychiatrist at the University of Jena in Germany, he used the German term "Elektroenzephalogramm" to describe the brain. He suggested that changes in brain currents depend on the functional state of the brain, such as sleep, fatigue, anesthesia, and epilepsy. This is a revolutionary idea that helped create a new branch of medicine called neurophysiology[46].

Since the advent of personal computing in the 1970s, engineers have been trying to shrink the communication gap between humans and computer technology. The process started with The development of graphical user interfaces (GUI), computers, and mice [98] has led to more Intuitive technology, especially with the advent of computational intelligence. Today, the ultimate boundary between humans and computers is building a bridge through the use of brain-computer interfaces (BCI), which enables computers to be deliberately controlled by monitoring brain signals.

1.2.1 EEG Signal Properties

EEG equipment is widely used to record brain signals in BCI systems Because it is non-invasive, has high temporal resolution, allows the user to freely move, and has a relatively Low cost[75]. Although BCI can be designed to use EEG signals for control in many ways, Motion Image (MI) BCI, where the user imagines the movement of the limbs in order to control the control system, has been extensively studied[63][4][20]. This interest is due to their extensive application potential in the fields of neurorehabilitation, neuroprosthetics and games, among which decoding the user's idea of imaginative movement will be invaluable [3].

Due to the weak electrical signal produced by the neurons with an order of the microvolt; usually between 0.5 and 100 µV as it can be seen in Figure 1.3[83], an amplification device is needed. Given the electrical conductivity of the skull and scalp, EEG signals are considered as the result of the sum of the potentials of a large number of neurons. The EEG signal is also altered by the skull and cerebrospinal fluid. In effect, the signal received by the electrodes on the surface of the skull is modified through the skull and scalp. Considering these two major reasons, the localisation of the signal source has proven to be a challenge.[3]

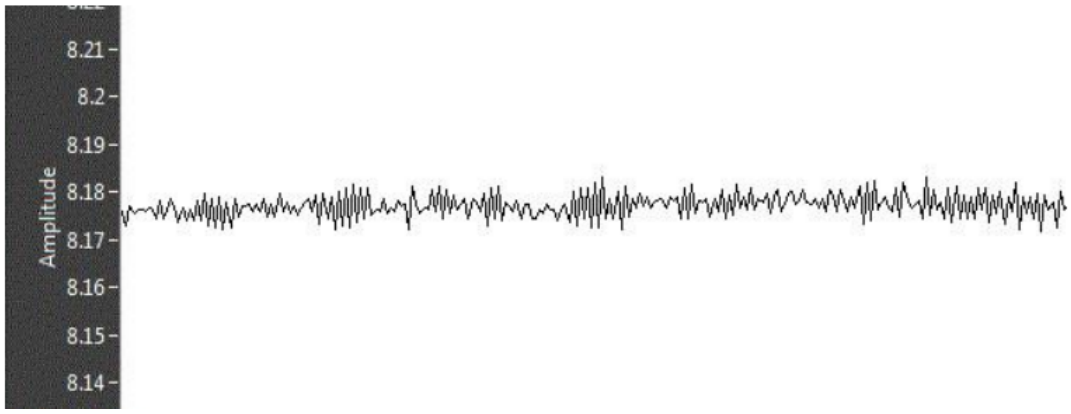


FIGURE 1.3: EEG signal

1.2.2 EEG Signal Spectral Components

EEG signals are spectrally defined at low frequencies (0-100Hz), the important information are generally found between 0.1Hz and 40Hz.The signal is subdivided into five important wave bands defined as follows:

- **Delta Waves (between 0.1Hz and 4Hz)**
  the dynamics of the signal in this band is the slowest but possesses the highest amplitude. They represent the gray matter of the brain. They appear in all phases of sleep. Physiologically, they reflect the deep third stage of sleep.
- **Theta waves (between 4Hz and 8 Hz)**
  In this band the signals represent the activity of the subconscious. They are observed during relaxation and drowsiness. Physiologically, they reflect light slow sleep.
- **Alpha waves (between 8Hz and 12 Hz)**
  These waves represent white matter and establish the link between the conscious and the subconscious. They describe the activities made in quiet status and in a status of concentration.
- **Beta waves (between 12Hz and 30 Hz)**
  In this wave band, the signals are related to the sensory activity of the individual. They are present during mental exertion. Physiologically, they reflect the normal electrical activity of the brain when conscious and alert.
- **Gamma waves (above 30 Hz)**
  These waves are linked to complex brain activities like cognitive processing.

Figure 1.4 represents EEG bands in respect to their frequency locations:

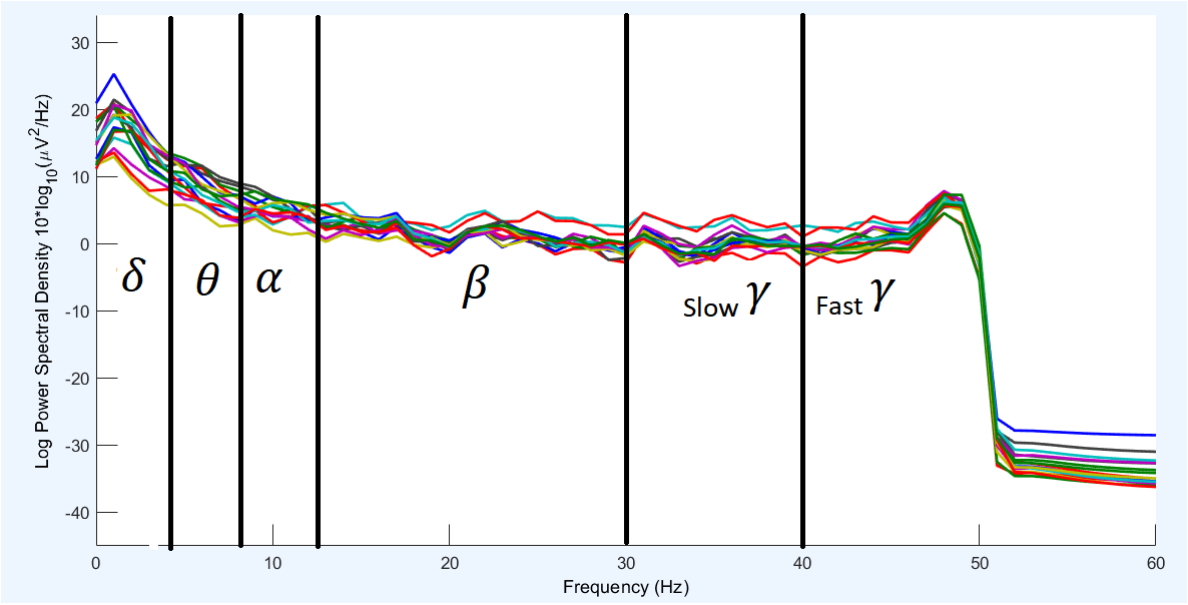


FIGURE 1.4: EEG wave bands

1.2.3 Evoked Potentials

Evoked potentials represent the response of the nervous system to an external stimuli namely visual, auditive or acoustic. EP can also be considered a response to an internal event such as emotion or imagination [3].

EP are then used to measure the electrical activity in certain areas of the brain that are affected by the stimuli. Therefore, the change in response to the external stimuli is small without neglecting the effects of the artifacts and noise on the quality of the signal. To overcome this problem, multiple records of the EEG signal associated with the same task need to be done. The average of the signals is then taken into consideration for the study in order to increase the signal to noise ratio (SNR). Figure 1.5 shows the effect of an example of an external stimuli.

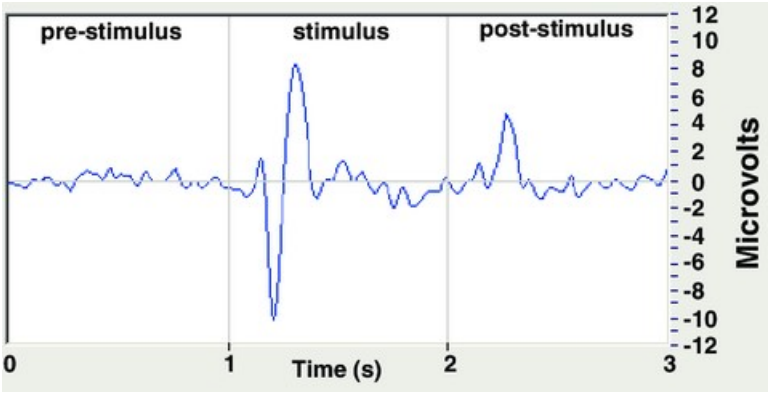


FIGURE 1.5: Example of an Evoked Potential

1.2.4 Electrode Placement Nomenclature

In order to maintain standardized test methods ensuring that the results of a subject's study could be compiled consistently and with reproducible results, an international nomenclature named the 10-20 configuration has been established as shown in Figure 1.6[4].

The 10–20 system is an internationally recognized method for describing and applying the location of electrodes in the context of an EEG examination. The system is based on the relationship between the location of an electrode and the underlying areas of the brain, specifically the cerebral cortex.

The "10" and "20" refer to whether the actual distances between adjacent electrodes are 10 or 20 percent of the total distance between the skull and the fore-aft or between the right and the left. Each site has a letter identifying the lobe and a number identifying the location of the hemisphere.

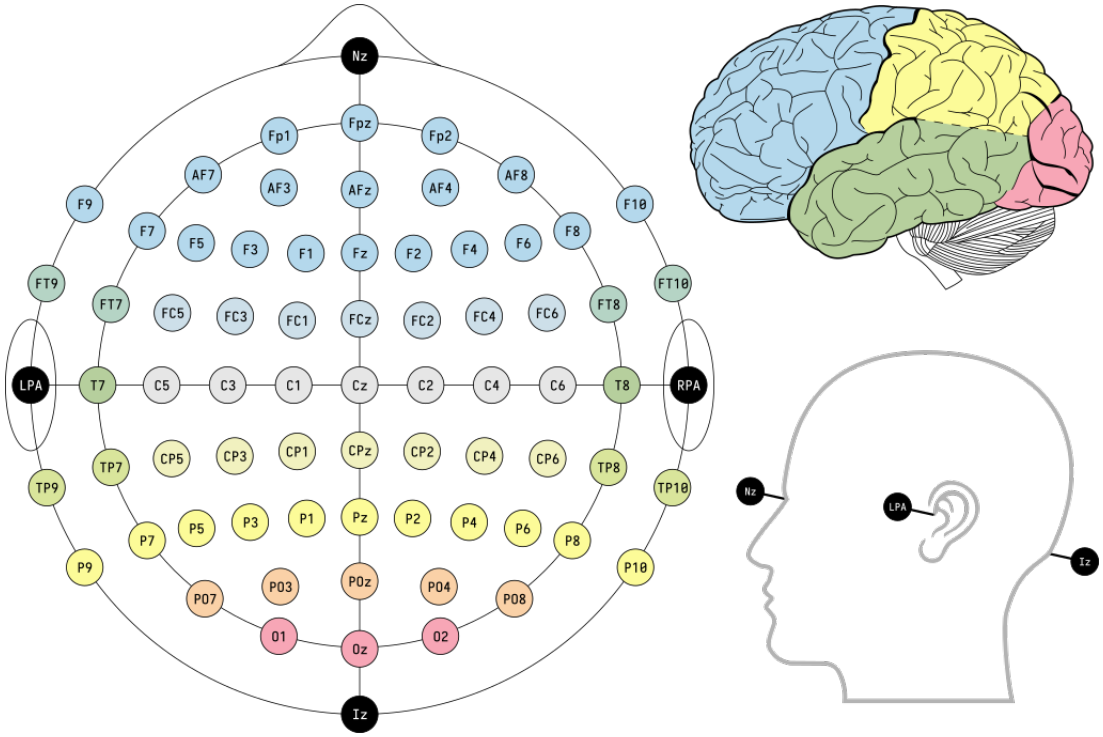


FIGURE 1.6: EEG electrode positions in the 10-20 system along with associated landmarks and lobes of the brain

- **Abbreviations in letters**
  Describe the location of the electrode sites relative to the brain: F = frontal, C = central, T = temporal, P = parietal, O = occipital, z = mid-line (the line that connects the nasion to the inion).
- **Abbreviations in numbers**
  They denote the location of the electrodes according to the hemisphere in which they are located. Odd numbers are used for electrodes located in the left hemisphere and even numbers are used for the right hemisphere.

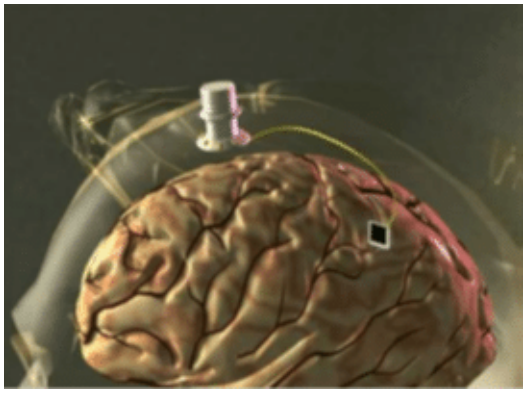
[4]Source: https://en.wikipedia.org/wiki/10–20$_{s}ystem_{(}EEG$)

1.3 EEG Signal Acquisition

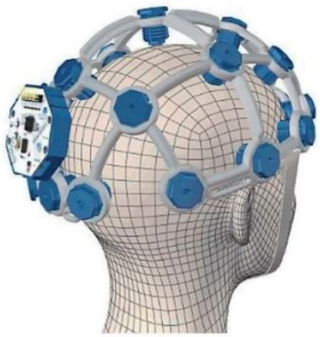
From physical and mental health support to Brain-Computer-Interfaces (BCI). Electroencephalogram (EEG) signals provide a Brain activity observation where the electrical discharges of millions of neurons are recorded with a high temporal resolution and portability advantage, which helps achieve our objective.

Two major BCI approaches exist for the measurement of the EEG signals; invasive and noninvasive BCI. The invasive approach is a complex and a dangerous method for measurement since the electrodes are directly implemented into the brain, this process needs to be implemented and supervised by a neurologist, the neuron's activities can be read without any external interference which detects high-quality signals as can be seen in Figure 1.7a[53] below:

On the other part, in the non-invasive approach, the electrodes are placed on the scalp as shown in Figure1.7b[17] below:



(A) Invasive EEG measurement



(B) Non invasive EEG recording with dry electrodes

FIGURE 1.7: EEG Measurement Types.

This approach can be easy to implement and the signal recording does not need professional assistance, but the resulting signals have more noise contents and they have reduced spatial resolution but due to its easiness, safety and cost effectiveness,it is widely used in the literature[54].

The quality of the EEG signal can be ensured by measuring the contact impedance between the skin (the scalp) and the electrode. An increase in noise and poor signal quality is seen as high contact impedance, as a consequence, a low contact impedance needs to be insured before the signal acquisition.

The impedance can be reduced by using conductive gel paste, but requires a lot of skin preparation .This involves extensive skin wear and dead cells removal from the surface of the skin[54].

The EEG recording for the non-invasive approach can be done by using either wet gel electrodes or dry electrodes, the difference between the wet and dry electrodes is described in the section below:

Wet Electrodes

The main characteristics of wet gel electrodes is described as follows:

- The contact between the electrode and the scalp is assured with electrolyte gel.
- Generally, the electrodes are made of gold, silver or tin.
- Low impedance due to high contact.
- Installation time is quite long (a least 30 min).
- The application of the gel creates a contact surface allowing the electrode to remain connected, which helps the resilience to the movement in order to maintain the signal away from the disturbance during any movement made by the subject.
- On the other hand, a risk of creating an electrolyte gel bridge between two close electrodes or more might occur as a result of this, the obtained measurement will be identical.
- impedance between a wet gel electrode and the scalp need be less than 5kHz.

Dry electrodes

For the dry electrodes , the main characteristic are described as fellow:

- Does not require any gel for the connection.
- The electrodes are generally made of graphite or carbon.
- High impedance between the electrode and the scalp.
- High sensitivity to the movement that might increase the signal to noise ratio (SNR).
- Quick installation time (less than 20min), however , the impedance reduction might require more time.

1.4 EEG Signal Artifacts

As we mentioned in the sections above, the information contained in EEG signals is defined to be at low frequencies. In addition to this, EEG signals are very weak in amplitude therefore they are easily disturbed, and contain artifacts that make EEG processing quite challenging, the two types of artifacts are defined as follows:

1.4.1 Ambient Noise

Ambient noises are external disturbances external to the brain and body. These disturbances are generally created by electromagnetic fields governed by radiation from the sector of other electrical equipment, which may not be superimposed on the useful frequency band of EEG signals and can be easily removed by low pass or filtering. However, in some cases, sub-harmonics might be generated by the sector and seen in the useful

spectrum of the EEG, to overcome this problem, a notch filter can be added or an experimental precaution can be done such as keeping distance from electrical equipment and the sector.

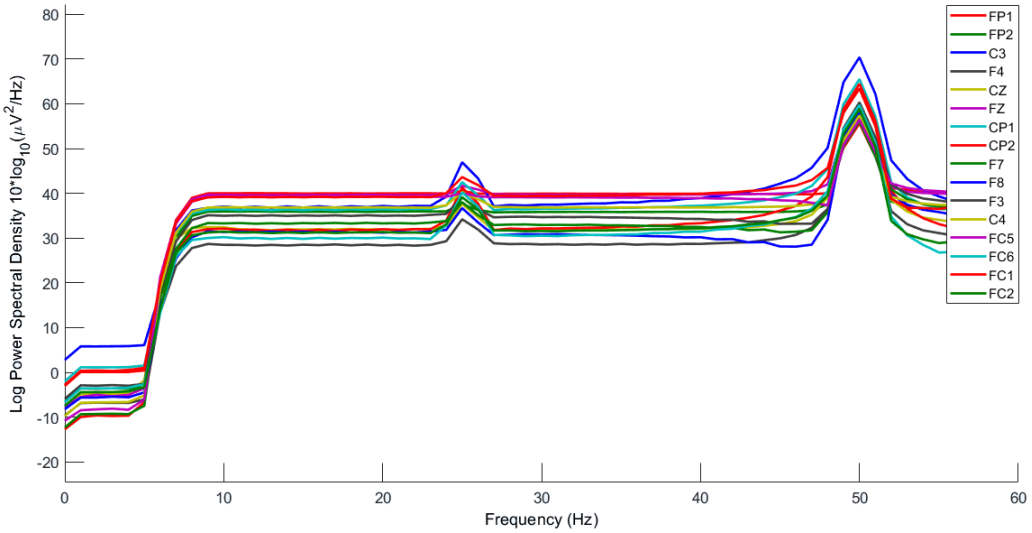


FIGURE 1.8: EEG spectrum with 50Hz power-line noise and 25Hz sub-harmonic

1.4.2 Physiological Noises

As for this type of artifacts, we first look for artifacts that come out of the brain. The brain consists of billions of constantly running neurons, resulting in the electrical activity of different neurons superimposed and experiencing distortion. The original EEG signal contains a measurement of combination of all the sources of the brain.

On the other hand, physiological muscle signals, muscle noise, heart pulse, eye movements, and line noise are very common physiological artifacts. Unlike environmental noise, EOG artifacts appear in the frequency band 0-10 Hz a.d is included in the EEG band. Nevertheless, this artifact can be processed by source separation algorithms.

1.5 Brain Computer Interface

In recent years, researchers were able to develop a wide range of applications with aim to improve life quality for those with muscular or motor-neuron disabilities using Brain Computer Interface (BCI) devices with a wide potential for applicability in fields such as neurorehabilitation [6], neuroprosthetics and gaming [75]. EEG-based BCI provides the opportunity to control external devices in a non-muscular communication using the interneuron brain activity measured by EEG signals with a real time implementation.The major use of BCI can be easily seen in establishing the communication between a muscle or multiple muscles and the environment of people that are suffering from serious motor function problems such Amyotrophic Lateral Sclerosis (ALS), Cerebral Palsy, brain or spinal cord injury [87], or any other struggle or disease that interrupted the transmission of the brain command to the patient's muscles and his external environment.

The electrophysiological signals generated from the brain can be used to control and command real devices such as wheelchairs [88], lower-limb exoskeletons [59]. In [63] and [87], an EEG based brain-machine interfacing is developed for the navigation of the mobile robot "E-PUCK" in four directions. A BCI system can be controlled using EEG signals in different ways such as Motor imagery (MI) where the thoughts of an imagined movement are decoded and then transformed into a real electric command.

(A) Brain Controlled Wheel [59].

(B) BCI for virtual wheelchair control[107].

FIGURE 1.9: Example of BCI for mobile robot control .

1.6 Types of EEG-Based BCI

EEG-based BCI can be classified into two types: Evoked and Spontaneous, also referred to as exogenous and endogenous respectively [75], in the following1.10 diagram the most used BCI in the literature are illustrated:

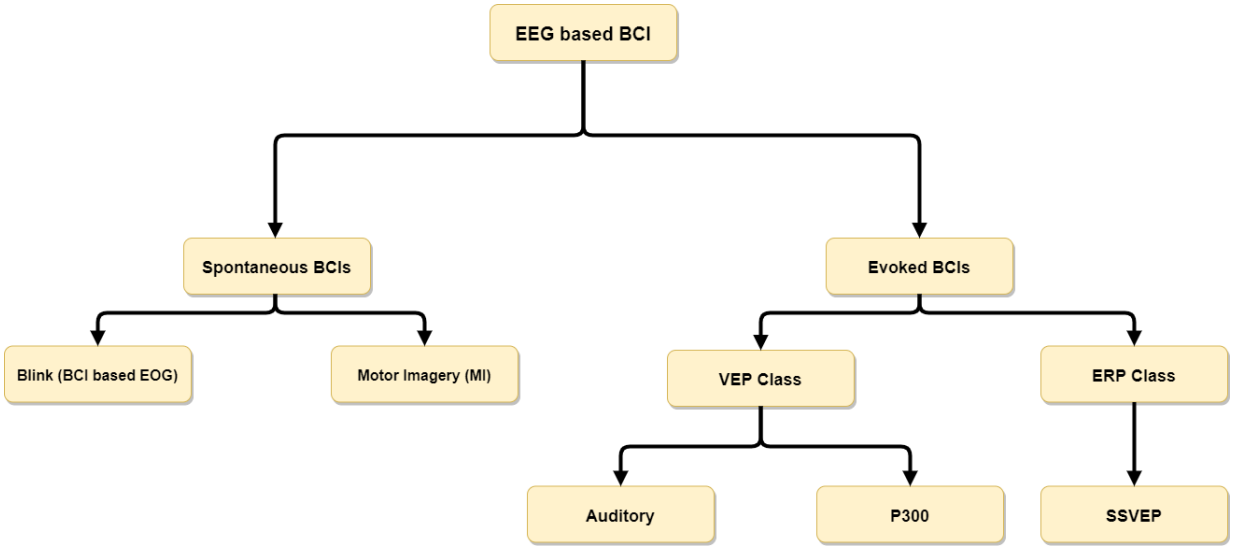


FIGURE 1.10: BCI Types

1.6.1 Sensory Evoked BCI

In Evoked Systems, the intention of the movement is identified using stimuli that evoke responses of an external stimulation such as visual, auditory, or sensory stimulation. EEG evoked systems are also separated into two main categories : EEG systems based on Event-Related Potentials (ERP), and systems based on Visually Evoked Potentials (VEP), where the brain signals are generated in response to a visual stimulus.

Event Related Potentials (ERP)

The Event Related Potential is defined as an electrophysiological response to a stimulus, which can be an external sensory, cognitive, or a motor event and generally uses an oddball-paradigm which consists of a sequence of standard irrelevant stimuli with interspersed rare deviant but relevant stimuli (the oddballs)[56]. One of the advantage of the use of ERP is that it's does not require neurofeedback training since it is elicited autonomously as a response of the brain to external stimulation, when it comes to this type of evoked BCI, the P300 Speller and Dynamic Auditory are the most common.

- **P300 Speller** [5] and in many cases follows an oddball. P300 is characterized by its amplitude, latency, and energy which can reflect the information associated with attention, error awareness and memory performance [56]. The first use of P300 for BCI was in [32],in which they introduced it as a positive deflection that occurs 200–700 ms after stimulus onset which is typically recorded over central–parietal scalp locations.

  The realization of such BCI is demonstrated by the P300 Speller matrix which is presented as a 6 by 6 matrix of characters as shown in the figure1.11[94] below. Many protocols can be considered for the application: like making the user spell the word displayed on the top on the screen [94], or attend to a specific letter shown in parentheses and count how many times the letter flashed.

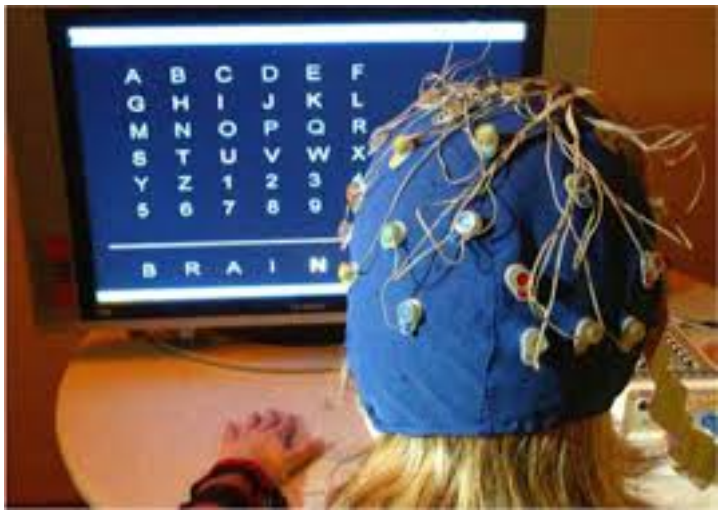


Figure 1.11: P300 BCI

- **Dynamic Auditory Speller**
  Auditory Speller presents another type of BCI based on ERP which applies a spelling interface as external stimuli. In [92] the user was surrounded by six speakers at ear height and faced a screen, during the experiment, the subject's task was to focus their attention to one (target) of the six directions and mentally count the number of appearances. During the calibration phase the target was given to the subject explicitly. During the writing phase the target direction was not given explicitly, but had to be inferred from the spelling interface and the stimulus onset asynchrony was set to 175ms.

  In [19], the auditory stimuli were delivered from five distinct spatial locations using loudspeakers positioned in a row in front of the user's head.

Visually Evoked Potentials (VEP)

Visually Evoked Potentials (VEP), or Visually Evoked Responses (VER) refer to the electrical potentials caused by visual stimuli. Steady-State Visually Evoked Potentials (SSVEPs) are one of the most widely researched areas of VEP based BCIs.

SSVEP is a stable small amplitude signal that appears in the occipital lobe in response to repetitive visual stimulation. The occipital lobe is the center of integration of this visual information. This type of signal has the property of oscillating at the same frequency as that of the stimulation where different options are displayed to the user as stimuli flickering at unique frequencies higher than 6hz, from which the user selects an option by focusing on the associated stimulus, so that the stimulus that the user observed can be identified and trigger the associated command, the performance of such a system depends on the number of stimuli, the modulation schemes, and the hardware used for the stimuli [75][38].

In [48], four different stimuli with individual frequencies (5.45 hz for up, 8.75 hz for down, 12 hz for right and 15 hz for left) are considered for 4 direction movement control as shown in the figure 1.12[48] below:

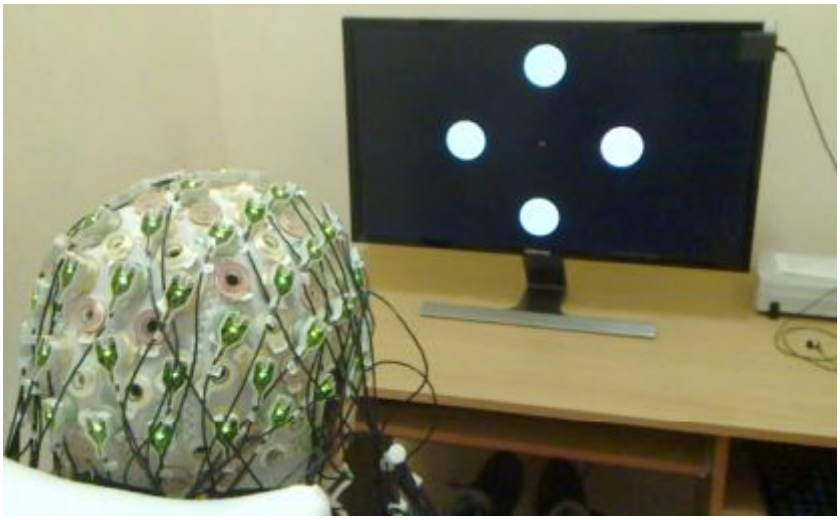


Figure 1.12: SSVEP BCI

1.6.2 Spontaneous BCI

EEG-based Spontaneous BCI is a type of Brain Computer Interface that does not require any external stimulation, the decision of any controlling action is based on activity produced as a result of mental or muscle activity, there are two types of Spontaneous BCI: BCI based on eye blinking (EOG) and BCI based on Motor imagery (MI).

EOG-Based Spontaneous BCI

EOG or Electrooculography is a measurement technique for recording the average of the resting amplitude between the cornea and the retina during the adaptation from light to darkness, and can be measured by analyzing the surrounding muscles of the eye. Eye blinking is typically classified into three categories: spontaneous (or normal) eye movements which occur frequently, a reflexive eye blink which is evoked by an external stimulus, and the voluntary eye blink. EOG-based BCI is a Spontaneous paradigm where eye blinking is analyzed in tandem with EEG signals (to detect eye closure events). In [81], an EOG-Based approach was developed to control a robot arm in four directions (left, right, top, bottom).

In [47], a hybrid BCI was designed for the control of a robotic arm as well as the movement of a wheelchair, the proposed system combined both EEG and EOG for controlling both systems as it can be demonstrated in the Figure2.1 below.

FIGURE 1.13: BCI based EOG

MI-Based Spontaneous BCI

In Motor Imagery (MI), the control action is synthesized based on the user's imagination of a specific movement for a certain task without actually performing it [3], BCI based on MI presents an operational option for patients with lack of independence and communication, this type of BCI is well implemented for patients with locked-in syndrome (LIS) [116], moreover, MI-Based Spontaneous BCI technology is well implemented for non medical domains such as the navigation and control of complex systems (wheelchair [107],virtual robot [87]).

MI-Based Spontaneous BCI rose out of its advanced approach with low requirement advantage and little to no body movement requirement, which made it a widely researched approach [6]. When the imagination of the movement of a limb is used to activate the cortical zone which is dedicated to it [75][69], the responses are called Event Related Desynchronization (ERD). It is also possible to teach the user to control the activation of a particular zone or a specific frequency band, in that case it is called Voluntary Cerebral Modulation, the response to which is called Event Related Synchronization (ERS). ERD is concealed in a form of a power drop in a frequency band and in a certain area, it is the witness of the desynchronization of a set of neurons. Conversely, the ERS results in an increase in power, thus reflecting a synchronization of the oscillatory of a group of neurons [69].

The EEG signals generated by MI tasks are considered as spontaneous type signals since they are generated based on the user's imagination according to a specific protocol. This spontaneity is interesting in that it makes MI technique perfect for supporting healthy and disabled users by streamlining their brains with no requirement of external stimulus signals in which they can potentially harm themselves and/or cause negative effects.

On the other hand, processing high-dimensional MI signals is problematic because of reasons related to: non-linearity, non-stationary, weak SNR, and high complexity of the signals [3].Due to this complexity of the EEG signal, as well as the signal quality and the user's mental state, recording the EEG data for testing and making sure that the data set is "valid" becomes a major challenge.

One of the other challenges that is caused by the MI data is the statistical distribution which is highly variable across subject-to-subject and session-to-session, this variation is due to the existence of alternative physiological and psychological features for each person at each time, as a result, the subsequent BCI system will be subject-dependent study [3].

The common class of movement for MI EEG systems are: left hand movement, right hand movement, the movement of the tongue [75][3] and the movement of the right and the left feet, even though the the movement of the feet is often labeled as a single class, with no distinction between the left and right foot movement because, as mentioned in [75], it is impossible to distinguish between left and right foot motor imagery, or between the movement of particular fingers because the cortex area associated with these different movements is too small to produce distinguishable ERD and ERS signal.

1.7 Fatigue Detection Based on EEG Signals

Whether at work, commuting, or at home, there are many activities in life that require full alertness, continuous attention, and high vigilance in order to be achieved successfully and safely[110]. These tasks, such as driving various vehicles, operation surgeries, and manual labor, when coupled with the lack of alertness or fatigue caused by a number of factors, such as chronic kidney disease, high or low blood sugar or thyroid hormone levels, liver failure, brain infection, disorders or injury, heart or breathing problems or just extended periods of performing this one task can lead to a decrease in the performance and risking damage in property, and even the loss of lives at times[84]. As a consequence, the study and early detection of vigilance is essential in tasks that demand sustained attention over an extended period, including air traffic control, military surveillance, seaboard navigation, industrial process/quality control, medical systems and long-distance driving etc [57].

Fatigue is defined as the transition between waking and sleeping states and it and generally accompanies drowsiness[79]. Further, fatigue accumulates gradually, The effects of fatigue can be resumed in a decrease in awareness and a decline in attention, slowing the reaction time and reducing the person's judgment, senses and performance[7].

Mechanisms that can identify fatigue are necessary to prevent related accidents. The studies [101][43] deployed indicators to measure physiological changes, such as changes in blinking rate and heart rate, as a means of evaluating cognitive capabilities[49].

Multiple studies showed that EEG signals may be a reliable indicator of fatigue where significant changes in EEG signals' power spectrum are associated to variations performance and state of an individual.

EEG spectrum has been found to change with the onset of fatigue. EEG has been widely used to judge individuals' alertness level during monotonous tasks or ones requiring sustained attention. Relative energy of different energy bands (alpha, beta, beta/alpha ratio and (alpha+theta)/beta ratio) has often been used as an indicator of fatigue [31]. The relative energy parameter (alpha+theta)/beta has also been found to proportionally to the alertness level[25].

Karolinska Sleepiness Scale

Measuring the level of fatigue of an operator is a subjective measurement since it is based on the user's own personal estimation which calls for a scale to unify the measurement of drowsiness. The most commonly used fatigue or drowsiness scale is the Karolinska sleepiness scale (KSS) which is a nine-point scale that has verbal anchors for each step as can be seen in the Table 1.1 below. Researchers have determined that major fatigue related physiological signals are prevalent in operators with a KSS rating between 6 and 9.

Rating	Verbal descriptions
1	Extremely alert
2	Very alert
3	Alert
4	Fairly alert
5	Neither alert nor sleepy
6	Some signs of sleepiness
7	Sleepy, but no effort to keep alert
8	Sleepy, some effort to keep alert
9	Very sleepy, great effort to keep alert, fighting sleep

TABLE 1.1: Karolinska Sleepiness Scale.

1.8 Conclusion

Cerebral activity translates the brain's cognitive tasks. There are several alternative measurement techniques of this activitiy, which differ only in the physical principle used for the acquisition. EEG presents an efficient measurement with an advantage of a fast time resolution. These signals have spectral peculiarities including a band extending from [0.5 100] Hz containing several sub-bands δ, α, β and γ . The acquisition of these signals is done with electrodes positioned according to the international 10-20 system and are generally put to the areas responsible for cognitive activity to be studied. The EEG signals recorded by the electrodes contain mainly two types of noise, one created by the equipment and another physiological generated by the various activities inside the body.

In the next chapter, we are going deep into the processing of EEG signals where we derive the multiple methods described in the literature, and their applications for Brain Computer Interfaces' realization in order to control an external device such as mobile robot as well as for drowsiness/fatigue detection.

2 State of The Art on EEG signal Processing

Abstract

In this chapter, we detail the required methods for EEG signals processing for human fatigue detection and Brain Computer Interface design. Section 2.1 develops the different filters used in the literature for the de-noised and source separation of the EEG signals, while Section 2.2 provides a literature review on the feature extraction methods already used for EEG based applications. Section 2.3 discusses the used machine learning algorithms for such applications, as well as the various evaluation metrics. Section 2.4 presents the deep learning (DL) algorithms used for this aim, its architectures and developments. And Section 2.5 describes the Genetic Algorithm used for dimensionality reduction. Finally, Section 2.6 develops on the Genetic Algorithm, its composition, and how it is applied along with DL methods to extract meaningful data from features.

2.1 EEG Signal Pre-Processing

Filtering the signal is considered to be one of the important steps in signal processing. It can be used to eliminate background noise and suppress interference signals by eliminating unwanted frequencies, DC offset and line frequency noise (50Hz or 60Hz depending on the country) are also removed using a specific filter such as a band pass filter. For power line noise, a notch filter is often applied to ensure the removal of the line frequency.

Two types of filtering are considered; frequency filtering the spatial filtering as can be seen in the following 2.1 diagram:

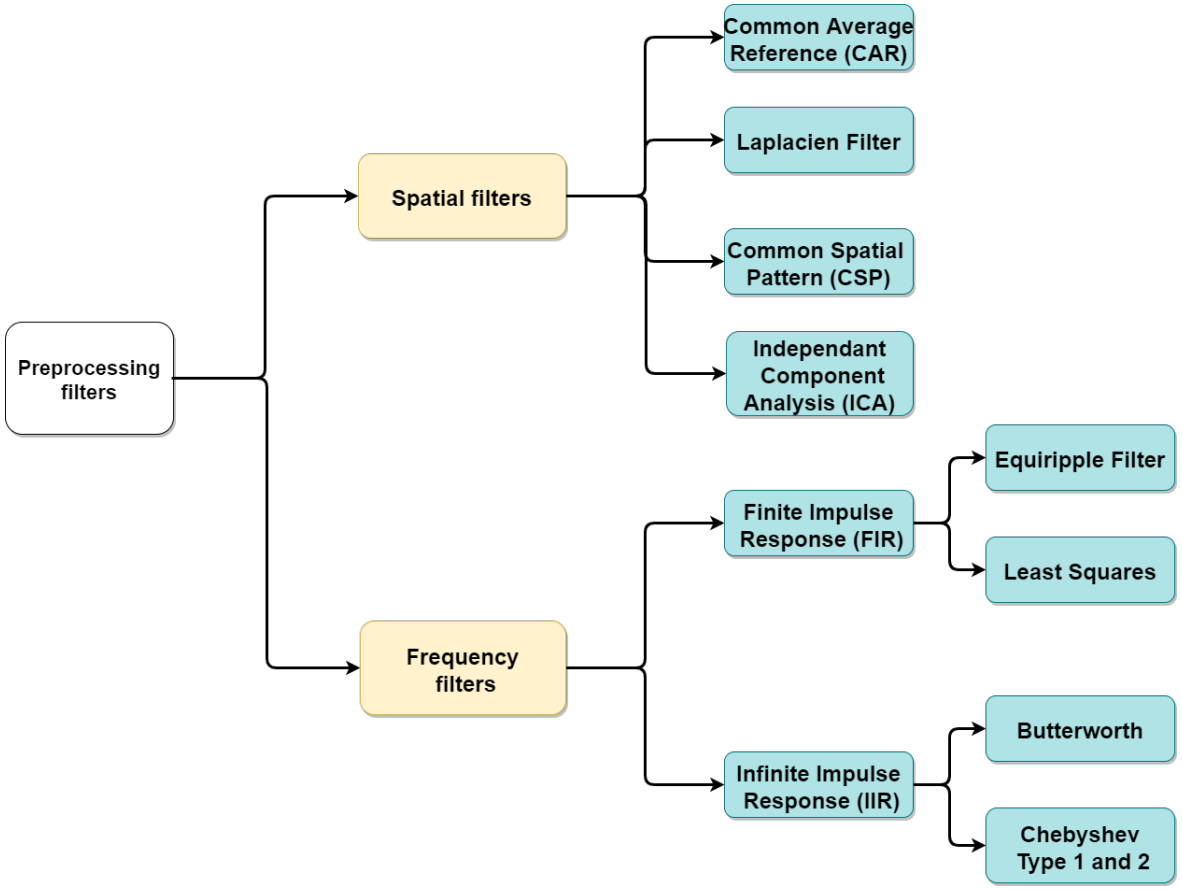


FIGURE 2.1: Pre-processing Diagram.

2.1.1 Frequency Filtering

Butterworth Filter

The Butterworth filter is an active infinite impulse response filter designed to have as flat a frequency response as possible (no fluctuations) in the passband and zero roll-off response in the stopband. It is one of the most commonly used digital filters in motion analysis and audio circuits, it is also well used for EEG processing [75].

The Butterworth filter was first introduced in 1930 by the physicist and British engineer Stephen Butterworth where he described its hardware realization along with other filter types in his book (on the theory of filter amplifiers) [15].

The filter design can be implemented digitally based on matching two methods of Z-transform and bilinear transformation. These two methods can be used to simplify the design of analog filters. If we consider a Butterworth filter with an all-pole filter, the pulse variance and the matched z-transform of the two methods are both considered equivalent.

The transfert function of the butterworth filter is given as:

$$G^2(w) = \left|H(jw)^2\right| = \frac{G_0^2}{1 + (\frac{jw}{jw_c})^{2n}} \tag{2.1}$$

where n is the order of the filter, w_c is the cut-off frequency and G_0 is the DC gain, by increasing the order of the filter, the transfert function gets closer to approximating a rectangular filter as it can be seen in Figure 2.2 below[1].

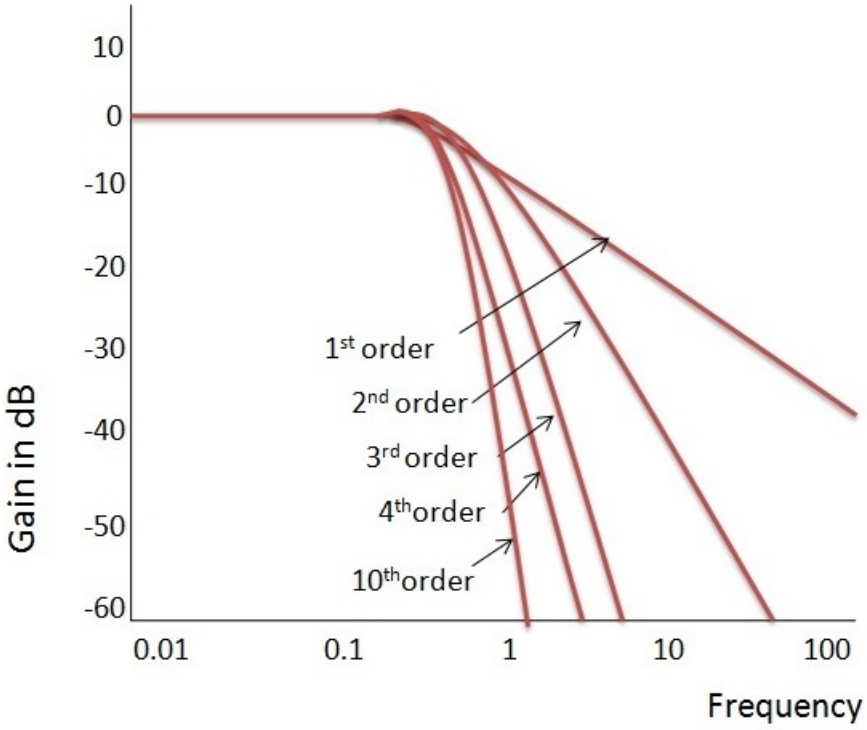


FIGURE 2.2: Low pass butterworth presented in different order.

2.1.2 Spatial Filtering

Spatial filtering has been widely used in EEG-based BCI in order to increase the signal to ratio of the eeg signal, in [70], Independent Component Analysis (ICA) was proposed for artifact rejection, Common Spatial Pattern (CSP) which is generally used for feature extraction is also considered as a spatial filtering method.

Independent Component Analysis (ICA)

Independent Component Analysis allows the estimation of hidden sources, and it is considered a powerful method with widespread use in the fields of source separation and artifact rejection.

On the surface, ICA is related to Principal Component Analysis (PCA), PCA is used to compress information and reduce the dimension of the input data while ICA aims to separate information by transforming the input space into a maximally independent basis as it can be seen in the figure 2.3.

[1]Source: https://www.electronicshub.org/butterworth-filter

This method assumes that the recorded signals are a composition for a non-Gaussian independent signal denoted S, the signal vector X can be written as follow:

$$X = W.S \tag{2.2}$$

where W is the mixing weights matrix. There are several variants of ICA, they differ by the criterion used to calculate the matrix W. The most used criteria are the minimization of mutual information between non-Gaussians sources, the maximization of entropy and the maximization of Kurtosis.

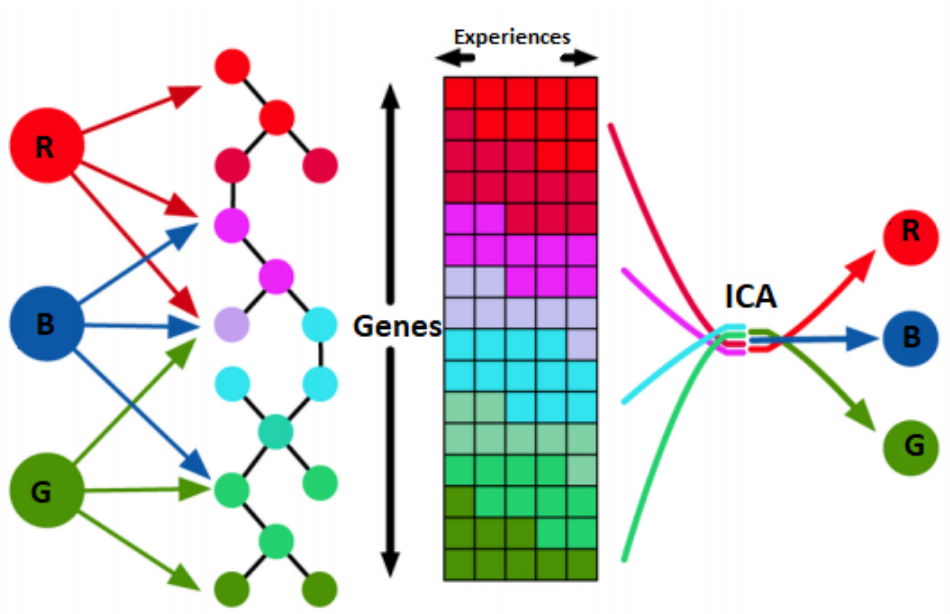


FIGURE 2.3: ICA source separation in case of 3 colour mixture.

2.2 Feature Extraction

Feature extraction represents a major step for motor imagery classification, several categories of characteristics can be extracted from EEG signals:

- **Time Domain Characteristics**
- **Frequency Domain Characteristics**
- **Time-Frequency Domain Characteristics**
- **Spatial Characteristics**

The diagram 2.4 below demonstrates the most common characteristics used in the literature:

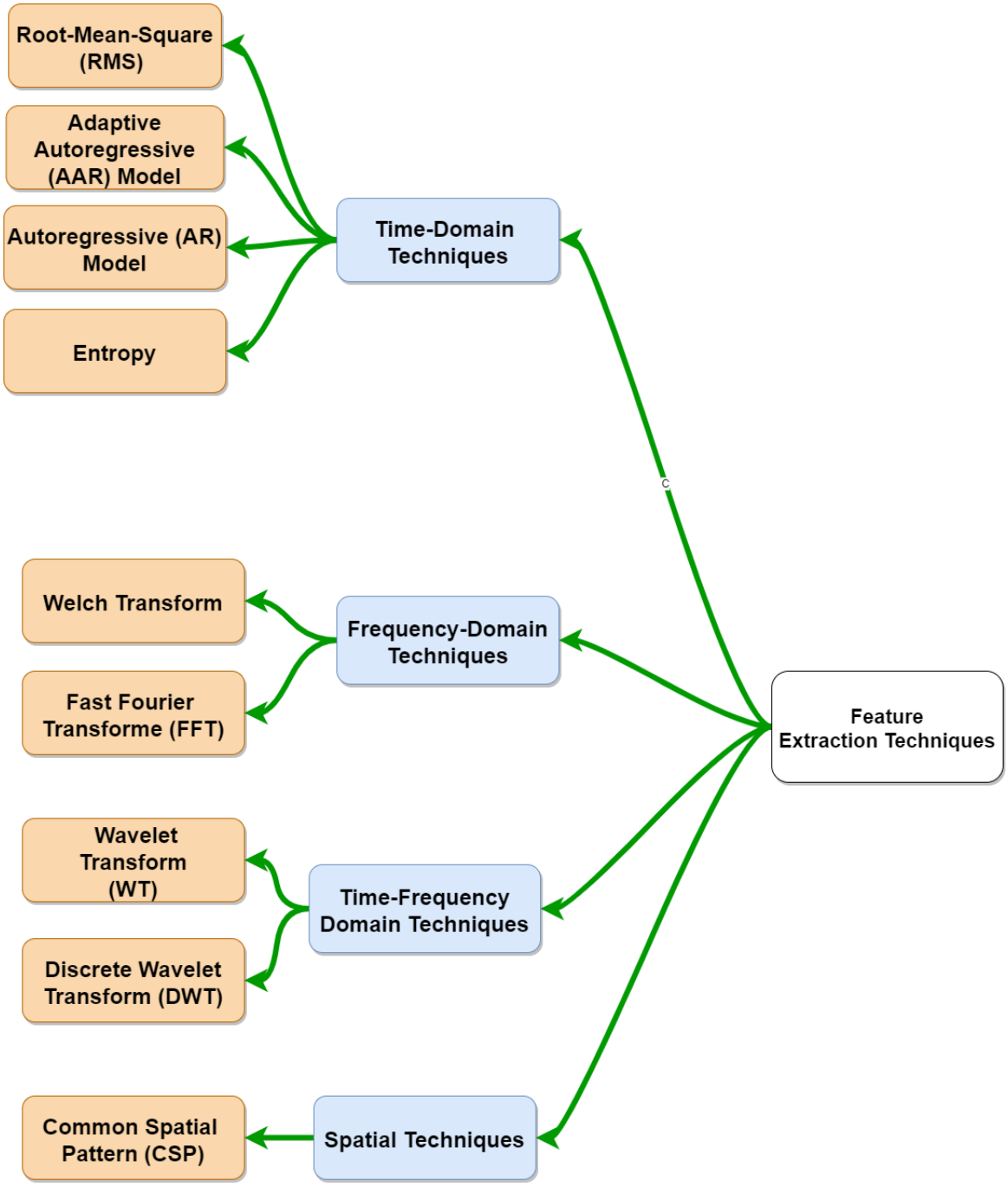


FIGURE 2.4: Feature Extraction.

In this section, we will describe the characteristics used in our work. In [21][30][102], a Common Spatial Pattern method is used for feature extraction, in [64], Power Spectral Density was used as features and input to an RNN network.

In [89], PSD and entropy were used separately as characteristics of the model and considered as an input to an SVM classifier.

2.2.1 Power Spectral Density (PSD)

Fast Fourier Transform (FFT)

Fast Fourier Transform (FFT) is considered as a common technique used for EEG data manipulation. The method's process consists of computing either the discrete Fourier Transform (DFT) or the Continuous Fourier Transform (CFT) for the first step and then converting the signal from the time domain to the frequency domain by applying Fourier analysis.

DFT assumes that a signal is repeated over a certain period of time and then converts a finite sequence of equally-spaced samples of the signal into a same-length sequence of equally-spaced samples of the Discrete-Time Fourier transform (DTFT), which is a complex-valued function of frequency.

DFT can be written as follows:

$$x\left[k\right] = \sum_{n=0}^{N-1} x\left[n\right] e^{\frac{-2j\pi kn}{N}} \tag{2.3}$$

To determine the DFT of a discrete signal $x\left[n\right]$ (where N is the size of its domain), we multiply each of its values by e raised to some function of n. We then sum the results obtained for a given n. If we used a computer to calculate the Discrete Fourier Transform of a signal, it would need to perform N (multiplications) x N (additions) = $O(N$š) operations.

As the name implies, FFT is an algorithm that determines the DFT of an input significantly faster than computing it directly. In computer science lingo, FFT reduces the number of computations needed for a problem of size N from O(N2) to O(NlogN).

CFT on the other hand assumes that the signal does not have a period of repetition which means that the signal is extended to infinity, it is considered the simplest method of time-frequency transforms for the infinite functions and signals.

The CFT of a signal can be computed as follows:

$$S(f) = \int_{-\infty}^{\infty} x(t) e^{-jw_0 t} dt \tag{2.4}$$

Welch PSD

The Welch method, also called the periodogram method, was firstly proposed by Peter D. Welch [89], and is a method of Spectral Density estimation for both stationary and non-stationary signals. It is widely used in applied mathematics and engineering with a main purpose of estimating the power of signals at different frequencies.

The Welch method is considered an improvement to the standard periodogram estimation with a concept of using periodogram spectrum estimation, which is the result of converting the signal from the time domain to the frequency domain. The method allows reduction in the power spectrum estimation in exchange for reducing the frequency resolution. Due to the noise caused by imperfect and limited data, this method is usually needed to reduce the noise which can be achieved by filtering the data before applying this method.[89]

Welch's method splits the data into overlapping segments where the data segment is divided into small segments L with a length of M each with an overlap of a D points, each segment is than windowed using the hamming window (Figure 2.5 , after that the modified periodogram for each segment is calculated, and then averages the periodogram to calculate the estimated value of the PSD with a motive of reducing the variance that is associated with the periodogram estimate of the power spectrum, this variance is proportional to the value of the spectrum itself so as the number of data increases, it does not decrease and in this approach we can take advantage of averaging independent estimates to reduce it.

Welch's PSD estimation is computed as follows:

$$S(w) = \frac{1}{L * \sum_{m=1}^{M} |h(m)|^2} \sum_{l=1}^{L} \left| \sum_{m=1}^{m} h(m)x(m + (l-1)D)exp(-jwm) \right|^2 \tag{2.5}$$

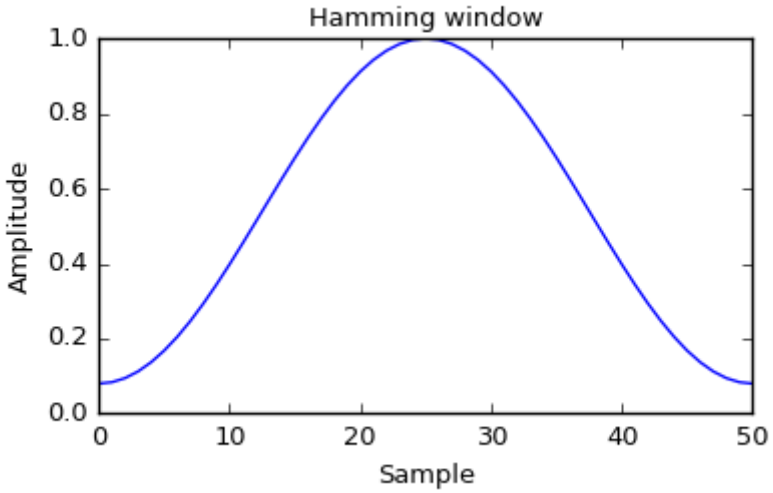


FIGURE 2.5: Hamming window.

2.2.2 Entropy

In 1945, Shannon introduced the notion of Entropy associated with a source which is modeled by a discrete random variable X, as the mean of the amount of information provided by the realizations of this variable. Since that date, this notion has had very great use in the field of information processing and particularly in data encoding and compression in telecommunications.

The Entropy associated with a discrete scalar random variable X with realizations $\{x_1 \cdots x_N\}$ and the probability distribution $\{p_1 \cdots p_N\}$ measures is defined by:

$$H[x] = -\sum_{i=1}^{n} p(i)ln(p_(i))) \tag{2.6}$$

2.2.3 Common Spatial Pattern (CSP)

Common Spatial Pattern or CSP is a method of projecting multi-channel EEG data in a subspace of reduced spatial dimension using a projection matrix W. Each row of the projection matrix W is a spatial filter. CSP in its basic version only uses two classes. It maximizes the variance between them and exploits the covariance matrix for the design of the CSP.

The projection matrix is calculated based on the simultaneous diagonalization of the two mean covariance matrices of each class. three approaches can be distinguished to obtain the projection matrix: geometric approach, by optimization or by solving an eigenvalue problem generalized.

Multiple extensions of CSP exist for multi-class projection. In [113] One-Versus-The-Rest CSP extension is proposed for feature extraction.

A detailed analysis of CSP and OVR CSP is in Appendix A.

2.3 Machine Learning Methods

Machine Learning (ML), a subfield of Artificial Intelligence, has revolutionized the biomedical research field offering tools to address the high complexity of EEG signals and further decode the information they have to offer.

The Figure 2.6 below shows the common ML methods used for EEG classification, they are divided based on their training algorithm, as well as on the way it is applied, either individually or in an Ensemble.

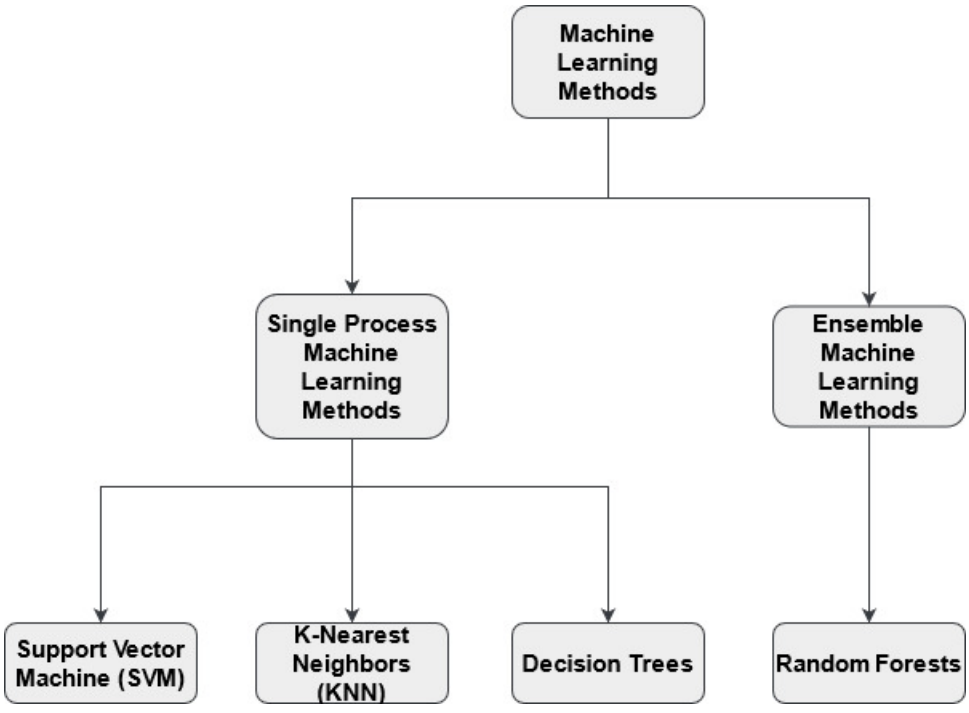


FIGURE 2.6: A diagram of the used Machine Learning methods.

2.3.1 Machine Learning Metrics

In order to evaluate a ML model's performance, various metrics are used depending on the application and the desired result. In order to see the whole picture and be able to fully evaluate a model, using many metrics is recommended, as each covers only part of the result.

Confusion Matrix

Not a metric but one of the key concepts in classification performance evaluation, the confusion matrix is a tabular visualization of the model's predictions against the actual data labels. Every row in the confusion matrix represents the model's number of predictions for each class respectively while the columns represent the classes' actual values. This matrix helps understand if and how the classification model is confused when making predictions. This not only allows us to know what mistakes were made, but especially the type of mistakes that were made[66], an example of it is shown in the figure below:

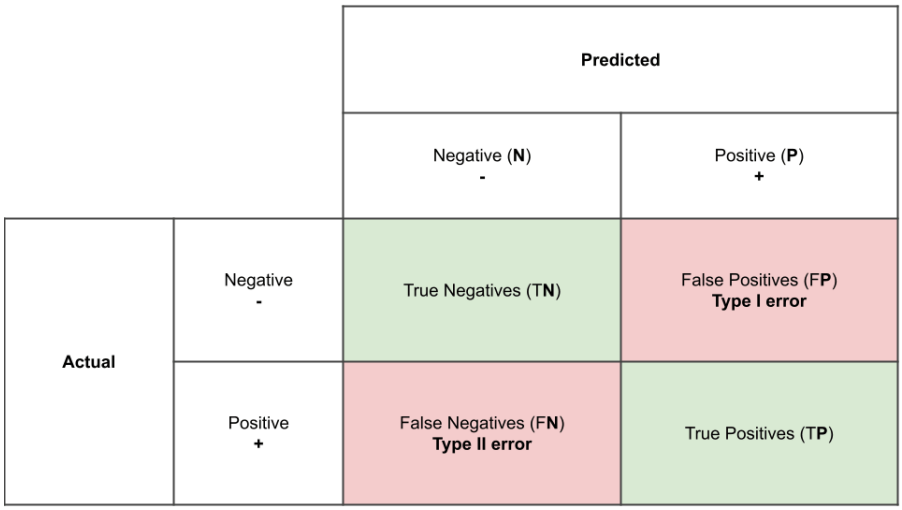


FIGURE 2.7: Confusion Matrix for 2 categorical classification.

When training a model for a classification problem, it is important to understand the concepts of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN), in order to understand its classification metrics. After the model is successfully trained, the correct and incorrect results are broken down by class and compared with the real values to obtain these four terms[66], which are defined by:

- **True Positives (TP)** represent the cases where the model's prediction is positive, and the real value is indeed positive.
- **True Negatives (TN)** represent cases where the model's prediction is negative, and the real value is indeed negative.
- **False Positives (FP)** represent cases where the prediction is positive, but the real value is negative.

- **False Negatives (FN)** represent cases where the prediction is negative, but the real value is positive[66].

Classification Metrics

- **Classification Accuracy**
  One of the simplest ML metrics, it is obtained by dividing the number of the ML model's correct predictions by the total number of predictions, and multiplied by 100 to obtain a percentage.

$$Accuracy = \frac{Correct\ Predictions}{Total\ Number\ of\ Predictions} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.7}$$

  Although Accuracy gives a general idea of the model's performance, there are many cases in which it does not convey the necessary information or throws the user off course, one such case is when one of the classes is predominant over the others, thus, it is a valid evaluation metric on its own only if there is no class imbalance[66]. Therefore, evaluating class specific classification metrics is necessary too.

- **Precision**
  Precision is defined as:

$$Precision = \frac{TP}{TP + FP} \tag{2.8}$$

  Precision is a valid choice of evaluation when we want to be very sure of our prediction, and when False Positives are unwanted in the end result[66].

- **Recall**
  Recall is defined as:

$$Recall = \frac{TP}{TP + FN} \tag{2.9}$$

  Recall is a valid choice of evaluation when we want to capture as many positives as possible[66].

- **F1 Score**
  F1 Score is the harmonic mean of precision and recall, and it is defined as:

$$F_1 = 2.\frac{Precision.Recall}{Precision + Recall} \tag{2.10}$$

- **ROC Curve**

  The Receiver Operating Characteristic curve is a plot of the True Positive rate (TPR) against the False Positive rate (FPR), it indicates how well the model is separating the positive classes from the negative classes. The ROC Curve is a

useful metric when deciding on a classification threshold because it is Classification-Threshold-Invariant unlike the other metrics (Accuracy, F1 Score, ...), the classification threshold simply decides which classification results get classified as positive class or negative class, which, depending on the classifier's application, is not always 0.5[66].

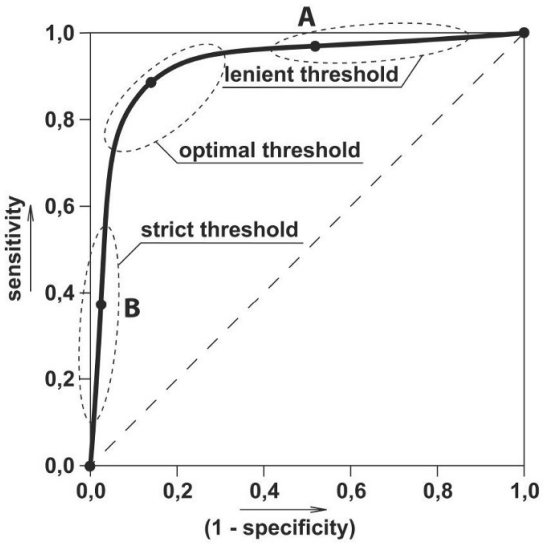


FIGURE 2.8: ROC Curve and the different types of classification thresholds.

2.3.2 Underfitting and Overfitting

One of the main problems encountered in the ML process are Underfitting and Overfitting, which are majorly responsible for poor ML models' performance [52]. In order to fully understand these terms, a few concepts must be clarified:

- **Generalization in Machine Learning** refers to how well the model adapts to never before seen data.
- **Bias** is assumptions made by the ML model to better adapt it to the data, that can possibly make it prejudiced towards one class or another.
- **Variance** is the amout of training error change we get from changing the training data.
- **Good Fit** is said when the ML model is well trained on the data, and making little to no error in its predictions, this is the desired result when training ML models, and it is right between Underfitting and Overfitting[52].

The concepts of Underfitting and Overfitting are highly linked to these four concepts, they are defined by:

- **Underfitting**
  A ML model is said to have Underfitting when it cannot capture the full pattern of the data, thus not being trained enough to be of use. Underfitting lowers the

model's accuracy significantly, two common causes for it is not having enough data to build an accurate model, or when a linear model's trained on non-linear data. Underfitting is characterized by high Bias and low Variance.

In order to avoid Underfitting, one can simply increase the model's complexity, its number of features, clean the data, or increase the number of epochs/training duration[52].

- **Overfitting**
  Conversely, a model's said to be Overfit, when it is trained on too much data, thus training it on the noise and inaccuracies in the dataset. Its causes include using non-parametric and non-linear methods when they not needed, giving the model more freedom and causing the end result to be an unrealistic model, or giving the model a long training duration (high number of epochs). Overfitting is characterized by low Bias and high Variance.

  Overfitting's soluions include increasing the training data, reducing the model's complexity, using early stopping (stopping the training right before the model starts overfitting on data), Ridge/Lasso Regularization, and using Dropout in the case of Neural Networks (explained more in Section 2.4.6)[52].

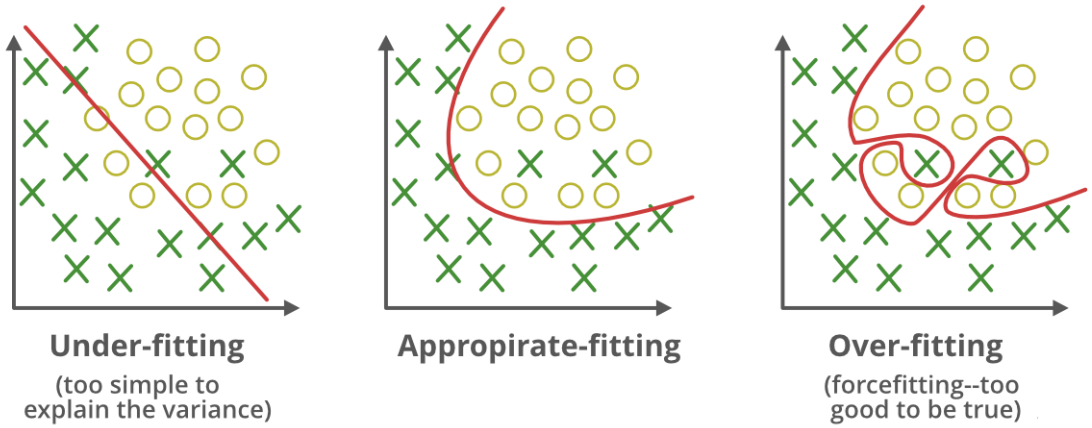


FIGURE 2.9: A comparison between an appropriately fit model, an Underfit model, and an Overfit one.

2.3.3 Cross-Validation

In order to better estimate the ML model's accuracy and avoid overfitting, Cross-Validation is used, especially in cases where data may be limited or particularly biased. Cross-Validation is done by dividing the dataset into a fixed number of partitions, running the process on each of them, and then averaging the overall accuracy or error estimation.

Cross-Validation techniques can be classified into two categories, Exhaustive and Non-Exhaustive methods, two examples of Non-Exhaustive ones ones that are particularily useful are:

- **Holdout Cross-Validation**
  Holdout Cross-Validation is the simplest of all the methods, the dataset is divided into training and testing parts only once, with a high ratio of training data to testing data, usually around 70:30 or 80:20, but can be different depending on the problem at hand.

  This method is done by first randomly shuffling the data before splitting it, which, depending on the nature of the dataset, can give different results every instance that it is done, this makes the method inherently unstable[99][22].

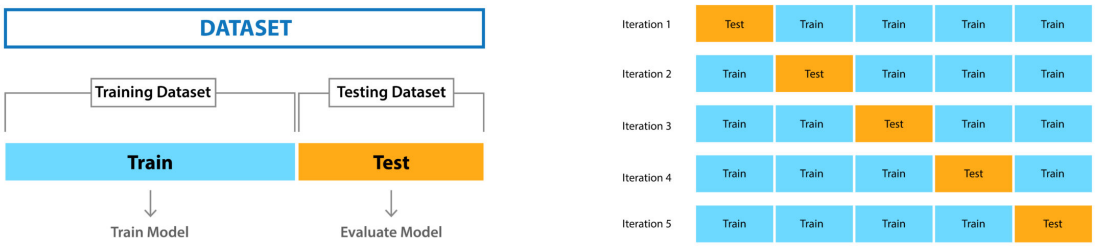
- **K-Fold Cross-Validation**

  This method further improves upon the Holdout method, and guarantees that the model's evaluation is not dependent on the way it was shuffled and split[99].

  This method is done by randomly splitting the dataset into k folds (k being a parameter for tuning based on the data), and for each fold, the model is train on the other k-1 folds and its accuracy is tested, this process is repeated for every fold and then the testing accuracies are averaged which results in the K-Fold Cross-Validation accuracy that serves as the evaluation metric for the classifier. The Holdout method, can be considered as one particular case of K-Fold Cross-Validation that has K set to 1[99][22].

  The drawback of this method is that it is time consuming, especially for higher values of k, because the model has to be trained k times from scratch, which means it the method needs k times as much time as the Holdout method to calculate the model's accuracy[99].

Both Cross-Validation methods are illustrated in the figures[2] below:



(A) Holdout Cross-Validation. (B) K-Fold Cross-Validation.

FIGURE 2.10: An illustration of both Holdout and K-Fold Cross-Validation.

2.3.4 Support Vector Machine (SVM)

Support Vector Machines is a supervised Machine Learning model mainly used for classification, most commonly used with data pertaining to 2 categories and especially effective with high dimensional data.

[2]Source: https://www.mygreatlearning.com/blog/cross-validation/sh1

Its algorithms aim to construct a hyperplane that acts as a boundary for classifying data points by separating them in two and then seeing what side of the boundary an unclassified point falls on, this hyperplane is represented by a function called a kernel, which can either be linear, polynomial of any order, or a radial basis function (Gaussian)[95][8].

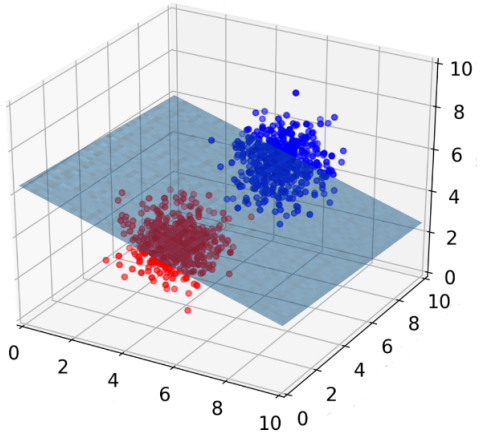


FIGURE 2.11: Linear SVM classifying three dimensional data into two categories.

Given the infinite number of possible defining boundaries or hyperplanes dividing the data, the algorithms construct it in a way such that the distance between it and the points in each class is maximal, in order to decrease the chance of false classification. All by using support vectors, which are virtual vectors going from the origin to every data point, and the distance between the support vectors and the decision boundary is called the margin. The closest support vectors to the margin are crucial in defining the hyperplane, in fact, the SVM algorithm can construct the hyperplane entirely without referring to the non-marginal support vectors, which is one of the algorithm's advantages, making it faster than the other algorithms that have to use all training points to make a prediction[95][8].

2.3.5 K-Nearest Neighbors (KNN)

Being one of the simplest Machine Learning algorithms to implement, the K-Nearest Neighbors algorithm predicts the label of new data instances based on the labels of its closest training set neighbors. This is done by normalizing the numeric data, finding the distance between these instances and all training data points, sorting the distances and fining the nearest K points, and finally classifying the unknown data points based on the most instances of the nearest K points[95]. This process differs slightly in the case of a regression. The figure below shows an example of KNN, where depending on which circle is used (representing k=3 or k=5 respectively), the unlabeled data point can either be classified as red or blue:

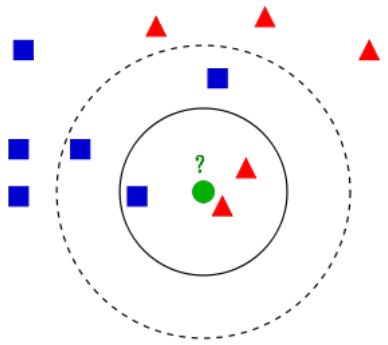


FIGURE 2.12: Example of KNN classification.

The idea behind this algorithm is based on the assumption that close-by points are likely to be more relevant making their labels similar. One of this algorithm's advantages is that depending on the training set's nature, finding neighbors can be an extremely fast process [95]. This algorithm still needs more computational time on average than the other algorithms [104].

The only parameter in this algorithm is K: an integer representing the number of neighbors. Increasing K decreases accuracy and leads to underfitting, while decreasing it causes overfitting, it must be fine-tuned with a trial and error process.

Data normalization is integral when applying KNN, due to the distances in it being Euclidean, which means the algorithm's efficacy can decrease in the case of heterogeneous training data [80], in the case of noise, irrelevant features, or inconsistently scaled features [71]. Many techniques for normalization exist[42], among them are Min-Max Normalization and Z-Score Normalization.

2.3.6 Decision Trees

A decision tree is a classification method that predicts the label associated with an instance by representing the differentiating criteria of the data features as branches emanating from a root node to other nodes, which eventually end in leaves representing the class labels. Binary decision trees are most commonly used although they're applicable to other problems[95]. A binary decision tree is a structure based on a sequential decision process. Starting from the root, features are evaluated and one of the branches is selected. This procedure gets repeated until a final leaf is reached, which represents one of the classification targets we're looking for. In comparison to other algorithms, decision trees are simpler; however, if the dataset can be split while keeping an internal balance, the process is intuitive and fast in its predictions. Moreover, decision trees can work with unnormalized datasets because their internal structure is not influenced by the values assumed by each feature[8].

When making decision trees, two different methods are used to find the best feature to split a dataset on, Gini Impurity and Information Gain:

- **Gini Impurity.**
  Which is defined as:

$$I_{Gini}(j) = \sum_i p(i|j).(1 - p(i|j)) \tag{2.11}$$

- **Information Gain**
  This theory uses the Cross-entropy impurity index which is very similar to the Gini Impurity, it is defined as:

$$I_{Cross-Entropy}(j) = -\sum_i p(i|j).logp(i|j) \tag{2.12}$$

However, without proper limitations, a decision tree could potentially grow until a single sample (or a very low number) is present in every node. This situation drives to overfit the model, and the tree becomes unable to generalize correctly[8].

2.3.7 Random Forests

All the aforementioned methods' algorithms train models on single instances, aiming to minimize a certain loss function and looking for the best possible solution. An alternate approach employs a set of weak algorithms and runs them in parallel or sequentially, the results can then be used as an ensemble (thus the nomenclature), based on a majority vote or the averaging of results[8].

Random Forests is one such ensemble method that uses an Ensemble method called Bootstrap Aggregating (Bagging). By generating multiple versions of Decision Trees and using them to get an aggregated predictor. The resulting versions do a majority vote when predicting new data. The "Bootstrap Aggregating" name comes from the process done, the training set is replicated resulting in bootstrap versions that are used as new learning sets. It can increase accuracy in many cases, especially when changing the training set causes significant changes in the resulting model[10]. This method is usually applied to decision trees, although it can be used with other types of methods[8].

2.4 Deep Learning Methods

Just like Machine Learning is a subfield of Artificial Intelligence, Deep Learning is a subfield of Machine Learning. The "Deep" in the name is a reference to the existence of multiple layers forming DL models, that ranging from a few to hundreds. The efficiency of Deep Learning shows when using a sufficiently large dataset/computational power, which makes them perfect for EEG signals, with the existence of the OpenBCI Mark IV that can supply as much data as needed[2].

2.4.1 Artificial Neural Networks (ANN)

Deep Learning models are represented by what is called Artificial Neural Networks (ANN), the name is inspired by Neurobiology, because of its resemblance to the real biological neural networks contained in the brain[2].

Artificial Neural Networks, or simply Neural Networks, are formed by data transforming layers, represented by weights, stacked on top of each other. And in order to train them, four things are needed:

- **Layers** constituting the Network, this means the number of layers, the type of layers, their activation functions as well as the Network's input and output dimensions and type.
- **Input Data**, a database used for training and/or evaluating the model using Cross-Validation.
- **Loss Function**, which determines how well the Network's learning.
- **Optimizer**, which determines how learning proceeds based on the Loss Function[22].

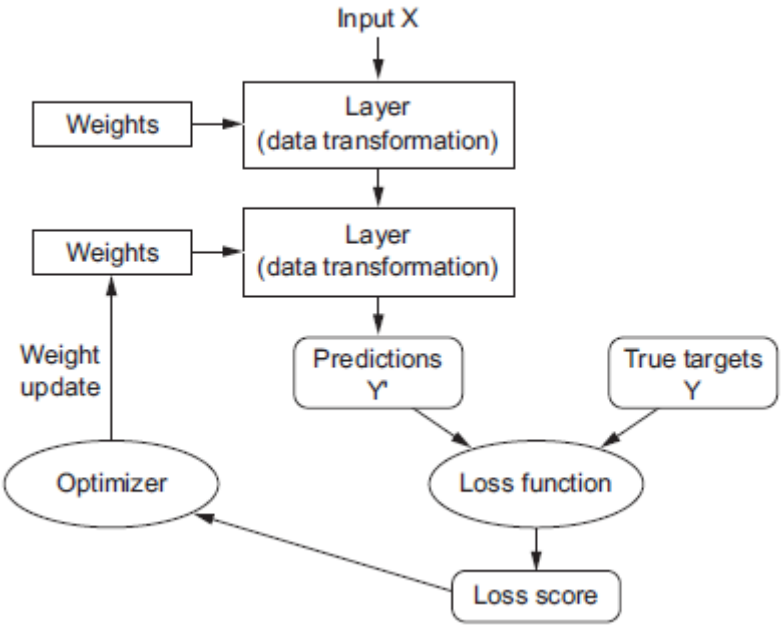


FIGURE 2.13: Anatomy of the Artificial Neural Network.

The figure above shows the structure of an Artificial Neural Network's learning process: Data inputs are given to the Network, they are passed through its layers (weight matrices), giving the Network's predictions, the Loss Function then measures the discrepancy between the predictions and the true values, thus measuring how good the results are, and based on that, the Optimizer updates the weights inching them closer to their optimal value with every training epoch. This process is repeated until the specified number of epochs has passed or the stopping condition has been fulfilled[22].

Artificial Neurons

The primary building block of Neural Networks, not all Neurons come in this exact structure but it is the most common. Its architecture is simple, as shown in the figure below, it

takes in a number of inputs $(x_1, x_2, ..., x_i)$ and multiplies each by their respective weights $(w_1, w_2, ..., w_i)$, a bias term b is often added to the result as an extra weight that shifts the weighted sum's value with a constant to better fit the data if need be. This weighted sum is then passed through a non-linearity called the Activation Function, and the result is outputted from the Neuron[2].

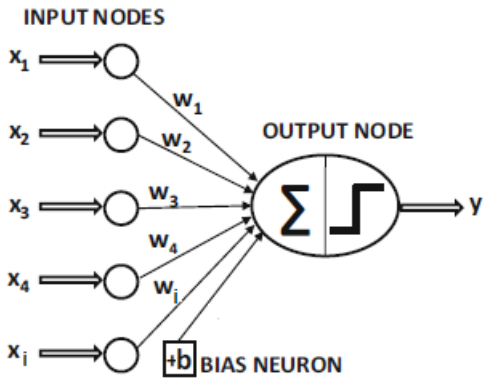


FIGURE 2.14: Structure of a single Artificial Neuron.

This process can be resumed with the equations:

$$\begin{cases} z &= \sum_{i=1}^{n} w_i x_i + b &= W^T X + b \\ y &= f(z) \end{cases} \tag{2.13}$$

Such that X are the inputs, y is the output, W is the Weight matrix, b is the bias and f is the Activation Function.

Activation Function

The Neuron's Activation Function is a non-linearity that is applied to the weighted sum that fires off into the next layer, deciding what is to be transferred. It is used mainly to allow the Neural Network to learn more complex patterns in the data, because without it the output is just a linear weighted sum that can only mimic simple patterns. There are many Activation Functions, and their choice is a critical part of designing a Neural Network, it depends on the Network's desired function and the type of data used[2].

Some examples of Activation Functions are illustrated in Figure 2.15 below:

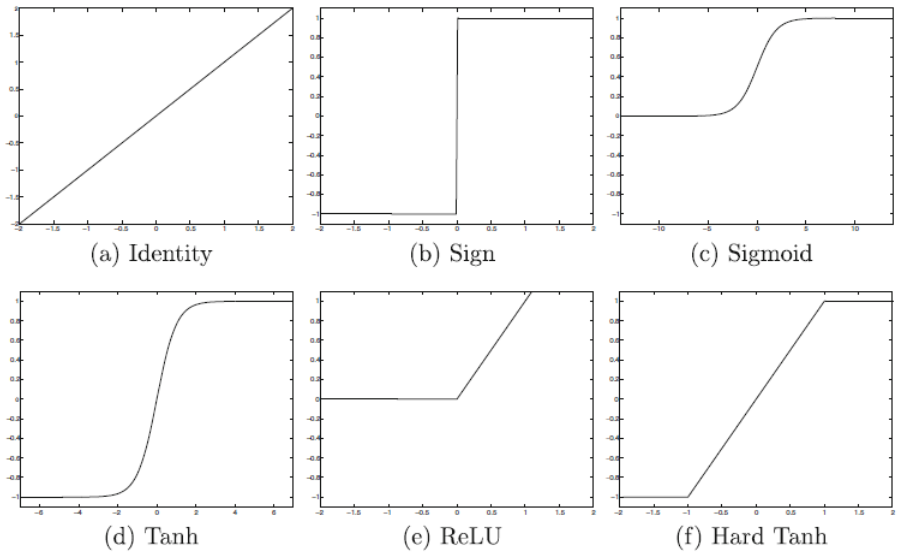


FIGURE 2.15: Some of the most commonly used Activation Functions.

Activation functions are deliberately used depending on the task, and they can be classified into Hidden Layer Activation Functions and Output Layer Activation Functions. Both Feedforward Neural Networks (2.4.2) and Convolutional Neural Networks (2.4.5) often employ ReLU Activation Functions in their Hidden Layers, while Recurrent Neural Networks (2.4.3) can have either a Sigmoid or Tanh Activation Function[14]. The Output Layer's Activation Function depends on the problem, Regression problems use linear Activation Functions, while Classification problems use Softmax or Sigmoid Activation Functions.

Layers

The layer is a processing module that contains the accumulated knowledge it has learned in the form of Weights, that are one or several matrices learned using Stochastic Gradient Descent. It takes in data and transforms it using them, and outputs the result[22].

There are many types of layers, depending on the corresponding data structure and data processing needed. Simple data instances, stored in 2D tensors (vectors), are processed by Fully Connected Layers, Sequence data (time series), stored in 3D tensors, are processed by Recurrent Layer (An LSTM Layer for example), and image data, stored in 4D tensors, is processed by 2D convolution layers, as well as other image processing layers[22] which will be discussed in later sections.

Fully Connected Layers process the simplest kind of data samples, in them, every input to the layer's neurons is connected to every activation function passing to the next layer. Their data transformation process can be represented using the equations:

$$Z = W^T X + B \tag{2.14}$$
$$Y = f(Z) \tag{2.15}$$

Such that X are the inputs, Y are the outputs, W and B are the Weight and Bias matrices and f is layer of Activation Functions.

Neural Networks composed only of these layers are dubbed Fully Connected Neural Networks.

Loss Function

The Loss Function, also called the Objective/Cost Function, is the quantity that will be minimized when training the Neural Network's weight matrixes. It represents a measure of success for the objective at hand.

Choosing the right Loss Function is paramount for having optimal weights and good model performance, the network's weights will be updating based on this function, which means that any inconsistency between the current objective and the Cost Function will result in the model doing unwanted things, because of the shortcuts it tends to take when minimizing the loss[22].

Examples for common Loss Functions for known problems include:

- **Binary Cross-Entropy** for binary classification problems:

$$BCE = \frac{1}{n}\sum_{i=1}^{n} y_i[log(\hat{y}_i) + (1 - y_i)log(1 - \hat{y}_i)] \tag{2.16}$$

- **Categorical Cross-Entropy** for multi-class classification problems:

$$CCE = -\sum_{i=1}^{n}[y_i log(\hat{y}_i)] \tag{2.17}$$

- **Mean Squared Error (MSE)** for regression problems:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}[y_i - \hat{y}_i]^2 \tag{2.18}$$

Optimizer

In order to update the Neural Network's Weights and optimize the Loss Function, an Optimizer is used. Gradient Descent-Based Algorithms are the most common for this, they are all variations of the Gradient Descent Algorithm, each introducing a small change on the original.

The Optimizer defines how the DL model learns and algorithm choice plays an immense role in the eventual performance of the resulting model[22].

Gradient Descent

Gradient Descent is one of the most widely used algorithms for Neural Network optimization, and one of the most popular optimization algorithms of all time. It is often used in one of its many variations[91]. Its equation is:

$$\theta_{j+1} = \theta_j - \alpha \frac{\partial(Loss)}{\partial(\theta_j)} \tag{2.19}$$

Such that:

- θ is the parameter/weight to optimize.
- I is the Loss Function.

Gradient Descent updates its parameters iteratively all the while minimizing a given Cost Function to its minimum. It reduces the Cost Function's value by moving in the opposite direction of its steepest ascent. It is dependent on derivatives of the Loss Function for finding minima[91]. This process is represented in Figure 2.16[3] below:

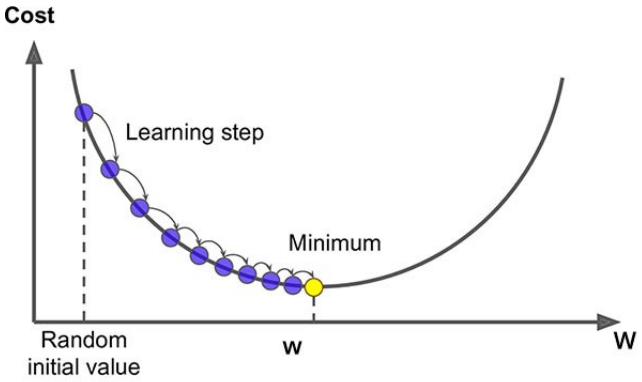


FIGURE 2.16: Gradient Descent optimization process.

Alpha is the learning rate, it represents how big/small the steps the Gradient Descent takes into the direction of the local minimum are, which figures out how fast or slow the weights will move towards their optimal values. If this value is too small, parameter convergence will be slow, and if it is too large, the loss function might oscillate or deviate from the minimum. A compromise must be found mainly using a trial and error process[91]. These three cases are illustrated in Figure 2.17[4] below:

[3]Source: https://morioh.com/p/15c995420be6
[4]Source: https://medium.com/mlearning-ai/optimizers-in-deep-learning-7bf81fed78a0

FIGURE 2.17: Learning Rate effect on Gradient Descent.

The amount of data the algorithm processes in each of its iterations depends on which of its variations is used, they are:

- **Batch Gradient Descent**

  Batch Gradient Descent uses the entire training dataset to calculate the gradient of the Cost Function to the parameters which consumes a large amount of processing power and takes a long time, although it is easy to implement[91].

- **Stochastic Gradient Descent (SGD)**

  Another one of the Gradient Descent's variations. It updates the model parameters on the training data one by one[91]. This method requires less memory, and for that same reason allows the processing of much bigger datasets, but on the other side, its frequency can have a negative effect, resulting in noisy gradients which may cause the error to increase instead of decreasing it, it results in much higher variance, and its frequent updates are computationally expensive[22][91].

- **Mini-Batch Gradient Descent**

  Mini-Batch Gradient Descent is a combination of both SGD and Batch Gradient Descent. Splitting the training dataset into batches and performing a parameter update using each of those batches at a time. This creates a middle ground between SGD's robustness and Batch Gradient Descent's efficiency. This method leads to more stable convergence, has more efficient parameter calculations and requires less memory than Batch Gradient Descent, its only downside is that it does not guarantee convergence, and depends on the batch size, which is an important hyperparameter in the learning process[91]

- **Adaptive Moment Estimation (Adam)**

  One of the most commonly used Gradient Descent Optimization algorithms. This method computes adaptive learning rates for each parameter, storing both the decaying averages of past gradients, and the decaying average of the past squared gradients. Adam is easy to implement, computationally efficient, and requires little memory[2][91].

Its algorithm relies on the following equations:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{2.20}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{2.21}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{2.22}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{2.23}$$

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \tag{2.24}$$

2.4.2 Feedforward Neural Networks

One of the simplest types of ANN, Feedforward Neural Networks are Neural Networks in which information only moves in one direction: from the input nodes through the hidden nodes and out the output nodes. There are no loops in the network or in the connections between layers, like there are in Recurrent Neural Networks[40].

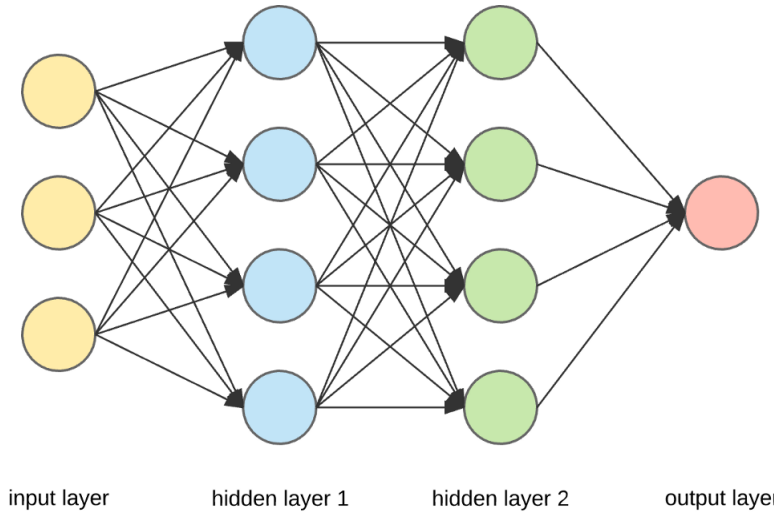


FIGURE 2.18: A Feedforward Neural Network with an input layer, an output layer, and 2 hidden layers.

2.4.3 Recurrent Neural Networks (RNN)

Recurrent neural network is an artificial neural network with cyclic connections. They are composed of interconnected units (neurons) that interact nonlinearly, and they contain at least one cycle in the structure. These units are connected by arcs-synapses with weights. The output of a neuron is a non-linear combination of its inputs as can be seen in Figure 2.19[33].

Recurrent neural networks are particularly suitable for variable size input data and time series analysis.

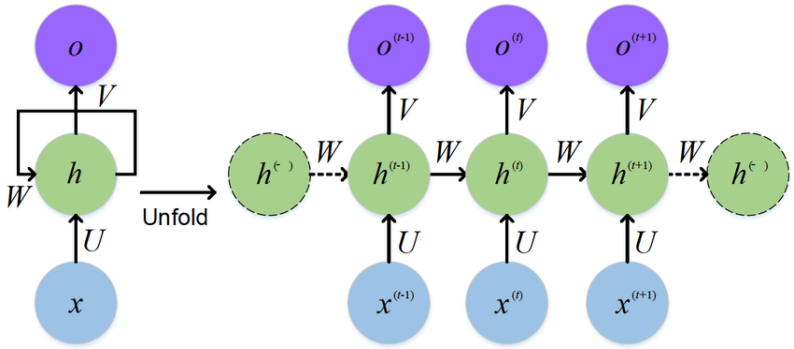


FIGURE 2.19: A Recurrent Neural Network.

2.4.4 Long Short-Term Memory (LSTM)

Due to the problem of vanishing gradient which risks overfitting, training standard RNN is challenging. To overcome this challenge, Hochreiter and Schmidhuber introduced a long-term Short-term memory (LSTM) model [44]. This model replaces the hidden layer of the standard RNN with a memory cell, and uses a cell, that memorizes data over random time intervals, and three gates: an input gate, an output gate, and a forget gate, whose job is to regulate the flow of data in and out of the cell.

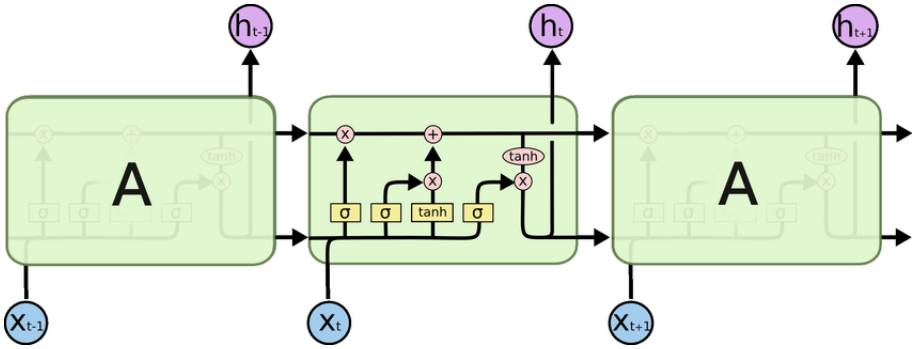


FIGURE 2.20: LSTM Cell.

Bidirectional LSTM or bi-LSTM networks were first proposed in [93], they are variation of normal the LSTM unit. Bi-LSTM models are trained not only from the inputs to the outputs, but also from the outputs to the inputs. Bi-LSTM models first feed input data to an LSTM model, and then repeat the training via another LSTM model but on the reverse order of the sequence of the input data, thus analyzing the data in both directions.

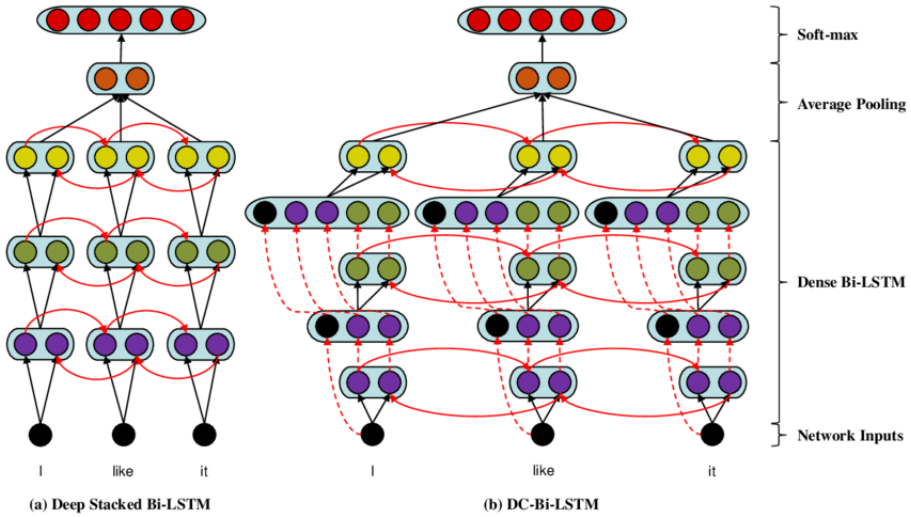


FIGURE 2.21: A standard LSTM and a bi-LSTM.

2.4.5 Convolutional Neural Networks (CNN)

In the past decade, Convolutional Neural Networks (CNN) have become the main standard operator for Computer Vision and Deep Learning. CNN is a Feedforward Artificial Neural Network with alternating convolution and pooling/subsampling layers, followed by Fully Connected Layers. Deep 2D CNN with many hidden layers and millions of parameters are capable of learning very complex objects and patterns, provided that they can be trained on a large scale visual database with real labels. With proper training, this unique ability allows them to be the main tools for various engineering applications of 2D signals (such as images) And video frames. However, in many other applications, the 1D-CNN may be a more viable option Especially for time series data.

1D-CNN were recently proposed fully study time series, and they immediately became the norm for multiple applications. Another major advantage is that Real-time and low-cost hardware implementation is feasible because of simplicity and compactness of one-dimensional CNN that only perform one-dimensional Convolution (scalar multiplication and addition), as well as one-dimensional Pooling.

1D-CNN can recognize local patterns in a sequence because the same filter (input transformation) passes over each plot (Figure 2.22. The filter having characterized a pattern at a certain position will be able to recognize the same pattern at a different position which makes the transformation invariant in time.

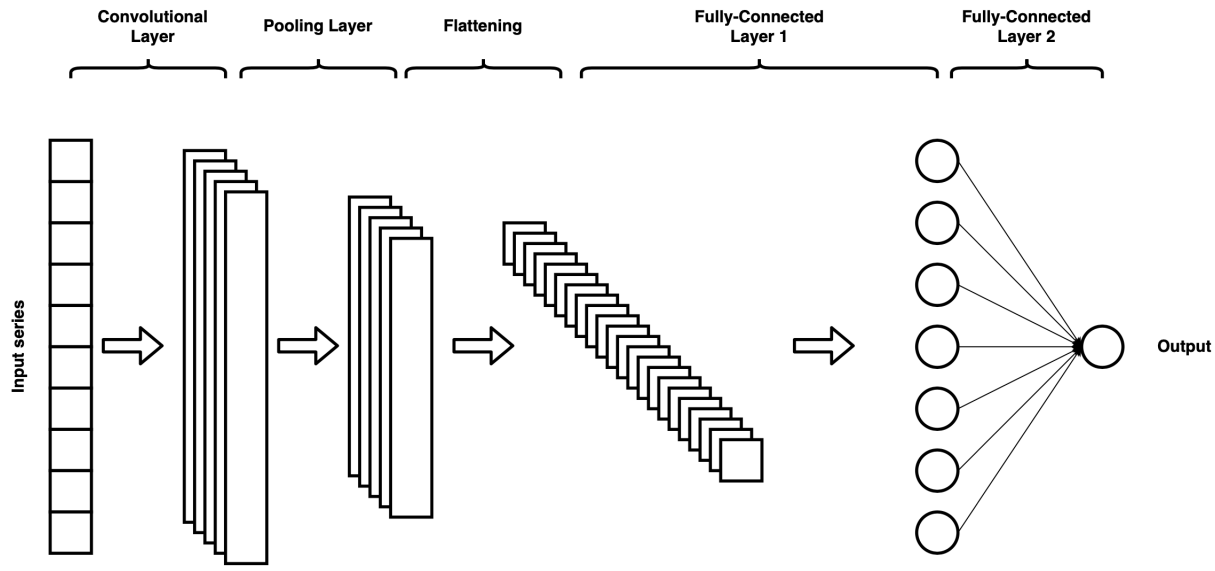


FIGURE 2.22: 1D-CNN Architecture.

2.4.6 Underfitting and Overfitting

Unfortunately, Neural Networks suffer from Underfitting and Overfitting too, the two concepts have already been explained in 2.3.2.

The solution for Underfitting in Neural Networks is to simply increase the model's capacity, which is its ability to fit more functions with more weights, giving it more range for mapping inputs to outputs. This is usually done by adding more layers and/or more Neurons to layers[13].

The ease of addressing an underfit model makes the more common problem Overfitting[13], which can be dealt with in a wide range of solutions called Regularization techniques.

2.4.7 Regularization

Regularization is when subtle changes are made to the learning algorithm so that the resulting model can generalize better, this improves the model's performance on unseen data[22][13].

A few different Regularization techniques exist in order to reduce Overfitting in DL models, they include:

- **Dropout**
  Dropout is one of the Regularization methods used for avoiding Overfitting in Neural Networks. When training, a certain number of layer outputs are randomly dropped out, meaning they are ignored temporarily removing them from the Neural Network, along with all their connections. Creating a new layer that has a different number of Neurons and input and output connections.

The hyperparameter specifying the Dropout rate is a probability, representing the rate at which layer outputs are kept. A common value for it is 0.5 for hidden layers and a value close to 1, such as 0.8, for output layers[22][12].

- **Early Stopping**
  Generally, when training a Neural Network, at a certain point the model will reach minimum Loss but then rebound due to it no longer generalizing based on the training data but learning the statistical noise in the dataset. This makes the model less useful and worsens its performance on other unseen data.

  We then need to train the Neural Network long enough so that it learns the data's pattern, but not long enough so that It starts overfitting. This is called Early Stopping, and it is another Regularization method used to avoid Overfitting.

  One approach to stopping early is putting the number of epochs among the hyperparameters and training the Neural Network multiple times all the while registering the resulting model's performance and/or Loss, and then choosing whichever suits us best. This approach is time consuming and requires a large amount of computational power, especially for larger datasets.

  Another approach is training the model only once for a large number of epochs, and evaluating the model's performance using a Holdout Cross-Validation dataset on every epoch. This can give insight into the model's performance's evolution over time, the training process can then be stopped when the model's performance starts to decline[22][11]. Figure 2.23[5] below is an example of the resulting graph's form:

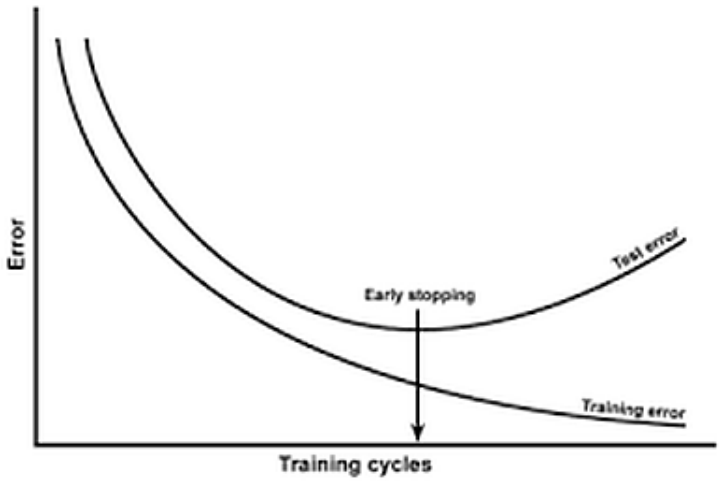


FIGURE 2.23: Neural Network training over many epochs (Training cycles) and Early Stopping point.

Other regularization techniques exist, such as activity regularization, introducing Weight Constraints, or training the Neural Network with Noise[13].

[5]Source: https://www.kdnuggets.com/2019/12/5-techniques-prevent-overfitting-neural-networks.html

2.4.8 Hyperparameter Optimization

Tuning hyperparameters is essential for having a well fit and performing DL model. This can be done using optimization algorithms such as Particle Swarm Optimization (PSO)[109], Genetic Algorithms (GA)[76], or Bayesian Optimization[51]. However, these algorithms come with the cost of additional computing time and even more parameter optimization, and in many cases it is better to use manual methods like Grid Search, Random Search[121], or just a trial and error process by generating a large amount combinations and taking the best one.

2.5 Dimensionality Reduction

Dimensionality Reduction, is the process of transforming high dimensional data into low dimensional data, to both reduce the number of features because high dimensional data is very time consuming to manage, as well as to derive meaning from the few remaining features[106], which is this section's objective. With the features being brain regions, this meaning is the exact regions that witness Energy/Entropy changes when transitioning into or from a state of fatigue. The type of Optimization algorithms that suit this objective are Metaheuristic Methods:

2.5.1 Metaheuristic Methods

Metaheuristic methods are a subfield of stochastic optimization, utilizing their objective functions in an abstract way without further insight into their inner structure, essentially treating them as a black-box. These methods are often inspired by used in a wide variety of problems, making them the most general type of algorithm among all optimization algorithms[45][73].

Many of these methods are inspired by natural systems, such as the thermodynamic process underlying annealing (the atoms in the molten material randomly move about, but, as the temperature drops, attempt to settle into the lowest possible energy state), the behavior of a colony of ants searching for food, or in our case, evolution and natural selection, creating varied and powerful algorithms[73][100].

These algorithms include but are not limited to: Particle Swarm Optimization (PSO), Simulated Annealing (SA), Ant Colony Optimization (ACO), and Genetic Algorithm (GA) which will be explained in detail in the next section.

2.5.2 Genetic Algorithm

The processes of natural evolution, as seen in a population of individuals competing for survival, has inspired the metaheuristic optimization algorithm known as the Genetic Algorithm[45].

This algorithm's areas of application include Chemistry, Medicine, Data Mining and Data Analysis, Geometry and Physics, Economics and Finance, Networking and Communication, and Electrical Engineering and Circuit Design[111].

In this algorithm, a population of individual solutions are represented by a binary sequence called a genome, similar to that of a human DNA, a first generation is randomly generated, natural selection then selects who survives and gets to reproduce, based on an objective/cost function called a fitness function, individuals would be selected from the pool of genomes favoring those who had better fitness using a selection function, the resulting members would then be combined into new ones using a crossover function, which will mix and recombine the survivors' genes into a new population, essentially mating the two candidate solutions, a mutation function would then be applied to randomly alter some genes to further increase diversity among them and to avoid local minima, the next generation is now born. This cycle is then applied across multiple generations (iterations), however many is needed to get the needed genome which represents the fitness function's optimal point and/or satisfies the algorithm's terminal condition, many of the functions mentioned above have parameters that need to be fine-tuned in order for the algorithm to properly converge in a convenient time frame, all of which will be detailed in the following sections[45][100][111].

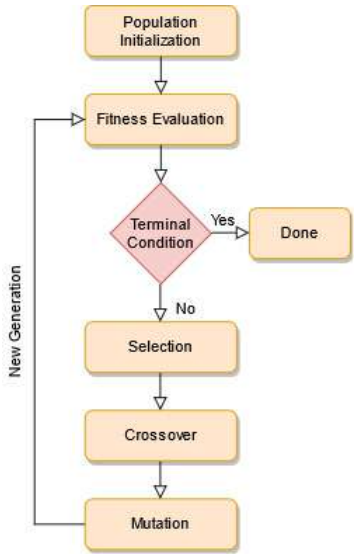


FIGURE 2.24: The Genetic Algorithm Cycle.

This algorithm is comprised of various functions, all of whom will be detailed in the next section in order of execution.

Genetic Representation

In order to apply the Genetic Algorithm to our problem/fitness function, a genetic representation function should be employed in order to encode the candidate solutions in the form of a binary sequence (called a genome, genotype or chromosome) which acts as the population (also called phenotypes). This representation differs in accordance with the candidate solution's form, and may in cases involve non-binary representation, when using integers/real numbers they can simply be converted to binary, in the case of a list the different items can be encoded using integers, or in that of 3D coordinates the binary sequence can be omitted for one containing the coordinates themselves, many more encoding schemes exist, but binary remains the most common one[105]

Seeing as this algorithm will be applied to regions of the brain, or features thereof, and only concerns their presence or absence in a ML/DL algorithm, each of the regions/features are represented by a binary sequence with a length of 16 bits.

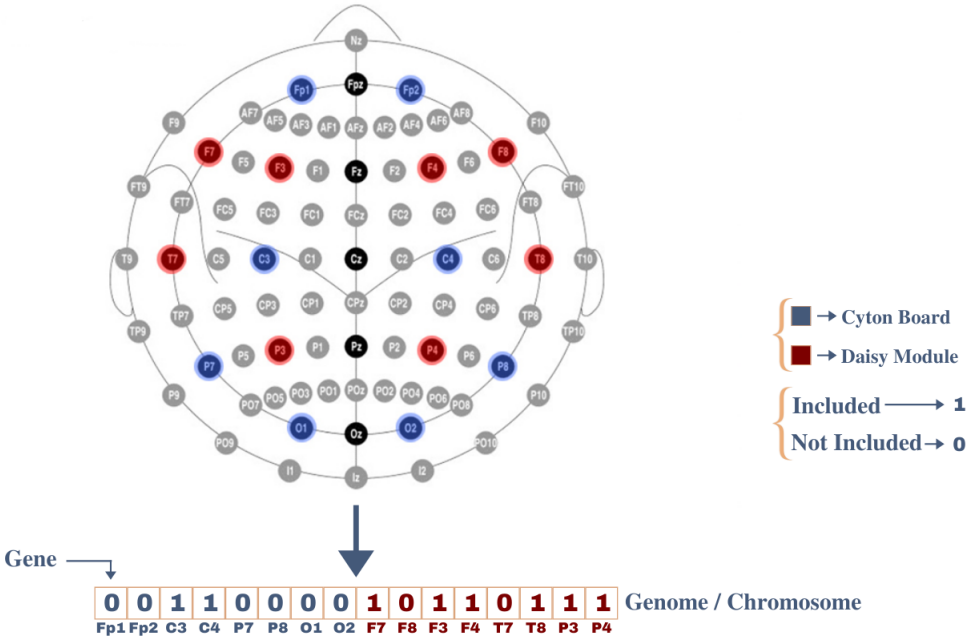


FIGURE 2.25: Genetic Representation of The Genome.

Population Initialization

When launching the algorithm, there is no existent population, thus the need to initialize one by random generation of genomes. The population size is generally in the range of 50-1000 depending on the problem at hand and the computational time needed for each individual[111][105].

Apart from random initialization, a heuristic one exists, initializing the initial population based on a known heuristic for the problem, and it has been seen to result in similar solutions that lack diversity and decrease the overall exploration capability of the algorithm[74]. This method will not be used because the nature of the problem requires proper convergence.

Fitness Function

The Fitness Function evaluates an individual's aptitude to survive within the population to be able to reproduce. It is the most important part of the Genetic Algorithm, any flaw in this function can prevent the algorithm from converging. GA being a Metaheuristic method means that there are countless possibilities for Fitness Functions because the inside content does not matter so long as the result is outputted [111][105].

Selection

Selection is where parents are chosen from the population for mating, and where the survivors whose genes get carried over are determined based on their fitness, it's applied to every member of the population[105].

- **Tournament Selection**
  This method draws k candidates from the population randomly and selects the members from the group with the best fitness, k is a parameter to tune.
- **Fitness Proportionate Selection (Roulette Wheel Selection)**
  This method involves selecting individuals based on a probability of being selected. This is done by sorting the population by fitness, generating a random probability, iterating through the sorted population, summing each individual's score up until it is equal to or greater than the probability, they are then selected as a parent.
- **Truncation Selection** This method selects a percentage of the fittest individuals to either mate with the rest of the population or with each other[105].

Crossover

In the analogy of this algorithm to natural selection, crossover represents reproduction and biological crossover, where parents are selected and offspring are produced using their genetic material[86]. There are many types of crossover, depending on the problem at hand.

- **Single-point Crossover**
  In this crossover, a random crossover point is selected and the two parent genomes' segments are swapped to get new offspring[105].
- **K-Point Crossover**
  This is a generalization of the single-point crossover wherein alternating segments are swapped to get new offspring[86].
- **Random Crossover**
  Child genes are treated separately and chosen randomly from either one of the corresponding parent genomes[105].
- **Uniform Crossover**
  Each gene is treated separately and a probability is used to decide which of the parents' genes is included in the offspring[86].

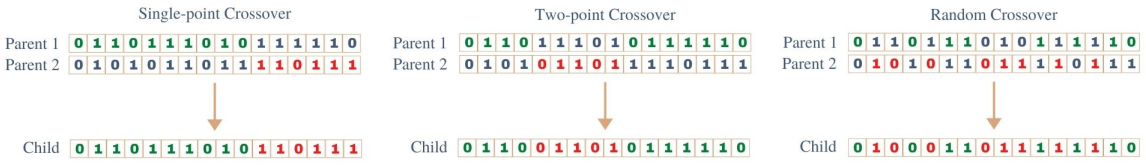


FIGURE 2.26: An illustration of Single-Point, Two-Point and Random Crossover.

Other types of crossover exist, such as mixed crossover, parent centric crossover, sequential crossover, and random mixed crossover[86].

The type used in this case is a single-point crossover, the crossover probability will be discussed in the results section.

Mutation

Like its counterpart in nature, mutation helps diversify the population and expand its search area, with the added benefit of helping the algorithm avoid local minima by preventing the population from becoming too similar. It is implemented on every child in the population by giving each of their genes a small probability of being altered at random, depending on the genetic representation[111][105].

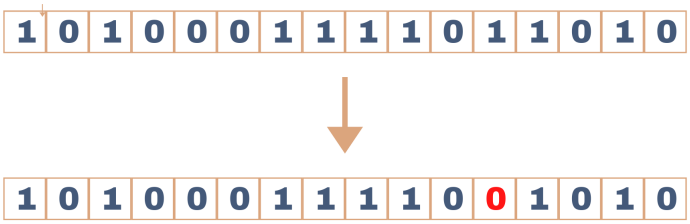


FIGURE 2.27: Mutation as applied on a Genome.

The genomes used are binary sequences, which means the individual bits are inverted when mutating a genome, the mutation probability will be discussed in the results section.

2.6 Conclusion

For the many applications involving the use of EEG signals, processing them is always necessary. In this chapter we have seen the different processing tools used in our work. First, we have defined frequency and spatial filtering for the pre-processing of signals. Then the extraction of characteristics that are classified by their domain: temporal, frequential, time-frequency, or spatio-temporal. The classification step is done with Machine Learning algorithms such as SVM, KNN, Decision Trees, and Random Forests classifiers and/or Deep Learning algorithms like CNN or LSTM as detailed. The evaluation process is done using classification metrics like accuracy, recall, precision, and the confusion matrix. Meta-Heuristic algorithms were also introduced for use in EEG systems for feature selection and dimensitioniality reduction.

In the following chapter, we will introduce the model and materials used for the realization of the Brain Computer Interface (BCI) based on Motor Imagery, we will detail the proposed protocol , demonstrate the implementation of the described methods and we will discuss the obtained results.

3 Brain Computer Interface Implementation for Mobile Robot Control

Abstract

Along this chapter, we present the implementation process for BCI design. The chapter starts with an introduction in3.1. Then in Section 3.2, we describe the materials used in our project as well the necessary mathematical model for robot trajectory evaluation. Subsequently, we detail our proposed experimental protocol for dataset collection in Section 3.3. In addition, the public dataset used in this work for comparison purpose is also described along the section. Further in In Section 3.4, we will outline the offline implementation architecture of BCI realisation, and its result will be showed in section 3.5, thereafter Section 3.6 demonstrates the real time implementation of our proposed architecture.

3.1 Introduction

BCI design requires seven main processing phases in order to ensure an acceptable results. Starting from choosing the type of BCI to work with, then collecting the data which can be provided from recording a private dataset using a specific instrumental sensor or using a publicly available dataset such as the BCI competition dataset. The next step is the preprocessing of the raw data by eliminating background noise and undesired frequencies. After that comes extracting application-specific features from the cleaned version of the data, these features can be computed in time domain, frequency domain, spatial domain or a combination of these such as the time-frequency features, a selection of the most salient features from the extracted ones is then done by applying one of the various methods proposed in the literature. These obtained characteristics are then classified using Machine Learning algorithms to deduce decisions and send commands to the desired device. The final step of this process is to provide feedback to the user based on the decisions after executing the commands by the device, visual feedback is the most commonly used [75], but other forms of feedback can be used such as auditory or haptics [3].

3.2 Materials

3.2.1 EEG Acquisition Device

The acquisition process of the EEG signals is achieved with the OpenBCI Ultracortex Mark IV headset available in the Complexes Systems Control and Simulator Laboratory ($L(CS)^2$) of the Military Polytechnic Institute in Algiers. This device offers open source access to all materials and data. This device is capable of recording research-grade brain activity (EEG), muscle activity (EMG), and heart activity (ECG) by sampling up to 16 channels of EEG signals and up to 35 different locations using dry electrodes for recording, which significantly reduce the time needed for preparation. The placement of the electrodes on the human scalp in the context of EEG is established according to the 10-20 international configuration detailed above. Figure 3.1 presents the 16 channel configuration of the Ultracortex headset that we have used in our project.

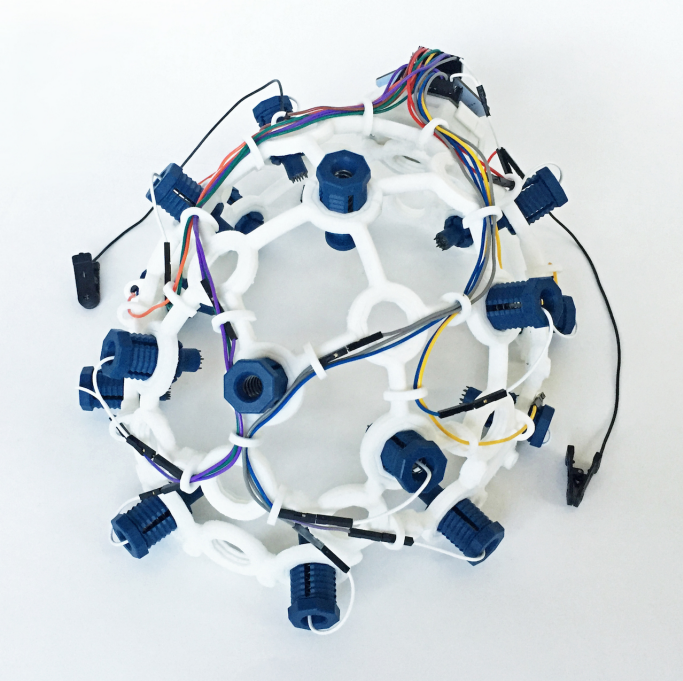


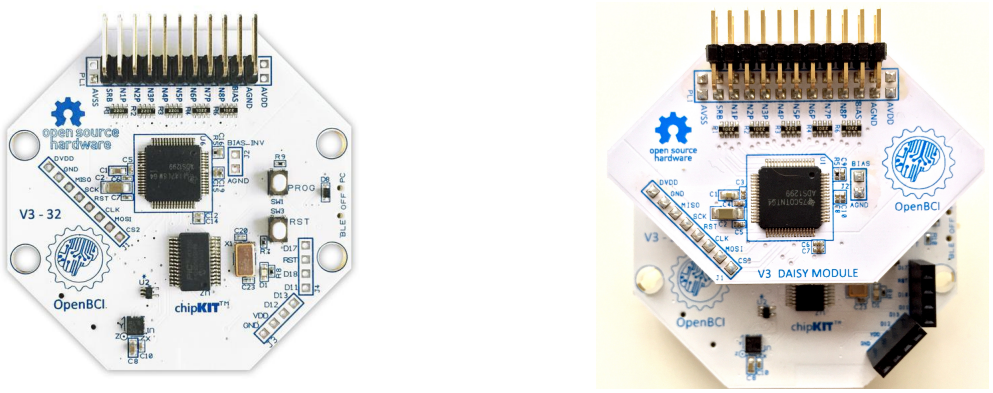
FIGURE 3.1: ULTRACORTEX Mark IV headset with Daisy Module included (16 channel)

Acquisition Module

The main component of the Mark IV headset is the Cyton board (Figure 3.2a) which includes a micro-controller, an amplifier for electrical recordings, wireless connectivity and a SD card recording options for up to 8 recording channels from which electrophysiological data can be collected.

The Daisy Shield on the other hand allows data acquisition from eight additional channels as has been seen in Figure 3.2b. Therefore, with this combination, the OpenBCI system allows collecting EEG signals from 16 channels with a sampling frequency of 125 Hz, data

resolution of 24 bits and compatibility with active and passive electrodes, among other features.



(A) Cyton Module (B) Daisy Module

FIGURE 3.2: OpenBCI Modules.

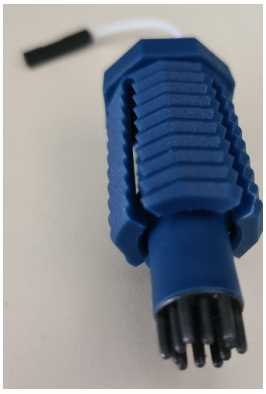
The Table shows the main characteristics of the OpenBCI Modules.

Features	Characteristics
Microcontroller	Atmel Atmega32-8p with Arduino uno bootloader with 8GPIO pins
Accelerometer	ST LIS3DH 3 axis accelerometer with 16 bit data output
Input Channels	8 biopotential input channels, 16 input channels for Daisy Module.
AC DC Converter	ADS 1299 Texas instrument with a High gain, and a Low noise ADC, 24 bit channel resolution.
Type of the extracted signal	EEG, EMG, ECG.
Sampling Rate	From 125Hz up to 16 kHz.

TABLE 3.1: OpenBCI Module Characteristics.

Electrodes

OpenBCI provides four different dry electrodes: flat (flat), pointed (spiky), 5mm combs and comfort electrodes. These electrodes are made of AgCl silver chloride.



(A) 5 mm pointed electrode.

(B) Comfort electrode.

(C) Spiky electrode.

(D) Flat electrode.

FIGURE 3.3: OpenBCI electrodes.

Ground and Reference Placement

Grounding is used for common mode rejection. The main purpose of grounding is to prevent power line noise from interfering with small biopotential signals of interest. According to the design, the amplifier should not be affected by large changes in the active and reference point potentials. For reference, this electrode is usually placed on the ear, or in the case of "total ear", placed on a pair of electrodes, one for each ear. Ideally, the measured potential difference is the voltage drop from the active electrode (connected to Vin+ on the amplifier) to the reference electrode (connected to Vin- on the amplifier). For the openbci system , the right earlobe is considered as the reference and the left earlobe as the ground.

OpenBCI Recording Software

The recording, streaming and the visualisation of the eeg data is done with the OpenBCI GUI software proposed by the OpenBCI Community. The data is recorded via a bluetooth communication between the bluetooth dongle and the headset. After setting the connection a live-time displaying of the data is provided. The data is saved in the computer with a .txt format to be used for an offline processing using Matlab or python. A live-time streaming of data to a third-party such as Matlab, Python, Ros or Java can be done using the LSL streaming layer.

FIGURE 3.4: OpenBCI GUI Software

The Openbci GUI is compatible with all the OpenBCI boards like ; Cyton, Ganglion, the Wifi Shield and the Daisy board that we have used in our experiment. The software provides multiple tools and visualisation option such as :

- Time series plot.
- Frequency spectrum plot.
- Headset energy distribution plot.
- EEG rhythms (α, β..) power represented in histograms.
- 50 Hz and 60 Hz notch filter
- Impedance testing
- Pre-defined pass band filtering
- signal smoothing

3.2.2 Mobile Robot Description

The E-puck 2 mobile robot available in the $(L(CS)^2)$ laboratory is used in this work. It is a small robot used for education and research. It was originally designed by Francesco

Mondada and Michael Bonani at EPFL (Lausanne, Switzerland) in 2004. A more recent version e-puck 2 as it is represented in Figure 3.5 was realized in 2018 which is more powerful and better equipped with sensors with WiFi, USB connectivity, long distance Time of Flight distance sensor, RGB led, and more. The e-puck is open hardware and open source platform.

FIGURE 3.5: E puck2 robot

E-puck 2 Robot Properties

Although the e-puck was developed for education, its low price and large number of sensors make it interesting for research. It has been used in collective robotics, evolutionary robotics, and artistic robotics. In [78] a control of synchronization regimes in networks of mobile interacting agents based of epuck robot was achieved and in [62] a multiple epuck robots was formed as swarm groupe of robotics system and a Supervisory control theory was applied to control the proposed system. The main specification of the e-puck 2 robot is described in the table 3.2 below:

TABLE 3.2: E-puck 2 Proporties.

Feature	Characteristics
Size, Weight	70mm diameter,55mm height, 150g
Processor	32-bit STM32F407 168Mhz
Motors	2 stepper motors with a 50:1 reduction gear and 20 steps per revolution
Wheels	diameter=41mm, distance between wheels=53mm
Speed	Max=1000 steps/s (about 12.9 cm/s)
Distance sensor	8 infra-red sensors measuring ambient light and proximity of objects up to 6 cm.
IMU	3D accelerometer, 3D gyro, 3D magnetometer.
Camera	VGA color camera , typical use: $160 * 120$
Audio	4 omni-directional microphones (digital) for sound localization speaker capable of playing WAV or tone sounds
LED	4 red LEDs and 4 RGB LEDs around the robot, green light, 1 strong red LED in front
Communication	USB Full-speed, Bluetooth 2.0, BLE, WiFi
Programming	Free C compiler and IDE, Webots simulator , onboard debugger (GDB)

Robot Localisation

Since trajectory tracking and evaluation requires knowing the exact position of the center of the robot, Webot simulation provides the angular displacement of the left and right wheel, the position of the center is computed following the development below: The odometers provide the angular displacement $\delta d_D(t)$, and $\delta d_G(t)$ of each of the rear wheels during the period t. From these two values, the elementary displacements of the vehicle in rotation δd and along its path $\delta\theta$ (Figure 3.28), are given by :

$$\delta d(t) = \frac{R(\delta d_G(t) + \delta d_D(t))}{2} \tag{3.1}$$

$$\delta\theta = \frac{R(\delta d_G(t) - \delta d_D(t))}{2L} \tag{3.2}$$

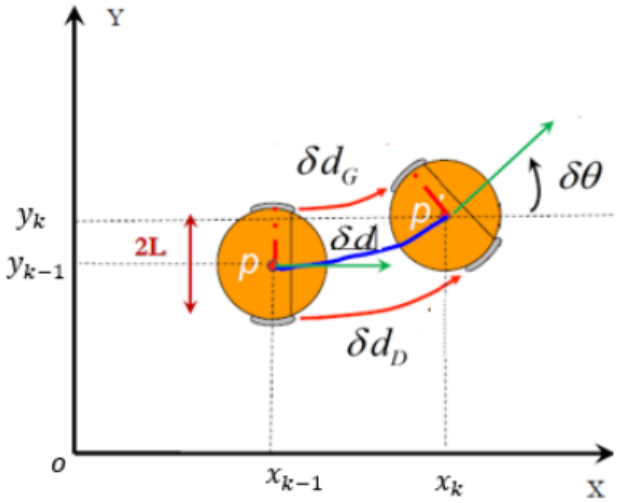


FIGURE 3.6: Robot Odometry Scheme

We seek to express recursively, with respect to the universal reference frame R, the position of the mobile system (x_k, y_k, θ_k) at time k as a function of the position $(x_{k-1}, y_{k-1}, \theta_{k-1})$ at time $k-1$ and the measured elementary displacements. The kinematic model of the mobile robot is described according to the following system of equations:

$$\begin{cases} \dot{x} &= v\,cos(\theta) \\ \dot{y} &= v\,sin(\theta) \\ \dot{\theta} &= w \end{cases} \tag{3.3}$$

Since the robot model is nonlinear, for which there is no general discretization method [9]. In the particular case we are interested in, we can approximate $\dot{x}$ by a finite difference, if the sampling period T_e is small (same for $\dot{y}$ and $\dot{\theta}$) then :

$$\dot{x} = \frac{x_k - x_{k-1}}{T_e} \tag{3.4}$$

We finally obtain the equations below that leads to the identification of the position of the centre of our robot:

$$\begin{cases} x_k &= x_{k-1} &+ \delta d\,cos(\theta_{k-1}) \\ y_k &= y_{k-1} &+ \delta d\,sin(\theta_{k-1}) \\ \theta_k &= \theta_{k-1} &+ \delta\theta \end{cases} \tag{3.5}$$

The equations above describe the displacement of a moving system in a straight line along the direction θ_{k-1} along length δd , and then rotates in place by $\delta\theta$.

3.3 Datasets Description

3.3.1 Private Dataset Building

Experimental protocol

Two male subjects (23 ± 1 years) were selected from the LCS^2 laboratory. A quick questionnaire was established to make sure the subjects have no medical contraindications such as a prior brain injury, use of prescription medication, severe concomitant or any psychological ,minor psychiatric disorders or any intellectual problems. For a proper acquisition, the subjects have to be in an alert state defined with a good sleep quality and a minimum mental activity. They were also requested to stop drinking caffeine such as cafe, tea or any stimuli 24h before the acquisition. The experiment was performed from 13h to 15h30 in a very quiet room with a controlled temperature. The participants received a comprehensive instruction about the protocol and the required tasks. Each participant needs to be trained for at least one week or 5 sets of training sessions before the official acquisition where the data are actually collected and used. The protocol for movement imagination consists of :

- **Left movement :** the left movement is defined by imagining squeezing an object with the left hand.
- **Right movement :** the right movement is defined by imagining squeezing an object with the right hand
- **Forward movement:** the forward movement is defined by imagining the movement of the tongue, this movement was previously defined as the imagination of feet movement, but due to problems of such imagination and realisation, we had to change this specific protocol, another advantage for this change is that the signals that are responsible for the tongue can be separable from the other classes in contrary to the feet imagination.
- **Stop:** the protocol for stopping the movement is defined with eye movement. A voluntary eye movement made by opening and closing both of the eyes is a short amount of time (2 to 3 seconds), the effect of the voluntary eye movement is affected in both EEG and EOG signals. The motive for this protocol is to detect with efficiently the stop and to be highly distinguishable.

Electrodes Placements

The main focus of the acquisition is to provide the EEG from from the region that is responsible for the motor imagery, as detailed in the 1 we chose the following electrodes placement based on the 10-20 system: F3,Fz,F4,FC5,FC1,FC2,C3,CZ,C4,CP1,CP2. And since the experiment protocol requires the EMG and EOG signal for the stop movement, as a result we have collect the signal for this class using the electrodes located in : FP1,FP2,F7,F8 and as it be seen it the Figure 3.7 below.

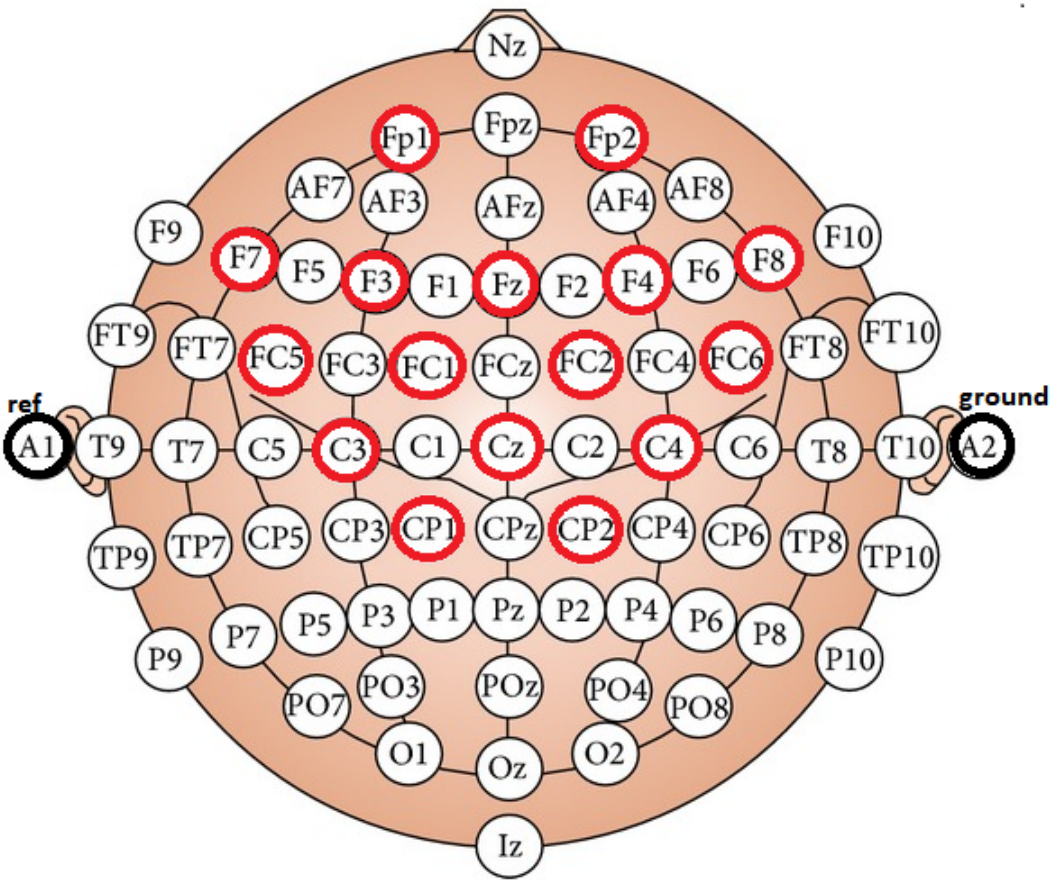


FIGURE 3.7: Electrode placement for the motor imagery

3.3.2 Public Dataset

In our work, we tested our approach and model with the BCI competition IV 2a dataset.The EEG acquisition for the dataset was made for nine able-bodied subjects, three females and six males, with a mean age of 23.8 years (standard deviation of 2.5 years). The Cue-based com paradigm takes on 4 different tasks of motor imagery, the motor imagery of left hand movement (class 1), right right hand movement (class 2), both feet movement (class 3) and tongue movement (class 4). Two sessions were recorded on different days for each subject. Each session contains 6 sets of acquisition separated by a short pause. Each set of acquisition contains 48 trials (12 trials for each of the classes), which leads to a total of 288 tests per session. EEG signals were recorded using Twenty-Two Ag/AgCl electrodes, the locations of each electrode was based on the 10-20 international system, As shown in the figure 3.8 below, the electrodes were more concentrated in the motor cortex.

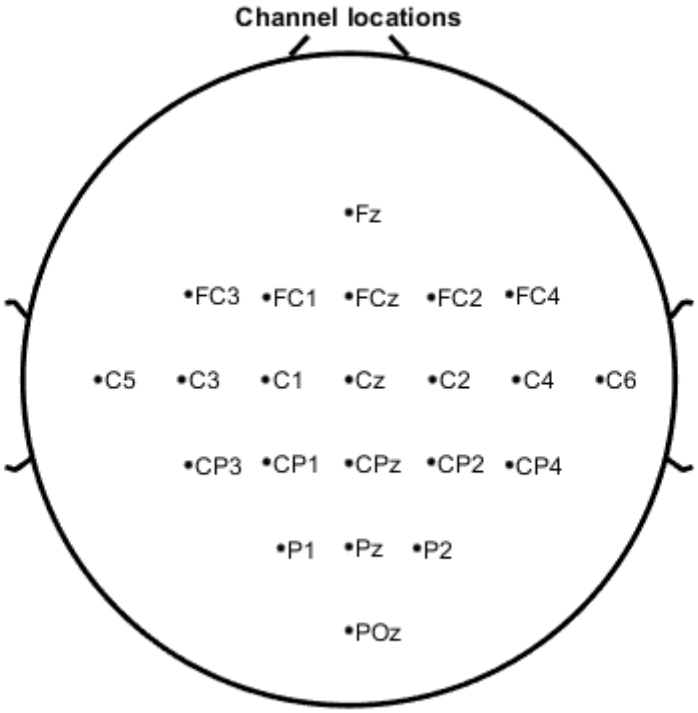


FIGURE 3.8: Electrode placement of the BCI IV 2a dataset for EEG acquisition

The reference for the recording signals was placed in the left mastoid and the ground was placed in the right mastoid. All signals were recorded monopolarly with the left mastoid serving as reference and the right mastoid as ground.The signal were sampled at 250 Hz and band-pass filtered between 0.5 Hz and 100 Hz. The sensitivity of the amplifier is set to 1 mV. An additional 50 Hz notch filter was enabled to suppress line noise. All the subjects were seated in a comfortable armchair in front of a computer screen during the acquisition,The data was visually inspected by an expert and the tests containing artifacts have been labeled.The EOG were recorded for the artifacts removal (Figure 3.9 and the study of the influence of the EOG signals for the EEG, as consequence these channels will not be included in the classification.

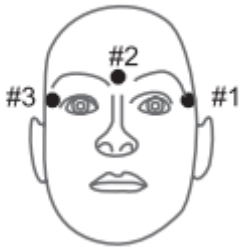


FIGURE 3.9: electrode placement for the three monopole EOG channels

The training paradigm includes repetitions of cue-based (synchronized) trials on four different moving image tasks, namely left hand, right hand, and foot imagination And tongue movement. At the beginning of each beep (at t = 0 s), a short beep will sound and a fixed cross will be displayed in the middle of the screen. After 2 seconds (t = 2 seconds), a visual cue (arrows pointing to the left, right, up or down) appeared on the screen for 1.25 s. Each position of the arrow requires the subject to perform a corresponding imaginary movement, namely to the left, right, tongue or foot respectively. Specifically, they were asked to maintain their imagination of movement for 3 to 6 seconds. After 6 s (t = 6 s), the fixed fork disappeared, allowing the subject to relax. The next test will start after a rest period of 1.5-2.5 seconds. The exact timing scheme is shown in the figure 3.10 below.

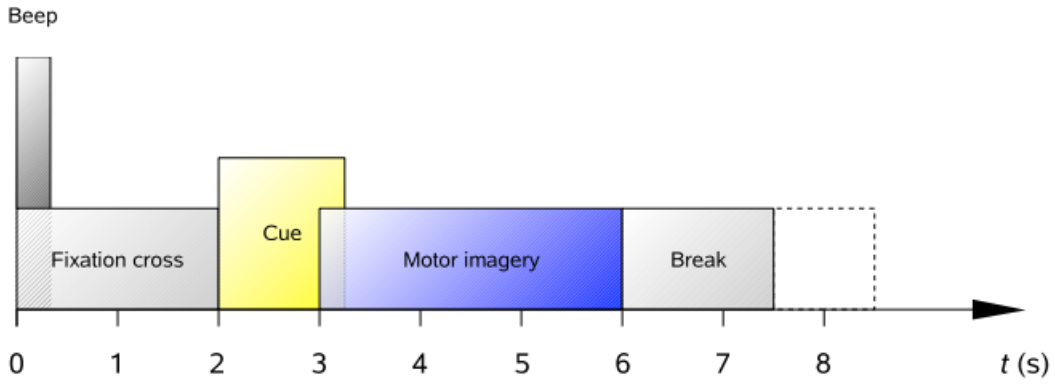


FIGURE 3.10: Diagram of the acquisition protocol

3.4 Offline Implementation Architecture

3.4.1 BCI Type

The main objective of our work is to control a mobile robot in multi direction by using the intention (imagination of the movement)of the user. BCI based on motor imagery is the most suitable in our case since it doesn't require any external stimuli, the realisation of this brain computer interface is based on the EEG signals as the measurement for the brain activity. The motive is to provide a non-invasive technique in order to not cause any harm to the user.

3.4.2 EEG Acquisition

The acquisition of the EEG signals is done with the ULTRACORTEX Mark IV 16 channels Daisy board headset with a sampling frequency of 125 Hz. The impedance between the electrode and the scalp is measured before starting the acquisition, this latter needs to be lower than 50 khz in order to provide a good acquisition with a high signal to noise ratio (SNR), the impedances were measured at 30hz as provides by the OpenBCI GUI software the signals are measured with a full respect of the experimental protocol. The table 3.3 shows the average of impedance obtained before starting the acquisition.

Channel	Brain Region	Average impedance (kOhm)	Channel	Brain Region	Average impedance (kOhm)
1	Fp1	10.4	9	F7	8.5
2	Fp2	12.5	10	F8	5.4
3	C3	18.7	11	F3	7.6
4	F4	9.6	12	C4	15.1
5	Cz	30.2	13	Fc5	5.2
6	Fz	26.4	14	Fc6	8
7	Cp1	14	15	Fc1	10.3
8	Cp2	18.2	16	Fc2	14.5

TABLE 3.3: Impedance Measurements.

3.4.3 Preprocessing

The acquisition of EEG signals by using non-invasive techniques has an influence of external noises such as the powerline but also the internal noises or artifacts like the eye blink and muscle movement, a Butterworth pass band filter [1 40] is applied to reduce the effect of the noises.
Since the motor imagery activities are related to α band [8 12] and the β band [12 30] , these rhythms are extracted by applying a pass band filter with respect to their defined frequencies.the Figures 3.11 and 3.12 illustrate the frequency spectrum before and after the preprocessing.

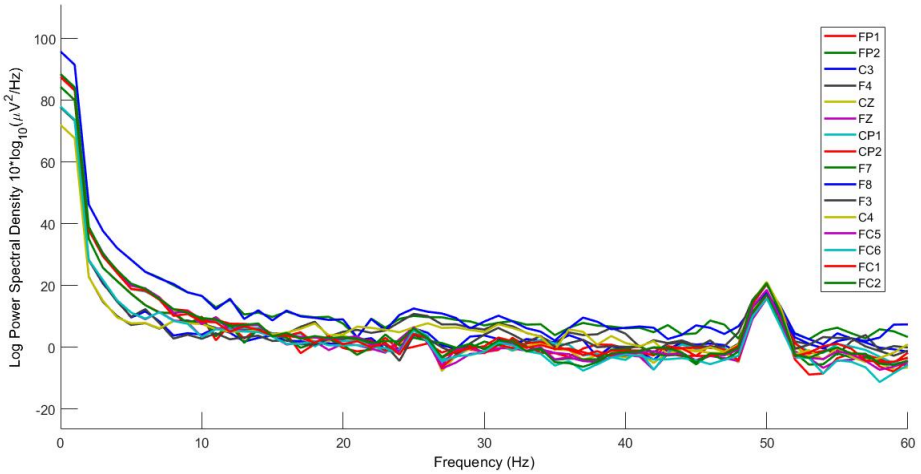


FIGURE 3.11: EEG Spectrum before filtering

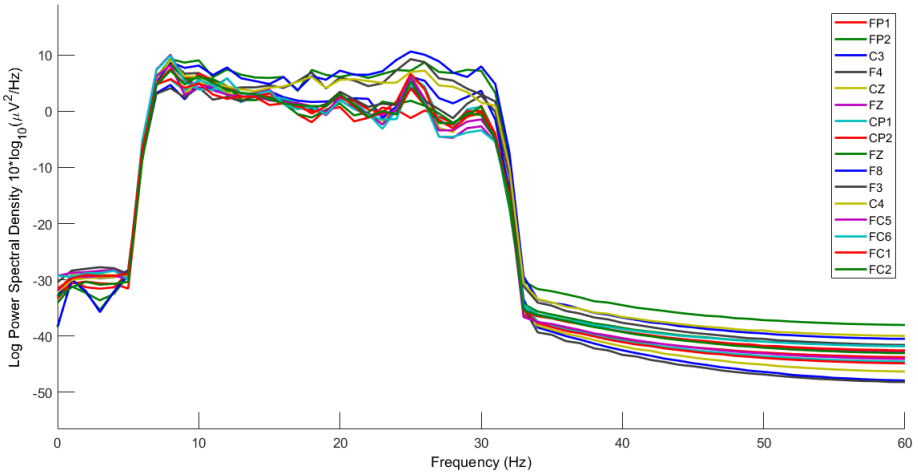


FIGURE 3.12: EEG Spectrum after filtering

3.4.4 Data Segmentaion

Time sequencing presents an important step for data processing, two type for segmentation can be defined:

Overlapping window

The first window is set with a fixed length, and the next window slides over the current window, such that the time interval between two windows is less than the width of the latter but greater than the processing time of the computer as can be seen in the Figure 3.13 below.

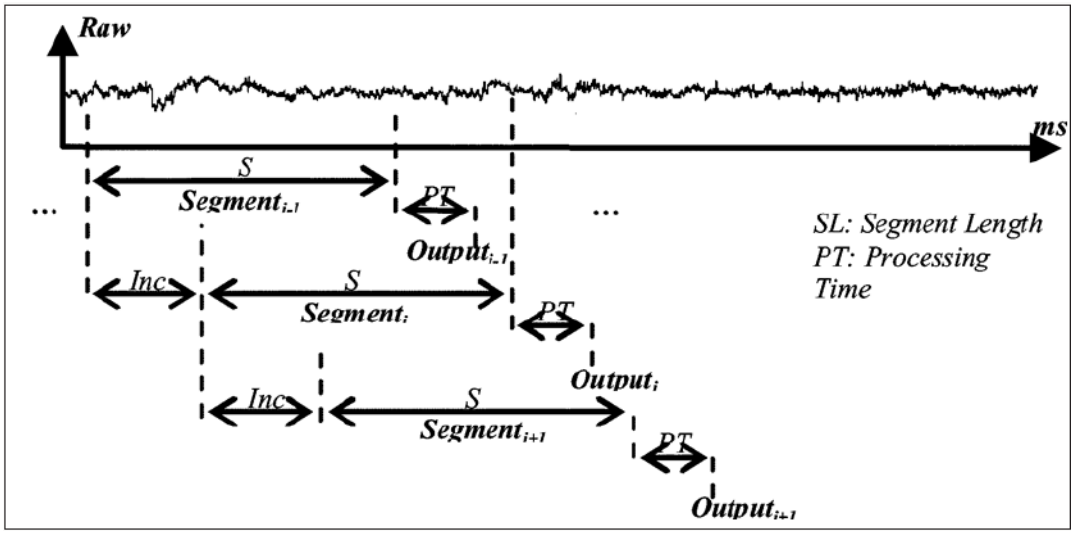


FIGURE 3.13: overlap window segmentation

Disjoint Segmentation

the data is decomposed into a predefined window, the length of the window needs to be bigger than the sampling time.in the figure below, we illustrate the denoised EEG signals with a fix segmentation of 200 ms in Figure 3.14 .

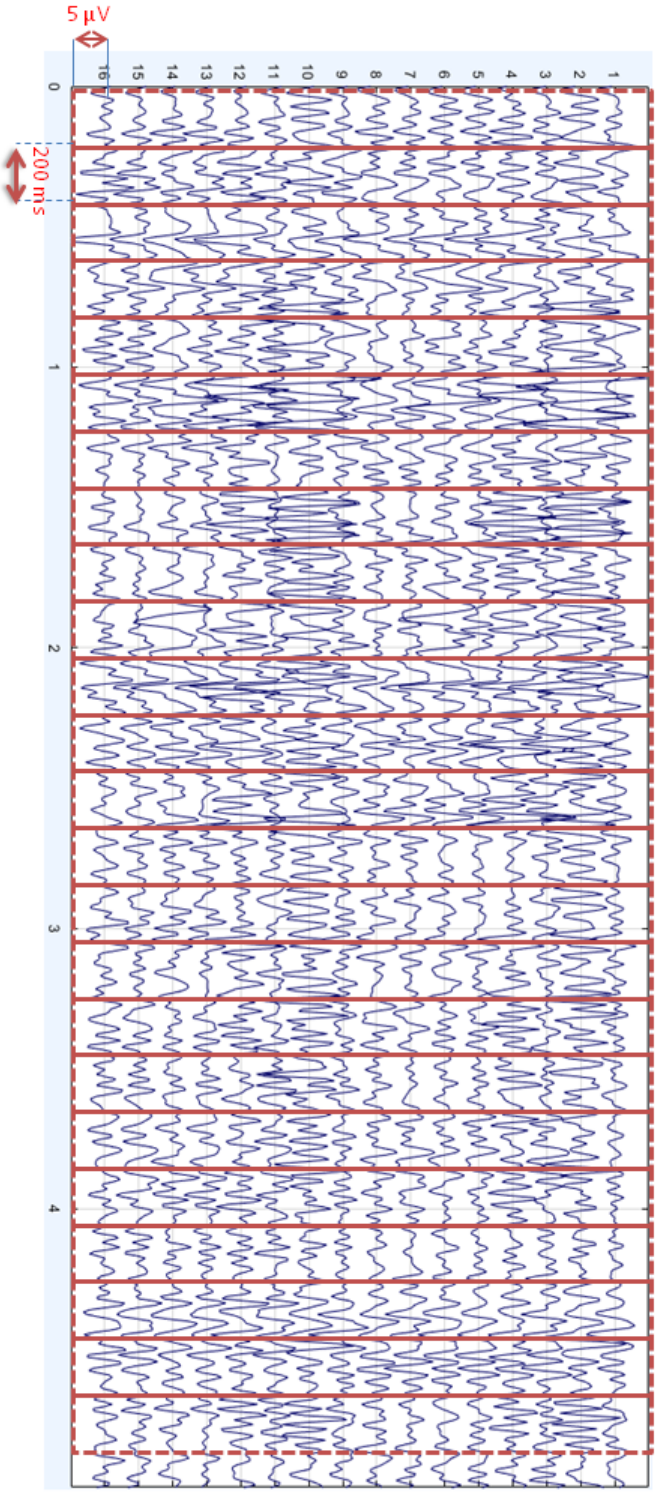
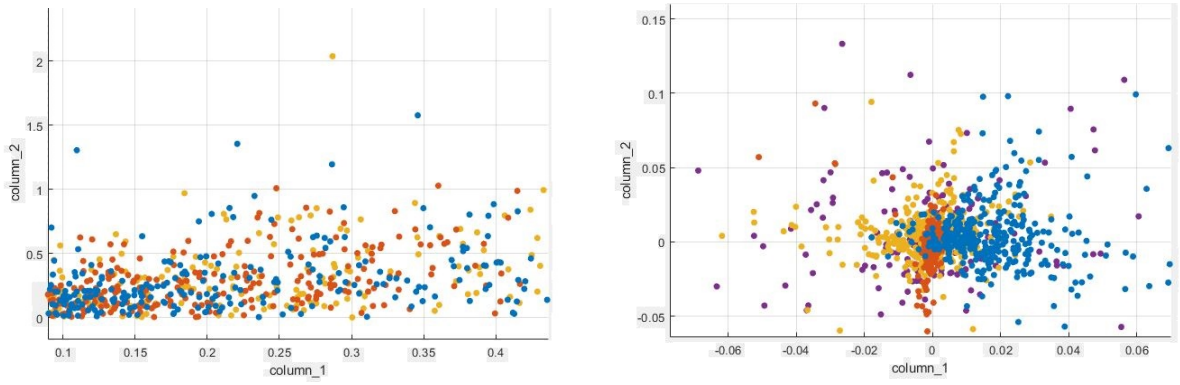


FIGURE 3.14: 200ms EEG data segmentation after filtering

3.4.5 Features Extraction

The power spectral density for the alpha and the beta rhythms is extracted by using the welch method as describe in 2.2.1. The contribution of our work is applying the common spatial pattern (CSP) to the obtained PSD where we computes the maximum ratio of variances between different classes of data by applying filters on it. The One Versus the Rest variant of the csp is applied to the 4 classes of movement. The PSD is divided into different sub-bands to extract information from different portions and then the selection of most relevant features from the extracted features. The Figure 3.15a and 3.15b show the data distribution before and after applying CSP respectively.



(A) PSD data before CSP

(B) PSD data after CSP

FIGURE 3.15: CSP features.

As it can be seen in the results above, the discriminability of the classes increased which made the dots of data more separable. The separability of the data is manifested with an angle between a couple of classes. We can also see that the data of the fourth class possess a high amplitude and the scatter plot distribution is distant from the other data which can be explained by the high energy generated from the EMG signals caused by the movement of the eyes. After applying the spatial filtering and extracting feature from the PSD, relevant features are extracted from the CSP, for each direction of an imagined movement, the variances of only a small number of signals most suitable for discrimination are used for the construction of the classifier. The signals that maximize the difference of variance of a class versus the rest of classes are the ones that are associated with the largest eigenvalues and they are related to the first and last rows of the projection matrix. In the next sections, we discuss the variation of the number of the selected features and the performance of the model in each case.

3.4.6 Feature Classification

For the classification , machine learning classifiers were chosen to classify the selected features. The final architecture of the processing is shown in the Figure 3.16 below:

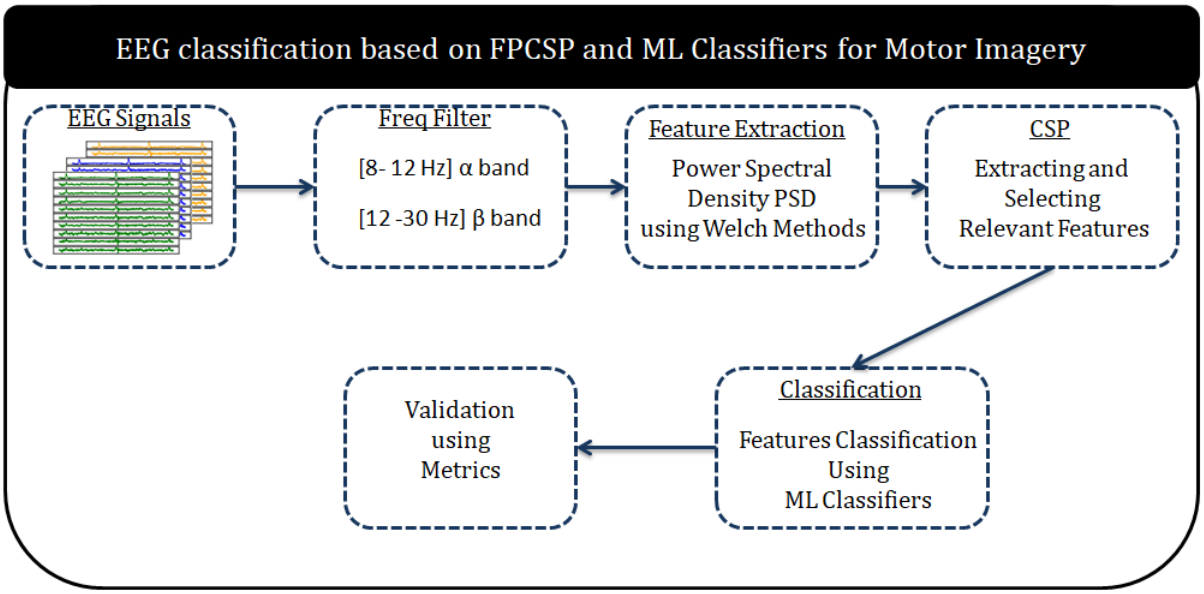


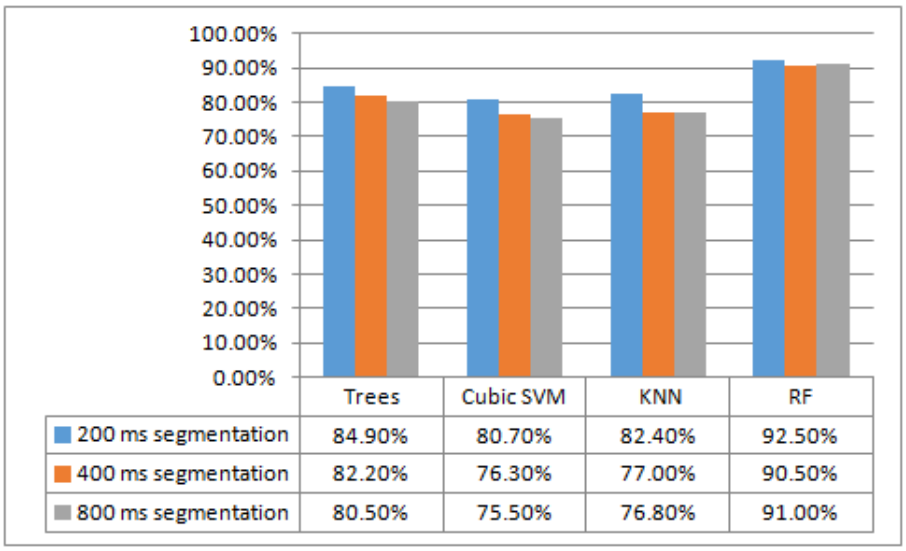
FIGURE 3.16: Offline EEG Processing

3.5 Results and Discussions

The signal and classification process was applied in Matlab 2018b. In order to final the accurate model, two parameters were tuned in order to visualise their influence to the : the segmentation window and the number of the selected features of the OVR CSP , in the other hand , multiple classifier has been chosen and finally validate the best obtained model in order to be test in the real-time.

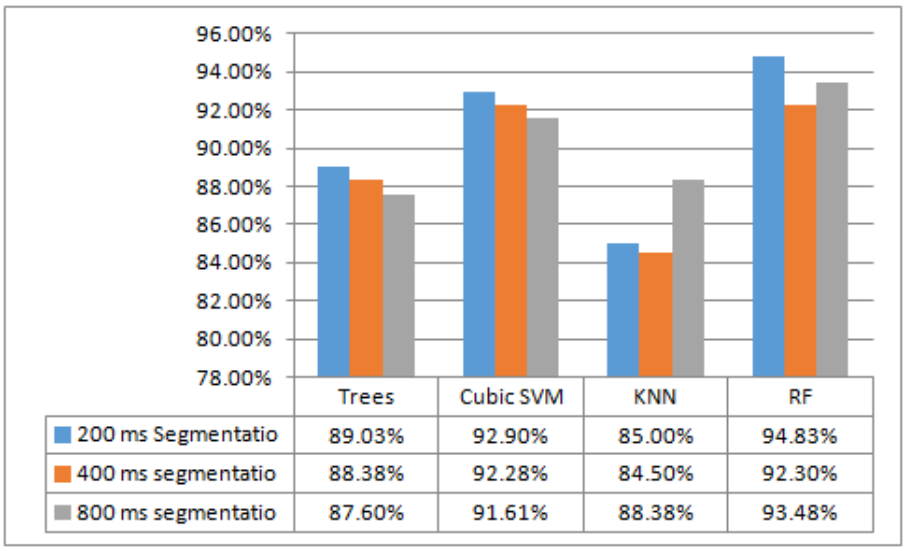
3.5.1 Study of the Influence of the Disjoint Window Length

In order to find the best length of the disjoint segmentation of data , we have chosen three values : 200, 400 and 800 ms as a length of the disjoint window. Four classifiers have been chosen for this study which are : Trees,SVM, KNN And Random Forest, we chose the accuracy as the validation metric, even though the confusion matrix and the Roc curve will be computed. The data is divided into 80% for the train and 20% for the test, the test accuracy is computed for the test data. The validation accuracy is computed using the K-fold cross validation as described in 2.3.3, for the continuity of the implementation we set k=5. The figures below show the test and validation accuracy for the different window's length. The results shown in Figure 3.17 and 3.18 presents the mean accuracy of both subjects



	Trees	Cubic SVM	KNN	RF
200 ms segmentation	84.90%	80.70%	82.40%	92.50%
400 ms segmentation	82.20%	76.30%	77.00%	90.50%
800 ms segmentation	80.50%	75.50%	76.80%	91.00%

FIGURE 3.17: Cross validation accuracy for different window length



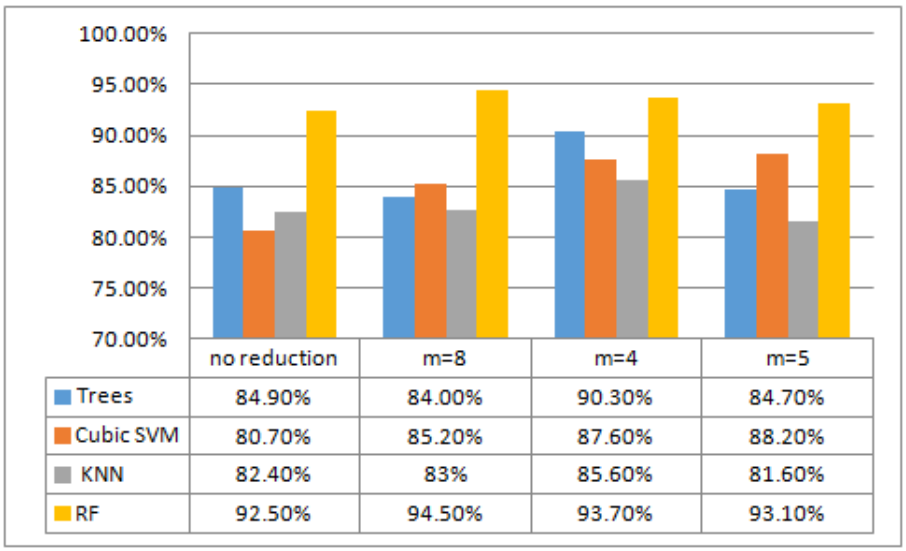
	Trees	Cubic SVM	KNN	RF
200 ms Segmentatio	89.03%	92.90%	85.00%	94.83%
400 ms segmentatio	88.38%	92.28%	84.50%	92.30%
800 ms segmentatio	87.60%	91.61%	88.38%	93.48%

FIGURE 3.18: Test accuracy for different window length

The Histograms below showed the influence of the variation of the length of segmentation window for the performance of the model. For this study, no reduction of the CSP has been considered. As it can be seen , the best accuracy was obtained for a 200 ms segmentation for the majority of the classifiers, as for the classifiers, Random Forest (RF) gave the best results with 87.3% in validation and 90.12 % for the test. The high accuracy of the RF can be explained by the characteristics of the classifier, in facts, Random forest adds extra randomness to the model while growing the tree. Instead of searching for the most important feature when splitting nodes, it searches for the best feature in a random subset of features. This leads to a wide range of diversity, which usually results in better models. As a conclusion, the window length will be set to 200 ms for the continuity of our work.

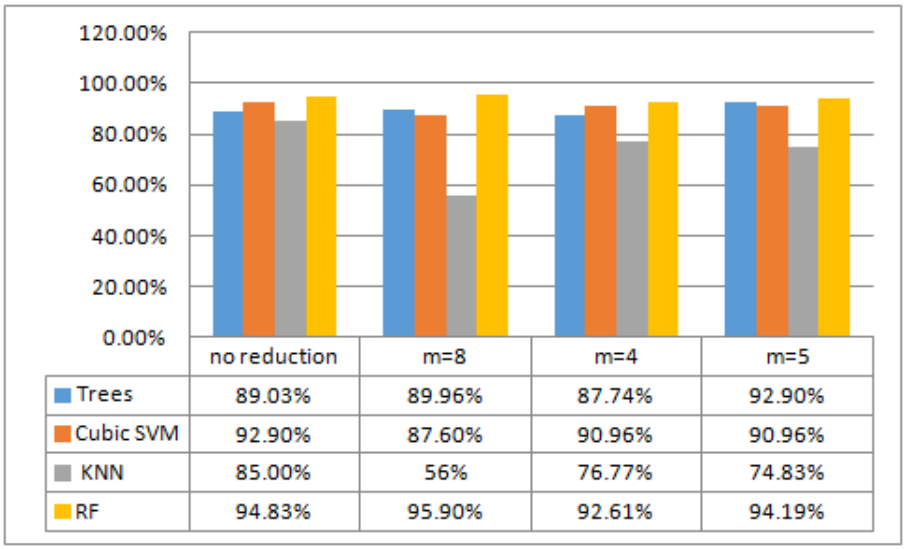
3.5.2 Study of the Influence of the Dimensiality Reduction of CSP

For this study, we tune the number of the number of rows taken from the CSP projection matrix, thus four option are taken : no reduction and m=8,5,4, the results are shown in the Figure 3.19 and 3.20.



	no reduction	m=8	m=4	m=5
Trees	84.90%	84.00%	90.30%	84.70%
Cubic SVM	80.70%	85.20%	87.60%	88.20%
KNN	82.40%	83%	85.60%	81.60%
RF	92.50%	94.50%	93.70%	93.10%

FIGURE 3.19: Cross validation accuracy for different dimension of feature selection



	no reduction	m=8	m=4	m=5
Trees	89.03%	89.96%	87.74%	92.90%
Cubic SVM	92.90%	87.60%	90.96%	90.96%
KNN	85.00%	56%	76.77%	74.83%
RF	94.83%	95.90%	92.61%	94.19%

FIGURE 3.20: Test accuracy for different dimension of feature selection

From the study below, we illustrated the influence of the number of the selected features on the performance of our model. The input of the model is of size 32 in which it can be reduced by selecting the m first and last rows of the transformed signal. The study showed

that selecting the 8th first and last rows leads to an improvement in the RF performance from 87.3% to 90.35% in validation and from 90.12 to 90.98% in test accuracy.

RF Model Characteristics: The random forest has been tuned in order to obtained the suitable parameters that led to better performance, the Table 3.4 shows the parameters of the our RF model:

Parameter	Value
Maximum number of splits	2400
Number of learners	30
Learning rate	0.1
Subspace dimension	1

TABLE 3.4: Random Forests Model Parameters

Confusion Matrix: In the figure 3.21 below, the confusion matrix of the subject 1 is showed by applying the best parameters found above , the accuracy was 92.75 %, and as it can be seen the model has a high accuracy for the third and fourth class and a relative low performance of the first and second class, which caused by the confusion between the right and left movement since it require a high intention in order to separate the movements. The best performance was obtained by the fourth class which can valid the choice of the proposed protocol.

FIGURE 3.21: Confusion matrix

ROC Curve: The ROC curve of the obtained model is shown in the Figure 3.22 below.

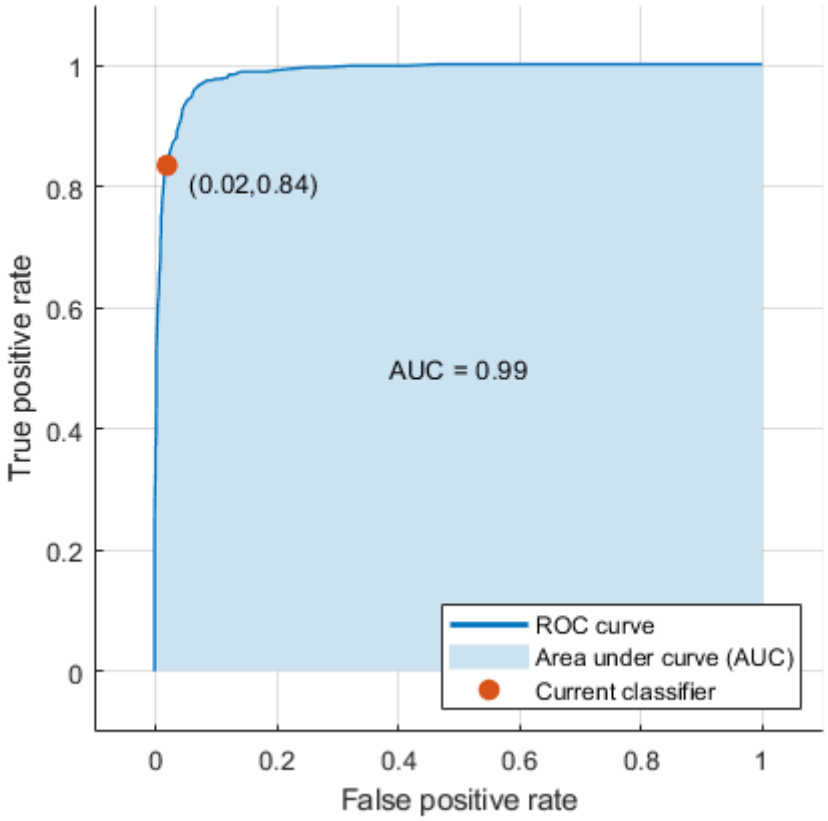


FIGURE 3.22: ROC Curve

3.5.3 BCI Competition IV 2a Dataset Results

As described in the Section above, we tried to implement our architecture to the BCI competition dataset, the table below show the accuracy of our model and a comparison with the best global results is shown in the Table 3.5 below, three cases has been chosen for the implementation, the 4 class imagery, and 3 class : right /left and tongue imagery since it matched with our protocol.

Model	Subj. 1	Subj. 2	Subj. 3	Subj. 4	Subj. 5	Subj. 6	Subj. 7	Subj. 8	Subj. 9	All (Mean)
Cubic SVM (3 classes)	93.5 %	88.8 %	90.09 %	96.07 %	87.56 %	90.81 %	95.02 %	94.7 %	93.4 %	92.22 %
Random Forests (4 classes)	82.4 %	77.95 %	87.6 %	80.8 %	79.4 %	83.61 %	86.74 %	84.91 %	86.42 %	83.31 %
Random Forests (3 classes)	90.5 %	87.3 %	91.12 %	95.6 %	90 %	90.04 %	91.5 %	94.8 %	93.77 %	91.59 %

TABLE 3.5: BCI Competition IV 2a results

The best result obtained by using the random forest classifier for the 4 movement class and cubic SVM for the 3 movement class, for this analysis we have applied with a 1s

segmentation with 8 vectors selected from the CSP matrix. As it can be seen, the model showed satisfactory results for the majority of the subjects, however The models trained on subjects 2 and 5 respectively have shown a poor accuracy due to the high amount of noise contained in their respective data, which confirms the statement in [27].

In the figure 3.23 we compare the performance of our model to the best obtained models in the literature, in which we see a good performance of our model

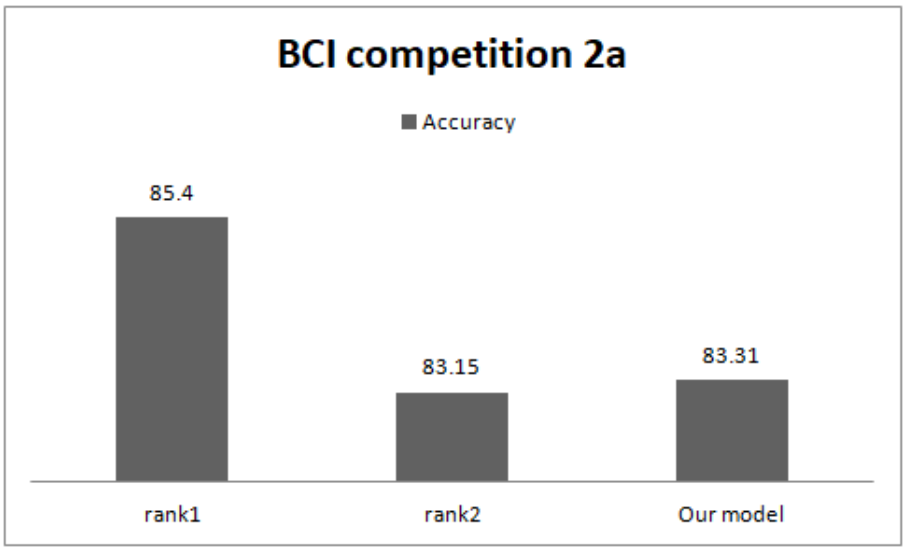


FIGURE 3.23: Comparison of BCI competition IV 2a results

3.6 Online Implementation

In order to validate the obtained results and test the model in a closed loop, starting from signal acquisition to the features extraction and classification , the output of the model presents the command of our mobile robot which in our case is the e-puck 2 mobile robot and a visual feedback is provided by the user as it is demonstrated in the Figure 3.24 below. The robot is implemented in Webot simulator which can also transfert the commands to a real robot using Bluetooth or WiFi connexion.

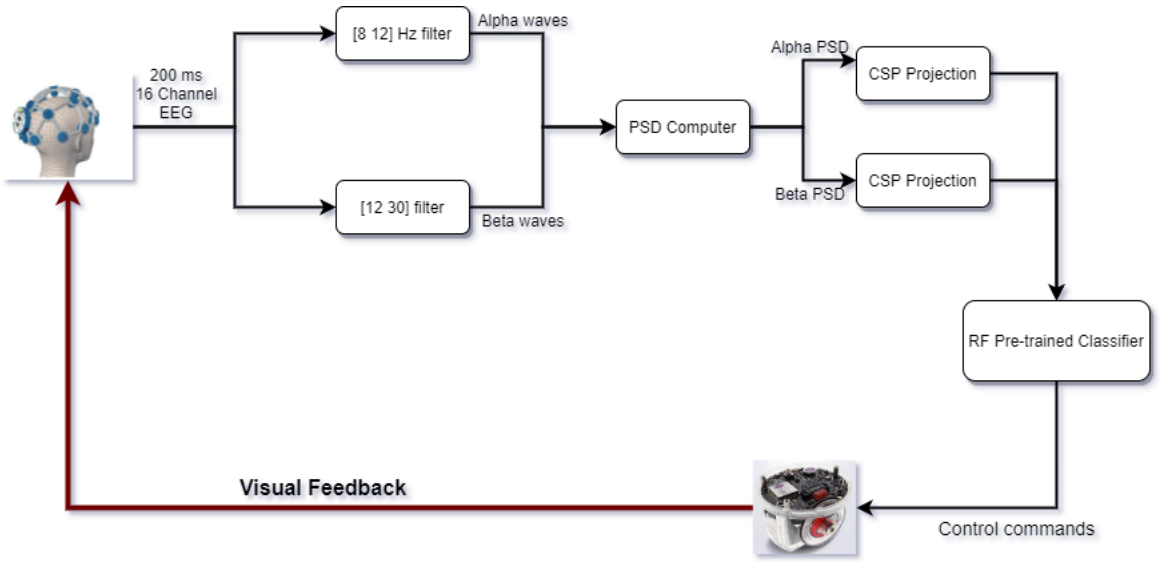


FIGURE 3.24: Closed loop diagram

For the control of the robot with the best model obtained in the classification step, we have developed a new code in MATLAB whose flowchart is shown in Figure3.25. First, a step of online acquisition of the EEG signal is performed using the Lab Streaming Layer (LSL) library and the openBCI GUI software. The sampling frequency chosen during the analog/digital conversion of the signal is the same during the training phase, i.e. 150 Hz and the recording window of the 16-channels signals is fixed at 200 ms. Then, a filtering stage is carried out for the elimination of the various noises and the extraction of the α and β wave, this is carried out by a programmed function which allows the application of a butterworth band-pass filter with respective frequencies [8-12] Hz and [12-30] Hz. Following the filtering step, the power spectral density is computed for both waves which creates a matrix of (25*32), the next step , the CSP features are computed by projection the PSD to the CSP matrix. The latter is the input of the pre-trained Random Forest model that will estimate the class of movement performed by the user.

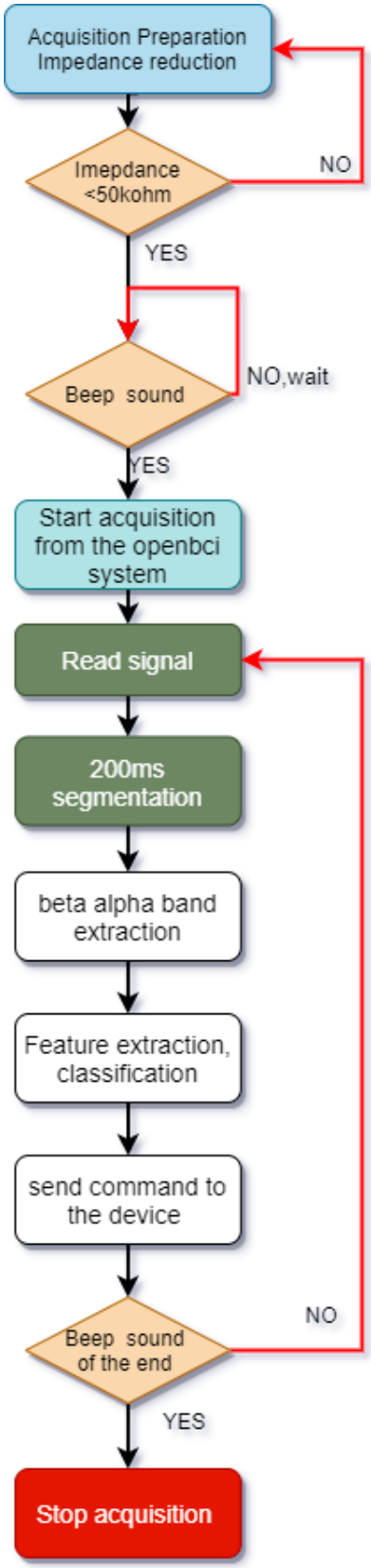


FIGURE 3.25: Real time Flow chart

For the implantation, we proposed a trajectory as it can be seen in the Figure 3.26 below, the user needs to provide real time commands to the robot with the aim of moving the robot from the initial position to the proposed final position passing by the designated path.

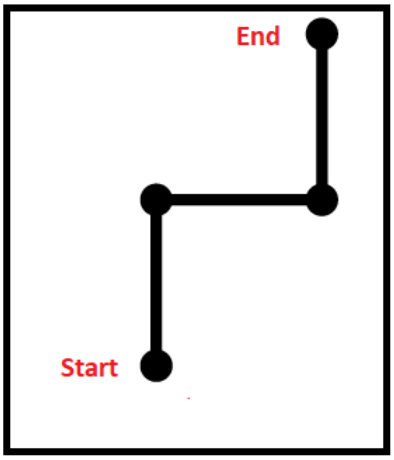


FIGURE 3.26: Proposed trajectory

Two scenarios are proposed for the experiment as shown in the figure 3.27. The operator had issues regarding the realisation of successive imagination which leads to lack of performance, as a consequence the second scenario was proposed to overcome the consuciotivity of movements.

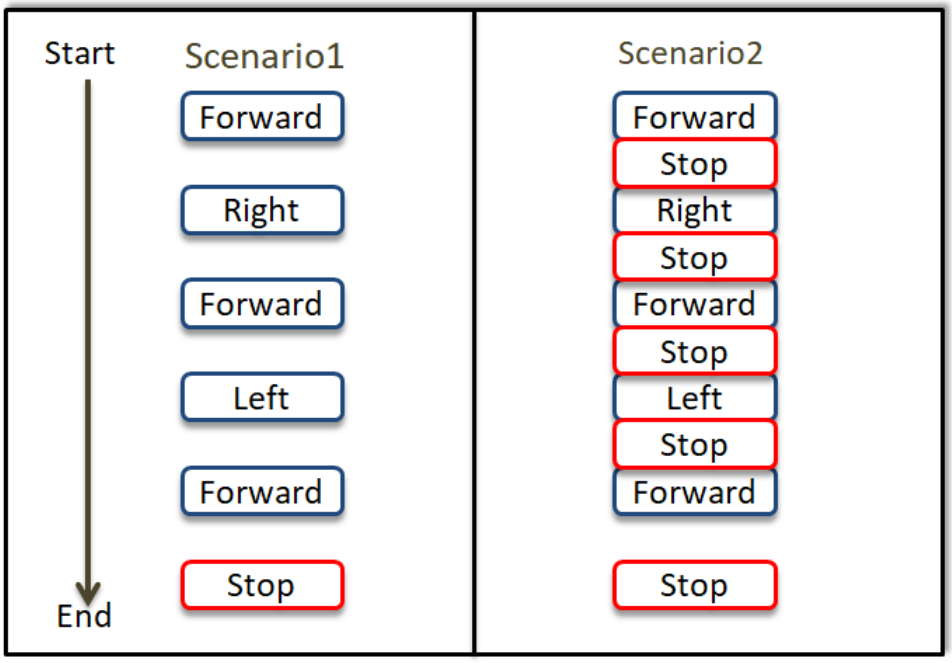


FIGURE 3.27: Real time scenarios

Time management The Table 3.6 below shows the required time for model to be trained and also the the needed time for the pre-trained time to delivered the command regarding the input signal.

TABLE 3.6: Processing timing.

Model	Trained Time	Pre-Trained
RF	12s	99ms
SVMcubic	11.2s	86.3ms

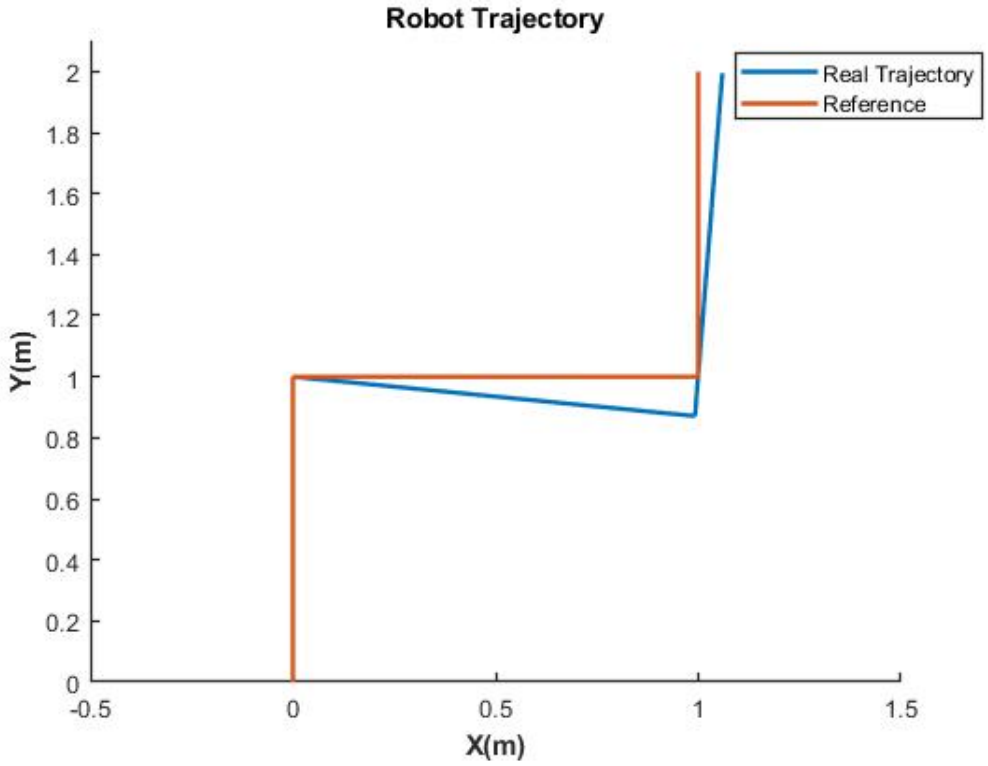


FIGURE 3.28: Robot Trajectory

Discussion The implementation of our model showed a satisfied results and performance regarding the complex nature of the EEG signal and the difficulties of real time realisation. As it can be seen from the Figure 3.28 the model showed a good performance in the forward movement and the stop movement. For the right class, the robot made a rotation angle bigger than the required angle (90 degree); this angle error can be explained by the visual difficulties of such movement and the required time to provide a perfect 90 degree rotation, we also remind that the robot is controlled only in velocity and the command provided from the user are quantitative commands since we can't provide an exact reference angle to the robot.

3.7 Conclusion

Along this chapter, we detailed the implementation steps of the Brain Computer Interface realization. We have also evaluated the obtained results of our proposed model. The evaluation process is done using both the online "BCI COMPETITION IV 2A" dataset and our acquired private dataset which is used for real time control.

For the online dataset, we achieved a state-of-the art performance with minimum training time and only by applying Machine Learning algorithms. For better performance, we can suggest individual subject study for Feature Selection and to extract other relevant features in order to improve the obtained results.

As for our private dataset, we have obtained promising results with a mean of 90% for the offline implementation, our model has been validated online where the output of our model was considered as the commands of the e-puck robot.

These results are obtained for subject candidates in alert state. However, if a user is tired, the proposed approach may give wrong results. Thus, we have targeted to integrate an additional module for fatigue detection.

In the next chapter, we will design a fatigue detection approach based on EEG signal using both machine learning and deep learning algorithms.

4 EEG Signals Classification for Fatigue Detection

Abstract

In this chapter, we develop the implementation process for drowsiness/fatigue detection. In Section 4.1, we review the various methods applied for fatigue detection, Section 4.2, details our proposed experimental protocol for data acquisition as well as the description for the public dataset used. In Section 4.3, we resume the application and results of the methods discussed in Chapter 2 for fatigue detection; in Subsection 4.3.1 we discuss the validity of the extracted features, in Subsection 4.3.2 we review the application of the previously discussed Pre-processing techniques. In Subsections 4.3.3, 4.3.4, and 4.3.5, we display the obtained results of the ML and DL algorithms on their respective datasets, and in Subsection 4.3.6, the results of Dimensionality Reduction are discussed. Finally, We close off this chapter with a conclusion 4.4.

4.1 Introduction

In the literature, two path of processing has been considered for the detection of drowniess, in 2, we have detailed the features extraction methods and machine learning algorithms that can be used for classification. In [50] the FFT of θ, α, β bands were used as part of a study on their proportionality with the states of fatigue and alertness, in [112], the same was been done with $(\theta+\beta)/\beta, (\beta/\alpha), (\theta+\alpha)/(\beta+\alpha)$ *and* $(\theta/(\beta))$ ratios as part of a study of the effect of fatigue on EEG spectrum, Independent Component Analysis (ICA) was used for artefact removal .in [90] CSP filter was applied to the pre-process EEG data and the result feature were fed to LDA classifier. In [108] Shannon entropy was computed on the EEG signal and then classified using PCNN classifier, the model was validated in real-time implementation. In [65], a comparison study of the SVM,KNN ,Random forest (RF) algorithms was made to classify the Shannon entropy of the EEG signals in both states (fatigue and non fatigue).

Recently, deep learning (DL) techniques have demonstrates strong capabilities in time series classification [60] With a main focus on the application of deep learning techniques to the classification of EEG signals[37]. These DL-based methods have achieved effective

classification results in various EEG-based recognition tasks, but still face many problems. Extracted features from deep learning technology are difficult to explain and analyze, and the network itself is also difficult to train the number of parameters is large, and the network depth is high. Deep learning technology still needs further research and development, including their Training efficiency and internal mechanism description.

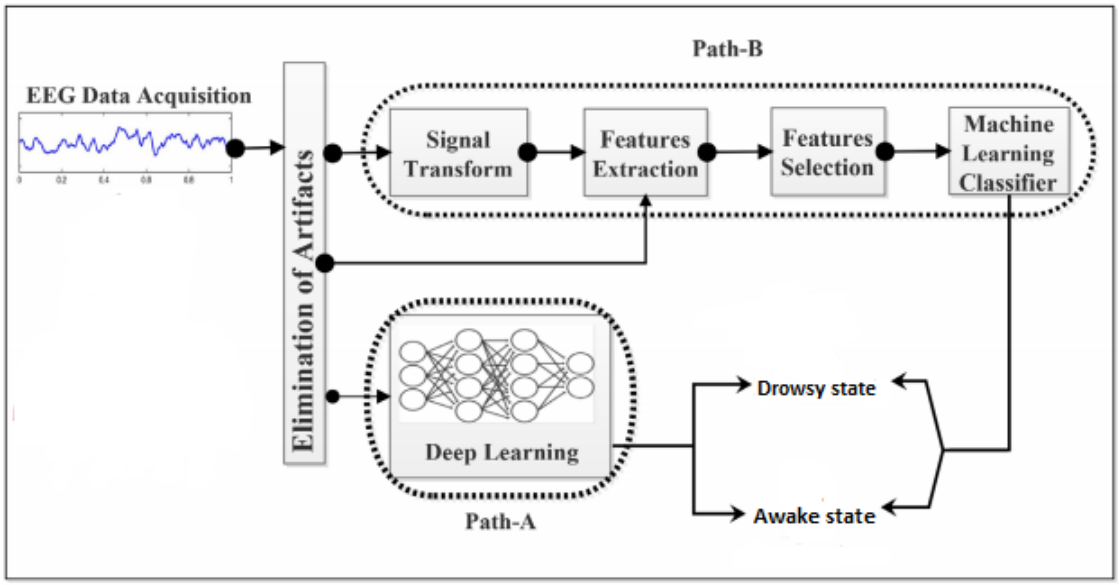


FIGURE 4.1: EEG processing scheme.

In DL methods (represented by path B Figure 4.1), raw EEG information was fed to the network in order to obtain the state of the subject [18], in [118] the drowsiness was detected using convolutional neural network (CNN) where the network identifies automatically the features of the first path and classify in two states. In [55], CNN and long short-term memory (LSTM) were both developed for offline drowsiness detection. Artifact removal present a major step for pre-processing of EEG data and it leads to performance improvement for the majority of the studies, in [41] ICA was applied to the EEG data in order to remove signal artifacts and extract the source components and the transformed signal was fed to the CNN network, and in [36] the source components were fed to the recurrent neural network RNN for drowsiness detection.

4.2 Dataset Description

4.2.1 Private Dataset

Workstation

The model were trained and implemented using a working station based on an I7-6700K Intel processor, an Nvidia GTX 1080 GPU and 32 GB Ram, and SSD hard drive. The station is available in the $L(CS)^2$ with a triple Full-HD monitors and is also used for flight simulation.

Experimental Protocol

9 male subjects (22-43 years old) were selected from the $L(CS)^2$ laboratory. A quick questionnaire was established to make sure the subjects have no medical contraindications such as a prior brain injury, use of prescription medication, severe concomitant or any psychological ,minor psychiatric disorders or any intellectual problems. For a proper acquisition, the subjects have to be in an alert state defined with a good sleep quality and a minimum mental activity. They were also requested to stop drinking caffeine such as cafe, tea or any stimuli 24h before the acquisition.

The participants received a comprehensive instruction about the protocol and the required tasks. Each participant needs to be trained to use the Xplane 11's based $L(CS)^2$ flight simulator and the attached command devices as it can be seen in the Figure 4.2. The motif for studying the driven or pilot fatigue is that the flight simulator as well as the drive simulator provides a good inducing of fatigue. The state of fatigue and alert state are defined as follows:

Alert state:

- The participant should have a minimum of 8 hours of sleep.
- The participant should have some previous experience with the flight simulator.
- The acquisition should occur in the morning.

Fatigue state:

- The operator should have from 4 to 5 hours maximum of sleep.
- The participant should operate withe the flight simulator for at least 30 minutes.
- The acquisition should occur in the afternoon.

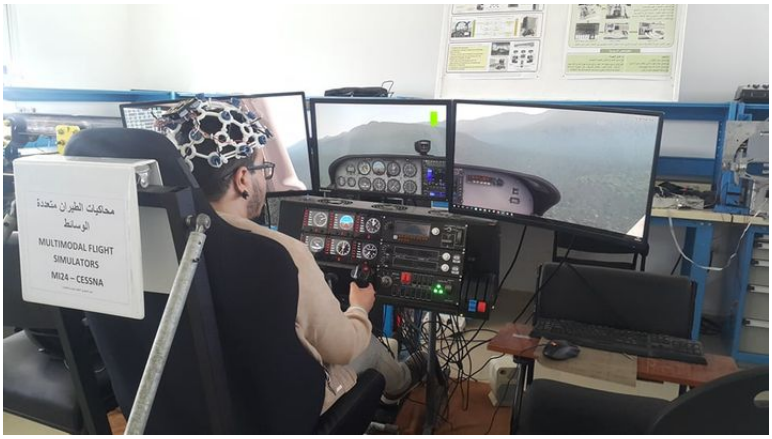


FIGURE 4.2: Pilot fatigue protocol

Electrode Placement

The effect of fatigue can not be seen in a specific area of the brain which require a collection of EEG data from the globality of the brain, features selection methods can

be used for the selection of electrodes related to fatigue detection. In our work, we use the Ultracortex Mark IV headset described in 3.2.1, as for the electrodes' placement, the configuration is shown in the figure below:

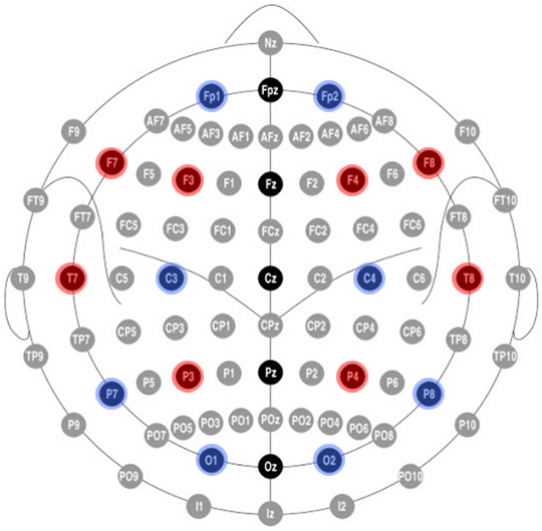


FIGURE 4.3: 10-20 system for fatigue detection

4.2.2 Public Dataset

In order to validate the obtained models' results, the "Min-Wang-Hu" public dataset (avaiable online)[1] has also been used in tandem with the private dataset. It contains EEG data that has been collected from twelve healthy subjects aged from 1-24 able-bodied persons with normal sleep schedules. Prior to the experiment, the participants practiced driving for 5 minutes in order to become acquainted with the experimental procedure. All experimental procedures were performed with a static driving simulator in a controlled lab environment, as shown in Figure 4.4 below:

FIGURE 4.4: Acquisition protocol of the Min Wang Hu dataset

[1]Source: https://figshare.com/articles/dataset/The$_{o}riginal_{E}EG_{d}ata_{f}or_{d}river_{f}atigue_{d}etection$/5202739

The driving simulation involves two protocols for each state, the first one lasts 20 minutes, with the last 5 minutes of EEG signals being recorded and labeled as the alert state, the second one lasts 40–100 minutes until the self-reported fatigue questionnaire results indicate that the subject is in a fatigue state, the last 5 minutes of EEG signals are then recorded and labeled as the fatigue state. All channel data were referenced to two electrically linked mastoids at A1 and A2, digitized at 1000 Hz from a 40-channel electrode cap based on the International 10–20 System and stored in a computer for offline analysis. Eye movements and blinks were monitored by recording horizontal and vertical EOG signals[65].

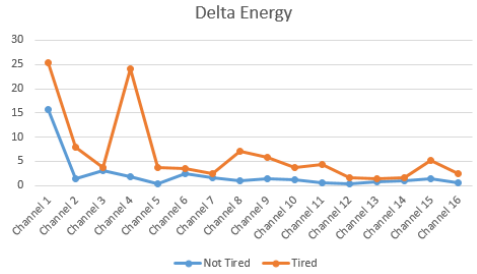
4.3 Results and Discussions

4.3.1 Energy and Fatigue Correlation Study

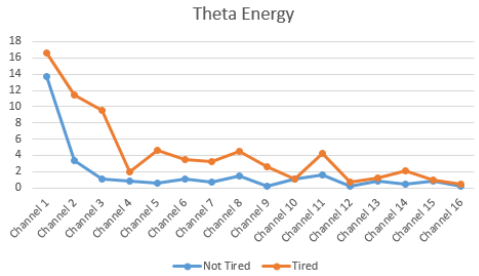
Frequency Band Energy Estimations

In order to be able to use the various frequency bands' PSD estimation for linear fatigue detection, a correlation between these energy estimates and the states of fatigue and non fatigue must be made. In order to do this, Energy estimations belonging to subject 3 were calculated for all frequency bands of two samples representing the two states, knowing that each sample has 16 channels, we can then observe the changes between the Energy values as the subject changes gets fatigued for each of the channels individually.

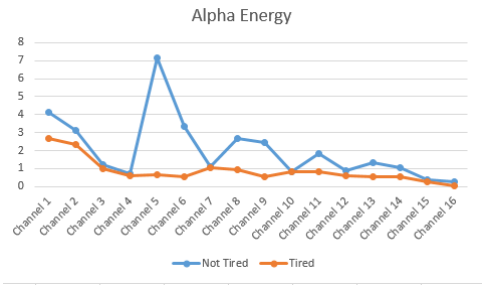
The results are illustrated in the figures below:



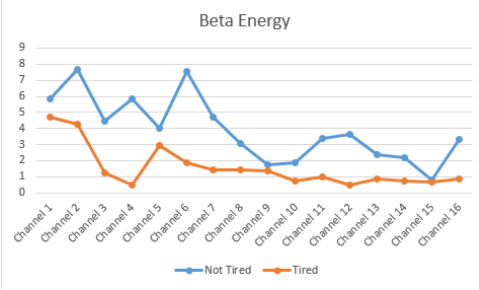
(A) δ Band Energy.



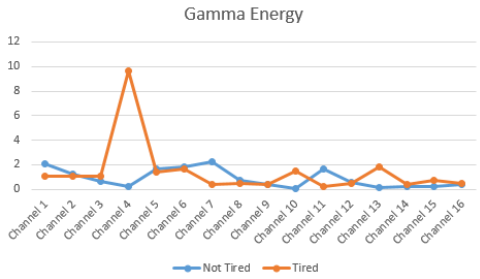
(B) θ Band Energy.



(C) α Band Energy.



(D) β Band Energy.



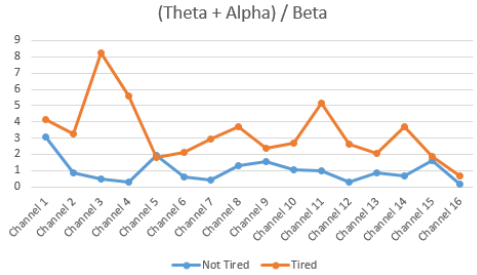
(E) γ Band Energy.

FIGURE 4.5: The four Band Energy estimations in both fatigue and alert states.

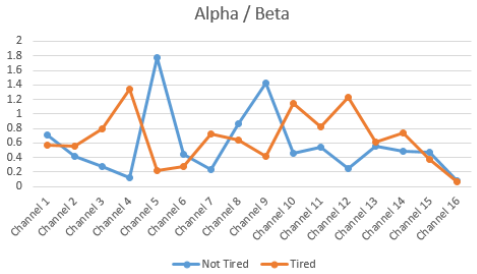
By observing the illustrations, we can deduce a correlation between all four frequency bands' energies and the state the subject is in, the Delta and Theta energies increase with the transition to fatigue, while the Alpha and Beta energies decrease, the Gamma band's energy on the other hand does not reflect any noticeable pattern. This is explained by the fact that Delta waves reflect deep sleep and Theta waves reflect light sleep, while Alpha waves reflect concentration and Beta waves reflect alertness (this has been detailed in 1.2.2).

Energy Ratios

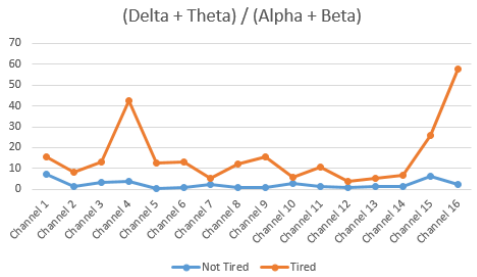
The same Energy ratios $((\theta+\beta)/\beta, (\beta/\alpha), (\theta+\alpha)/(\beta+\alpha)$ *and* $(\theta/(\beta)))$ have consistently been used in [50], [112], and others, as part of a simple fatigue detection study. The validity of these ratios will be tested simply using the resulting energy estimations from the previous part. The results are shown in the figures below:



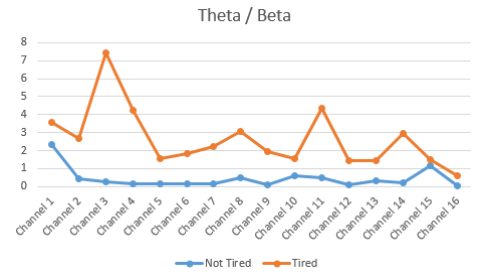
(A) $(\theta+\alpha)/\beta$.



(B) α/β



(C) $(\delta+\theta)/(\alpha+\beta)$.



(D) θ/β.

FIGURE 4.6: Different Band Energy ratio estimations in both fatigue and alert states.

Only three out of the four ratios are noticeably proportionate to the subject's state, which are $(\theta+\alpha)/\beta$, $(\delta+\theta)/(\alpha+\beta)$, and θ/β, all of whom notice an increase with the increase of fatigue, α/β on the other hand, gives inconsistent results, and fails to convey the transition in a meaningful way.

Therefore, there is a possibility of predicting noise merely through calculating the ratios $(\theta+\alpha)/\beta$, $(\delta+\theta)/(\alpha+\beta)$, and θ/β, but it remains an unreliable method because of the EEG signals' susceptibility to noise, even with pre-processing and/or source separation.

4.3.2 Pre-Processing Results

In order to make the brute signals (used in 4.3.4 and 4.3.5) and their features (used in 4.3.3) devoid of noise, and more easily learned, pre-processing was done using the EEGLAB toolbox available on Matlab, more information is available in Annex B.

- **Data Selection (Private Dataset only)**
  In the private dataset, due to the impedance inconsistency and the amount of noise both while starting the acquisition process and while finishing it, parts of the acquired data (taken in 50s intervals) had to be removed in a selection process, leaving only the middle 30s. Further tests showed this improved results when using multiple methods.

  This process is assumed to have already been applied to the online dataset, it is for this that it has not been applied for it.

- **Downsampling (Online Dataset only)**
  The online dataset comes in a sampling frequency of 1000Hz, making it too voluminous for pre-processing and training, it has thus been downsampled to 125Hz, a frequency similar to that of the private dataset.
- **Filtering**
  A 3rd order bandpass Butterworth filter in the range of [1Hz, 40Hz] has been used to filter the data's noise and artifacts, which are apparent at and around the 50Hz line noise.
- **Independent Component Analysis (ICA)**
  ICA was applied using EEGLAB to both datasets to remove artifacts resulting from various sources as well as to apply Source Separation, which is more prevalent the more electrodes there are.

  In the case of the online dataset, the resulting features were reduced from 40 channels into 33 ones because of the nature of the dataset, this is not the case for the private dataset, its end result has 16 channels.

4.3.3 Machine Learning Classification Results

To obtain results for the 4 ML methods mentioned previously, features had to be extracted from the resulting data from pre-processing. Samples are taken using a window size of 125Hz which corresponds to one second, and four out of the five Energy estimations are calculated (δ, θ, α, β), as well as three of the four ratios studied previously ($(\theta+\alpha)/\beta$, $(\delta+\theta)/(\alpha+\beta)$, and θ/β), and the Entropy. The γ band Energy and the α/β ratio were omitted as features due to their poor performance, which was confirmed with a trial and error process. The resulting feature selection is a fusion between band energies, band energy ratios and Entropy.

The result is then reorganized in a data transformation that labels the data and makes it easier to loop through the features using Matlab.

Validation and Testing Results

The Validation Accuracy was calculated using 5-Fold Cross-Validation on 70% of the dataset, and the Training Accuracy was calculated on the remaining 30%, using 70:30 Holdout Cross-Validation. The two tables below 4.1, and 4.2 show the obtained Validation and Testing Accuracy results respectively:

Method	Subj. 1	Subj. 2	Subj. 3	Subj. 4	Subj. 5	Subj. 6	Subj. 7	All
SVM	98.30 %	96.30 %	96.20 %	96.40 %	95.20 %	86.90 %	94.10 %	94.77 %
KNN	96.40 %	96.30 %	96.40 %	92.40 %	91.20 %	83.80 %	93.70 %	92.89 %
Decision Trees	97.50 %	87.30 %	93.80 %	79.30 %	96.20 %	76.20 %	93.10 %	89.06 %
Random Forests	99.20 %	95.50 %	95.70 %	93.30 %	95.70 %	88.90 %	95.80 %	94.87 %

TABLE 4.1: Machine Learning Validation Results

Method	Subj. 1	Subj. 2	Subj. 3	Subj. 4	Subj. 5	Subj. 6	Subj. 7	All
SVM	93.17 %	98.77 %	93.89 %	93.79 %	90.56 %	86.42 %	95.11 %	93.10 %
KNN	93.17 %	95.68 %	96.11 %	94.41 %	88.33 %	81.89 %	94.67 %	92.04 %
Decision Trees	92.55 %	88.89 %	92.22 %	72.67 %	97.22 %	76.54 %	91.56 %	87.38 %
Random Forests	94.41 %	96.91 %	94.44 %	93.79 %	91.67 %	87.24 %	96.44 %	93.56 %

TABLE 4.2: Machine Learning Testing Results.

Resulting Model Parameters

Parameter	Value
Training Environment	i7-7600K
Dataset size after pre-processing	15360 - 103680 (Total of 622208)
Model	SVM
Kernel	Cubic
Model	KNN
Number of neighbors	1
Distance	Euclidean
Normalization method	Min-Max Normalization
Model	Decision Trees
Max number of splits	100
Loss Function	Gini Impurity
Model	Random Forests
Learning Rate	0.1
Max number of splits	356
Number of learners	30

TABLE 4.3: Resulting Model Parameters.

4.3.4 Deep Learning Classification Results (Private Dataset)

To test the efficiency of 1D-CNN and LSTM in analyzing and learning from temporal data, the resulting data from pre-processing had to be reorganized in data formats that can be inputted into either 1D-CNN and LSTM. The training process is done using Matlab.

Testing Results

The Training Accuracy was calculated using 70:30 Holdout Cross-Validation. Table 4.4 below shows the obtained results:

Method	Subj. 1	Subj. 2	Subj. 3	Subj. 4	Subj. 5	Subj. 6	Subj. 7	All
1D-CNN	94.77 %	95.68 %	95.56 %	94.77 %	91.67 %	77.78 %	98.67 %	92.70 %
LSTM	96.08 %	77.78 %	92.78 %	71.24 %	88.33 %	61.32 %	97.33 %	83.55 %
bi-LSTM	94.77 %	79.01 %	95 %	79.08 %	92.22 %	61.73 %	99.56 %	84.43 %

TABLE 4.4: Private Dataset Deep Learning Training Results.

Hyperparameter Tuning

All three models' Hyperparameters were tuned using a trial and error process that involved generating multiple layers of different types and sizes in order, and training them with a fixed set of training Hyperparameters, once the architecture with the best result is found, the same process is then applied to training Hyperparameters, until a good combination that is a good fit for the data is found.

1D-CNN Model

The 1D-CNN Model's architecture involves a 1D image input, one Convolution Layer followed by a ReLU Activation Function, a Max Pooling Layer, then a Fully Connected Neural Network composed of 2 layers, the output is passed through a Softmax Activation Function with a classification threshhold of 0.5.

The model's architecture parameters and Hyperparameters are shown in Table 4.5 below:

Parameter	Value
Training Environment	i7-7600K & GTX 1080
Dataset size after pre-processing	240000 - 1620000 (Total of 9722000)
Learning rate	0.001
Number of epochs	30
Batch size	5
Optimizer	Adam
Input size	[1, 125, 16]
Convolution Layer number of filters	96
Convolution Layer filter size	[1, 3]
Max Pooling Layer window size	[1, 5]
Fully Connected Layer 1 number of neurons	32
Fully Connected Layer 2 number of neurons	2

TABLE 4.5: Private Dataset 1D-CNN Model Parameters and training options.

LSTM and bi-LSTM Models

Both LSTM models' architectures involve a sequence input, followed by an LSTM/bi-LSTM layer, outputting directly into a Fully Connected Neural Network composed of 2 layers, the output is passed through a Softmax Activation Function with a classification threshhold of 0.5.

The model's architecture parameters and Hyperparameters are shown in Table 4.6 below:

Parameter	Value
Training Environment	GTX 1080
Dataset size after pre-processing	240000 - 1620000 (Total of 9722000)
Learning rate	0.001
Number of epochs	20
Batch size	5
Optimizer	Adam
Input size	[16, 125]
LSTM/bi-LSTM Layer number of neurons	16
Fully Connected Layer 1 number of neurons	32
Fully Connected Layer 2 number of neurons	2

TABLE 4.6: Private Dataset LSTM and bi-LSTM Model Parameters and training options.

4.3.5 Deep Learning Classification Results (Online Dataset)

To test the efficiency of 1D-CNN and LSTM in analyzing and learning from temporal data, the resulting data from pre-processing had to be reorganized in data formats that can be inputted into 1D-CNN and LSTM ([1, 125, 16], and [16, 125] respectively). The training process is done using Matlab.

To test the architectures in use as well as to compare results to others in the literature, the "Min-Wang-Hu" dataset is used and the same data organization process is applied as in the previous section. The end architectures are trained on Matlab.

The same model architectures as used in 4.3.4 were used, only their parameters and Hyperparameters differ. The same trial and error method of tuning them was used.

Testing Results

The Training Accuracy was calculated using 70:30 Holdout Cross-Validation. Table 4.7 below shows the obtained results:

Method	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6
1D-CNN	100.00%	96.11%	100.00%	100.00%	100.00%	93.89%
LSTM	97.22%	98.89%	96.11%	99.44%	93.33%	95.56%
bi-LSTM	99.44%	98.89%	92.22%	99.44%	95.00%	90.00%
Subject 7	Subject 8	Subject 9	Subject 10	Subject 11	Subject 12	All
98.33%	100.00%	96.11%	99.44%	100%	99.44%	98.61%
93.89	98.89%	96.11%	93.33%	95.56%	95.56%	96.16%
94.44%	96.11%	94.44%	95%	95%	96.11%	95.51%

Table 4.7: Online Dataset Deep Learning Training Results.

1D-CNN Model

Parameter	Value
Training Environment	i7-7600K & GTX 1080
Dataset size after pre-processing	2475000 - 2483250 (Total of 29724750)
Learning rate	0.001
Number of epochs	40
Batch size	10
Optimizer	Adam
Input size	[1, 125, 33]
Convolution Layer number of filters	96
Convolution Layer filter size	[1, 3]
Max Pooling Layer window size	[1, 5]
Fully Connected Layer 1 number of neurons	64
Fully Connected Layer 2 number of neurons	2

Table 4.8: Online Dataset 1D-CNN Model Parameters and training options.

LSTM and bi-LSTM Models

Parameter	Value
Training Environment	GTX 1080
Dataset size after pre-processing	2475000 - 2483250 (Total of 29724750)
Learning rate	0.001
Number of epochs	60
Batch size	10
Optimizer	Adam
Input size	[33, 125]
LSTM/bi-LSTM Layer number of neurons	33
Fully Connected Layer 1 number of neurons	64
Fully Connected Layer 2 number of neurons	2

TABLE 4.9: Online Dataset LSTM and bi-LSTM Model Parameters and training options.

Result Comparison

Referring to the obtained performances, our model showed a satisfied results for both private and online dataset using both machine and deep learning algorithms.

Obtained results using the Min-Wang-Hu dataset are compared with the other performances in the literature using the same dataset. Min et al. [65] applied a multi entropy fusion for fatigue detection, Yin et al. [115] proposed a fatigue detection model based on entropy fusion analysis, Nugraha et al. [72] proposed a fatigue detection model based on EEG using emotive EPOC+, and Zhang et al. [120] using entropy and complexity measure for fatigue detection. The accuracy for each model is shown and compared in Figure 4.7 below:

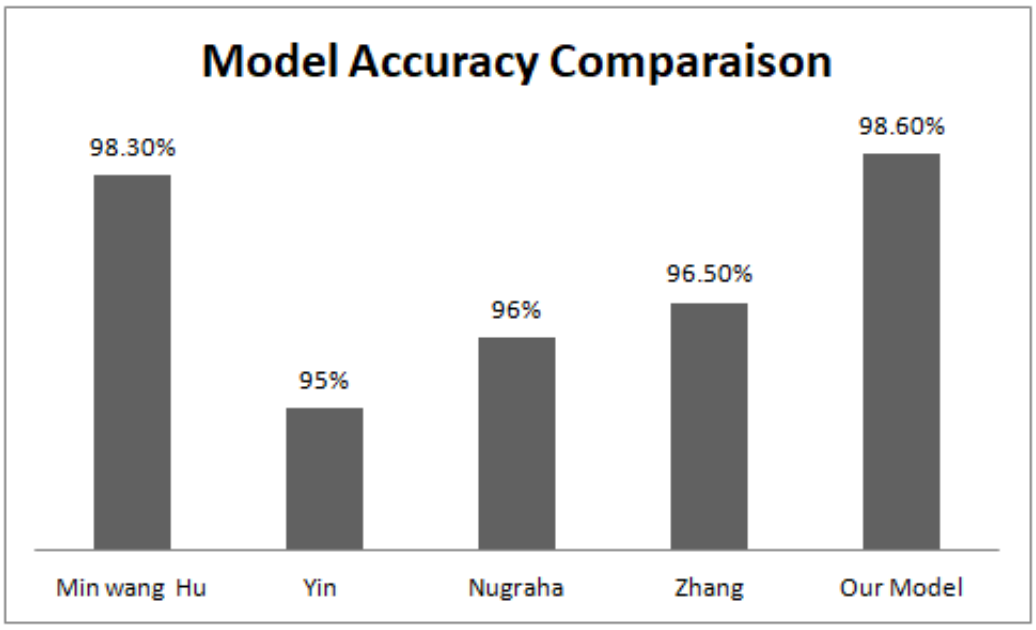


FIGURE 4.7: Model Comparaison.

4.3.6 Dimensionality Reduction Results

In order to determine the brain reginos that are the most responosible for conveying the states of fatigue and alertness, the Genetic Algorithm is applied on the previously pre-processed private dataset,

The GA has been realized on Matlab as one central algorithm employing a group of smaller functions dedicated to every task in the cycle, it is more computationally demanding than Deep Learning because each iteration involved training an entire population of DL models.

Fitness Function

In order to quantify an individual's fitness, the genome must be transformed into an accuracy. This is what the created Fitness Function does. Given the binary sequence, it removes the features corresponding to a 0 and reconstitutes the data, lowering the data's number of channels, training a ML/DL model on this data, and outputting its testing Accuracy.

Due to its performance being the best among the used architectures, 1D-CNN is used in the Fitness Function, thus the dataset must be reorganized inside the Fitness Function into 1D images of [1, 125, 16] dimensions, and the resulting Accuracy is calculated using 70:30 Holdout Cross-Validation.

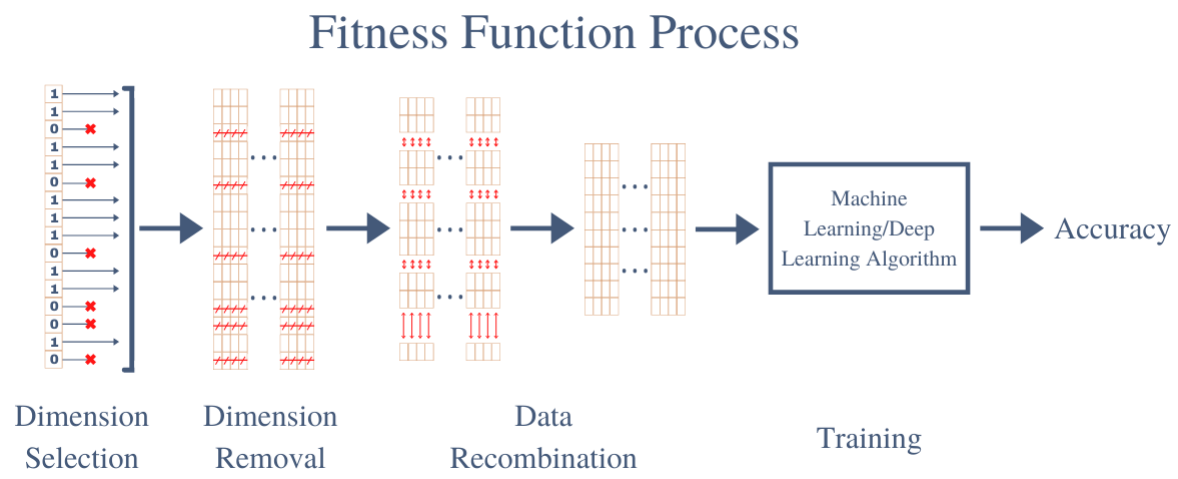


FIGURE 4.8: Fitnes Function Process.

Genetic Algorithm Parameters

The Genetic Algorithm has various parameters whose functions are described in 2.5.2, the criteria for tuning them are as follows:

- **Population size**
  A well chosen Population size helps the GA converge, it needs to be big enough to allow diversity in the Population, but not too big so as to require too much computational power.
- **Selection Tournament parameter**
  This parameter should not be as low as to make the chosen parents relatively unfit, and not high enough so that the same parents are chosen everytime, stagnating the process.
- **Crossover probability**
  The Crossover probability mostly depends on the application at hand, a trial and error process was then used to tune it.
- **Mutation probability**
  The Mutation probabiliy needs to be very low, so as not to destroy the solution by distorting the population too much and diverging the algorithm.

Based on the crieria and [111] and [45], the GA parameters were concluded. They are summed up in the following table:

Parameter	Value
Number of iterations	15
Genome Length	16
Population Size	20
Selection Tournament parameter	3
Crossover Probability	0.9
Mutation Probability	0.05

TABLE 4.10: Genetic Algorithm parameters.

Results

The Genetic Algorithm was applied to the data until it gained maximal accuracy by the 15th iteration, seeing as the testing process is stochastic by nature, it would give slightly different results every time. Thus, the process was applied multiple times for every subject, and the different combinations associated with each one were combined and averaged to obtain the regions that display fatigue the most for every subject. These results are resumed in Table 4.11 below:

Subject	Selected Electrodes/Brain Regions
1	O1, O2, F4, T7, P3
2	Fp2, C3, C4, O1, O2, F7, F8
3	Fp1, O1, F3
4	Fp1, C4, P7, F3, F4, P3
5	C3, C4, P4
6	C4, P7, P8, O1, O2, P3, P4
7	C3, C4, P7, F3, T4, P3

TABLE 4.11: Dimensionality Reduction Results.

The results show a strong Occipital, Frontal, and Parietal region influence in most subjects. Which is explained when analyzing the individual regions' functions, the Occipital lobe is associated with visuospatial processing, depth and distance perception, object and face recognition and memory formation, all of which are affected when in a state of fatigue. The Frontal lobe is responsible for voluntary movement and language, as an individual transitions into a state of fatigue, both of those functions slow down and become harder to control. And the Parietal lobe interprets the sensory world, the distortion of which is very common when fatigued.

4.4 Conclusion

Detection of fatigue based on EEG signals requires Pre-processing and Feature Extraction. In this chapter, we have studied the effects of drowsiness on the EEG power spectrum, the obtained result is then merged with entropy and considered as the input for the Machine Learning classifiers, we have obtained a good accuracy of 94% for the SVM and Random Forests classifiers.

Deep Learning methods are inspired from real life, and serve as a useful tool in handling EEG signals and a variety of data, but introduce a new level of complexity into customizing the model. Using them removes the need for Feature Extraction as they are capable of analyzing the temporal signal directly and classifying it, in our work we have designed 1D-CNN, LSTM and Bi-LSTM architectures which showed a high performance for both our private and the public dataset.

The Genetic Algorithm, also inspired from nature, is composed from various smaller algorithms that represent different steps in the natural evolutionary process. It is perfectly suited to handling EEG signals and extracting meaning from its binary composition. The GA used in the work aimed to reduced the number of electrodes for each subjects separately, the obtained results showed that with a smaller number of electrodes used as input based mainly around the Occipital, Frontal, and Parietal regions, better performance is achieved, this is due to the regions' inherent functions as part of the Cortex.

Final Conclusion

The objective of this work is to propose an architecture for brain electrical activity interpretation and exploitation using EEG signals. A BCI was performed to control the movement of a mobile robot was achieved according to the operator's imagination based on Motor imagery (MI) techniques.

The BCI based on MI depends strongly on the user's concentration, state of mind and the amount of the preparation and practice performed before the implementation. Therefore, collecting a large dataset from multiple subjects requires an enormous amount of time and a well established protocol.

Since the collected data usually suffers from various noises, artifacts as well as the inseparability between the proposed classes of data, it makes it more challenging to get a proper command to the external device. For this reason, we have proposed a novel architecture combining Power Spectral Density computing and a One-Versus-The-Rest Common Spatial Pattern variant as features characteristics inputted to the different machine learning algorithms.

The proposed approach showed a high performance in the offline implementation with an average accuracy of 90%. This performance has been validated using the online available dataset "BCI Competition IV 2a" where it showed a satisfied results of 83.31% comparing to the best obtained performance in the literature and in which verified the robustness of our approach.

The realization of such BCI requires an online closed-loop validation where the signals are extracted from the user and a command is sent to the external device based on the trained model. This validation requires a large training data, a high quality signal acquisition device while dealing with real time constraints. Taking into consideration these requirement, an online algorithm is proposed and validated where our subject had to control an e-puck mobile robot according to a specific scenario.

In the second application, an approach for fatigue detection based on EEG signals Power Spectral Density and Entropy fusion analysis. In order to build our dataset, 9 subjects have participated in the established experiment. The proposed approach has been also validated using an available online dataset.

To prove the effectiveness and robustness of the proposed method, four common Machine Learning classifiers were selected for training and testing data, after the data was filtered and Independent Component Analysis was applied to the resulting signals for Source Separation, PSD of the source rhythms was extracted using the Welch method, ratio of these

PSD values were also calculated and used as features. They were then studied, extracted and the useful ones were combined with the Entropy of the signal. Theses features are then considered as the input of our classifiers where it showed a mean performance of 93% .

With the progress of Deep Learning algorithms, the classification and analysis of EEG signals has become more accessible. The appearance of CNN, RNN and LSTM has automated the processing and showed better results in the offline implementation. In our application, we have proposed a 1D-CNN architecture which gave a mean of 92.7% for our private dataset and 98.61% for the online dataset. Two architectures of LSTM and Bidirectional LSTM or BI-LSTM were also implemented and compared with a performance higher than 95% for the online dataset and 84% for our private dataset.

In this work, a channel selection method was also used for optimizing electrodes using a Genetic Algorithm was proposed. The results indicated that this type of system has potential for detecting fatigue especially when applied to three distinct regions of the brain, the Occipital, Frontal and Parietal regions, since it achieved a better performance rate with far less electrodes.

Perspectives

For the continuity of this work, different perspectives can be considered. We propose:

- Experimenting with other applications exploring EEG signals.
- Real-time testing and implementation the fatigue detection.
- Validation of the MI-Based BCI on a large private dataset and operators.
- Combining fatigue and BCI based in one real-time system
- The test of hybrid DL architectures for processing temporal signals such as CNN-RNN, CNN-LSTM, CNN-RF, CNN-SVM.
- Real-time implementation using DL algorithms

Annex

A Common Spatial Pattern (CSP)

In this appendix, we will describe the common spatial pattern in its simple version, furthermore we will detail and develop the One Versus the Rest variant of the CSP.

In order to achieve a suitable brain-computer interface, it is important to extract relevant features that capture better the variant characteristic of each specific brain state.

A.1 CSP for Binary Class

CSP was first proposed in [103] in its binary version, where an optimal decomposition were found, which converts the two sets of measurements into their shared space. In this common space, the two groups of transformed data have the same principal components, and their corresponding eigenvalues add to 1. Because these principal components are optimal in terms of the proportion of variance that they can explain in the public space, they can be used to distinguish the two groups to the greatest extent.

For this analysis, the original EEG data of a single experiment is expressed as a matrix. As a fist step of the development, the normalized spatial covariance of the EEG is computed as follows:

$$C = \frac{EE'}{trace(EE')} \tag{A.1}$$

Where E denotes the raw eeg signal presented in $N * T$ matrix, and where N presented the number of channels and T the number of samples,E' is the transpose of the EEG matrix. Each of distribution is separated and their respective spatial covarinace is computed based on A.2 equation, finally the composite spatial covariance is given as:

$$S_c = S_1 + S_2 \tag{A.2}$$

where S_1 is the spatial covariance if the first class and S_2 the spatial covariance of the second class.S_c is then factored to $S_c = U_c \lambda_c U_c'$, U_c presents the matrix of eigenvectors and λ_c the diagonal matrix of eigenvalues sorted in descending order.

The next step is computing the whitening transformation computed as follows:

$$P = \sqrt{\lambda_c^{-1}} \tag{A.3}$$

if S_1 and S_2 are transformed, we can then write the transformation below:

$$C_1 = PS_1P' \ \ and \ \ C_2 = PS_2P' \tag{A.4}$$

Another property that can be described is that C_1 and C_2 shares the same eigenvectors, thus:

$$if\, C_1 = B\lambda_1 B' \ \ then \ \ C_2 = B\lambda_2 B' \ \ and \ \ \lambda_1 + \lambda_2 = I \tag{A.5}$$

where I is the identity matrix, one of conclusion that we can declare is that Since the sum of two corresponding eigenvalues is always one, the eigenvector of C_1 with the largest eigenvalue has the smallest C_2 eigenvalue, and vice versa on the contrary. This attribute makes the feature vector useful for classifying two distributions. Projecting the whitened EEG onto the first and last eigenvectors in B will give the features The best vector to distinguish between two populations EEG in the sense of least squares. The projection matrix is then defined as :

$$W = (B'P)' \tag{A.6}$$

the decomposition of a trial E is given as:

$$Z = WE \tag{A.7}$$

where the columns of W^{-1} presents the common spatial pattern.

A.2 OVR FPCSP for Multi Class

The need of multiclass classification leds to development of such extention of the CSP, the first multiclass extention of the csp was first proposed in [67] which is based on pairwise classification and voting but since this approach still in its binary version, other variant were developed ; the One Versus the Rest (OVR) [28] and the Simultaneous Diagonalization (SIM). In our apprach we have applied the OVR variant of CSP applied to power spectral density of the filtred EEG data in which we denoed as the FPCSP short of Filter Power Common Spatial Pattern. In this approach, spatial pattern are computed for each class against all others. we will derive the OVR FPCSP algorithm for four class case. Extension to more classes or less is trivial. We denoted filtered power spectral density of four classes EEG signal matrix as X_1,X_2,X_3 and X_4 with a $N * T$ dimension, where N presents the number of channels and T the number of samples in time. As a first step, we considered the spatial covariance of the EEG classes such as:

$$R_1 = X_1X_1^T \ \ R_2 = X_2X_2^T \ \ R_3 = X_3X_3^T \ \ R_4 = X_4X_4^T \tag{A.8}$$

and the composite covariance matrix as:

$$R = R_1 + R_2 + R_3 + R_4 \tag{A.9}$$

The composite covariance is then factored to :

$$R = V\lambda V^T \tag{A.10}$$

where λ is a $N * N$ matrix of eigenvalues and V in the $N * N$ unitary matrix of principal components, the whitening transformation matrix is defined as

$$W = \sqrt{\lambda^{-1}}V^T \tag{A.11}$$

The common Spatial patterns are extracted for each class, let considered the first class, we first defined the spatial covariance $R'_1 = R_2 + R_3$, which is transformed to :

$$S'_1 = WR'_1W^T \tag{A.12}$$

and the transformation related to R_1 is defined as

$$S_1 = WR_1W^T \tag{A.13}$$

From [35] S'_1 and S_1 share common principal components, and the sum of the corresponding eigenvalues is equal to one, the eigendecomposition of S_1 and S'_1 are:

$$S_1 = U\lambda_1U^T S'_1 = U\lambda'_1U^T \tag{A.14}$$

and

$$\lambda_1 + \lambda'_1 = I \tag{A.15}$$

Where U is the matrix of common principal components. Combining the equations above, we obtain the couple of equations below:

$$\lambda_1 = (W^TU)^T R_1(W^TU) = F_1R_1F_1^T \tag{A.16}$$

and

$$\lambda'_1 = (W^TU)^T R'_1(W^TU) = F_1R'_1F_1^T \tag{A.17}$$

and where the spatial filter is defined as

$$F_1 = U^TW \tag{A.18}$$

For this first class , the m common principal components of the m largest eigenvalues of λ_1 possesses a maximal variance. In the other part, the eigenvalues of λ'_1 should represent the variance accounted for the other classes and based on A.15 equation, when the variance of the signal components for the first class is maximized, the variance of the signal components for all other classes is minimized and thus we have

$$\lambda_2 + \lambda_3 + \lambda_4 = \lambda'_1 \tag{A.19}$$

and where $\lambda_2 = F_1R_2F_1^T$,$\lambda_3 = F_1R_3F_1^T$ and $\lambda_4 = F_1R_4F_1^T$ Evidently, the diagonal elements of λ_2,λ_3 and λ_4 represent the variance accounted by other classes on the common principal components, As a result of the development above the signal components corresponding to the first class is giving as:

$$Z_1 = F_1X_1 \tag{A.20}$$

and the decomposition of X_1 can be written as:

$$X_1 = P_1 Z_1 \tag{A.21}$$

where P_1 is the matrix of common spatial patterns and it is the pseudo-inverse of F_1 Therefore, X_1 is projected into the space of common spatial pattern as orthogonal components and where Z_1 is seen as a new time series that is equivalent to X_1 in the space of common spatial patterns. The final step is to considered m rows in Z_1 as feature vectors for X_1, we finally denotes the matrix of the selected vectors as Z_1^s which can be written as :

$$Z_1^s = F_1^s X_1 \tag{A.22}$$

We repeat the above procedures in order to find feature vectors Z_2^s,Z_3^s and Z_4^s

B EEGLAB

EEGLAB is wide use offline environment for EEG data analysis, transformation and visualisation. EEG is considered as Matlab toolbox and can be installed for the official EEGLAB website[1].

B.1 EEGLAB Quickstart

B.1.1 Basics Tools

Once openning the toolbox via Matalb, the EEGLAB window pop up, the data need to be loaded in mat form and the frequency rate is introduced as a parameter in the beginning of the session, the location of electrodes are also loaded in the beginning of session in order to provide component map in each location.

[1]For more information: https://eeglab.org

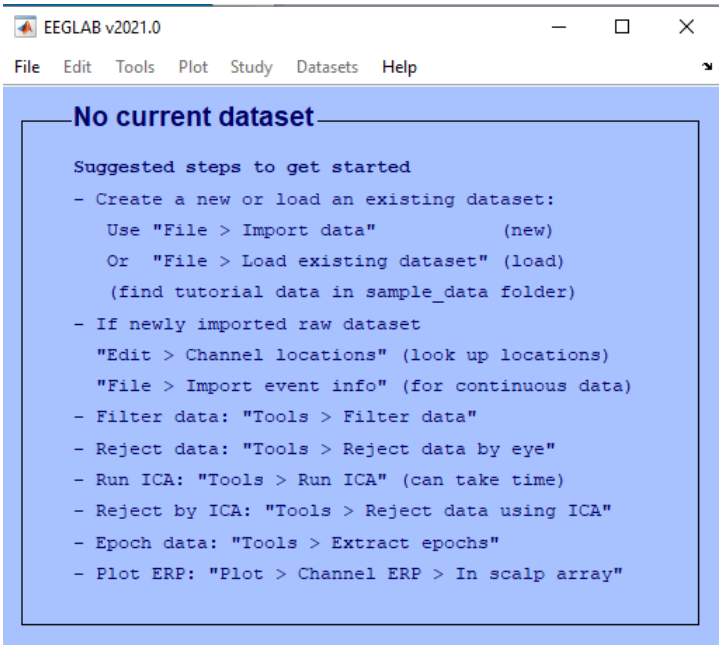


FIGURE B.1: EEGLAB Starting Page

After loading the data, the operator can starting exploring the toolbox, these are the main operation that can be done :

- Selecting a specific time segment of data.
- Selecting a number of data points.
- Visualise data in time domain.
- Visualise the spectrum of EEG signals.
- Frequency filtering, where either FIR filter or IIR filter can be used, the filter can be a pass-band or stop-band filter and where the parameters are introduced by the user

B.1.2 Independent Component Analysis (ICA)

One of the main tools of EEGLAB toolbox is the independent components computing using the *runica*() function, After calculating using the ICA algorithm, a window appears. This window contains first of all a topography of the projection of the component onto the electrodes (see Figure B.2,and secondly, information relating to independent components.

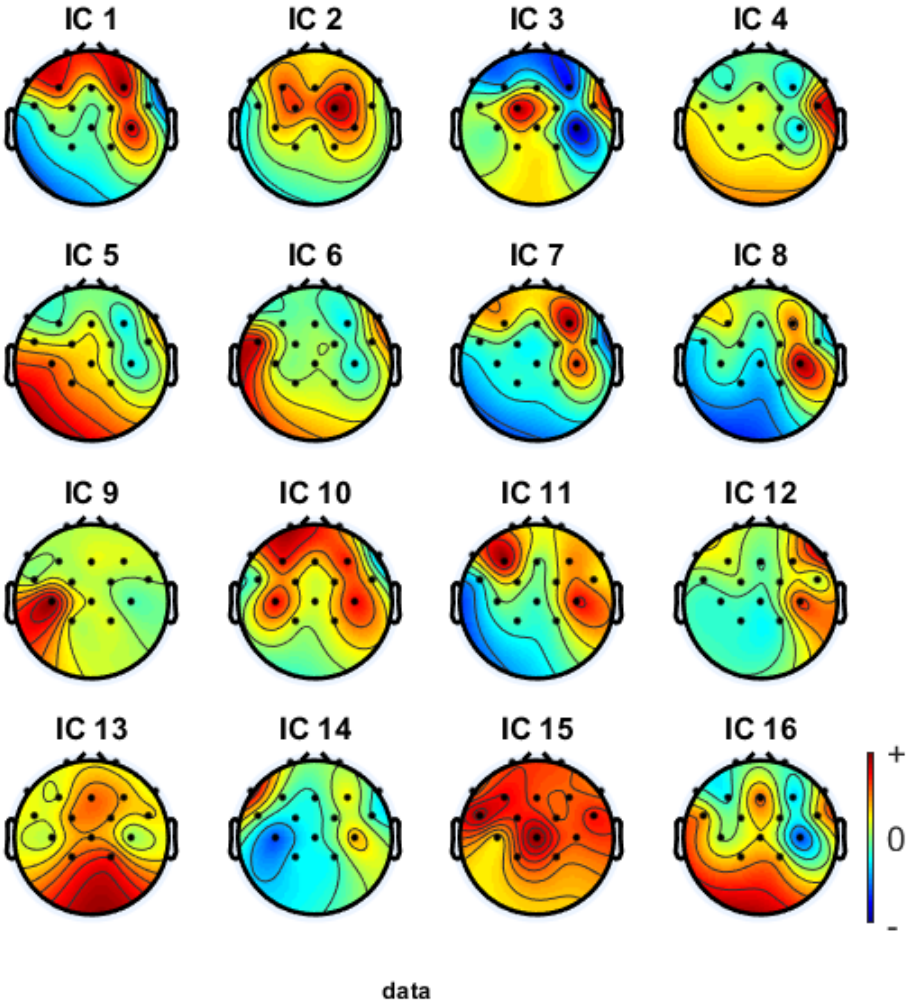


FIGURE B.2: Topography of 16 independent component

In B.3 we illustrate the effect of the independent component on each electrode through a projection. Green represents no activity, red and blue member of positive and negative contributions, respectively.

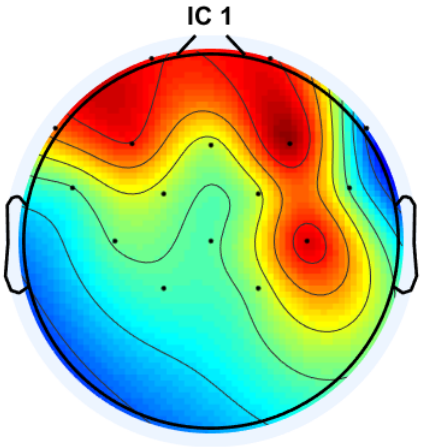


FIGURE B.3: Topography of the first independent component

in Graph B.4 the spectral energy of the independent component (mean with respect to the axis of the set of temporal windows) according to the C3 electrode placement is shown .

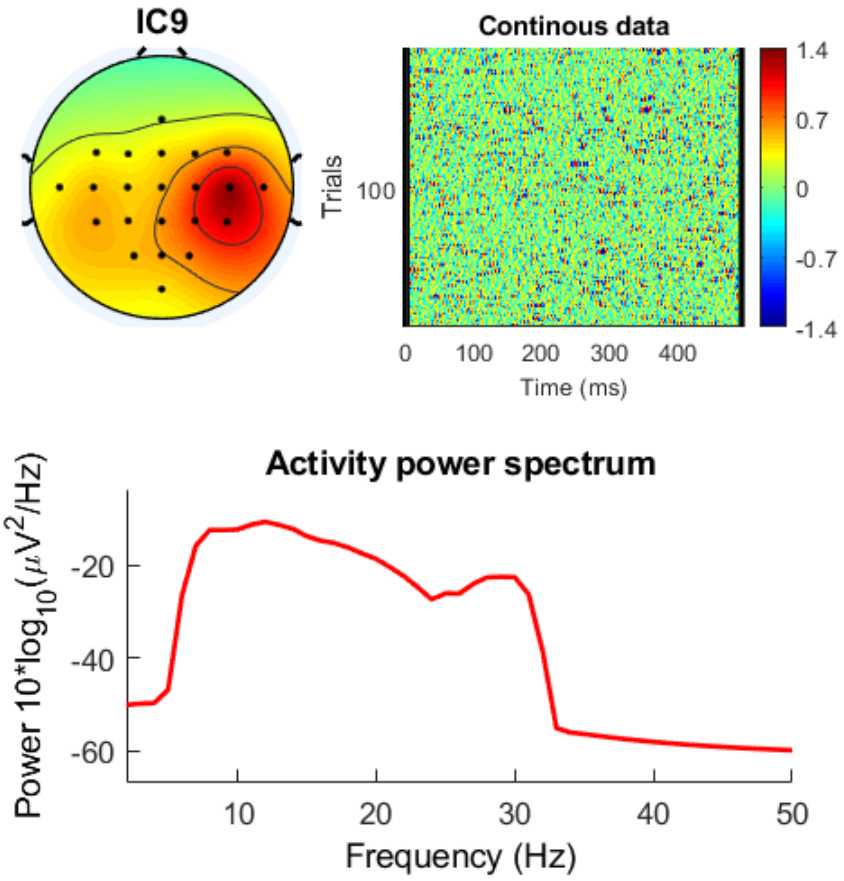


FIGURE B.4: PSD and Topography of the independent component of C3

Bibliography

[1] Enas Abdulhay et al. "Computer-aided autism diagnosis via second-order difference plot area applied to EEG empirical mode decomposition". In: *Neural Computing and Applications* (2018), pp. 1–10.

[2] Charu C Aggarwal et al. "Neural networks and deep learning". In: *Springer* 10 (2018), pp. 978–3.

[3] Ali Al-Saegh, Shefa A Dawwd, and Jassim M Abdul-Jabbar. "Deep learning for motor imagery EEG-based classification: A review". In: *Biomedical Signal Processing and Control* 63 (2021), p. 102172.

[4] Majid Aljalal et al. "Comprehensive review on brain-controlled mobile robots and robotic arms based on electroencephalography signals". In: *Intelligent Service Robotics* 13 (2020), pp. 539–563.

[5] Mahnaz Arvaneh, Ian H Robertson, and Tomas E Ward. "A p300-based brain-computer interface for improving attention". In: *Frontiers in human neuroscience* 12 (2019), p. 524.

[6] Xiaoping Bai et al. "The offline feature extraction of four-class motor imagery EEG based on ICA and Wavelet-CSP". In: *Proceedings of the 33rd Chinese Control Conference*. IEEE. 2014, pp. 7189–7194.

[7] T Balkin et al. *Effects of sleep schedules on commercial motor vehicle driver performance*. Tech. rep. United States. Department of Transportation. Federal Motor Carrier Safety ..., 2000.

[8] Giuseppe Bonaccorso. *Machine learning algorithms*. Packt Publishing Ltd, 2017.

[9] Philippe Bonnifait. "Localisation précise en position et attitude des robots mobiles d'extérieur à évolutions lentes". PhD thesis. Nantes, 1997.

[10] Leo Breiman. "Bagging predictors". In: *Machine learning* 24.2 (1996), pp. 123–140.

[11] James Brownlee. "A gentle introduction to early stopping to avoid overtraining neural networks". In: *Machine Learning Mastery* 7 (2018).

[12] Jason Brownlee. "A gentle introduction to dropout for regularizing deep neural networks". In: *Machine Learning Mastery* 3 (2018).

[13] Jason Brownlee. *How to Avoid Overfitting in Deep Learning Neural Networks*. 2018.

[14] Jason Brownlee. "How to Choose an Activation Function for Deep Learning". In: *Machine Learning Mastery* (2019).

[15] Stephen Butterworth et al. "On the theory of filter amplifiers". In: *Wireless Engineer* 7.6 (1930), pp. 536–541.

[16] Hanshu Cai et al. "A pervasive approach to EEG-based depression detection". In: *Complexity* 2018 (2018).

[17] Anthony Casey et al. “BCI controlled robotic arm as assistance to the rehabilitation of neurologically disabled patients”. In: *Disability and Rehabilitation: Assistive Technology* (2019), pp. 1–13.

[18] Siwar Chaabene et al. “Convolutional Neural Network for Drowsiness Detection Using EEG Signals”. In: *Sensors* 21.5 (2021), p. 1734.

[19] Moonjeong Chang et al. “Spatial auditory two-step input Japanese syllabary brain-computer interface speller”. In: *Procedia Technology* 18 (2014), pp. 25–31.

[20] Kyuwan Choi and Andrzej Cichocki. “Control of a wheelchair by motor imagery in real time”. In: *International conference on intelligent data engineering and automated learning*. Springer. 2008, pp. 330–337.

[21] Kyuwan Choi and Andrzej Cichocki. “Control of a wheelchair by motor imagery in real time”. In: *International conference on intelligent data engineering and automated learning*. Springer. 2008, pp. 330–337.

[22] Francois Chollet et al. *Deep learning with Python*. Vol. 361. Manning New York, 2018.

[23] Sergio A Cortez, Christian Flores, and Javier Andreu-Perez. “A Smart Home Control Prototype Using a P300-Based Brain–Computer Interface for Post-stroke Patients”. In: *Proceedings of the 5th Brazilian Technology Symposium*. Springer. 2021, pp. 131–139.

[24] Drew Dawson, Amelia K Searle, and Jessica L Paterson. “Look before you (s) leep: evaluating the use of fatigue detection technologies within a fatigue risk management system for the road transport industry”. In: *Sleep medicine reviews* 18.2 (2014), pp. 141–152.

[25] Dick De Waard and Karel A Brookhuis. “Assessing driver status: a demonstration experiment on the road”. In: *Accident analysis & prevention* 23.4 (1991), pp. 297–307.

[26] S Deivasigamani, Chinnaiyan Senthilpari, and Wong Hin Yong. “Machine learning method based detection and diagnosis for epilepsy in EEG signal”. In: *Journal of Ambient Intelligence and Humanized Computing* 12.3 (2021), pp. 4215–4221.

[27] Xin Deng et al. “Advanced TSGL-EEGNet for Motor Imagery EEG-Based Brain-Computer Interfaces”. In: *IEEE Access* 9 (2021), pp. 25118–25130.

[28] Guido Dornhege et al. “Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms”. In: *IEEE transactions on biomedical engineering* 51.6 (2004), pp. 993–1002.

[29] Messaoud Doudou, Abdelmadjid Bouabdallah, and Véronique Berge-Cherfaoui. “Driver drowsiness measurement technologies: Current research, market solutions, and challenges”. In: *International Journal of Intelligent Transportation Systems Research* (2019), pp. 1–23.

[30] Daniel Elstob and Emanuele Lindo Secco. “A low cost EEG based BCI prosthetic using motor imagery”. In: *arXiv preprint arXiv:1603.02869* (2016).

[31] Hong J Eoh, Min K Chung, and Seong-Han Kim. “Electroencephalographic study of drowsiness in simulated driving with sleep deprivation”. In: *International Journal of Industrial Ergonomics* 35.4 (2005), pp. 307–320.

[32] Monica Fabiani et al. “Definition, identification, and reliability of measurement of the P300 component of the event-related brain potential”. In: *Advances in psychophysiology* 2.S 1 (1987), p. 78.

[33] Weijiang Feng et al. "Audio visual speech recognition with multimodal recurrent neural networks". In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2017, pp. 681–688.

[34] Zachary V Freudenburg et al. "Sensorimotor ECoG signal features for BCI control: a comparison between people with locked-in syndrome and able-bodied controls". In: *Frontiers in neuroscience* 13 (2019), p. 1058.

[35] Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[36] Zhong-Ke Gao et al. "A recurrence network-based convolutional neural network for fatigue driving detection from EEG". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 29.11 (2019), p. 113126.

[37] Zhongke Gao et al. "EEG-based spatio–temporal convolutional neural network for driver fatigue evaluation". In: *IEEE transactions on neural networks and learning systems* 30.9 (2019), pp. 2755–2763.

[38] Felix Gembler, Piotr Stawicki, and Ivan Volosyak. "Autonomous parameter adjustment for SSVEP-based BCIs with a novel BCI wizard". In: *Frontiers in neuroscience* 9 (2015), p. 474.

[39] Xiaotong Gu et al. "EEG-based brain-computer interfaces (BCIs): A survey of recent studies on signal sensing technologies and computational intelligence approaches and their applications". In: *IEEE/ACM transactions on computational biology and bioinformatics* (2021).

[40] Tushar Gupta. "Deep learning: Feedforward neural network". In: *Towards Data Science* 5 (2017).

[41] Mehdi Hajinoroozi et al. "EEG-based prediction of driver's cognitive performance by deep convolutional neural network". In: *Signal Processing: Image Communication* 47 (2016), pp. 549–555.

[42] Henderi Henderi, Tri Wahyuningsih, and Efana Rahwanto. "Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer". In: *International Journal of Informatics and Information Systems* 4.1 (2021), pp. 13–20.

[43] Cristy Ho, Hong Z Tan, and Charles Spence. "Using spatial vibrotactile cues to direct visual attention in driving scenes". In: *Transportation Research Part F: Traffic Psychology and Behaviour* 8.6 (2005), pp. 397–412.

[44] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[45] JH Holland. *Genetic Algorithms, computer programs that "evolve" in ways that resemble even their creators do not fully understand*. 1975.

[46] Krister Holmes. "Hans Berger and the EEG". In: *Thresholds* (2014), pp. 88–97.

[47] Qiyun Huang et al. "An EEG-/EOG-based hybrid brain-computer interface: application on controlling an integrated wheelchair robotic arm system". In: *Frontiers in neuroscience* 13 (2019), p. 1243.

[48] Zafer İşcan and Vadim V Nikulin. "Steady state visual evoked potential (SSVEP) based brain-computer interface (BCI) performance under different perturbations". In: *PloS one* 13.1 (2018), e0191673.

[49] Kun Jiao et al. "Effect of different vibration frequencies on heart rate variability and driving fatigue in healthy drivers". In: *International archives of occupational and environmental health* 77.3 (2004), pp. 205–212.

[50] Difei Jing et al. “Fatigue driving detection method based on EEG analysis in low-voltage and hypoxia plateau environment”. In: *International journal of transportation science and technology* 9.4 (2020), pp. 366–376.

[51] Kirthevasan Kandasamy et al. “Tuning Hyperparameters without Grad Students: Scalable and Robust Bayesian Optimisation with Dragonfly.” In: *J. Mach. Learn. Res.* 21.81 (2020), pp. 1–27.

[52] Sai Nikhilesh Kasturi. *Underfitting and Overfitting in machine learning and how to deal with it*. 2020.

[53] Barjinder Kaur and Dinesh Singh. “The applicability of BCI in different domains”. In: *National Conference on Recent Trends in Computing and Communication Technologies*. 2016, pp. 82–86.

[54] Javeria Khan et al. “Multiclass EEG motor-imagery classification with sub-band common spatial patterns”. In: *EURASIP Journal on Wireless Communications and Networking* 2019.1 (2019), pp. 1–9.

[55] Souhir Khessiba et al. “Innovative deep learning models for EEG-based vigilance detection”. In: *Neural Computing and Applications* (2020), pp. 1–17.

[56] SC Kleih et al. “Motivation modulates the P300 amplitude during brain–computer interface use”. In: *Clinical Neurophysiology* 121.7 (2010), pp. 1023–1031.

[57] Moritz Körber et al. “Vigilance decrement and passive fatigue caused by monotony in automated driving”. In: *Procedia Manufacturing* 3 (2015), pp. 2403–2409.

[58] Andrea Kübler. “The history of BCI: From a vision for the future to real support for personhood in people with locked-in syndrome”. In: *Neuroethics* 13.2 (2020), pp. 163–180.

[59] Kyuhwa Lee et al. “A brain-controlled exoskeleton with cascaded event-related desynchronization classifiers”. In: *Robotics and Autonomous Systems* 90 (2017), pp. 15–23.

[60] Sangdi Lin and George C Runger. “GCRNN: Group-constrained convolutional recurrent neural network”. In: *IEEE transactions on neural networks and learning systems* 29.10 (2017), pp. 4709–4718.

[61] Ziming Liu et al. “A systematic review on hybrid EEG/fNIRS in brain-computer interface”. In: *Biomedical Signal Processing and Control* 68 (2021), p. 102595.

[62] Yuri K Lopes et al. “Supervisory control theory applied to swarm robotics”. In: *Swarm Intelligence* 10.1 (2016), pp. 65–97.

[63] Mufti Mahmud, Alessandra Bertoldo, and Stefano Vassanelli. “EEG based brain-machine interfacing: Navigation of mobile robotic device”. In: *Mobile Robots-Control Architectures, Bio-Interfacing, Navigation, Multi Robot Motion Planning and Operator Training* (2011), pp. 129–144.

[64] XianJia Meng et al. “A motor imagery EEG signal classification algorithm based on recurrence plot convolution neural network”. In: *Pattern Recognition Letters* 146 (2021), pp. 134–141.

[65] Jianliang Min, Ping Wang, and Jianfeng Hu. “Driver fatigue detection through multiple entropy fusion analysis in an EEG-based system”. In: *PLoS one* 12.12 (2017), e0188756.

[66] Shervin Minaee. "popular machine learning metrics. part 1: Classification & regression evaluation metrics,"" in: *Medium. url: https://towardsdatascience. com/20-popular-machine-learning-metrics-part-1-classification-regressionevaluation-metrics-1ca3e282a2ce* (2019).

[67] Johannes Müller-Gerking, Gert Pfurtscheller, and Henrik Flyvbjerg. "Designing optimal spatial filters for single-trial EEG classification in a movement task". In: *Clinical neurophysiology* 110.5 (1999), pp. 787–798.

[68] Khushboo Munir, Fabrizio Frezza, and Antonello Rizzi. "Deep Learning for Brain Tumor Segmentation". In: *Deep Learning for Cancer Diagnosis*. Springer, 2021, pp. 189–201.

[69] Muhammad Naeem et al. "Seperability of four-class motor imagery data using independent components analysis". In: *Journal of neural engineering* 3.3 (2006), p. 208.

[70] Yogendra Narayan. "Motor-Imagery EEG Signals Classificationusing SVM, MLP and LDA Classifiers". In: *Turkish Journal of Computer and Mathematics Education (TURCOMAT)* 12.2 (2021), pp. 3339–3344.

[71] Florian Nigsch et al. "Melting point prediction employing k-nearest neighbor algorithms and genetic parameter optimization". In: *Journal of chemical information and modeling* 46.6 (2006), pp. 2412–2422.

[72] Brilian T Nugraha et al. "CLASSIFICATION OF DRIVER FATIGUE STATE BASED ON EEG USING EMOTIV EPOC+." In: *Journal of Theoretical & Applied Information Technology* 86.3 (2016).

[73] Hime Aguiar e Oliveira Junior et al. "Global Optimization and Its Applications". In: *Stochastic Global Optimization and Its Applications with Fuzzy Adaptive Simulated Annealing* (2012), pp. 11–20.

[74] Eneko Osaba et al. "A study on the impact of heuristic initialization functions in a genetic algorithm solving the n-queens problem". In: *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*. 2014, pp. 1473–1474.

[75] Natasha Padfield et al. "EEG-based brain-computer interfaces using motor-imagery: Techniques and challenges". In: *Sensors* 19.6 (2019), p. 1423.

[76] Sampath Kumar Palaniswamy and R Venkatesan. "Hyperparameters tuning of ensemble model for software effort estimation". In: *Journal of Ambient Intelligence and Humanized Computing* 12.6 (2021), pp. 6579–6589.

[77] Damiano Paolicelli et al. "Magnetoencephalography and High-Density Electroencephalography Study of Acoustic Event Related Potentials in Early Stage of Multiple Sclerosis: A Pilot Study on Cognitive Impairment and Fatigue". In: *Brain sciences* 11.4 (2021), p. 481.

[78] Fernando Perez-Diaz, Ruediger Zillmer, and Roderich Groß. "Control of synchronization regimes in networks of mobile interacting agents". In: *Physical Review Applied* 7.5 (2017), p. 054002.

[79] Ross O Phillips. "A review of definitions of fatigue–And a step towards a whole definition". In: *Transportation research part F: traffic psychology and behaviour* 29 (2015), pp. 48–56.

[80] S Madeh Piryonesi and Tamer E El-Diraby. "Role of data analytics in infrastructure asset management: Overcoming data size and quality problems". In: *Journal of Transportation Engineering, Part B: Pavements* 146.2 (2020), p. 04020022.

[81] Cristian-Cezar Postelnicu, Doru Talaba, and Madalina-Ioana Toma. "Controlling a robotic arm by brainwaves and eye movement". In: *Doctoral Conference on Computing, Electrical and Industrial Systems*. Springer. 2011, pp. 157–164.

[82] Alexander D Rae-Grant, Peter J Barbour, and James Reed. "Development of a novel EEG rating scale for head injury using dichotomous variables". In: *Electroencephalography and clinical neurophysiology* 79.5 (1991), pp. 349–357.

[83] Vyza Yashwanth Sai Reddy. "Robot Control Using Brain-Computer Interface System". In: (2017).

[84] Nicolas Régis et al. "Formal detection of attentional tunneling in human operator–automation interactions". In: *IEEE Transactions on Human-Machine Systems* 44.3 (2014), pp. 326–336.

[85] Ziwu Ren et al. "EEG-based driving fatigue detection using a two-level learning hierarchy radial basis function". In: *Frontiers in Neurorobotics* 15 (2021).

[86] Amin Riazi. "Genetic algorithm and a double-chromosome implementation to the traveling salesman problem". In: *SN Applied Sciences* 1.11 (2019), pp. 1–7.

[87] Ricardo Ron-Angevin et al. "A two-class self-paced BCI to control a robot in four directions". In: *2011 IEEE International Conference on Rehabilitation Robotics*. IEEE. 2011, pp. 1–6.

[88] Ricardo Ron-Angevin et al. "Brain-Computer Interface application: auditory serial interface to control a two-class motor-imagery-based wheelchair". In: *Journal of neuroengineering and rehabilitation* 14.1 (2017), pp. 1–16.

[89] G Roy, AK Bhoi, and S Bhaumik. "A Comparative Approach for MI-Based EEG Signals Classification Using Energy, Power and Entropy". In: *IRBM* (2021).

[90] Raphaëlle N Roy, Sylvie Charbonnier, and Stéphane Bonnet. "Detection of mental fatigue using an active BCI inspired signal processing chain". In: *IFAC Proceedings Volumes* 47.3 (2014), pp. 2963–2968.

[91] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv preprint arXiv:1609.04747* (2016).

[92] Martijn Schreuder, Thomas Rost, and Michael Tangermann. "Listen, you are writing! Speeding up online spelling with a dynamic auditory BCI". In: *Frontiers in neuroscience* 5 (2011), p. 112.

[93] Mike Schuster and Kuldip K Paliwal. "Bidirectional recurrent neural networks". In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681.

[94] Abeer E Selim, Manal Abdel Wahed, and Yasser M Kadah. "Machine learning methodologies in P300 speller Brain-Computer Interface systems". In: *2009 National Radio Science Conference*. IEEE. 2009, pp. 1–9.

[95] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[96] Mohd Maroof Siddiqui et al. "EEG signals play major role to diagnose sleep disorder". In: *International Journal of Electronics and Computer Science Engineering (IJECSE)* 2.2 (2013), pp. 503–505.

[97] Siuly Siuly, Yan Li, and Yanchun Zhang. "Electroencephalogram (EEG) and its background". In: *EEG Signal Analysis and Classification*. Springer, 2016, pp. 3–21.

[98] Mads Soegaard and Rikke Friis Dam. "The encyclopedia of human-computer interaction". In: *The encyclopedia of human-computer interaction* (2012).

[99] Indratmo Soekarno, Iwan K Hadihardaja, M Cahyono, et al. "A study of hold-out and k-fold cross validation for accuracy of groundwater modeling in tidal lowland reclamation using extreme learning machine". In: *2014 2nd International Conference on Technology, Informatics, Management, Engineering & Environment*. IEEE. 2014, pp. 228–233.

[100] Kenneth Sörensen. "Metaheuristics—the metaphor exposed". In: *International Transactions in Operational Research* 22.1 (2015), pp. 3–18.

[101] Charles Spence and Cristy Ho. "Multisensory interface design for drivers: past, present and future". In: *Ergonomics* 51.1 (2008), pp. 65–70.

[102] SR Sreeja et al. "Classification of motor imagery based EEG signals using sparsity approach". In: *International Conference on Intelligent Human Computer Interaction*. Springer, Cham. 2017, pp. 47–59.

[103] EE SWARTZLA. *Introduction to Statistical Pattern Recognition-Fukunaga, K.* 1974.

[104] Kashvi Taunk et al. "A brief review of nearest neighbor algorithm for learning and classification". In: *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*. IEEE. 2019, pp. 1255–1260.

[105] Tech-Effigy. "The Genetic Algorithm - Explained". In: ().

[106] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. "Dimensionality reduction: a comparative". In: *J Mach Learn Res* 10.66-71 (2009), p. 13.

[107] Sergio Varona-Moya et al. "Wheelchair navigation with an audio-cued, two-class motor imagery-based brain-computer interface system". In: *2015 7th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE. 2015, pp. 174–177.

[108] Hong Wang et al. "Real-time EEG-based detection of fatigue driving danger for accident prediction". In: *International journal of neural systems* 25.02 (2015), p. 1550002.

[109] Yulong Wang, Haoxin Zhang, and Guangwei Zhang. "cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks". In: *Swarm and Evolutionary Computation* 49 (2019), pp. 114–123.

[110] Joel S Warm, Raja Parasuraman, and Gerald Matthews. "Vigilance requires hard mental work and is stressful". In: *Human factors* 50.3 (2008), pp. 433–441.

[111] Thomas Weise. "Global optimization algorithms-theory and application". In: *Self-Published Thomas Weise* (2009).

[112] Edmond Q Wu et al. "Detecting fatigue status of pilots based on deep learning network using EEG signals". In: *IEEE Transactions on Cognitive and Developmental Systems* (2020).

[113] Wei Wu, Xiaorong Gao, and Shangkai Gao. "One-versus-the-rest (OVR) algorithm: An extension of common spatial patterns (CSP) algorithm to multi-class case". In: *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*. IEEE. 2006, pp. 2387–2390.

[114] Siyu Yang et al. "Exploring the Use of Brain-Computer Interfaces in Stroke Neurorehabilitation". In: *BioMed Research International* 2021 (2021).

[115] Jinghai Yin, Jianfeng Hu, and Zhendong Mu. "Developing and evaluating a mobile driver fatigue detection network based on electroencephalograph signals". In: *Healthcare technology letters* 4.1 (2017), pp. 34–38.

[116] Shivanthan AC Yohanandan et al. "A robust low-cost eeg motor imagery-based brain-computer interface". In: *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2018, pp. 5089–5092.

[117] Xinyi Yong, Rabab K Ward, and Gary E Birch. "Facial EMG contamination of EEG signals: Characteristics and effects of spatial filtering". In: *2008 3rd International Symposium on Communications, Control and Signal Processing*. IEEE. 2008, pp. 729–734.

[118] Hong Zeng et al. "EEG classification of driver mental states by deep learning". In: *Cognitive neurodynamics* 12.6 (2018), pp. 597–606.

[119] Walid Zgallai et al. "Deep learning AI application to an EEG driven BCI smart wheelchair". In: *2019 Advances in Science and Engineering Technology International Conferences (ASET)*. IEEE. 2019, pp. 1–5.

[120] Chi Zhang, Hong Wang, and Rongrong Fu. "Automated detection of driver fatigue based on entropy and complexity measures". In: *IEEE Transactions on Intelligent Transportation Systems* 15.1 (2013), pp. 168–177.

[121] Xiang Zhang et al. "Deep neural network hyperparameter optimization with orthogonal array tuning". In: *International conference on neural information processing*. Springer. 2019, pp. 287–295.