HAROKOPIO UNIVERSITY

School of digital technology
Department of Informatics and Telematics
Master of Science in Informatics and Telematics
MSc in Web Engineering

Brain-Computer Interfaces using Machine Learning

A Thesis in the Field of Machine Learning

Theodoros Papadopoulos

Athens, 2019

HAROKOPIO UNIVERSITY

School of digital technology
Department of Informatics and Telematics
Master of Science in Informatics and Telematics
MSc in Web Engineering

Examining Committee

Iraklis Varlamis (Supervisor)
Associate professor, Harokopio University

Dimitrios Michail,
Associate professor, Harokopio University

Sophia Karagiorgou,
Adjunct Lecturer, Harokopio University

Theodoros Papadopoulos,

I hereby declare that:

1) To the best of my knowledge, my work does not insult persons, nor does it offend the intellectual rights of third parties.

2) I accept that the university's library may, without changing the content of my work, make it available in electronic form through its Digital Library, copy it in any medium and / or any format and hold more than one copy for maintenance and safety purposes.

Abstract

This thesis explores machine learning models for the analysis and classification of electroencephalographic (EEG) signals used in Brain-Computer Interface (BCI) systems. The goal is 1) to develop a system that allows users to control home-automation devices using their mind, and 2) to investigate whether it is possible to achieve this, using low-cost EEG equipment. The thesis includes both a theoretical and a practical part.

In the theoretical part, we overview the underlying principles of Brain-Computer Interface systems, as well as, different approaches for the interpretation and the classification of brain signals. We also discuss the emergent launch of low-cost EEG equipment on the market and its use beyond clinical research. We then dive into more technical details that involve signal processing and classification of EEG patterns using machine leaning.

Purpose of the practical part is to create a brain-computer interface that will be able to control a smart home environment. As a first step, we investigate the generalizability of different classification methods, conducting a preliminary study on two public datasets of brain encephalographic data. The obtained accuracy level of classification on 9 different subjects was similar and, in some cases, superior to the reported state of the art.

Having achieved relatively good offline classification results during our study, we move on to the last part, designing and implementing an online BCI system using Python. Our system consists of three modules. The first module communicates with the MUSE (a low-cost EEG device) to acquire the EEG signals in real time, the second module process those signals using machine learning techniques and trains a learning model. The model is used by the third module, that takes control of cloud-based home automation devices. Experiments using the MUSE resulted in significantly lower classification results and revealed the limitations of the low-cost EEG signal acquisition device for online BCIs.

***Keywords**: Brain-Computer Interface, Machine Learning, Deep Learning, Signal Processing, Electroencephalography, Spatial Filtering, Python, Smart Home*

Abstract (Greek)

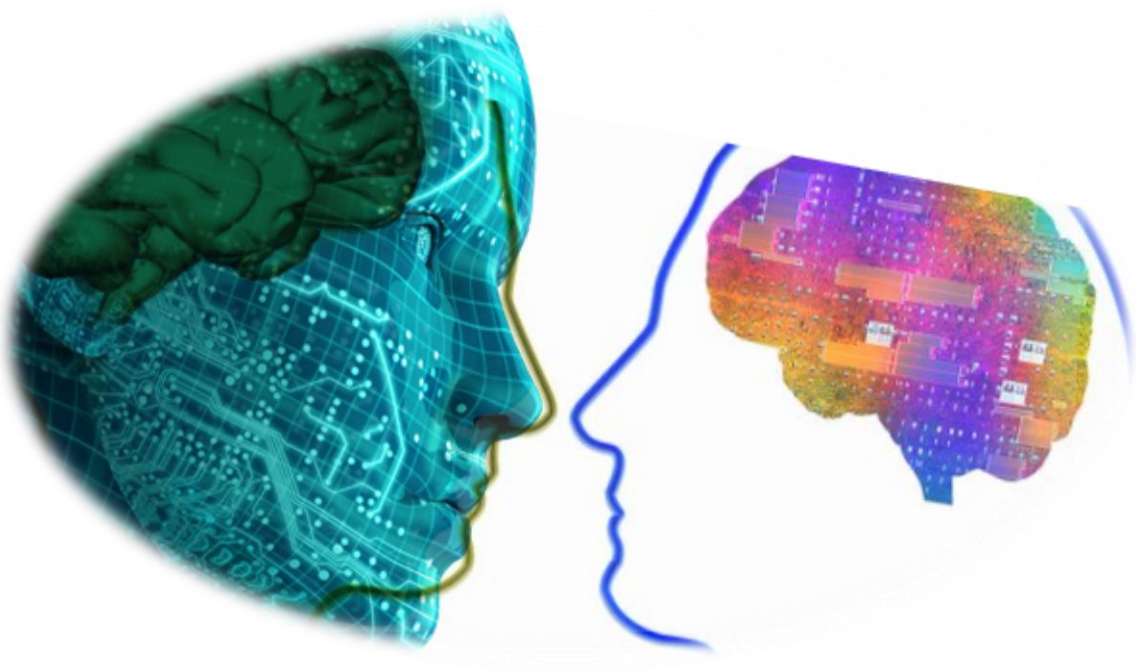
Η διπλωματική αυτή εργασία εξετάζει μοντέλα μηχανικής μάθησης για την ανάλυση και την κατηγοριοποίηση ηλεκτροεγκεφαλογραφικών σημάτων (EEG), που χρησιμοποιούνται σε συστήματα διεπαφής εγκεφάλου-υπολογιστή (BCI). Ο στόχος είναι 1) να αναπτυχθεί ένα σύστημα που να επιτρέπει στους χρήστες να ελέγχουν συσκευές «έξυπνου σπιτιού» χρησιμοποιώντας το μυαλό τους, και 2) να διερευνηθεί εάν αυτό μπορεί να επιτευχθεί χρησιμοποιώντας φθηνές εμπορικές συσκευές ηλεκτροεγκεφαλογραφίας. Η διπλωματική αυτή περιλαμβάνει τόσο θεωρητικό, όσο και πρακτικό μέρος.

Στο θεωρητικό μέρος εξετάζουμε τις βασικές αρχές των συστημάτων διεπαφής εγκεφάλου-υπολογιστή, καθώς και διαφορετικές προσεγγίσεις για την ερμηνεία και την κατηγοριοποίηση των εγκεφαλικών σημάτων. Συζητάμε επίσης, την εμφάνιση καταναλωτικών φορητών συσκευών ηλεκτροεγκεφαλογραφίας, καθώς και την πιθανή χρήση τους πέρα από την κλινική έρευνα. Στη συνέχεια, αναλύουμε τεχνικές πτυχές που αφορούν την επεξεργασία και κατηγοριοποίηση σημάτων ηλεκτροεγκεφαλογραφίας.

Σκοπός του πρακτικού μέρους είναι να αναπτυχθεί μια διεπαφή εγκεφάλου-υπολογιστή που θα είναι σε θέση να ελέγξει ένα περιβάλλον «έξυπνου σπιτιού». Για το σκοπό αυτό, αρχικά εξετάζουμε μεθόδους για την κατηγοριοποίηση των εγκεφαλικών νευρωνικών σημάτων και στη συνέχεια, διεξάγουμε μια προκαταρκτική συγκριτική μελέτη με την εφαρμογή τεχνικών προεπεξεργασίας και κατηγοριοποίησης σήματος σε δύο δημόσια σύνολα εγκεφαλογραφικών δεδομένων, επιτυγχάνοντας καλές αποδόσεις πρόβλεψης.

Στο τελευταίο μέρος υλοποιούμε ένα σύστημα διεπαφής εγκεφάλου-υπολογιστή, που αποτελείται από τρεις μονάδες. Η πρώτη μονάδα επικοινωνεί με τη συσκευή ηλεκτροεγκεφαλογραφίας για την απόκτηση των σημάτων EEG σε πραγματικό χρόνο. Στη συνέχεια, η δεύτερη μονάδα επεξεργάζεται αυτά τα σήματα χρησιμοποιώντας τεχνικές μηχανικής μάθησης και εκπαιδεύει ένα μοντέλο που τροφοδοτείται στην τρίτη μονάδα, η οποία αναλαμβάνει τον έλεγχο των συσκευών αυτοματισμού του «έξυπνου σπιτιού».

Λέξεις κλειδιά: *Διεπαφή Εγκεφάλου-Υπολογιστή, Μηχανική Μάθηση, Βαθιά μάθηση, Επεξεργασία Σήματος, Ηλεκτροεγκεφαλογράφημα, Χωρικά Φίλτρα, Έξυπνο Σπίτι*



Any sufficiently advanced technology is indistinguishable from magic

Arthur C. Clarke

Dedication

To my parents,

For the memories of a childhood in which I always return when I cannot stand a world faster than the speed of life.

For the long summers, the sea, the carefree and the love.

For life.

Acknowledgment

I would like to thank my supervisor, associate professor Iraklis Varlamis, for the continuous support throughout the development of my master thesis.

I would also like to thank all the staff of the secretariat because without them there would be no university.

My uncle, for buying me my first computer when I was 12 years old.

Eleni, who without her distractions, I would have traveled less and finished this thesis many months earlier, having nothing to do right now.

.

Table of contents

List of Figures

List of Tables

Chapter 1.

Introduction

Over eons, biology via evolution and natural selection has solved the problem of processing, prioritizing and interpreting massive amount of noisy and highly redundant information, coming from a rapidly changing external environment. The brain, a constantly evolving network of billions of interconnecting neural cells, processes internal and external stimuli and makes decisions in a very short time. The input of the brain includes sensory cells (associated with seeing, hearing, tasting, touching etc.), motor/muscle cells, and even some cells within the brain. These cells pick up the stimulus provided and take it in for further processing. In fact, it's not only the brain but our entire nervous system that contributes towards the larger network, which is our consciousness.

During the last decades, humanity is developing artificial constructions, aiming to solve the exact same problems. Building a system that mirrored the simulations of the human brain, was the starting point of artificial neural networks (ANNs). But despite the recent technological advances that boosted the usage of artificial neural networks in multiple domains of human decision making and automation, and gave a thrive to artificial intelligence, there is still a huge gap in terms of abilities and effectiveness between the human brain and the artificial constructions of the man. There is also a big efficiency gap, which results in today's supercomputers that run on megawatts to consume huge amount of power, while the human brain relies only on water and some calories in order to function.

In recent years, researches in the field of Artificial Intelligence are resuming the study of the models and the architecture of the human brain as a means for the development of AI, introducing structural and operational principles of the brain into the design of machine learning algorithms. Convolutional feedforward networks, which now dominate computer vision, take inspiration from the architecture of the primate visual hierarchy. Other research approaches are inspired by the recurrent connections in the human brain, and their key role for associative learning and pattern classification.

On the other side, neuroscientists use AI tools to learn more about the workings of the brain. In order to decode and understand the brain signals, ANNs and other machine

learning algorithms are being applied for pattern recognition, anomaly detection, and brain neural signals interpretation.

Recently, another research approach aiming to develop the technology for human enhancement by merging human with artificial intelligence is gaining attention. The ability to cooperatively use the two types of intelligence, by providing the appropriate interfaces that will allow human brain to communicate unobtrusively with a computer program and vise versa, will unlock the great potential of the human brain and will open new research opportunities in human-driven artificial intelligence. Brain-computer interfaces, combining knowledge and techniques both from neuroscience and Artificial Intelligence, are already used in medical applications but gradually, non-medical application paradigms also emerge. Translating thoughts into action, without acting physically or allowing direct brain-based communication between humans may seem to belong to the realm of science fiction today, and yet science fiction has a good track record for predicting inventions and developments we now take for granted.

Investments to the fields of human-driven intelligence and Brain-computer Interfaces (BCIs) are steadily increasing and more companies and research laboratories are entering the field. According to a market analysis conducted by OMR (OMR, 2018), global BCI market is expected to witness a significant growth rate at a CAGR of 22.8% during the forecasted period (2018-2023). Recently Elon Musk entered the industry, announcing a $27 million investment in Neuralink, a venture with the mission to develop a BCI that improves human communication in light of AI, and Facebook announced its plans for a Brain-Computer Interface technology, that would allow for more efficient digital communication between brains.

Brain-Computer Interfaces

Human-computer interaction (HCI) is a research field focused on the interfaces between people (users) and computers. Humans interact with computers in many ways. Those include the prevalent graphical user interfaces (GUI) of today, the emerging voice user interfaces (VUI) which are used for speech recognition and synthesizing, as well as, other types of interaction mechanisms aiming to make human-computer interaction more natural, such as touch-screens, gesture-based interfaces, or eye-tracking driven interfaces.

Over the last twenty years, neural engineering has emerged as a new field that merges neuroscience and information technology and has resulted in neurotechnology, able to link brain activity with man-made devices. A brain-computer interface (BCI) is a combination of hardware and software that permits the capture of cerebral activity associated with a user's intent or emotions and the translation of this recorded activity to specific control signals. These control signals can be used to control an external device, such as a computer or an external smart-home device. In other words, a BCI allows direct communication between the brain and an external device.

A modern definition defines BCI as a system which captures a biosignal measured from a person in real-time and predicts an abstract aspect of the person's cognitive state.

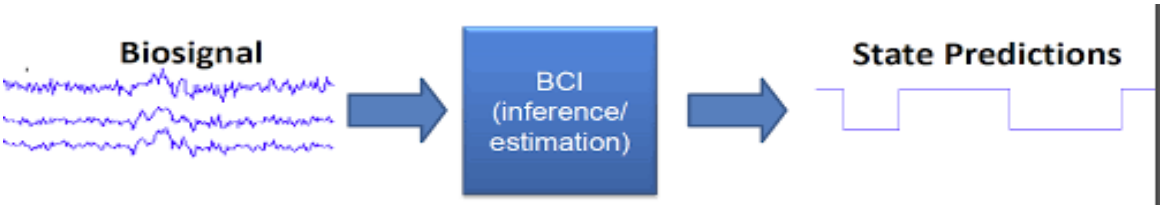


Figure 1: A Brain-Computer Interface system

Image from BCILAB tutorials and presentations: ftp://sccn.ucsd.edu/pub/bcilab/

Brain-computer interfaces combine knowledge and techniques from neuroscience, signal processing, and machine learning domains.

The original development of BCI systems targeted severely paralyzed people and was focused on the ability to communicate with the external environment, without the

necessity of muscular control. Recently, the advent of the first commercial, inexpensive, dry-electrode devices, made it feasible to target new fields and apply this technology outside the laboratory. In the near future, it is possible to develop BCIs that will be able to facilitate hands-free applications, allowing the mind-controlling of machines.

Despite strong efforts, current BCIs are still facing several challenges that limit their usefulness for everyday applications. These challenges are related to increasing bit rates (Allison et al., 2012b), optimizing sensors, signal processing, and classification techniques, but also to the type of control signal and overall systems design. Many of these issues directly affect BCI performance, which is a field of active research and addressed at multiple levels of the BCI loop.

Thesis Objective

This thesis includes both a theoretical and a practical part.

The theoretical part has three main objectives. The first is to give an overview of the underlying functionality of BCI systems, focusing specifically to EEG systems. The second is to give an overview of the state of the art of machine learning algorithms, focusing on supervised classification algorithms and neural networks. The third is to evaluate different EEG pipelines, by conducting experiments on publicly available datasets of brain signals.

The practical part aims in designing and implementing an asynchronous active BCI system, using a commercially available EEG device able to control external home-automation devices. In our experiment, we employed Muse, a portable headband launched by InteraXon in 2014 to collect data of brain waves. We also developed an interface with the Amazon Echo (Alexa) device, in order to send control commands for home automation devices.

In order to implement the proposed BCI solution, we focus on the classification of EEG signals generated during motor imagery tasks (e.g. imagining the movement of a limb). This method has been reported as a reliable way for developing Motor Imagery (MI)-based BCI, which means BCIs that can recognize imagined movements, such as left or right-hand imagined movements. This is usually done by associating different limb movements to software commands.

In order to complete this thesis, we had to carry out several tasks such as:

- Review of the EEG decoding and BCI systems literature.
- Review of the Machine learning methodology.
- Develop, train, and evaluate different EEG decoding Pipelines in Python.
- Comparison of the obtained results with state-of-the art-results in literature.

In addition, we performed an initial attempt to:

- Develop an EEG Pipeline using Deep Learning Neural Networks.
- Develop an active BCI system, able to control home-automation devices in a real-time scenario.

Thesis Structure

This thesis consists of 8 chapters structured as follows:

Chapter 2 provides a brief overview of the underlying functionality of BCI systems. This chapter begins by describing the building blocks of the human brain and the electrochemical interactions that produce its electrical signals. Furthermore, presents various methods for measuring brain activity, and discusses in more detail the functional principles of Electroencephalography.

Chapter 3 elaborates on EEG-based BCI systems and their fields of application. Different EEG-based activity patterns used in the development of BCI systems are presented and the emergence commercialization of EEG devices is discussed, along with a presentation of their basic characteristics and limitations. Finally, some examples of novel applications, which provide evidence for the promising potential of BCI technology both for medical and non-medical uses, are presented.

Chapter 4 gives an overview of the state-of-the-art machine learning algorithms, focusing on supervised learning used for EEG classification. Furthermore, it discusses deep learning approaches based on artificial neural networks.

Chapter 5 discusses in more depth the fundamental steps for the design of an EEG pipeline suitable for BCI systems, based on oscillatory EEG activity. It analyses the importance of proper data recording and data pre-processing, detailing the processing steps and providing an overview of the most important features extraction algorithms for EEG signals.

Chapter 6 presents our methodology for the evaluation of different EEG classification pipelines and discusses the results of experiments conducted on two publicly available datasets.

Chapter 7 presents the setup and methodology for the implemented BCI system and details on the hardware & software stack and the software architecture.

Chapter 8, summarizes and concludes the work presented on this thesis. This final chapter provides also ideas for future development of this work.

Chapter 2.

Theoretical background and basıc concepts

Introduction

A brain-computer interface (BCI) acquires brain signals and provides to the Human Neural System a new output that is not neuromuscular or hormonal. In order to design such a system, a thorough understanding of the complexities of the human brain, both in structure and in function, are required. It's also important to investigate different approaches for acquiring the input signals and select the source that is more suitable for the use case of the designed BCI-system.

This chapter first presents the neurophysiological and anatomical basis of the function of the human brain. Next, we discuss various methods for capturing the brain activity and explain the advantages and drawbacks for each one. The chapter concludes with a brief introduction to Electroencephalography (EEG), the most commonly used method for brain signal acquisition and the one we are using for the development of our BCI system in this thesis.

The Building blocks of the brain

The human brain is the main organ of the human central nervous system (CNS). The brain consists of hundreds of thousands of cells, so-called neurons. Recent studies estimate that there are approximately 100 billion neurons in the human brain, which are all heavily interconnected.

A typical neuron consists of three parts: the **soma** or cell body, several **dendrites**, and the **axon**. The soma is usually compact; the axon and dendrites are filaments that extrude from it. The soma may give rise to numerous dendrites, but never to more than one axon. Dendrites typically branch profusely, getting thinner with each branching, and extending their farthest branches a few hundred micrometers from the soma.

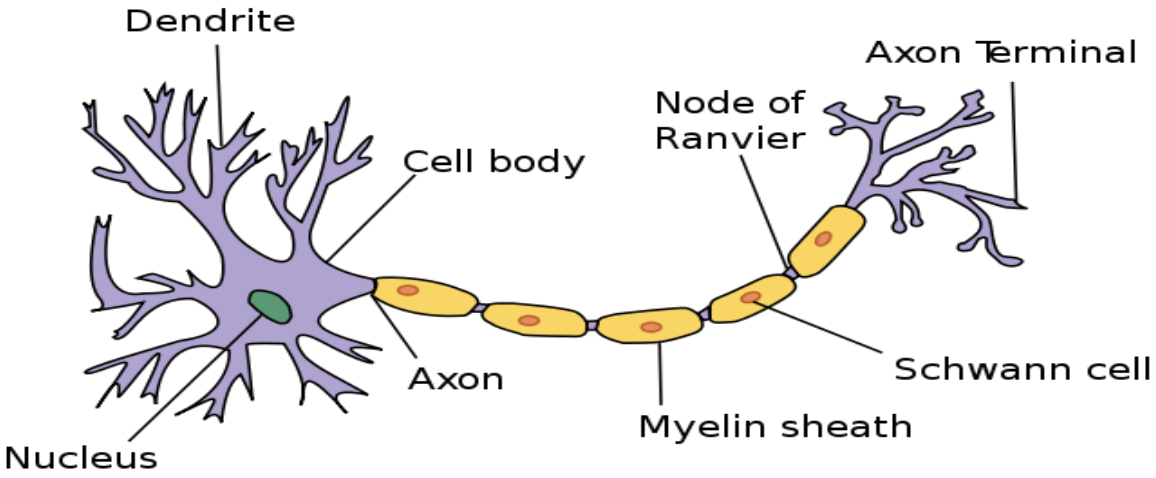


Figure 2: A typical neuron

Picture from Wikipedia under [CC BY-SA]

Methods for measuring brain functions

BCI must operate on observable effects of brain activity. Most BCI systems, operate on electrical effects of neural processes using Electroencephalography (EEG) and can detect large-scale neural dynamics (e.g. 10.000 neurons firing in near-synchrony). Other methods based on the metabolic changes of the brain during neural activity (e.g. PET, fMRI) can also be used, but are less suitable for BCI systems, because of their low temporal resolution.

From the methods for capturing signals that represent brain activity, Electroencephalography is the most commonly used and is the method used in this work. In order to give a more systematic and detailed description than the other methods, we dedicate the next section especially to it. In the remainder of this section, we summarize the main methods for capturing brain activity.

Electroencephalography

Electroencephalography (EEG) is an electrophysiological monitoring method to record the electrical activity of the brain. It is typically noninvasive, with the electrodes placed along the scalp, although invasive electrodes are sometimes used such as in electrocorticography (ECoG). One important strength of EEG is its high temporal resolution, which is in the range of milliseconds (Nunez & Williamson, 1996). This highly temporal resolution of EEG allows the capture of underlying physiological changes of the cognitive processes, most of which, occur within tens of millisecond.

Magnetoencephalography

Magnetoencephalography (MEG) is a non-invasive technique that measures the magnetic fields generated by neural activity. Like EEG, MEG has excellent time resolution and is often considered to capture deeper neural activity much better than EEG. One important advantage of MEG over EEG is the fact that the measured signals are not distorted by the body. However, the signal strengths are extremely weak and specialized shielding is required to eliminate the magnetic interference of the external environment.

MEG scanners are large, stationary and expensive and they require heavy technical maintenance and training resources.

Functional Magnetic Resonance Imaging

Functional Magnetic Resonance Imaging (fMRI) measures the metabolic changes that take place in the active parts of the brain, and more specifically, the changes in blood flow associated with neural activity. Increased neural firing increases the need for oxygen, which is delivered by the neighboring blood vessels. Because the magnetic properties of oxygenated blood are different from those of non-oxygenated blood, the increase in oxygenated blood is measured by fMRI as a distortion of the magnetic field generated by protons. fMRI has excellent spatial resolution (a few millimeters), but poor time resolution (comparable to EEG). Apart from the low time resolution, the disadvantages of fMRI include the fact that it measures the brain function indirectly and that it requires large-scale non-portable and expensive equipment.

Positron emission tomography

Positron emission tomography (PET) is an invasive nuclear imaging technique based on gamma radiation of a decay, which is inserted into the body of the subject. Like fMRI, PET monitors the metabolic activity (for example, blood flow, oxygen, and glucose metabolism) of neurons, and therefore provides an indirect measure of neural activity. While PET has a high spatial resolution, in the order of few millimeters, it is lacking in time resolution. The temporal resolution of PET varies from minutes to hours (Nunez & Williamson, 1996). Its main drawbacks, beyond the low temporal resolution, is the injection of a radioactive substance into the bloodstream and that it requires a large-scale non-portable and expensive equipment.

Electroencephalography

Overview

Electroencephalography (EEG) measures electrical activity on the scalp. This activity is the sum of the post-synaptic potentials generated by thousands of neurons having the same radial orientation (typically pyramidal neuron cells) with respect to the scalp.

Electroencephalography has been discovered in 1929, by Hans Berger. In a set of experiments in which electrodes were placed on the scalp, Berger described the electroencephalogram (the plotting of the changes in voltage over time) and suggested that brain electrical currents reflected the functional status of the brain such as sleep, anesthesia, and epilepsy. Over the following decades, EGG has been widely used in both scientific and clinical applications (e.g. to evaluate neurological disorders or to monitoring depth of anesthesia during a surgery).

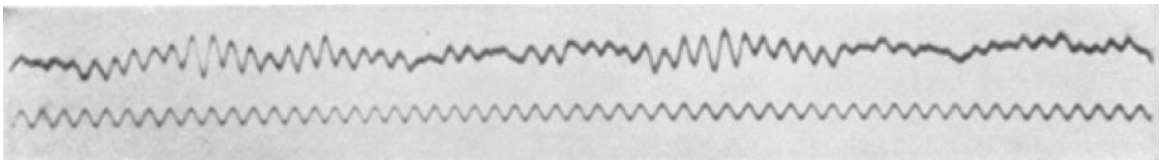


Figure 3: One of the first recordings of EEG signals made by Berger
By Hans Berger - Berger H. Über das Elektrenkephalogramm des Menchen. Archives für Psychiatrie. 1929; 87:527-70., Public Domain,
https://commons.wikimedia.org/w/index.php?curid=2900591

The electrical signals of the brain

The neural cells exchange signals with each other via the **synapses**, the structures that permit a neuron to pass an electrical or chemical signal to another neuron or to the target effector cell. Synaptic signals from other neurons are received by the soma and dendrites; signals to other neurons are transmitted by the axon. A typical synapse, then, is a contact between the axon of one neuron and a dendrite or soma of another. Synapses can act as inhibitory or excitatory gateways, preventing or propagating impulses across neurons. The synaptic transmission is triggered by the release of neurotransmitters

(dopamine, epinephrine, acetylcholine, etc.), which causes a voltage change across the cell membrane.

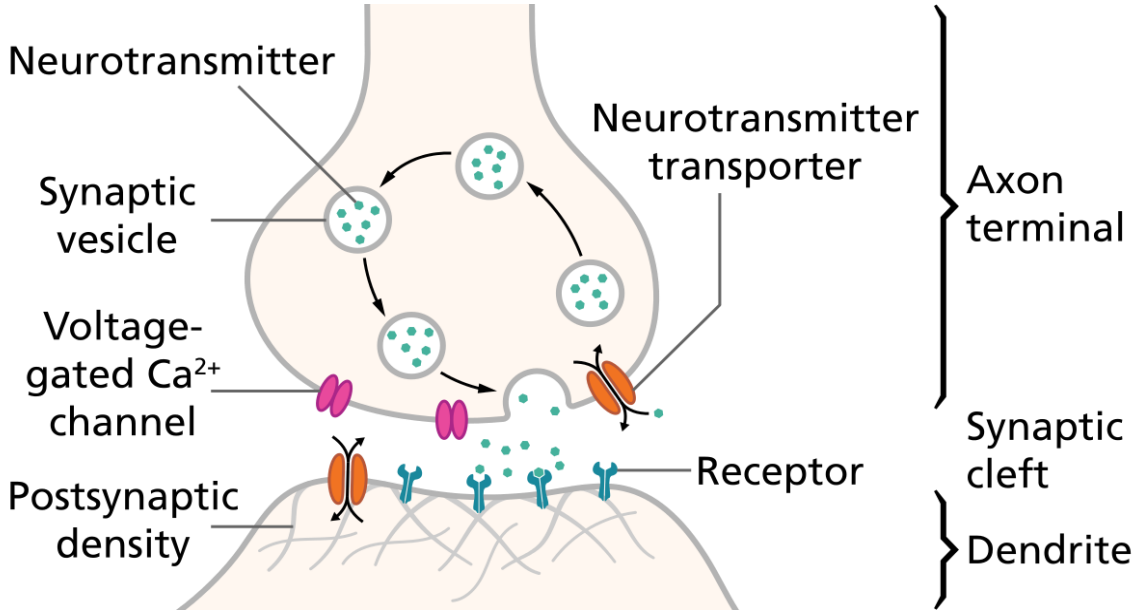


Figure 4: Structure of a typical chemical synapse

Image from Wikipedia

There are two main types of electrical activity associated with neurons, action potentials, and postsynaptic potentials.

- **Action potentials** (AP) are discrete voltage spikes that travel from the beginning of the axon at the cell body to the axon terminals, where neurotransmitters are being released.
- **Postsynaptic potentials** (PSP) are the voltages that arise when the neurotransmitters bind to receptors on the membrane of the postsynaptic cell, causing ion channels to open or close and leading to a graded change in the potential across the cell membrane.

If the PSP reaches the threshold conduction level for the postsynaptic neuron, the neuron fires and an AP is initiated (Atwood and MacKay, 1989). Because neurons rarely

fire at precisely the same time, action potentials in different axons will typically cancel, and the only way to record the action potentials from a large number of neurons is to place the electrode near the cell bodies and to use a very high impedance electrode that is sensitive only to nearby neurons. Whereas the duration of an action potential is only about a millisecond, postsynaptic potentials typically last tens or even hundreds of milliseconds. In addition, postsynaptic potentials are largely confined to the dendrites and cell body and occur essentially instantaneously rather than traveling down the axon at a fixed rate. Under certain conditions, these factors allow postsynaptic potentials to summate rather than cancel, making it possible to record them at a great distance (i.e., at the surface of the scalp).

Not all electrical fields generated by the brain are strong enough to spread all the way through tissue, bone, and skull towards the scalp surface. Research indicates that it is primarily the synchronized activity of pyramidal neurons in cortical brain regions which can be measured from the outside. Pyramidal cells can be found in all cortical areas (occipital, temporal, parietal, frontal cortices), where they are always oriented perpendicular to the cortical surface. The cell body is heading away from the surface (towards the grey matter), while their dendrite is heading towards the surface (for more details see Luck, 2014 and Buzsáki et al., 2012). This unique orientation of the pyramid cells generates an electrical field with a very stable orientation. EEG activity, therefore, represents a sum of the activity of millions of neurons having a similar spatial orientation. By contrast, the electrical fields from cells in deeper brain structures (such as brain stem or cerebellum) that don't have this specific orientation, are more likely to spread into various directions and cancel out instead of projecting in a stable way towards the scalp surface - even if hundreds of thousands of neurons in these deeper regions show synchronized activity.

Oscillatory activity

In general, EEG signals have a broad spectral content, but also expose oscillatory activity in specific frequency bands. Although the raw data obtained from EEG are formatted as a function of time, neural oscillations are usually visualized in terms of frequency using the Fourier transform and are measured in units of Hertz (Hz).

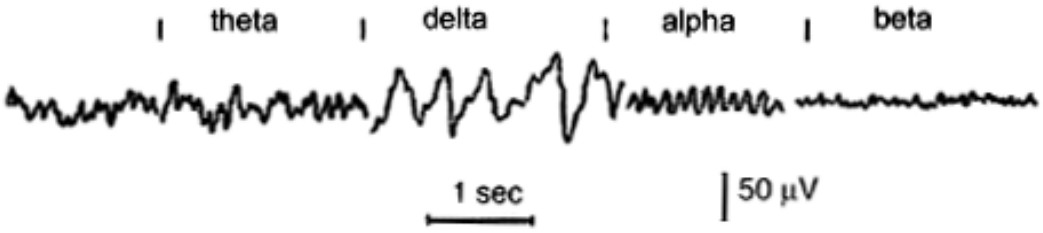


Figure 5: Neural Oscillations

Neural oscillations are visualized in terms of frequency.

From a diagnostic perspective, neural oscillations can be used as indicators of specific neurological phenomena such as sleep state, state of consciousness, perception and information processing, memory, abnormal neural function, such as epilepsy, and Parkinson's. There are 5 frequency bands categorizing the EEG signals based on their frequencies, named after Greek letters:

- delta (1–4 Hz), which are high amplitude waves mostly linked with slow-wave sleep.
- theta (4–8 Hz), which are mostly observed when the subject is in a meditative, daydreaming or in early stages of sleep.
- alpha (7.5–12.5 Hz), which are associated with cognitive functions such as relaxation and disengagement and are therefore the most commonly used in mood/meditation applications.
- beta (13–30 Hz), which are commonly associated with concentration or attention or more generally when. a subject is actively engaged in an activity.
- gamma (> 30 Hz), which are thought to play a crucial role in information processing, concentration, and learning.

Electrode arrays and placement

EEG recordings are performed using electrode arrays, comprising of a varying number of electrodes, depending on the scope of the experiment. Conventional EEG

systems use a conductive gel or paste for the electrodes, but many cheaper commercial systems use dry electrodes, reducing the preparation time, and making EEG more suitable for generic use.

For faster application, and consistent results between different recordings, EEG electrodes are mounted in elastic caps, meshes or rigid grids, ensuring that the data can be collected from identical scalp positions across sessions or respondents. Electrode locations and names are specified by an international system, named '10–20 system'. This system ensures that the naming of electrodes is consistent across laboratories and experiments. In the 10-20 system, electrode names begin with one or two letters indicating the general brain region where the electrode is placed (F = frontal, Fp = frontopolar; C = central; P = parietal; O = occipital; T = temporal). Each electrode name ends with a number or letter indicating the distance to the midline. Odd numbers are used in the left hemisphere, even numbers in the right hemisphere. Larger numbers indicate greater distances from the midline, while electrodes placed at the midline are labeled with a "z" (for zero).

Because the EEG voltage reflects the potential (or current) between two sites, different montages can be used for the electrodes.

- *In bipolar (or sequential) montage* the potential difference between a pair of electrodes is measured.
- *In unipolar (or referential) montage* the potential of each electrode is compared to a common neutral (reference) electrode. Typical reference positions are the tip of the nose, the cheek, and the right and left mastoids (the bony part behind left/right ears).
- *In an average referential montage*, the average of all electrodes is used as a reference point.

Advantages of EEG

EEG has several benefits compared to other imaging techniques. The most central benefit of EEG is its excellent time resolution, that is, it can take hundreds to thousands of snapshots of electrical activity across multiple sensors within a single second. This makes

EEG an ideal technology to study the precise time course of cognitive and emotional dynamics most of which, occur within tens of millisecond. The second reason that justifies EEG as such an advantageous technique for the study of neurocognitive processes is that it allows the direct measure of neural activity. EEG signals directly reflect biophysical phenomena occurring on neuron populations. This is a clear advantage over other methods such as fMRI that do not directly measure neural activity but introduce an extra relationship between what is measured (changes in blood flow in the case of fMRI) and the actual neural activity. Finally, EEG is non-invasive, and the required equipment is relatively cheap, portable and relatively easy to operate.

Disadvantages of EEG

The main disadvantage of EEG is its poor spatial resolution. Neural activity is conducted through the brain volume to the scalp and electrodes by *volume conduction* (the transmission of electric or magnetic fields from an electric primary current source through biological tissue towards measurement sensors). The concept of volume conduction carries important implications for surface EEG measurements because it means that (a) currents are not restricted to the immediate neighborhood of the source, and (b) the electrical activity measured between electrodes has more to do with their orientation to the actual generator than with the proximity of the electrodes to the generator. Because the skull is a poor conductor current tends to "splash off of it" and each electrode receives signals from millions of neurons, reducing any potential spatial localization. This is exacerbated by the fact that the conductivities of the head tissues, varies across individuals and within the same individual due to variations in age, disease state, and environmental factors. The inference of the location of the current sources from electrode voltage measurements on the scalp is known as **the EEG inverse problem** and is comparable to reconstructing an object from its shadow: only generic features (the shape) are uniquely determined, others must be deduced on the ground of additional information.

EEG is also very sensitive to subject movement and external noise. Electrodes used in EEG recording do not discriminate the electrical signals they receive. The recorded activity which is not of cerebral origin is termed artifact. Artifacts are noncerebral signals that often contaminate the recordings in both temporal and spectral domains within a wide

frequency band. The internal source of artifacts may be due to physiological activities of the subject (e.g., eyes movement, electrocardiographic activity, sweat or muscle artifacts) or its movement. External sources of artifacts are environmental interferences such as electrical noise from mains interference, bad contacts between electrode and skin, or interferences from recording equipment and cable movement.

Chapter 3.

EEG-based BCI Systems

Introduction

A brain-computer interface is a direct communication pathway between a brain and an external device. There are four criteria that need to be met, for a system to function as a BCI system:

1. **Direct**: The system must rely on activity recorded directly from the brain.
2. **Intentional control**: At least one recordable brain signal, which can be intentionally modulated, must provide input to the BCI (electrical potentials, magnetic fields or hemodynamic changes).
3. **Real-time processing**: signal processing must occur online and yield a communication or control signal.
4. **Feedback**: The user must obtain feedback about the success or failure of his/her efforts to communicate or control.

Although that, as we have already mentioned, there are various types of bio-signals that can be used to measure brain activity and act as an input for a BCI. In the rest of this thesis, we will discuss and describe BCI systems that are based on EEG signals in order to function. The main advantage of using EEG as an input signal for a BCI is that EEG (using dry electrodes) is the easiest and least invasive method.

The most difficult task in designing real-time BCI systems is interpreting the recorded data. First, EEG data are a superposition of the brain signals of interest, with a plethora of other signals from other brain regions, muscles, and from non-biological artifacts. Second, brain activity exhibits a huge variability across subjects. Since neural responses are different across subjects even for the same stimulus, almost all EEG-based brain-computer interfaces need some labeled, subject-specific, data to calibrate a new subject. As a result, a major challenge in developing high-performance and user-friendly BCIs is to cope with such individual differences, so that the calibration can be reduced or even completely eliminated (He, Wu, & Member, 2018). In order to overcome these

problems, state of the art BCI systems need to use continuous adaptive signal processing and machine learning algorithms, to extract meaningful information from brain signals. Until recently, the requisite technology to adapt and analyze the EEG in real-time, either did not exist or was extremely expensive. The improvements in processing power along with the developments in machine learning algorithms allow the introduction of small portable BCI systems suitable for every-day use.

BCI Applications

The ability to control the world around us, using only our mind, has been a feature of some of the best science fiction stories. Still, a lot of research is taking place around the world for advanced brainwave-based digital interfaces and in recent years remarkable strides have been achieved toward this goal. Brain-computer interfaces are already been applied in various fields of research.

One of the main areas of applied research concerns the use of BCIs in medical applications. Such systems are mainly targeting people with severe disabilities such as tetraplegia, locked-in syndrome etc. The aim of these BCIs is to either restore movement of individuals with paralysis or provide some special devices to assist them (POSTELNICU, TALABĂ, & M.I, 2010). In other cases, the research targets the restoring of communication with the external environment. The brain-controlled speller is one of the most famous BCI paradigms as it allows communication disabled people to spell letters or words. MindDesktop (Ossmy et al., 2017) is a medical purpose BCI allowing people with severe disabilities to operate any Windows-based operating system. Another area of medical BCIs is focusing on motor neuroprosthetics. For example, artificial arms or legs are being controlled by a portable BCI system, thus allowing individuals with a paralyzed or missing body part to move or grasp.

Gaming is among the most promising applications for BCI systems mainly because the number of potential users in BCI games is very high. BCI can be used either as a primary controller of a game or as an extra channel for special actions. For example, the user may imagine body part movement or concentrate on a certain object to generate brain signals that reveal the user's intention. Studies have shown promise in achieving 2D or 3D navigation, including moving a computer cursor or walking in a virtual world. Several studies have demonstrated the use of BCI for controlling popular games such as Tetris (Pires, Torres, Casaleiro, Nunes, & Castelo-Branco, 2011) or World of Warcraft (Van De Laar, Gurkok, Plass-Oude Bos, Poel, & Nijholt, 2013). An example of use in Virtual environments was demonstrated by the He's research group in studies that showed that human subjects could fly a virtual helicopter in a 3D virtual world using EEG signals recorded from the scalp (Doud, Lucas, Pisansky, & He, 2011). A different approach

promotes the use of BCI for mental state monitoring during gaming, in order to make adaptive and dynamic games.

Figure 6: Brain-computer interfacing controlled Tetris game

A volunteer is playing a BCI-controlled version of the Tetris computer game. He uses left- and right-hand motor imagery to move the falling pieces horizontally, mental rotation to rotate it clockwise and foot motor imagery to let it drop.

Another active field or research concerns BCIs that allows for continues mental state monitoring. There is a wide range of everyday application that can be enhanced by adding passive mental state monitoring. When aiming to optimize the design of user interfaces or, more generally, of a workflow, the mental state of a user during task execution can provide valuable information. This information can be exploited for the improvement of industrial production environments, the user interface of cars and for many other applications. Examples of these mental states are the levels of arousal, fatigue, emotion, workload or other variables the brain activity correlates of which are (at least partially) accessible by measurement (Blankertz et al., 2010).

More futuristic research proposals include the ability of "telepathic" communication between people, or the use of brain waves for user authentication. Published studies have demonstrated direct transmission of brain activity between two

humans (Rao et al., 2014), between two animals, and even between human and rat (Yoo, Kim, Filandrianos, Taghados, & Park, 2013). DARPA, the Pentagon's technology research division, is currently working on an initiative called "Silent Talk," which would let soldiers on secret missions communicate with their thoughts alone. Various studies have proposed the use of brain signals as a two-factor, changeable, authentication method resistant to shoulder-surfing.

Major Components of a BCI system

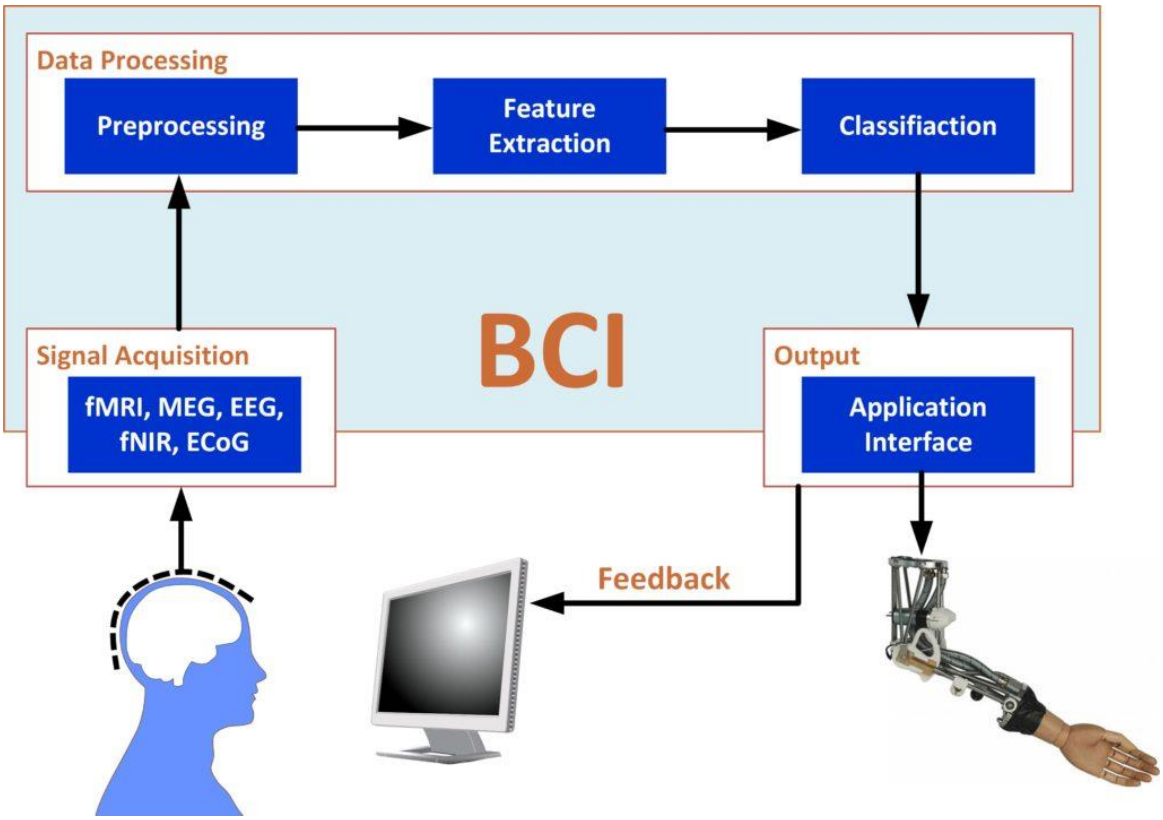


Figure 7: A typical BCI pipeline

A typical BCI pipeline. Image from https://www.mne-cpp.org/wp-content/uploads/2015/08/BCI_processing_pipeline-1024x717.jpg

From the technical point of view, a BCI system consists of at least 5 components:

- **Signal Acquisition**: Although there are various methods to acquire brain signals, the one most suitable for real-time applications is Electroencephalography (EEG). With EEG, brain signals are recorded on the scalp of the users using electrodes. This preferably happens in a non-invasive manner, using external scalp electrodes or even better, dry EEG electrodes making direct contact with the skin without requiring the application of electrode gel.

- **Signal Preprocessing**: The measured signals are quite weak. Even worse, existing electric network current or muscular movement and eye-blinks can greatly influence them. Therefore, complex algorithms are applied to a) filter and b) enhance the raw signal quality and increase the signal to noise ratio (SNR).
- **Features Extraction**: Signal processing techniques are applied to extract features that can be used later by a machine learning algorithm. Depending on the task, different methods can be used, e.g. spatial information to identify signal on certain areas of the brain, spectral information that represents frequency bands of interest, or temporal information representing the change of signal over time.
- **Machine Learning**: The extracted features are analyzed with modern machine learning methods to discriminate between different classes of commands. Most methods conform to a common framework of a training, evaluation and prediction function. Typical classifiers for BCI applications are Linear Discrimination Analysis, Support Vector Machines and various types of Neural Networks.
- **Control and Feedback**: the predicted target from the previous step is then used to control an external device or application. For most BCI systems, the device is a computer screen and the output is the selection of certain targets. Advanced applications include the controlling of external devices such as prosthetic, robotic or smart-home devices.

Types of BCI systems

According to Zander et al (2009) BCI systems can be classified in one of the following types:

- **Active BCIs**: which derives its outputs from brain activity which is directly consciously controlled by the user, independently from external events, for controlling, for instance, an application.
- **Reactive BCIs**: which derives its outputs from brain activity arising in reaction to external stimulation, which is indirectly modulated by the user for controlling an application.
- **Passive BCIs**: which derives its outputs from arbitrary brain activity without the purpose of voluntary control, for enriching a human-computer interaction with implicit information.

A BCI usually includes a piece of software responsible for signal analysis and pattern recognition to achieve the translation of raw brain activity to control signals. Depending on whether this analysis happens in real-time or not, BCI systems can further be split into synchronous, which only analyzes the signals during pre-defined time windows, and asynchronous, which always looks at the signals seeing if there is a command pattern present. An asynchronous BCI is always active and besides reacting to the predetermined mental tasks that control the system, is also able to identify (and ignore) rest states. Synchronous is much easier to implement, but asynchronous offers much more seamless interaction. (Heyden, 2016).

Another distinction is that between one-directional and two-directional BCI systems. **One-directional** refers to BCIs that only "read" brain activity and affect the external environment of the user. In **two-directional (**or closed-loop) BCIs, the system can also affect the brain, e.g., by electric stimulation of the brain's reward center. The development of closed-loop systems raises intriguing ethical issues and requires additional research and clear governing policies.

Recently, novel approaches have been proposed for BCIs that combine various, possibly diverse, types of signals. These are called **hybrid BCIs**. Input signals used in hybrid BCIs can either be two different types of brain imaging methods (e.g., EEG and

functional Near-Infrared Spectroscopy, fNIRS), or the combination of one brain signal with another physiological signal (e.g., heart rate, eye tracker etc.).

EEG devices

Encephalographic measurement devices are consisting of 1) electrodes, 2) amplifiers with or without filters, 3) an A/D converter and 4) a recording device. Electrodes read the signal from the head surface, amplifiers bring the microvolt signals into the range where they can be digitalized accurately, converter changes signals from analog to digital form, and a storage device (or more commonly a personal computer) stores obtained data.

Until recently, the collection of electroencephalographic (EEG) data used to be associated with expensive (>$25,000 USD), large electrode array systems. However, in the past ten years, there has been a rapid increase in the availability and number of "low-cost" EEG systems available to researchers (Krigolson, Williams, & Colino, 2017). At the lower end of the cost, there are devices costing less than 1,000 euros, although the research potential of such devices is ultimately limited by the few numbers of channels. A wide range of options, with up to 64 channels, is available at the middle price range, with devices costing between 1,000 and 25,000.

Apart from the cost, for EEG technology to become widely-adopted and more user-friendly, EEG systems must move from the bulky systems used in clinical settings to sleek, convenient compact systems. At the hardware level, there are currently three types of approaches that aim to improve EEG-based BCI portability and usability. First, the transition from the bulky EEG devices that consists of different components for the recording and the amplification of the EEG signal, to more compact, wearable devices with embedded parts. Second the replacement of gel or water-based EEG electrodes with dry electrodes that do not require the application of conductive gel. And third, the wireless connection between the amplifier and the recording device, which eliminates the need for cables and allows their use in a wide range of new settings. While such systems are already offered for end-users and consumers and can be used for BCI applications, the signal quality of such EEG hardware is not yet comparable to professional devices and their use is not yet recommended for serious applications.

Commercial EEG headsets

There are a lot of "low-cost" EEG headsets out in the market. These EEG headsets are often mentioned in different terms such as "Mind Controller", "Brainwave Controller", "EEG headbands", etc. Most of these devices have the lowest number of electrodes and low sampling rates.

OpenBCI is an open-source brain-computer interface device, created after a successful Kickstarter campaign in 2013. Today the company behind OpenBCI offers various versions of the device, allowing to choose between 4 and 16 channels and different boards with or without Bluetooth connectivity. The cost for an 8 channels OpenBCI board along with a headset is around 1,000 euros.

Emotiv offers 5 and 14 channel solutions. The internal sampling rate of the device is 2048 Hz, but the data is then down-sampled to 128 Hz before becoming available to the user. Emotiv's devices are also wireless, giving the possibility of more free movement to the user. The cost for the 14 channels device is around 700 euros. Currently, access to the raw EEG data of the Emotiv comes with an additional cost-per-usage, which can increase significantly the total cost of use.

The lower end of the consumer available devices includes devices with the lowest number of electrodes. Companies like NeuroSky and Muse offer neurofeedback solutions that are targeting meditation and monitoring uses, with limited research potential. Neurosky MindWave is at the lower end of usability of low-cost consumer EEG devices with only one electrode (placed at Fp1) and a 128Hz sampling rate. Muse offers a device with 4 channels and at a prize around 300 euros.

EEG-based paradigms for BCI systems

For the development of a BCI system, besides having the hardware to capture brain signals, it is also important to know which parts of the brain are responsible for certain mental processes and how the signals in the brain behave under these mental processes. Using certain mental processes to activate brain regions to control a device or a computer program is called **paradigms** (Strategien et al., n.d.).

There are two main paradigms that are used as control signals for BCIs. **Event-Related Potentials** and **Sensorimotor Rhythm Activation.** The former uses brain activity generated in response to specific visual or auditory stimuli while the latter uses activity spontaneously generated by the user or by the user's mental state.

Event-Related Potentials

Event-Related Potentials (ERPs), are characterized by a phase-locked, time-domain waveform that appears in response to stimulation. Typical features are time-domain signal, generally averaged across several repetitions of the stimulation in order to increase the signal to noise ratio.

ERPs are perhaps the most studied type of activity in EEG and has been used in cognitive science, cognitive psychology, and psychophysiological research. In ERP studies, the EEG is recorded from participants, as experimental stimuli are presented. In this context, the cognitive "events" of interest may include a particular class of stimulus, the absence of an expected stimulus (omitted stimulus paradigm), a correct or incorrect response, among many other possibilities, as long as a distinct time point for ERP time - locking can be defined. Segments of the EEG, each encompassing a fixed period of time before and/or after each instance of an event, is then averaged to yield an average ERP. Averaging over multiple trials eliminates unrelated background activity that is random with respect to the stimulus and thus averages to zero, given enough trials. The resulting ERP reveals brain activity that is related, and synchronized in time and phase, to the presented stimulus (Faust, 2012). The averaged ERP waveform consists of a series of positive and negative voltage deflections, which are called components. Years of research have helped

to link different components to specific cognitive processes, making ERPs a powerful technique for examining the nature of cognitive and neural processes.

There are many advantages of event-related potentials for the study and analysis of complex neuro-cognitive processes. ERPs are simple and fast to compute, mainly because ERPSs generally involve significantly fewer data. They have a highly temporal resolution, providing continues measure of processing, including both ex-stimulus and post-stimulus activity. This highly temporal resolutions of ERPs allow the measurement of brain activity with the precision of millisecond, which is particularly important if we consider that many aspects of attention and perception appear to operate on a scale of tens of milliseconds. Besides, ERP-based BCIs have the advantage that usually do not require subject-specific training sessions, making it suitable for generic-use (no learning sessions are required in order to adapt the system to new users). The main drawback regarding the use of ERPs for the development of BCI systems is their dependency on the external stimuli, which prohibits the development of voluntary controlling interfaces suitable for active BCIs.

Sensorimotor Rhythm

A different control signal for active BCIs is the sensorimotor rhythm (SMR) that is based on the neural oscillations. These oscillations appear naturally in ongoing EEG activity and are representative of a wide range of different cognitive states (e.g. sleep stage, meditation, etc.) or can be induced by a specific task, for example, a hand movement or the performance of mental calculus. Sensorimotor Rhythm Activation is characterized by a change in signal power in specific frequency bands. The SMR modulation manifests as a decrease in the alpha (also known as mu rhythm) and beta frequency bands accompanied by an increase in the gamma frequency band. Typical features are extracted using fast Fourier transform based algorithms, or more simply variance/covariance of the signal after frequential filtering.

BCI systems based on oscillatory activity are functioning by detecting patterns in mental states which lead to changes in the oscillatory components of EEG signals, i.e., that lead to change in the power of EEG signals in some frequency bands. The increase of EEG signal power in a given frequency band is called an Event-Related Synchronization (ERS), whereas a decrease of EEG signal power is called an Event-Related Desynchronization

(ERD). Unlike ERPs, ERD/ERS are not phase locked to a stimulus presentation and, therefore, cannot be identified by averaging the EEG amplitudes, instead, band-power is measured in frequency bands of interest, localized in some specific brain areas. As such, they naturally need to exploit both the spatial and spectral information. The original EEG signals are converted to time-frequency signals by applying a function of short time Fourier transforms (STFTs) and are localized using either channel selection or spatial filtering techniques.

Imagining a movement or performing an action mentally is known as Motor Imagery (MI). Studies based on fMRI revealed that imagery and executed movements had similar activation patterns (Lotze et al., 1999). Motor Imagery is very commonly used in BCI systems because it allows the development of asynchronous active BCIs. The main advantage of MI-based BCIs is that it allows the user to control the system spontaneously, by imaging the execution of a movement. On the other hand, they suffer from high variability across and within subjects and therefore require excessive user training and long calibration times in order to achieve reasonable performance.

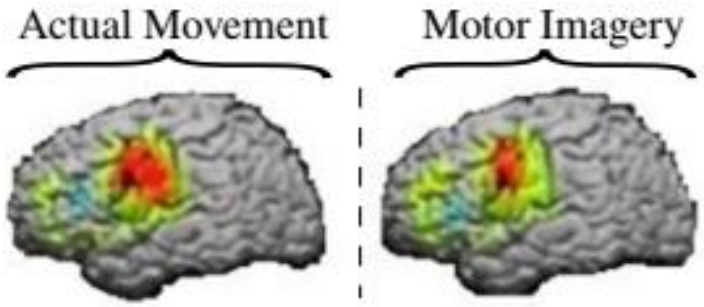


Figure 8: fMRI scans of actual and motor imagery

Chapter 4.

Machine Learning

Introduction

Machine learning (ML) is a field of artificial intelligence that uses statistical techniques in order to give information systems the ability to "learn" from data, without being explicitly programmed. The concept of learning here means the progressively improving of their performance on certain future tasks. We say that learning a general function or rule from specific input-output pairs is called inductive learning (Russell, Norvig, & Davis, 2010). Predictive modeling is the problem of developing a model using historical data, which will be able to make predictions on new (unseen) data. Typically, a model includes a machine learning algorithm that learns a function from a training dataset in order to make predictions.

According to the feedback that is available to learn from, machine learning algorithms can be grouped into three main types:

- **Unsupervised learning**: where a learning agent discovers patterns in the input data without any explicit feedback. The most common unsupervised learning task is clustering: the detection of potentially useful cluster of input instances.
- **Reinforcement learning**: Where an agent “learns” from a series of awards or penalties in a series of interactions with the environment. In each step of the learning process, the agent is informed about the state of the environment and decides which action to perform. For every action, the agent gets feedback from the environment in the form of numerical rewards, that is positive if the action was correct, or negative when the action was incorrect. This procedure helps the agent to form a policy for associating the correct actions to states. To maximize the long-term reward, the agent must explore its environment and update its policy to incorporate the discovered knowledge into the policy.
- **Supervised learning**: Where the learning agent has access to training data consisting of examples of input-output pairs and learns a function that maps from input to output. A supervised learning algorithm analyzes the training data and

produces an inferred function, which can be used for mapping new unseen examples. The discovered function is represented in a structure referred to as a **model**. There are two major categories of supervised learning, regression, and classification. **Regression models** are based on the analysis of relationships between variables and trends in order to make predictions about a continuous target variable, while the task of **classification models** is to assign discrete class labels to observations. In supervised classification, the class labels in the dataset, which is used to build the classification model, are known.

In the following sections, we will describe in more details the theory and the main algorithms for supervised classification, as it is the learning model that has been used to develop the BCI System for this thesis.

Supervised Classification

More formally, the task of a supervised classification learning algorithm is this:

Given a training set of N tuples of examples – class variable pairs

$$(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N) ,$$

where each y_i was generated by an unknown function $y = f(x)$

discover a function h that approximates the true function f.

An **example** is a collection of features that have been measured from some object, fact or event. We typically represent an example as a vector $x \in R^n$ where each entry x_i of the vector is called a **feature**. A **training set** is a collection of tuples of examples, each one associated with a **label** or **class**. Classification is made by choosing the best possible function h that assigns each feature vector x to a class y_i based on the samples in the training set. The function h is called a **hypothesis**. Learning is a search through the space of possible hypothesis for one that will perform well on unobserved data. The simplest kind of classification problem is **binary classification**, when there are only two values for the output class.

Performance Evaluation for Supervised Classification

To evaluate the performance of a classification algorithm, we usually measure the accuracy of the model, which is defined as the proportion of examples for which the model produces the correct output. The accuracy measured on the training dataset is called **training accuracy**. The inaccuracy of predicted output values is termed training error or the **training error rate**.

Usually, we are interested in how accurately a classification algorithm can predict outcome values for previously unseen data. Measurements of accuracy or error on the training data do not provide much information about predictive ability on new data. We, therefore, evaluate the algorithm on a previously unseen dataset, called the **test dataset**. That accuracy is often called **testing accuracy** or simply **accuracy**. The error rate on the test dataset is called test error or **generalization error**. We say that an algorithm **generalizes** well if it correctly predicts the value y for previously unobserved examples.

In the end, the factors determining how well an algorithm will perform are its ability to a) make the training error as small as possible and b) make the gap between training and test error as small as possible. These two goals correspond to the two central challenges in machine learning: underfitting and overfitting. **Underfitting** occurs when the model is not able to obtain a sufficiently low training error, indicating that the learning algorithm cannot adequately capture the underlying structure of the data. **Overfitting** occurs when the gap between training and test error is too large, indicating that the learning function corresponds too closely or exactly to the training set, and therefore fails to fit additional data or predict future observations reliably.

In some cases, there are parameters in the classifier that must be tuned. These are usually chosen by splitting the data into three instead of two sets: Training, testing, and **validation**. The classifier is trained on the training set for different parameter configurations (also called **hyperparameters**) and evaluated on the validation sets. The one that performs best is then chosen and tested on a previously untouched test set to yield its generalization performance by obtaining the performance characteristics such as accuracy, sensitivity, specificity, F-measure, and so on.

Resampling

When the amount of data is limited, it is common practice to re-sample the data, that is, partition the data into training and test sets in different ways. By doing so, a more reliable estimate of the true generalization error of the inducer is estimated. There are various methods to re-sample the data, and the choice between them depends both on the size of the dataset and the specificities of the learning problem.

- **Single Random Sampling:** This method is mainly used when the dataset is substantially large. The dataset is divided randomly into two separate subsets, the training set, and the test (or holdout) set. A general rule-of-thumb suggests an 80-20 split, thus 80% of the examples for training and the rest 20 % for testing. The training set is used for learning a classifier and the test set is used for evaluating the classifier. The training set should not be used in the evaluation as the classifier is biased toward the training set. That is, the classifier may overfit the training data, which results in very high accuracy on the training set but low accuracy on the test

set. Using the unseen test set gives an unbiased estimate of the classification accuracy.

- **Multiple Random Sampling:** In cases where the available dataset is small, splitting the dataset into training and test subsets, can be unreliable because the test set would be too small to be representative. One approach to deal with the problem is to perform the abovementioned random sampling multiple times. Each time a different training set and a different validation set are produced. This results also to multiple accuracies. The final estimated accuracy on the data is the average of the resulted accuracies. When randomly selecting training or validation sets, we may want to ensure that class proportions are maintained in each selected set. This can be done via stratified sampling (first stratify instances by class, then randomly select instances from each class proportionally).

Cross-Validation

Another widely used approach when the dataset is small is the n-fold cross-validation method. In this method, the available data is partitioned into n equal-sized disjoint subsets. Each subset is then used as the validation set and the remaining n-1 subsets are combined as the training set to learn a classifier. This procedure is then repeated n times, which gives n accuracies. The final estimated accuracy of learning from this dataset is the average of the n accuracies. In stratified cross-validation, stratified sampling is used when partitioning the data, to maintain class proportions in each set.

An extreme case of cross-validation is the leave-one-out cross-validation (or LOO for short) method. In this method, the number of folds is set to the total number of examples in the dataset so that each example is given a chance to be the held out. In such a case, if the original data has m examples, then it yields to a m-fold cross-validation. This method is mostly used when the available data is very small, but it is not efficient for a large dataset as it has very high computational costs.

Another special case of cross-validation, frequently used while evaluating EEG datasets, is the leave-one-group-out cross-validation (or LOGO for short). In this method, the examples are partitioned based on an external group property and the number of folds is set equal to the number of groups. This method is used on EEG datasets containing data from different subjects, to perform cross-subject evaluation.

Supervised Classification Algorithms

Linear Classifiers

Linear classifiers are a family of algorithms that learn a linear function to distinguish classes by separating input vectors using linear (hyperplane) decision boundaries. Linear classifiers work well for problems with many variables (features), reaching accuracy levels comparable to non-linear classifiers while taking less time to train and use. Logistic Regression and Linear Discrimination Analysis are the main algorithms belonging to this category.

Classification via Logistic Regression

The goal of the Logistic Regression algorithm (LR) is to find the best fitting and most parsimonious model to describe the relationship between an outcome (dependent variable) and a set of independent variables. LR can be applied to more than two categories but in its simplest form called **Binary Logistic Regression**, there is only one binary output variable with two possible values.

Logistic regression is a special case of the generalized linear model and thus analogous to linear regression. In linear regression, we try to find the parameters (theta values) to minimize a special cost function$J(\Theta)$, so that a hypothesis of the form$h_\theta\ (x) = \theta_0\ + \theta_1 x_1 + \cdots +\ \theta_n x_n$, outputs a value for $h_\theta\ (x)$ as close as possible to y for all the input-output pairs of x and y.

Logistic regression uses the same basic formula, but instead of the continuous output, it is regressing for the probability of a categorical outcome. LR measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities which are restricted to (0,1) through a logistic sigmoid function of the form $f(z) = \frac{1}{1+ e^{-z}}$. In the case of binary LR, if the probability in the modeled class (usually the positive class) is above some cut point (the default is 0.50), the example is predicted to be a member of the modeled class otherwise the example is predicted to be a member of the other class. For multinomial LR the probability scores are calculated for all the possible classes, instead of just the positive class.

LDA

Linear Discriminant Analysis (LDA) is a classification method originally developed in 1936 by R. A. Fisher. LDA is based upon the concept of searching for a linear combination of variables (predictors) that best separates two classes. LDA classifies samples by projecting them onto a vector and thus obtaining a scalar value $y = \underline{w}^T \cdot \underline{x}$. The goal then is to find a vector $\underline{w}$ which maximizes the separability of the projections of a given training set $\underline{X}$. This is achieved by first maximizing the difference between the projected means, and then minimizing the interclass variance.

In the simplest case of binary classification, the two classes are assumed to be normally distributed with different means but identical full rank covariance matrix ($\sum c1 = \sum c2$). Suppose the true means μ_i $(i = 1,2)$ and the true covariance matrix $\sum$ are known, then the normal vector w of the Bayes optimal separating hyperplane of the LDA classifier is given as $w = \sum^{-1}(\mu_1 - \mu_2)$.

LDA makes more assumptions than linear regression, requiring a normal distribution of the data, with equal covariance matrix for classes. However, when these assumptions are met, LDA is more powerful than logistic regression.

Support Vector Machine

Support Vector Machine (SVM) classifies samples by constructing a linear hyperplane separating classes with a maximized margin on either side. In this regard SVM is a linear classifier. However, SVM can also perform a non-linear classification by constructing a hyperplane in a higher dimensional space, allowing the separation of data that are not linearly separable in the original input space. This is achieved by mapping the original data with a kernel function that computes the inner-product between two projected vectors. The most commonly used kernel is the Gaussian kernel, also called RBF kernel.

The basic SVM algorithm is designed for the classification of examples into two possible classes. For multi-class classification problems, the most common approach is called *One-against-all,* and it works by first finding a discrimination hyperplane between each class and all the rest and then combining the results using a maximization rule (each example is assigned to the class corresponding to the SVM that outputs the largest score).

Nearest Neighbor Classifiers

The intuition underlying Nearest Neighbor Classification is quite simple, examples are classified based on the class of their nearest neighbor. It is often useful to take more than one neighbor into account, so the technique is more commonly referred to as k-Nearest Neighbor (k-NN). This type of learning is a type of instance-based learning, or lazy learning, where the function is only approximated locally, and all computation is deferred until classification. Because classification is based directly on the training examples it is also called Example-Based Classification or Case-Based Classification. During the classification stage for a given testing example, the k-NN algorithm directly searches through all the training examples by calculating the distances between the testing example and all the training data. The distance between two examples is calculated by a similarity measure (or distance function). The Euclidean distance is the most widely used distance function but there are several other types of distance functions, such as cosine similarity, Minkowsky and Chi square.

Simple Bayesian Classifiers

Simple Bayesian classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem for classification on new examples using the conditional probability model. The naive Bayes classifier is the simplest of these models.

Naive Bayes classifier is making strong (naive) assumptions about the input data, by considering each of the input features to contribute independently to the probability that an example belongs to a specific class. It then assigns probabilities to each class using Bayes' theorem and selects the one with the maximum a posterior probability rule.

Different types of naive Bayes classifiers rest on different assumptions about the distributions of features. These assumptions are called the event model of the Naive Bayes classifier. In general, the Naive Bayes classifier is not linear, but if the likelihood factors $p(\ xi\ \mid \mathrm{x})$ are from exponential families, the naive Bayes classifier corresponds to a linear classifier in a particular feature space. An important advantage of Naive Bayes classifier is that it only requires a small number of training data to estimate the parameters necessary for classification.

Decision Trees

Decision trees is one of the simplest and yet successful families of machine learning algorithms as they are capable of modeling complex nonlinear decision boundaries. Tree models where the target variable can take a discrete set of values are called classification trees. A classification tree is a multistage decision process which instead of using the complete set of features jointly to decide, uses different subsets of features at different levels of the tree.

There are many specific core algorithms for building decision trees. The most famous is ID3 by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses Entropy and Information Gain to construct the decision tree.

Decision-tree can create over-complex trees that do not generalize well from the training data (overfitting). Mechanisms such as pruning are often used to avoid this problem, by eliminating nodes based on statistical significance tests.

Deep Learning and Artificial Neural Networks

As already mentioned, neuroscience is based on the hypothesis that mental activity consists primarily of electrochemical activity in networks of brain cells called neurons. Inspired by this hypothesis, some of the earliest AI work aimed to create artificial neural networks (Russell, Norvig, & Davis, 2010). Today, artificial neural networks (ANNs) are deployed on a large scale, particularly for image and visual recognition problems and belong to an architectural approach in Machine learning, called Deep learning. Deep learning is part of the broader family of machine learning methods based on learning data. The term Deep is referring to the many layers involved.

Unlike the biological brain where any neuron can connect to any other neuron within a certain physical distance, artificial neural networks have discrete layers of neurons, connections, and directions of data propagation. Each level of neurons learns to transform its input data into a slightly more abstract and composite representation. Each of the connections has a number associated with it called the connection weight and each of the neurons has a number and a special formula associated with them called a threshold value and an activation function respectively. These are the parameters of the neural network. When a neural network is being trained, it learns to adjust its weights and threshold values to arrive at the correct output.

One of the key ideas behind deep learning is to extract high-level features from the given dataset. Thereby, deep learning aims to overcome the challenge of the often-tedious feature engineering task.

MultiLayer Perceptron

The most widely used Neural Network architecture is the **MultiLayer Perceptron** (MLP) or deep feedforward network. An MLP is composed of at least three layers of neurons: an **input layer**, one or more hidden layers, and an output layer. The number of layers is called the **depth** of the model. Each neuron's input is connected with the output of the previous layer's neurons whereas the neurons of the output layer determine the class of the input feature vector. (Lotte et al., 2007)

MLP is a feedforward artificial neural network in which connections between the nodes do not form a cycle and the information moves only forward, from the input nodes, through the hidden nodes and finally to the output nodes. Except for the input nodes, each node simulates a neuron using a nonlinear activation function that models the firing of action potentials of biological neurons.

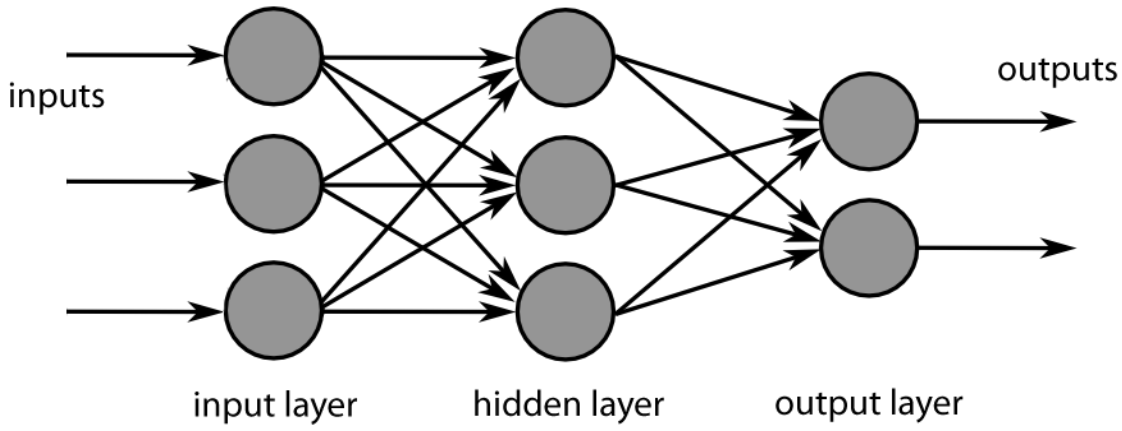


Figure 9: Layers of the MultiLayer Perceptron

Picture from Wikipedia with CC BY-SA license.

Learning occurs in the perceptron by changing connection weights after each piece of data is processed, based on the amount of error in the output compared to the expected result. In this sense, training of a neural network is like training any other machine learning model using gradient descent. The largest difference between the linear models and neural networks is that the nonlinearity of a neural network causes most interesting loss functions to become nonconvex. This means that neural networks are usually trained by using iterative, gradient-based optimizers that merely drive the cost function to a very low value (Goodfellow, Bengio, & Courville, n.d.). This is carried out through backpropagation, an algorithm that adjusts the weight of the network during the training phase.

Feedforward networks with at least one layer of hidden units have been proved to be universal function approximators: Given a sufficient number of hidden units, a network can approximate any continuous function of the inputs in the output units (Hornik, 1991).

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a family of neural networks for processing variable-length sequential data. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations. In a sense, RNNs introduce the concept of “memory” which holds information about what has been calculated so far. This allows it to exhibit temporal dynamic behavior for a time sequence. In theory, RNNs can make use of information in arbitrarily long sequences, but in practice, they are limited to looking back only a few steps because of the so-called vanishing gradient problem (Hochreiter & Urgen Schmidhuber, 1997).

An RNN maintains a recurrent hidden state whose activation at each time is dependent on that of the previous time step. Training an RNN is similar to training a traditional Neural Network. It makes use of the backpropagation algorithm, but because the parameters are shared by all time steps in the network, the gradient at each output depends not only on the calculations of the current time step but also on those of the previous time steps. This is called Backpropagation Through Time (BPTT).

In the last few years, RNNs have been particularly useful to a variety of problems: speech recognition, language modeling, machine translation, grammar learning, speech synthesis, and recognition. Approaches based on RNNs have, for example, set records for the accuracy of phoneme recognition (Graves, Mohamed, & Hinton, n.d.) and speech synthesis (Fan, Qian, Xie, & Soong, 2014). Essential to these successes is the use of a new special kind of recurrent neural network called Long Short-Term Memory networks or LSTMs.

LSTMs solve the so-called vanishing gradient problem of back-propagated error signals that either shrink rapidly or grow out of bounds in traditional RNNs (Hochreiter & Urgen Schmidhuber, 1997). Because of the vanishing gradient problem, RNNs have difficulties to learn long-range dependencies. To overcome this problem, an LSTM is augmented by recurrent gates called "forget" gates that prevent backpropagated errors from vanishing or exploding. There are several architectures of LSTM units. A common architecture is composed of a memory cell, an input gate, an output gate, and a forget gate. In LSTMs, errors can flow backward through unlimited numbers of virtual layers unfolded

in space, allowing the network to learn tasks that require memories of events that happened many time steps earlier.

A slightly newer variation on the LSTMs are the Gated Recurrent Unit Networks (GRUs). First proposed in 2014 (Cho et al., n.d.), GRUs are simplified versions of LSTMs that merge the cell and hidden state, and combine the forget and input gates into a single update gate.

Convolutional Networks

Convolutional neural networks (CNNs) are a kind of neural network specialized for processing data that has a known grid-like topology (Goodfellow et al., n.d.). Their design is inspired by the brain with the connectivity pattern between neurons resembling that of the mammal's visual cortex. CNNs were originally developed and are still most commonly applied for the analysis of visual imagery. CNNs use a sequence of 3 basic types of layers: convolution, pooling, and activation.

Convolution layers are used to extract spatial features. In a convolutional layer, neurons receive input from only a restricted subarea of the previous layer. This restricted input area of a neuron is called its **receptive field**. The layer's parameters consist of a set of learnable filters, which have a small receptive field but extend through the full depth of the input volume. During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input and producing a 2-dimensional activation map of that filter. As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input. **Weight sharing** scheme is used in Convolutional Layers to reduce the number of parameters, based on the logical assumption that, if one feature is useful to compute at some spatial position, then it should also be useful to compute and at a different position.

The **activation layer** controls how the signal flows from one layer to the next, emulating the firing of the neuron in our brain. Output signals which are strongly associated with past references would activate more neurons, enabling signals to be propagated more efficiently. Among the various activation functions that can be used, the one most

frequently used in CNNs is Rectified Linear Unit (ReLU), which is favored for its faster training speed.

Convolutional networks may include local or global **Pooling layers**, which combine the outputs of neuron clusters at one layer into a single neuron in the next layer. Pooling (also called subsampling or downsampling) reduces the dimensionality of each feature map but retains the most important information. For example, **max pooling** uses the maximum value from each of a cluster of neurons at the prior layer. Another example is average pooling, which uses the average value from each of a cluster of neurons at the prior layer. This is done in part to tackle overfitting by providing an abstracted form of the representation, as well as, to reduce the computational cost by reducing the number of parameters to learn.

The output from the convolutional and pooling layers represent high-level features of the input. A **Fully Connected layer** is then used on these features for classifying the input into various classes based on the training dataset. Neurons in a fully connected layer have connections to all activations in the previous layer, as in regular (non-convolutional) artificial neural networks.

CNNs vary in the number of convolutional layers, ranging from **shallow** architectures with just one convolutional layer to very **deep** architectures with more than 1000 layers. CNNs paced the way for major breakthroughs in Image Classification and are the core of most Computer Vision systems today. More recently CNNs have been used for classification problems in Natural Language Processing and EEG decoding.

Chapter 5.

The EEG Pipeline for MI-BCIs

Introduction

From the perspective of system design, a BCI is deployed in four phases:

- **The training phase** consists in 1) Acquiring the training EEG signals (i.e., training examples) and 2) Optimizing an EEG signal processing pipeline by tuning the feature parameters and training a classifier.
- **The evaluation phase**: Where the results of various classification pipelines are evaluated, and their parameters are re-adjusted based on the outcomes.
- **The calibration phase**: Because neural responses are different across subjects to even the same stimulus, almost all EEG-based brain-computer interfaces require some subject-specific data to calibrate a new subject.
- **The use (or production) phase**: This consists in using the best-trained classification model obtained after the test phase and calibrated during the calibration phase, to recognize the mental state of the user from previously unseen EEG signals, or in other words to operate a BCI system.

In this chapter we review the so-called EEG pipeline i.e. the main stages used for the training and evaluation phase, describing in more details the main approaches and considerations for Motor Imagery based BCIs.

In general, the training phase of a BCI consists of the following processing stages: a data recording stage, where neural data is recorded; a signal processing stage, where the recorded data is preprocessed and cleaned; a feature extraction stage, where meaningful information is extracted from the neural data, and a classification stage, where a decision is interpreted from the data. Following the training phase, is the evaluation phase where the trained classification models are evaluated and their parameters are tuned until the results are satisfactory.

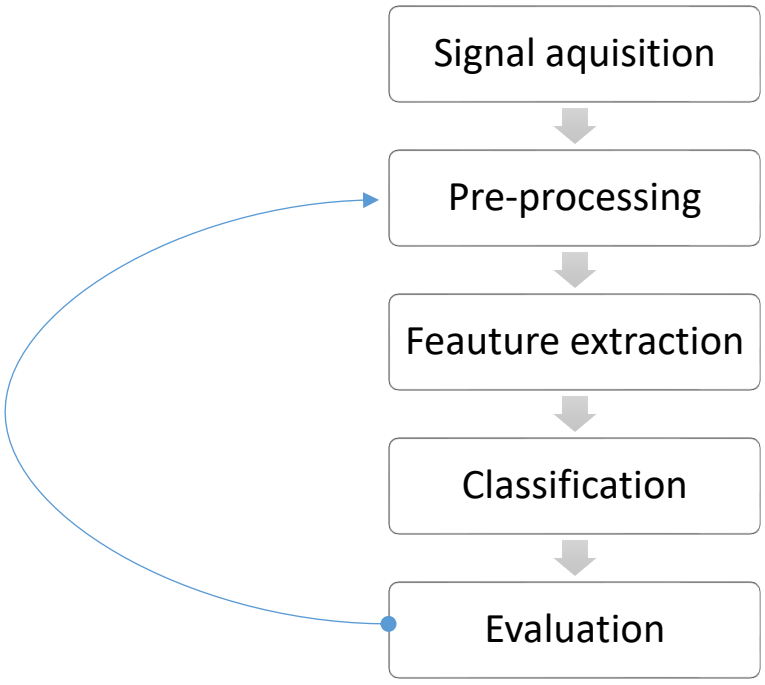


Figure 10: The outline of a BCI training and evaluation pipeline

Data Recording

The design of any BCI systems entails the recording of training data. For MI-based BCIs, a cue (usually visual) is presented to the subject while the EEG recording device is capturing the brain signal. The signal is then synchronized with event markers that encode the class of the stimulus presented to the subject and is sent to a storage device for offline analysis. The accuracy of synchronization of the recorded data with the event markers is essential for the subsequent analysis and specialized protocols are often used to lessen the transmission lags.

Signal digitization and amplification

Low noise and high-resolution processing are critical factors for accurate and robust EEG measurements. The amplitude of an EEG signal typically ranges from 1 to 100 μV in a normal adult. Since the electrical signals are very small, they must be amplified before conversion to digital. After amplification, an A/D converter is interfaced to the recording system so that each sample can be saved in its digital representation. A/D conversion transforms the analog signal to a series of discrete, discontinuous data points separated by equal intervals of time. Important characteristics are the resolution and the sampling rate. The **resolution** of the converter is determined by the smallest amplitude that can be sampled. A/D converters for EEG usually use minimally 12 bits (discriminating 4,096 value levels). The **sampling rate** or sampling frequency is measured in Hertz (Hz) and defines the rate, per second, at which the original analog signal is sampled. Therefore, a sampling rate of 512 means that the original signal was sampled 512 times per second. The higher the sampling rate, the higher the temporal precision of the resampled signal's representation.

While having high temporal precision is generally desirable, it has the drawback of yielding large data files, which in turn are much slower to process. Therefore, it is often necessary to reduce the sampling frequency. However, it is not possible to choose the new sampling frequency in an arbitrary manner; there is a rule, called the **Nyquist rule**, which determines the extent to which we can reduce the sampling frequency. According to the Nyquist rule, the sampling frequency must be at least twice the highest frequency that we

wish to analyze. In most MI-BCIs, we are usually interested in activity below 35Hz, which implies that a sampling frequency of 512 is not necessary. Generally, for MI analysis a sampling frequency of 128Hz or 256Hz is sufficient (CREx, 2018).

Environment & Subject Preparation

There are specific protocols that define the procedures and the criteria for the recording of brain signals. For data acquisition of suitable quality, the choice of the right representative group of volunteers should be considered. The position of subjects during the EEG measurements should be comfortable enough to avoid unwanted activities. The keeping of the same conditions and instructions across subjects and recording periods for the whole length of the experiment period is desired.

Recording data for BCI training can often be a monotonous and lengthy task, and it is easy for the subject to get tired and lose focus, deteriorating the quality of the recording. To encourage the subjects and keep them focused, a form of feedback is provided after each trial, so the subject knows how he or she is doing. The operation of a BCI is not intuitive and users need to learn how to voluntarily control their neural activities. Especially in case of motor imagery based BCIs, a rather long training period is required until the users gain skill in the imagery task and achieve optimal performance (Alimardani, Nishio, & Ishiguro, 2016).

Special electrically shielded rooms minimize the impact of external electric background, in particular, 50/60 Hz alternating current line noise. For usual purposes, a shielded room is not necessary but for advanced research purposes when the maximal amount of information is desired, the shielded room must be used. In these cases, amplifiers run on batteries and an optical cable leads to the recording device or PC which is standing outside from the shielded space.

Data Recording Structure

A typical EEG experiment consists of multiple session recordings per subject. Different sessions for a particular subject can be recorded on different days, but a session must always be completed within the same day. Sessions contain multiple runs each

containing multiple trials. A run is a series of experimental trials, where a trial is the repetition of the same procedure. For MI-BCI recording a trial is the recording of MI tasks, whenever a cue or stimulus is presented to the user. The interstimulus interval (often abbreviated as ISI) is the temporal interval between the offset of one stimulus to the onset of another. The inter-trial interval (ITI) is the time from the end of one trial to the beginning of the next.

Neuroimaging experiments result in complicated data that can be arranged in many ways. So far there is no consensus on how to organize and share data obtained in neuroimaging experiments and there are various formats such as HD5, EGI, and EDF. The European Data Format (EDF) is a simple and flexible format for exchange and storage of multichannel biological and physical signals. It was developed by European 'medical' engineers who met at the 1987 international Sleep Congress in Copenhagen. After its official introduction in 1992, the European Data Format (EDF) became the standard for EEG and PSG (Sleep) recordings. An extension of EDF, named EDF+, was developed in 2002. EDF+ is largely compatible with EDF and supports interrupted recordings, annotations, stimuli, and events.

Data Preprocessing

Data preprocessing is an important step in any data mining or machine learning project. Preprocessing refers to any transformation between data collection and data analysis (Cohen, 2014). In general, we perform data preprocessing because the original data may be incomplete (e.g. lacking attribute values, lacking certain attributes of interest), noisy (e.g. containing errors or outliers) or inconsistent (e.g. containing discrepancies in codes or names). In other cases, preprocessing steps may merely reorganize the data to facilitate analysis.

In EEG signal analysis, preprocessing steps may include more specific tasks such as extracting epochs from continues data, removing whole channels, or rejecting epochs with unwanted artifacts and reduce the noise to signal ratio.

Epoching

Epoching refers to the segmentation of continues EEG data based on the experiment's events. Epoching increases the dimension of the EEG data from two-dimensional (channels x time) to three-dimensional (trials x channels x time). Important decisions that need to be addressed when epoching EEG data is the selection of the event to use for time locking (the time 0) as well as the duration of each epoch. Both decisions depend on the specificities of the experiment. For the time locking event, we usually select the earliest event of each trial because it allows us to shift the data to a later time, if we choose to do so, during the analyses phase. Similarly, for the selection of the duration of each epoch, the maximum time, that is the duration of a trial, is the safest option. On the other hand, if we perform a specific analysis that for example computes only ERP components then we can reduce the duration of an epoch to that of the specific components plus a baseline period (e.g. -200ms to 500ms for a P300 component analysis).

Reduce class imbalance

A dataset is imbalanced if the classification categories are not approximately equally represented. Most machine learning algorithms work best when the number of examples for each class are roughly equal. When the number of instances of one class far

exceeds the other, problems can arise. Especially for EEG datasets, analyses based on phase are more sensitive to class imbalance than analyses based on power or the ERP.

The reduction of class imbalance usually involves identifying the class with the minimum appearances and then selecting trials from the other classes so that we end up with an equal number of trials for each class (**under-sampling**) or duplicating the minority classes till we have an equal number of examples for each class (**over-sampling**). By under-sampling, we risk removing some of the majority class instances which is more representative, thus discarding useful information, while, by over-sampling, we risk to overfit the classifier to a few examples.

To overcome these problems, a sampling-based algorithm called SMOTE (Synthetic Minority Over-Sampling Technique) was introduced (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). SMOTE is a combination of over-sampling and under-sampling, but the oversampling approach is not based on replicating minority class but instead on constructing new minority class data instances via a KNN algorithm. SMOTE has been reported (Fergus, Hignett, Hussain, Al-Jumeily, & Abdel-Aziz, 2015) to further improved sensitivity, specificity, and AUC results when used to increase the number of seizure and non-seizure records on a seizure detection EEG-based analysis.

Frequency Filtering

Frequency filtering (in contrast to spatial and other types of filtering) refers to the attenuation of signal components of a particular frequency (band). The common rationale behind filtering is to attenuate noise in the recordings while preserving the signal (of interest). In electrophysiology, neither noise nor signal is clearly defined. Typically, there is an overlap of signal components and noise components in the same frequency band. (Widmann, Schröger, & Maess, 2015). The low signal-to-noise ratio of EEG recordings makes filtering a useful tool for the analysis of EEG data. However, filtering can also result in various unintended adverse filter effects (distortions such as smoothing) and filter artifacts, biasing or even invalidating the results. Usually, we prefer to run these frequency filters before any other type of correction, on the continuous EEG signal.

There are 4 basic types of frequency filters:

- **Low-pass filter**: all frequencies below a defined frequency are passed and all frequencies above this limit are rejected.
- **High-pass filter**: the inverse of the low-pass filter in which all frequencies above a defined frequency limit are passed and all below are rejected.
- **Band-pass filter**: all frequencies between defined lower and upper-frequency limits are passed.
- **Band-stop filter**: also referred to as a « notch filter » is the inverse of the band-pass filter; all frequencies between a defined lower and upper-frequency limit are rejected. Notch filters are commonly applied to suppress electrical noise from mains interference (50 or 60 Hz).

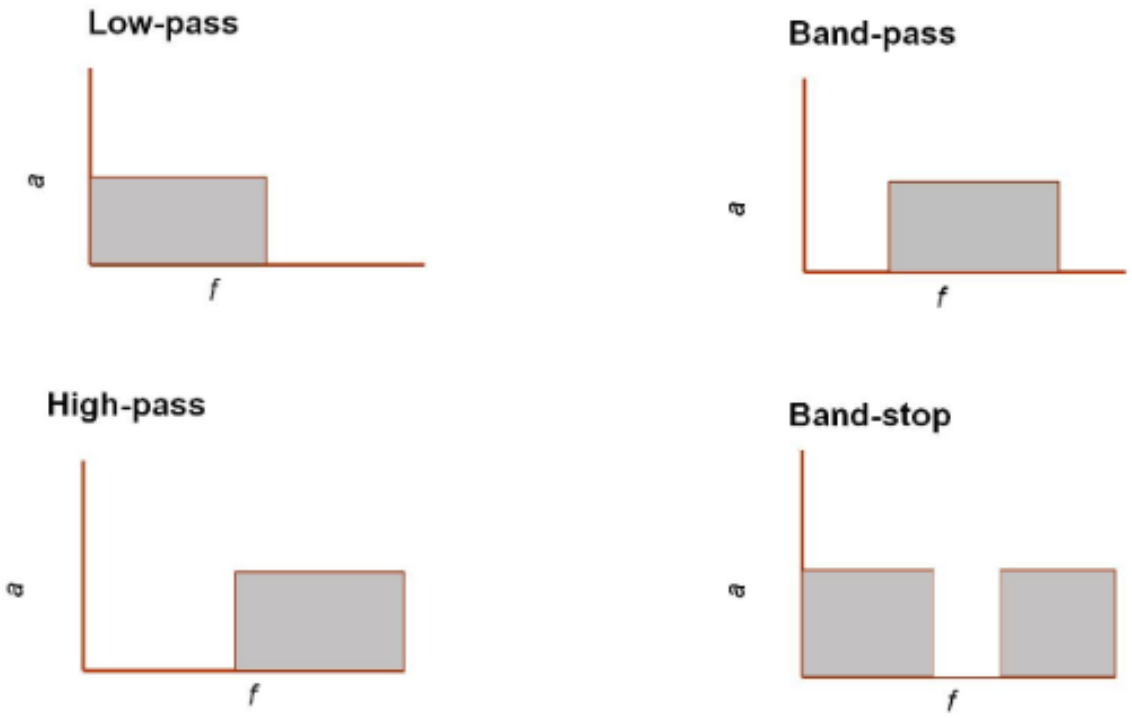


Figure 11: Four basic filter types

Figure presents the frequency responses of each filter type (f = frequency, a = amplitude) .(CREx, n.d.)

Rejecting of bad epochs/trials

In some cases, extracted epochs or entire trials are strongly affected by muscular noise or external artifacts. In cases that the noise or artifacts cannot be reduced using frequency filtering, those segments can be marked as bad and excluded from further analysis. Bad trials/epochs detection is usually based on a combination of different criteria, such as extreme amplitudes (deviation criterion), lack of correlation with any other trial (correlation criterion) and unusual high-frequency noise (noisiness criterion).

Feature Extraction

Usually, not all the data in the feature space are important for classification. Especially for EEG signals, which are captured as high-dimensional multivariate time series, we usually try to isolate a few relevant values called “features” which capture the information embedded in EEG signals that is relevant to describe the classes we want to identify. To reduce the volume of data, we usually opt for:

1. **Feature selection**: where only some of the original features are kept either based on information gain using entropy, or on searching for the best (near optimal) features, using some heuristic searching algorithm such as genetic algorithms. For EEG, selecting features can also mean to select channels and only use data extracted from the selected channels.
2. **Feature extraction**: where the original feature space is transformed into one with lower dimension, using methods that include but are not limited to: general purpose algorithms, such as, principal component analysis (PCA), Independent component analysis, linear discriminate analysis, Fourier transforms, wavelet transform, discrete cosine transform , as well as , other more specific to the analysis of EEG signals, such as common spatial patterns (CSP) and Riemannian transformations. All features extracted are usually arranged into a vector, known as a **feature vector**.

EEG Dimensionality reduction

Raw EEG signals are high-dimensional and thus are not suitable as a direct input for most classifiers. EEG data require dimensionality reduction before they could be classified mainly because of i) the low EEG signal-to-noise ratio, and ii) the redundancy from the strong statistical correlation between signals recorded from close positions in the scalp. Another reason for not using raw EEG data as features vector is due to the so-called “curse of dimensionality” which states that *as the number of features or dimensions grows, the amount of data we need to generalize accurately grows exponentially*. There is no one-size-fits-all ratio between the number of features and the number of available examples, but a general rule of thumb defines this ratio between 1/5 and 1/10. In the case of a typical

EEG two seconds trial with 32 electrodes and a sampling rate of 256Hz, this means that we would need at least 81920 (5x2x32x256) trials per class, to be able to train a classifier. Considering the high cost (both in money and time) to acquire EEG training data, as well as, the cost of analysis, the direct use of EEG signal as a features vector is usually avoided. Instead, we extract specific features from the EEG signal. These features can be related to three main sources of information: 1) **Spatial information** that focus on the signal coming from specific brain areas, 2) **Frequency information** that describe the variation of power in specific relevant bands and 3) **Temporal information** that describe the variation of the EEG signal in specific time windows. In many cases, a combination of these sources is used. One of the most used approaches for MI-BCIs is using spatial-frequency features extracted by the filter bank common spatial pattern algorithm (FB-CSP).

Spatial Filters

Spatial filtering is useful not only because it reduces the dimension from many EEG channels to a few spatially filtered signals, but also because it can help to minimize volume conduction effects. As already mentioned, the EEG signals measured on the surface of the scalp are a blurred image of the signals originating from within the brain, as the underlying brain signal is spread over several EEG channels. Spatial filtering can help to recover this original signal by gathering the relevant information that is spread over different channels.

There are two main categories of spatial filters. **Fixed spatial filters** with their weights fixed in advance, according to predefined neurophysiological knowledge, or **data-driven filters**, that are, optimized on training data. Among the fixed spatial filters, we can notably mention the bipolar and Laplacian, which are local spatial filters that try to locally reduce the smearing effect and some of the background noise (McFar- land et al, 1997). On the other hand, data-driven spatial filters are optimized for each subject according to training data. The weights of data-driven spatial filter can be estimated either in an unsupervised way, without the knowledge of the actual class the trial data belongs to, or in a supervised way, with each training example being labeled with its class.

Among the unsupervised spatial filters, generic algorithms such as Principal Component Analysis (PCA) and Independent Component Analysis (ICA) have been reported to offer rather good results. xDAWN (Souloumiac, Souloumiac, Attina, & Gibert,

2009) is another dimensionality reduction algorithm specifically designed to separate brain waves for ERP classification by enhancing the discrimination between signal and noise and reducing the dimension of the EEG signals. xDAWN is commonly used on ERP experiments to calculate the principal components of the average evoked responses.

Common Spatial Patterns (CSP) and Filters based on Riemannian Topologies are commonly used as supervised data-driven filters in EEG Classification problems. CSP separates a multivariate signal into additive subcomponents. The general idea behind CSP is for each subcomponent to maximize its variance for one class while minimizing it for the other (Hülsmann, J., Jirku, M., Dyck, 2017). The CSP algorithm is highly successful in calculating spatial filters for detecting Event-Related Desynchronization/Synchronization. The core idea behind algorithms using Riemannian geometry is to manipulate covariance matrices in the manifold of symmetric positive-definite (SPD) matrices and use them directly as features in a classifier that respects their intrinsic geometry (Rodrigues et al., 2017). Tangent space projection is used to convert covariance matrices in Euclidean vectors while conserving the inner structure of the manifold. The tangent space projection is an operation that project matrices from the manifold in a vector space named Tangent space. This tangent space is Euclidean and locally homeomorphic to the manifold and Riemannian distance computations in the manifold can be well approximated by Euclidean distance computations in the tangent space. (Alexandre Barachant, 2014).

The Filter Bank Common Spatial Patterns

The most widely used spatial-frequency features extraction algorithm for classification of Motor Imagery EEG is Filter Bank Common Spatial Patterns (FB-CSP) algorithm. There are three steps in the FB-CSP method: 1) a group of band-pass filters is applied to the raw EEG data to obtain specific frequency bands, 2) the CSP algorithm is applied to every filter result to extract the optimal spatial features and 3) the best-resulting features among the multiple spatial filters obtained are then selected using feature selection algorithms. Finally, a classifier is used on the extracted features.

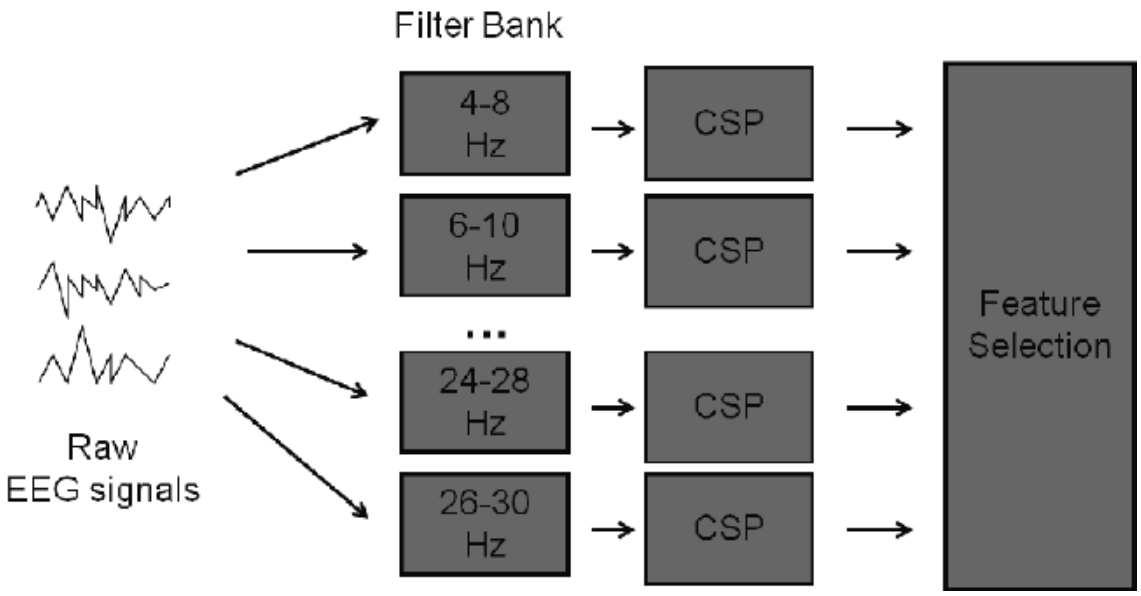


Figure 12: Filter Bank Common Spatial Patterns

The principle of Filter Bank Common Spatial Patterns (FBCSP) (Lotte, 2014)

Classification

For BCI, the most used classifiers so far are discriminant classifiers and notably Linear Discriminant Analysis classifier. LDA is preferred for BCI mainly because of its low computational requirements and its simplicity which makes it good at generalizing to unseen data. Another very popular classifier for BCI is the Support Vector Machine (SVM). SVM is known to have good generalization properties, to be insensitive to overtraining and to the curse-of-dimensionality. The kernel generally used in BCI research is the Gaussian or Radial Basis Function (RBF) kernel.

Among the Non-linear classifiers used in BCI systems, Bayes quadratic and Hidden Markov Model are also known for their performance. Both classifiers produce non-linear decision boundaries. The advantages of these classifiers are that they are generative and reject uncertain samples more efficiently than discriminative classifiers (Lotte et al., 2007).

KNN algorithms are not very popular in the BCI community, probably because they are known to be very sensitive to the curse-of-dimensionality, which made them fail in several BCI experiments (Lotte et al., 2007). A more promising distance-based classification based on distance computation makes use of the Riemannian geometry, to calculate the (Riemannian) distance to mean. Since the covariance matrices are symmetric positive definite (SPD) and lie on a Riemannian manifold, a popular approach is to view each covariance matrix as a point in the Riemannian space and use its geodesic to the Riemannian mean as a feature in classification. This approach is called the **Minimum Distance to Riemannian Mean** (He et al., 2018). The computational complexity of algorithms based on this ground is of concern for high-density EEG data. This happens because Riemannian algorithms rely on eigendecompositions, whose number of operations is on the order of n^3, where n is the number of electrodes.

Classifiers can also be combined to reduce the variance and thus increase the classification accuracy. The combination of various classifiers is termed **Ensemble learning**. Ensemble learning approaches allow the production of better predictive performance compared to a single model. That is why ensemble methods placed first in many prestigious machine learning competitions.

Deep learning classification methods can also be used for BCIs, although, deep Neural Network architectures are not hardware friendly and it takes a lot of resources to

train deep networks fast. Convolutional Neural Networks (CNNs), which have been used in computer vision and speech recognition to perform automatic feature extraction and classification, have successfully been applied to EEG-based BCIs. An attractive property of CNNs that was leveraged in many previous applications is that they are well suited for end-to-end learning, i.e., learning from the raw data without any a priori feature selection. End-to-end learning might be especially attractive in brain-signal decoding, as not all relevant features can be assumed to be known a priori. (Schirrmeister et al., n.d.). Deep RNN architectures have also be used to extract the sequential relationships from EEG signals. Albeit its suitability for processing time series data, RNNs models are not performing as well as expected in classifying motor imagery EEG data (Fedjaev, 2017). A mixed approach (Bashivan, Rish, Yeasin, & Codella, 2015) combines RNNs with CNNs. In this approach, the EEG time series were transformed into spectral images before being used in the deep recurrent-convolutional network. The authors suggest that such representation of data preserves temporal, spectral and spatial information.

Model Evaluation

An evaluation defines how we go from trials per subject and session to a generalization statistic score (AUC score, f-score, accuracy, etc.). For evaluation of EEG pipelines, there can be three different evaluation approaches:

- within-recording-session evaluation
- across-session within-subject evaluation
- across-subject evaluation,

Evaluations within the same session or within the sessions of the same subject yield the best results but are less useful for generic-purpose BCIs. In contrast, the inter-subject variability of EEG recordings poses a challenge in across-subject BCI classification. As a matter of fact, in cross-subject learning, not all available subjects may improve the performance on a test subject (Nasiri Ghosheh Bolagh, Bagher Shamsollahi, Jutten, & Congedo, 2016).

There are many methods to evaluate a classifier, and there are also many different measures to compare the performance of a set of classifiers. The best approach for cross-subject evaluation is to use the Leave One Subject Out Cross-validation (LOSO-CV) strategy, where in each fold, the training set of examples used for learning consists of all the subjects but one. The one excluded from training is then used only to assess the performance of the classifier on completely unseen data.

The measures of accuracy and confusion matrix are very prevalent to evaluate the performance of a classification system, but accuracy may not be a useful measure in cases where there is a large class skew, or there are differential misclassification costs between the classes. More recently, receiver operating characteristic (ROC) curves have been used to evaluate the trade-off between true and false-positive rates of classification algorithms. A ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots True Positive Rate (or recall) vs False Positive Rate. A ROC Curve compares the classifiers' performance across the entire range of class distributions and error costs. However, often there is no clear dominating relation between two ROC curves in the entire range; in those situations, the area under the ROC curve, or simply AUC, provides a single-number "summary" for the performance of a learning algorithm. AUC (stands for "Area under the ROC Curve")

measures the two-dimensional area underneath the entire ROC curve. This is equal to the probability that a classifier will rank a randomly chosen positive example higher than a randomly chosen negative example. AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0; one whose predictions are 100% correct has an AUC of 1. From the literature, it has been shown that AUC is a more statistically consistent and more discriminating, measure than accuracy and should be preferred over accuracy (Ling & Zhang, 2003). This thesis mainly uses ROC AUC to assess the performance of the classification models. Accuracy is reported only when necessary to compare performance with other published results.

Chapter 6.

Evaluation of Public Datasets

Introduction

In this chapter, we experiment with various motor imagery pipelines by applying them on two public datasets related to left/right hand motor imagery. The datasets are BCI Competition IV Dataset 2b (Leeb, Brunner, Müller-Putz, Schlögl, & Pfurtscheller, 2008) and Physionet Motor Imagery (Schalk, McFarland, Hinterberger, Birbaumer, & Wolpaw, 2004). The objective of the experiments is to evaluate different decoding pipelines for motor imagery classification, in order to decide on an optimal configuration which can possibly be used later in the development of our BCI system.

To tune the performance of the classification, we developed a pre-processing analysis script in Python that searches in a space of possible pre-processing configurations for the optimal parameters. Using the best possible pre-processing configuration, we then fine-tuned the hyper-parameters of the classification algorithm, achieving a classification score (AUROC) of 0.855 for the first dataset and a score of 0.963 for the second dataset.

Datasets

For the evaluation of the different pipelines and configurations we selected two publicly available datasets of EEG Motor Imagery, namely:

- BCI Competition IV (2008) - Graz dataset 2b, 9 subjects using 3 EEG channels (BCI42B)
- PHYSIONET MI dataset, 109 subjects using 64- EEG channels (PhysionetMI)

Table 1: Evaluation datasets

Dataset	***Type***	***Channels***	***Trials***	***Sessions***	***Subjects***	***Epoch***
BCI42B	MI	3	120	5	9	3 - 7.5
PhysionetMI	MI	64	40-60	1	109	1 - 3

Dataset B from BCI Competition 2008 (BCI42B)

The dataset was originally released as dataset 2b of the BCI Competition IV and contains 2-class MI recordings from 9 different subjects. For each subject, 5 sessions were recorded. Each session consisted of six runs with ten trials each and two classes of imagery (left and right hand). This resulted in 20 trials per run and 120 trials per session. Three bipolar recordings (C3, Cz, and C4) were recorded with a sampling frequency of 250 Hz. They were bandpass- filtered between 0.5 Hz and 100 Hz, and a notch filter at 50 Hz was enabled. The electrode position Fz served as EEG ground.

It should be noted that prior to the first motor imagery training, the subject executed and imagined different movements for each body part and selected the one, which they could imagine best (e. g., squeezing a ball or pulling a brake). Furthermore, the placement of the three bipolar electrodes was slightly different for each subject.

Each trial started with a fixation cross and an acoustic warning tone. Some seconds later a visual was presented for 1.25 seconds. Afterward, the subjects had to imagine the corresponding hand movement over a period of 4 seconds. Each trial was followed by a

short break of at least 1.5 seconds. A randomized time of up to 1 second was added to the break to avoid adaptation

Physionet Motor Imagery dataset.

The Physionet Motor Imagery dataset consists of over 1500 one- and two-minute EEG recordings, obtained from 109 subjects. Subjects performed different motor/imagery tasks while 64-channel EEG was recorded using the BCI2000 system ("BCI 2000," n.d.). From the different motor/imagery tasks, we used only the corresponding for imagery and executed left or right fist movement. We used only the first 9 subjects for our analysis, both for reasons of consistency and comparison with the first dataset, but mainly due to limited resources for the training (hardware & time).

Past studies on the same datasets

While the two datasets we used for our evaluation are freely available, we didn't manage to find similar studies with reproducible results to benchmark our methodology. There are some studies on the same datasets but because of different setups, subsets of data, and performance metrics are not directly comparable with our evaluation results. Nonetheless, we are presenting here some relevant past studies and their reported results.

Kai Keng Ang et al applied the Filter Bank Common Spatial Pattern algorithm on 2 datasets of the BCI competition 2008 (Ang, Chin, Wang, Guan, & Zhang, 2012). In the case of dataset 2b, FBCSP using the Mutual Information-based Rough Set Reduction (MIRSR) algorithm for feature selection and Naïve Bayesian Parzen Window (NBPW) classifier, reached a kappa value (an indicator that measures the agreement between two raters classifying a set of items into mutually exclusive categories) of 0.599.

J.Sleight et al, classified trials of imagined and executed movement of the PhysionetMI dataset (Sleight, Pillai, & Mohan, 2009). Their study was aiming to distinguish between trials of imagined and executed movements and achieved a 0.566 accuracy for cross-subject evaluation.

Hülsmann, J et al, used the Common Spatial Patterns with LDA to distinguish between imagined hands and foots movements (Hülsmann, J., Jirku, M., Dyck, 2017). The

reported intra-subject classification results, using multiple randomly selected subjects for training and a random set of remaining subjects for testing, was around 60%.

Alomari et al applied various pre-processing steps and used SVM classification on the PhysionetMI dataset (Alomari et al., 2013). In their study they used only trials of executed movement on a subset of 6 subjects, reporting a cross-validation accuracy of 97%.

Evaluation Design

Initial experiments with both datasets, using different classification algorithms, revealed that beyond the algorithms, an important role for the classification performance had the pre-processing configuration, namely: the segmentation of the trials (epoching), the spectral filters, as well as, the selection of feature extraction algorithms.

As a result, we developed a Python script to automatically apply different combinations of time-windows, spectral filters, feature extraction methods and classification algorithms to each dataset. The script allows for different configuration of pre-processing hyper-parameters to be applied by setting flags and passing command line arguments.

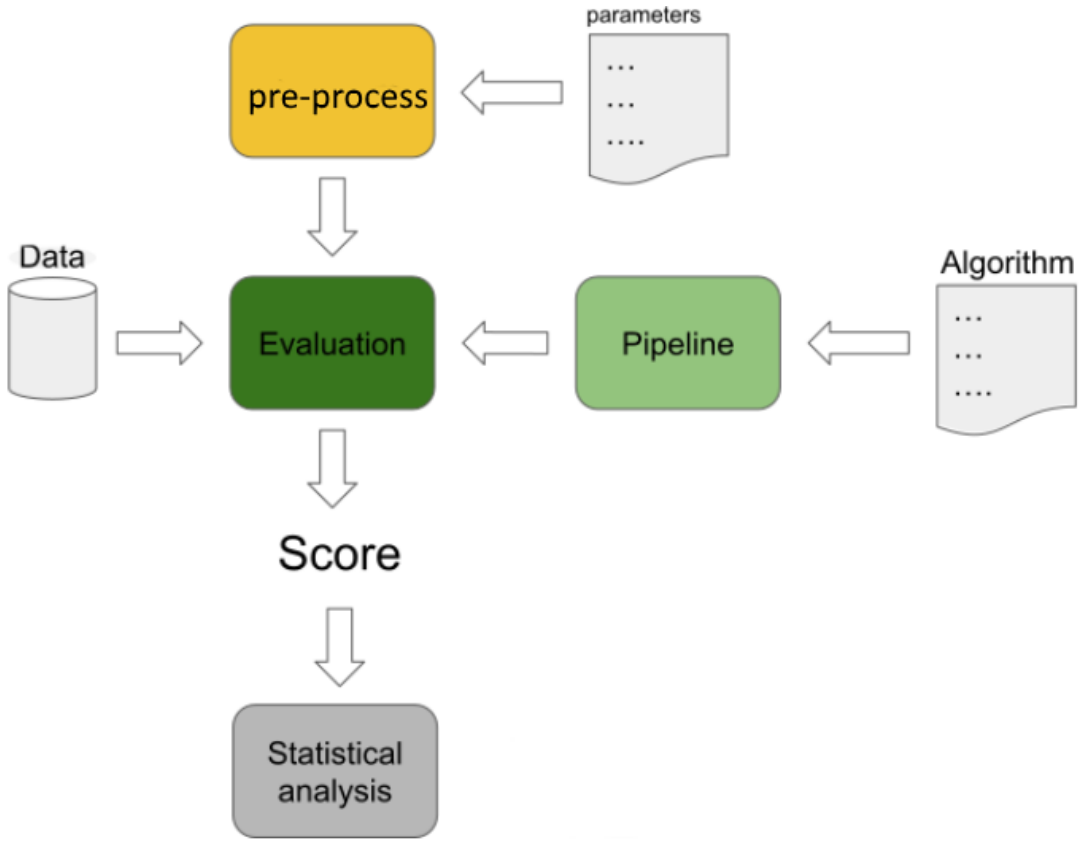


Figure 13: Evaluation design for the Analysis of Public datasets

The architecture of the analysis script used in the evaluation of different pre-processing configurations and classification algorithms

The varying parameters we used for the pre-processing are:

Spectral filters:

Instead of directly using a wide filter band, we applied various bandpass filters on raw EEG by using a set of overlapping subbands. These subbands are constructed using a high-pass frequency in [1,3,7] Hz and a low-pass frequency ranging between 30–60 Hz with an overlapping rate of 5 Hz. This way we constructed 3 different groups of band-pass filters:

- **Fb1k** = 1–k Hz, k in [30,35,40,45,50,55,60] Hz.
- **Fb3k** = 3 -k Hz k in [30,35,40,45,50,55,60] Hz.
- **Fb7k** = 7 – k Hz, k in [30,35,40,45,50,55,60] Hz.

In total 22 different band-pass filters were tested as we included also a pass with no filtering.

Epoching

In order to evaluate the influence of different time segments of each trial on the classification success rate, a sliding time window with a step of 200ms and varying durations applied to the raw data resulting in:

- 40 different time windows for the first dataset (BCI42B) starting between 0 and 1 seconds with steps of 200ms and a duration between 1 and 4.5 seconds with steps of 500ms. Epochs overlapping the maximum trial duration (4.5 seconds) were automatically rejected.
- 22 different time windows for the second dataset (PhysionetMI) starting between 0 and 1 seconds with steps of 200ms and a duration between 1 and 3 seconds with steps of 500ms. Epochs overlapping the maximum trial duration (3 seconds) were automatically rejected.

Features extraction algorithms:

For the features extraction we tested eight different algorithms:

- Covariance estimation with tangent space mapping. This simply performs a covariance matrix estimation for each given trial.

- ERP Covariances with and without projection to Tangent Space. Estimation of special form covariance matrix dedicated to ERP processing.
- XDawn Covariances, with and without projection to Tangent Space. Estimation of special form covariance matrix dedicated to ERP processing combined with xDAWN spatial filtering. This is similar to ERP Covariances, but data are spatially filtered with xDAWN. A complete description of the method is available in (Alexandre Barachant, 2014).
- Common Spatial Patterns (CSP) with 4, 6 ,8 components.

Classifiers

We tested the following classifiers on both datasets:

- Logistic Regression
- LDA
- SVM with linear, poly and Rbf kernels
- Riemannian Minimum Distance to Mean (RMDM), which computes a geometric mean for each class from the training data and then assigns an unlabeled trial to the class corresponding to the closest mean.

We used the sklearn Python library to combine the features extraction and classification algorithms into evaluation pipelines. In total, we tested 25 pipelines on each dataset over 880 (22*40) pre-processing configurations for the first dataset (BCI42B) and 484 (22*22) pre-processing configurations for the second dataset (PhysionetMI).

Table 2: The 25 pipelines that have been used to evaluate the 2 public datasets

Pipeline Name	**Features Extraction**	**Classifier**
RG + LR	Covariances, projection to Tangent Space	Logistic Regression
RG + LDA	Covariances, projection to Tangent Space	LDA
CSP4 + LR	Common Spatial Patterns (4 components)	Logistic Regression
CSP4 + LDA	Common Spatial Patterns (4 components)	LDA
CSP6 + LR	Common Spatial Patterns (6 components)	Logistic Regression

CSP6 + LDA	Common Spatial Patterns (6 components)	LDA
CSP8 + LR	Common Spatial Patterns (8 components)	Logistic Regression
CSP8 + LDA	Common Spatial Patterns (8 components)	LDA
ERPCov + TS + LR	ERP Covariances, projection to Tangent Space	Logistic Regression
ERPCov + TS + LDA	ERP Covariances, projection to Tangent Space	LDA
ERPCov + TS + SVM Rbf	ERP Covariances, projection to Tangent Space	SVM with Rbf kernel
ERPCov + MDM	ERP Covariances	Riemannian Minimum Distance to Mean
XdawnCov+ TS + LR	Xdawn Covariances, projection to Tangent Space	Logistic Regression
XdawnCov+ TS + LDA	Xdawn Covariances, projection to Tangent Space	LDA
XdawnCov+ TS + SVM Rbf	Xdawn Covariances, projection to Tangent Space	SVM with Rbf kernel
XdawnCov+ MDM	Xdawn Covariances	Riemannian Minimum Distance to Mean
RG + SVM Linear	Covariances, projection to Tangent Space	SVM with Linear kernel
RG + SVM Poly	Covariances, projection to Tangent Space	SVM with Poly kernel
RG + SVM Rbf	Covariances, projection to Tangent Space	SVM with Rbf kernel
CSP4 + SVM Linear	Common Spatial Patterns (4 components)	SVM with Linear kernel
CSP4 + SVM Poly	Common Spatial Patterns (4 components)	SVM with Poly kernel
CSP4 + SVM Rbf	Common Spatial Patterns (4 components)	SVM with Rbf kernel
CSP8 + SVM Linear	Common Spatial Patterns (8 components)	SVM with Linear kernel
CSP8 + SVM Poly	Common Spatial Patterns (8 components)	SVM with Poly kernel
CSP8 + SVM Rbf	Common Spatial Patterns (8 components)	SVM with Rbf kernel

Script Implementation

The script for evaluation of different pre-processing configurations was developed in Python and it uses the open sourced libraries NumPy, Pandas, scikit-learn, pyRiemann, and MNE. The latter (Larson et al., 2018) is a community-driven software for processing neural signals including EEG. NumPy and Pandas are general-purpose Python packages, offering libraries and tools for general scientific computation while scikit-learn is a machine learning library for the Python programming language. pyRiemann (A. Barachant, 2015) is a python toolkit for covariance matrices manipulation and classification through Riemannian geometry.

The evaluation script starts by applying (or not) a specific band-pass filter to the row data. It then creates epochs using a specific time-window. The number of features is then reduced using a specific feature selection algorithm and then a classification algorithm is applied. This process is repeated for all the combination of pre-processing configurations and classification algorithms.

At the end of every run, the script exports the current pre-processing configuration and the results for each pipeline to a folder named according to the dataset and the evaluation type (cross-subject or cross-session). In this folder, the results are stored as csv files named according to specific pre-processing parameters (e.g. applied spectral filters). The automatic evaluation script run continually for long periods of time, on two online servers dedicated to this task, evaluating over 2 million combinations.

Cross-Validation methodology

For the evaluation of the pipelines, we employed a leave-one-subject-out cross-validation (LOSO-CV) procedure, i.e. for predicting the labels of a particular subject we only use training data from other subjects. The Area Under the ROC curve (AUROC) is used as the scoring metric of each evaluation.

For the BCI42B dataset, we applied both cross-subject and inter-subject (cross-session) evaluation in order to assess the impact of inter-subject variability of EEG signal in the classification performance. This resulted in 990,000 distinct scores for each type of

evaluation (25 Pipelines * 880 hyper-parameter configurations * 9 subjects * 5 sessions per subject) and a total of **1,980,000** evaluation scores for the BCI42B dataset.

For the PhysionetMI dataset, we applied only cross-subject evaluation, but we run the evaluation script 3 times. One using only trials of imagined movement, one using only trials of executed movement and one using trials of imagined and executed movement. This resulted in 108,900 distinct scores for each category (25 Pipelines * 484 hyper-parameter configurations * 9 subjects) and a total of **326,700** evaluation scores for the PhysionetMI dataset. On this dataset, we also applied deep learning classification using a Shallow Feed Forward Neural Network and tried Automated Machine Learning (ensemble) approaches using TPOT a Python tool that optimizes machine learning pipelines using genetic programming.

Analysis Results

BCI42B Dataset

For the BCI42B dataset, we found that the optimal pre-processing configuration consists of the combination of a [0.2 – 4.2sec] time-window for the epoch with the application of a [7-35Hz] band-pass filter. The best cross-subject classification performance, for the optimal configuration, was **0.8 (AUROC)** using pipelines based on Common Spatial Pattern for features extraction. After excluding 'bad' training subjects (subject selection) we got an even better performance of **0.846** for cross-subject classification. Finally, we fine-tuned the classifier's hyper-parameters using Grid Search, achieving a final performance of **0.855**. For this dataset, we also applied inter-subject classification reaching a performance of **0.813 (AUROC)** without subject selection and **0.864** after applying subject selection.

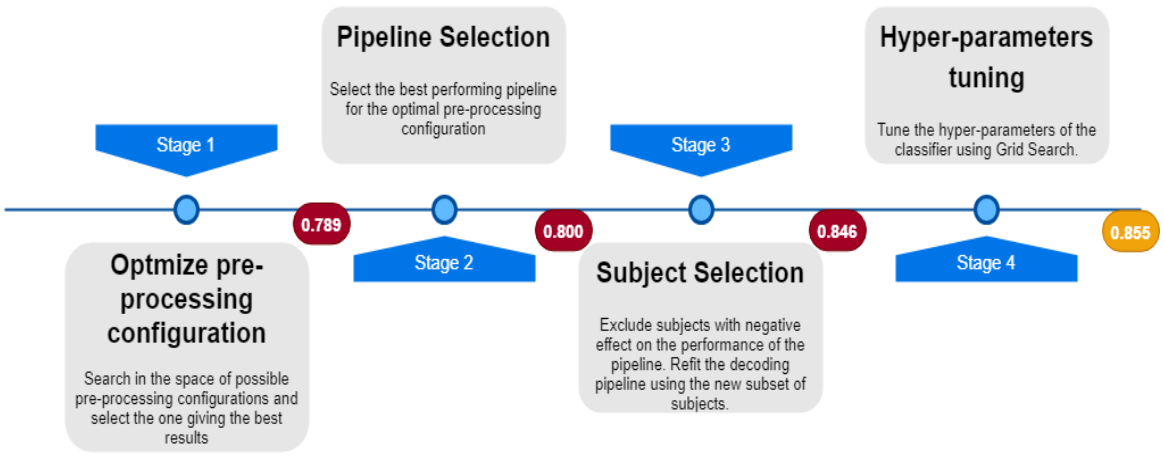


Figure 14: BCI42B - optimization steps and results

Detailed results

From the results of the pre-processing evaluation script, we found that for this dataset the optimal pre-processing configuration was the use of a [0.2 – 4.2sec] time-window for the epoch with the application of a [7-35Hz] band-pass filter. The top five

performing pipelines are using Common Spatial Patterns (CSP) for features extraction. The number of components for CSP (4,6 or 8) or the classification algorithm (Logistic Regression, LDA, SVM) do not appear to have any significant effect on the final classification result.

Table 3: BCI42B – Cross-Subject classification performance for the top 5 pipelines using optimal pre-processing configuration.

pipeline	Mean AUROC
CSP4 + LR	0.800
CSP6 + LR	0.800
CSP8 + LR	0.800
CSP4 + SVM Linear	0.800
CSP8 + SVM Linear	0.800

The following table presents the classification performance (AUROC) for all the pipelines averaged over four different setups for the pre-processing configuration:

- evaluations filtered at [7-35Hz], over all different time-windows (column 2).
- evaluations for the [0.2 – 4.2sec] time-window, over all filter setups (column 3).
- evaluations filtered at [7-35Hz], over the evaluations for the [0.2 - 4.2sec] time-window. The optimal pre-processing setup (column 4).
- all evaluations with all spectral and epochs setups (column 5).

Table 4: BCI42B – Cross-Subject mean classification score for all the pipelines over different pre-processing configuration

Pipeline	Spectral Filter	7-35Hz	All	7-35Hz	All
	epoch	All	0.2 - 4.2	0.2 - 4.2	All
CSP4 + LDA		0.781	0.757	0.799	0.742
CSP4 + LR		0.781	0.757	0.800	0.742
CSP4 + SVM Linear		0.781	0.757	0.800	0.742
CSP4 + SVM Poly		0.764	0.733	0.778	0.725
CSP4 + SVM Rbf		0.781	0.757	0.800	0.742
CSP6 + LDA		0.781	0.757	0.799	0.742

CSP6 + LR	0.781	0.757	0.800	0.742
CSP8 + LDA	0.781	0.757	0.799	0.742
CSP8 + LR	0.781	0.757	0.800	0.742
CSP8 + SVM Linear	0.781	0.757	0.800	0.742
CSP8 + SVM Poly	0.764	0.733	0.778	0.725
CSP8 + SVM Rbf	0.781	0.757	0.800	0.742
ERPCov + MDM	0.768	0.775	0.784	0.755
ERPCov + TS + LDA	0.755	0.770	0.765	0.749
ERPCov + TS + LR	0.767	0.782	0.779	0.760
ERPCov + TS + SVM Rbf	0.780	0.794	0.797	0.770
RG + LDA	0.776	0.752	0.792	0.742
RG + LR	0.777	0.754	0.793	0.743
RG + SVM Linear	0.776	0.753	0.792	0.742
RG + SVM Poly	0.743	0.716	0.756	0.708
RG + SVM Rbf	0.779	0.757	0.796	0.745
XdawnCov + MDM	0.764	0.769	0.779	0.751
XdawnCov + TS + LDA	0.755	0.768	0.762	0.749
XdawnCov + TS + LR	0.768	0.782	0.779	0.761
XdawnCov + TS + SVM Rbf	0.780	0.794	0.797	0.770
mean (all pipelines)	**0.773**	**0.760**	**0.789**	**0.745**

Score per subject

Analysing further the results, we found that subject number 2 (M = 0.641, 95% CI [0.6335, 0.6489]) and 3 (M = 0.603, 95% CI [0.5944, 0.6118]) had significantly lower results for the optimal pre-processing configuration. When we excluded subjects 2 and 3 from the evaluation, we got a higher score of **0.846** for the optimal pre-processing configuration and the best pipeline.

Table 5: BCI42B – Cross-Subject classification performance (AUROC) per subject

	All pre-processing configurations		**Optimal pre-processing configuration**	
subject	**mean score (all pipelines)**	**mean score (best pipeline)**	**mean score (all pipelines)**	**score (best pipeline)**
4	0.940	0.954	0.962	0.973
8	0.818	0.821	0.821	0.832
9	0.797	0.806	0.832	0.842
6	0.750	0.754	0.854	0.878
1	0.738	0.741	0.815	0.826

7	0.729	0.723	0.742	0.751
5	0.721	0.687	0.829	0.834
3	0.607	0.607	0.603	0.608
2	0.602	0.586	0.641	0.653
mean (all subjects)	**0.745**	**0.742**	**0.789**	**0.800**

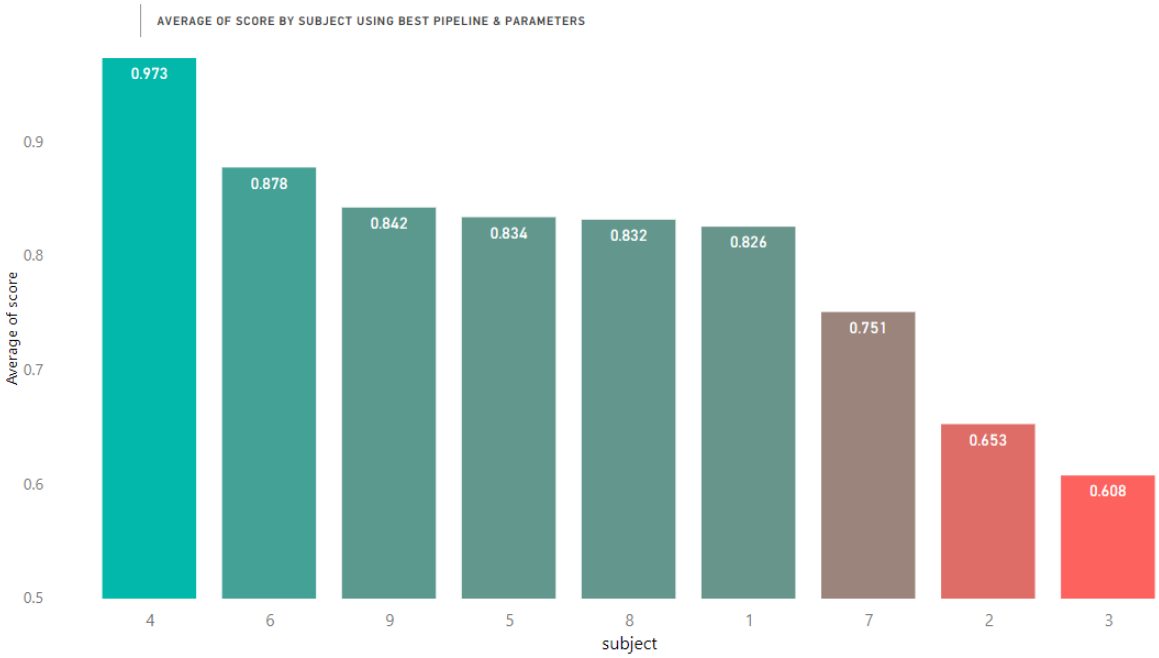


Figure 15: BCI42B dataset – Cross-Subject performance (AUC) per subject using optimal pre-processing configuration and the best pipeline.

Cross-subject vs inter-subject evaluation

Comparing the results between cross-subject and inter-subject evaluations, we found that there is a very small increase in the cross-validation score when we apply inter-subject (cross-session within the same subject) evaluation. The best score for the inter-subject evaluation was 0.813, an increase of only 0.013 against the cross-subject evaluation. We found a similar increase when we excluded the 'bad' subjects (0.846 for cross-subject vs 0.867 for inter-subject evaluation).

Table 6: BNCI - Cross-Subject vs Inter-Subject classification score (AUROC)

	All subjects	**Excluding subjects 2,3**
Cross-Subject	0.800	0.846
Inter-Subject	0.813	0.867

PhysionetMI Dataset

For the PhysionetMI dataset, we found that the optimal pre-processing configuration consists of the combination of a [0 – 3sec] time window for the epoch and the absence of any band-pass filters. The best cross-validation score, for the optimal configuration, was **0.94** (AUROC) using the ERPCov + TS + LR Pipeline. After excluding 'bad' training subjects (subject selection) we got an even better score of **0.958** for cross-subject evaluation. Finally, we fine-tuned the classifier's hyper-parameters using Grid Search, achieving a final score of **0.963**. This score is at the upper end of the scale, especially for Cross-Subject evaluations. To verify the results, we tested the best-optimized pipeline, on new unseen data of the same dataset, achieving an AUROC = **0.743**.

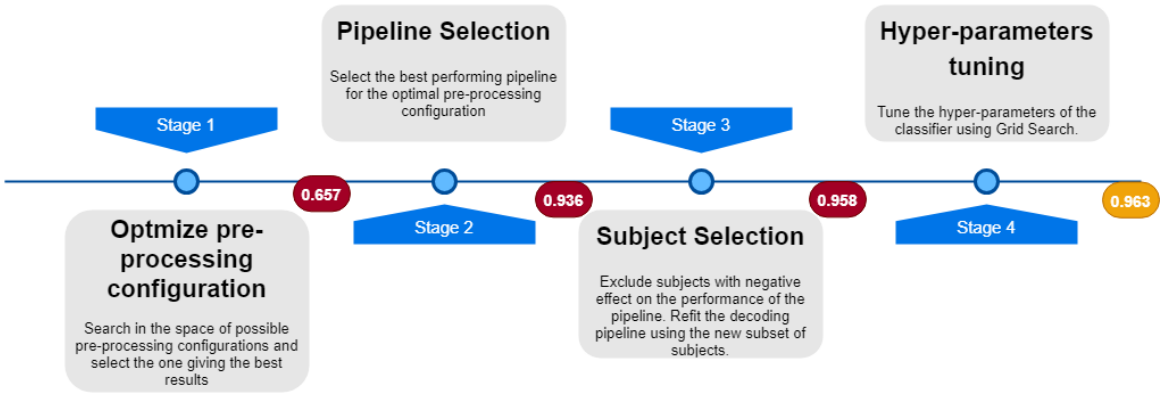


Figure 16: PhysionetMI optimization steps and results

Detailed results

From the results of the pre-processing analysis script, we found that for this dataset, the optimal pre-processing configuration was the use of a [0 – 3sec] time-window for the epoch and the absence of spectral filters. The best-performing pipelines were those that were using ERP or xDAWN covariances for features extraction. This is probably due to visual evoked potential associated with the experimental paradigm.

Table 7: PhysionetMI – Cross-Subject classification performance for the top 5 pipelines using optimal pre-processing configuration

pipeline	Mean AUROC
ERPCov + TS + LR	0.936
ERPCov + TS + SVM Rbf	0.927
XdawnCov + TS + SVM Rbf	0.922
XdawnCov + MDM	0.919
XdawnCov + TS + LR	0.900

The following table presents the score for all the pipelines averaged over four different setups for the pre-processing configuration:

- evaluations without a spectral filter over all different time-windows (column 2).
- evaluations for the [0 – 3sec] time-window, over all filter setups (column 3).
- evaluations without a spectral filter over the [0 -3sec] time-window. The optimal pre-processing setup (column 4).
- all evaluations with all spectral and time-windows (column 5).

Table 8: PhysionetMI - Cross validation score for all the pipelines

Pipeline	Spectral Filter	None	All	None	All
	epoch	All	0 - 3	0 -3	All
ERPCov + TS + LR		0.823	0.769	**0.936**	0.685
ERPCov + TS + SVM Rbf		0.799	0.731	0.927	0.664
XdawnCov + TS + SVM Rbf		0.820	0.667	0.922	0.584
XdawnCov + MDM		0.819	0.666	0.919	0.583
XdawnCov + TS + LR		0.784	0.658	0.900	0.571

ERPCov + TS + LDA	0.774	0.580	0.896	0.551
ERPCov + MDM	0.741	0.687	0.862	0.623
XdawnCov + TS + LDA	0.708	0.610	0.827	0.547
RG + LR	0.615	0.691	0.658	0.635
RG + SVM Rbf	0.621	0.691	0.650	0.651
RG + SVM Linear	0.607	0.674	0.645	0.624
RG + LDA	0.540	0.569	0.586	0.550
RG + SVM Poly	0.553	0.616	0.559	0.582
CSP4 + SVM Rbf	0.518	0.547	0.547	0.539
CSP4 + SVM Linear	0.510	0.569	0.532	0.545
CSP6 + LDA	0.517	0.575	0.530	0.555
CSP4 + LDA	0.509	0.572	0.529	0.546
CSP4 + LR	0.508	0.570	0.529	0.546
CSP6 + LR	0.518	0.575	0.529	0.555
CSP8 + SVM Rbf	0.515	0.565	0.516	0.548
CSP4 + SVM Poly	0.505	0.559	0.496	0.540
CSP8 + SVM Linear	0.506	0.592	0.493	0.558
CSP8 + LR	0.505	0.591	0.486	0.558
CSP8 + SVM Poly	0.495	0.580	0.485	0.545
CSP8 + LDA	0.504	0.591	0.481	0.558
mean (all pipelines)	**0.61256**	**0.6198**	**0.6576**	**0.57772**

Pipelines using xDAWN or ERP Covariances for features extraction performed better on raw unfiltered data. CSP-based pipelines, on the other hand, performed relatively better when applied to band-passed filtered data.

Table 9: PhysionetMI – mean score of features extraction ‘families’

Features Extraction	**filter**	None	All	None	All
	epoch	All	0 - 3	0 - 3	All
xDAWN Covariances		0.783	0.650	0.892	0.571
ERP Covariances		0.784	0.692	0.905	0.631
Covariances (RG)		0.587	0.648	0.620	0.609
CSP		0.509	0.574	0.513	0.549

Figure 17: Performance of pipelines with and without spectral filtering

From the figure, we can observe that CSP-based pipelines performed better on band-passed data, while Riemannian-based pipelines performed better on raw unfiltered data.

Score per subject

Analyzing further the results, we found that subject number 5 had significantly lower results for the optimal pre-processing configuration (M = 0.524, 95% CI [0.464, 0.584]). When we excluded subject 5 from the evaluation, we got a higher cross-validation score of 0.958 for the optimal pre-processing configuration and the best pipeline.

Table 10: PhysionetMI - Cross validation score per subject

	All pre-processing configurations		Optimal pre-processing configuration	
subject	mean score (all pipelines)	mean score (best pipeline)	mean score (all pipelines)	score (best pipeline)
7	0.657	0.827	0.710	0.984
2	0.620	0.769	0.713	0.990
8	0.586	0.618	0.623	0.870
9	0.584	0.685	0.645	0.909
4	0.584	0.733	0.671	0.915
1	0.567	0.677	0.700	0.970
3	0.534	0.611	0.678	0.988
6	0.514	0.596	0.656	1
5	0.554	0.649	0.524	0.796
mean (all subjects)	**0.578**	**0.685**	**0.658**	**0.936**

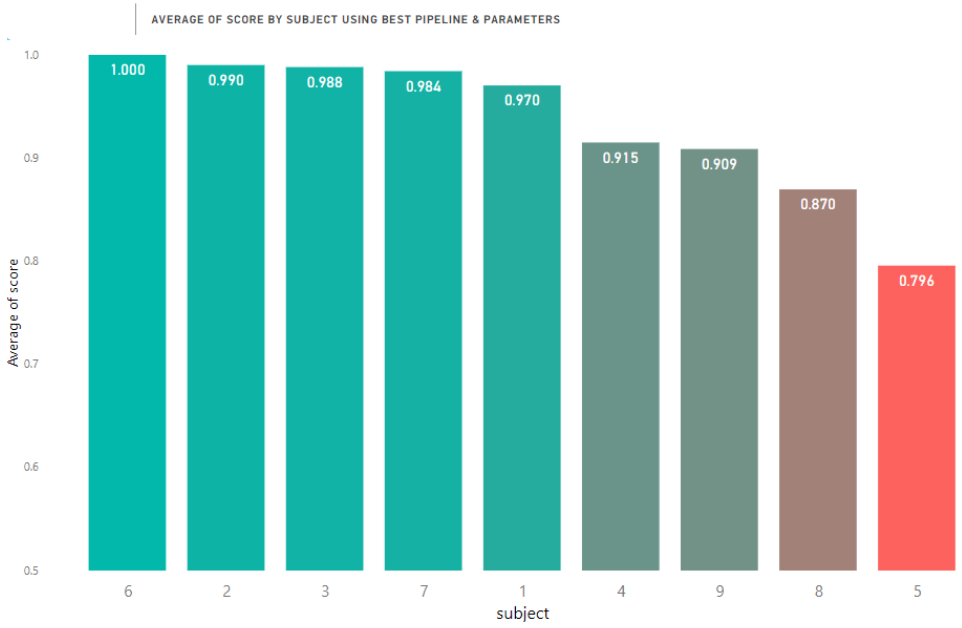


Figure 18: PhysionetMI dataset – Cross validation score per subject using optimal pre-processing configuration and the best pipeline.

Hyper-parameters tuning

As a last optimization step, we performed hyper-parameters tuning for the classifier. Our best classification pipeline is using Logistic Regression for the classification step. In order to choose the best hyper-parameters for the LR learner, we used grid search cross-validation in a finite space of possible values for the best parameters [C, penalty, solver]. The obtained best parameters allowed us to achieve our final best cross-validation score of **0.963** (AUROC).

Table 11: PhysionetMI - Hyper-parameters optimization for Logistic Regression Classifier

Parameter	**Best value**
C value	0.518
penalty	L2
solver	Limited-memory Broyden–Fletcher–Goldfarb–Shanno Algorithm (lbfgs)

Comparison of trial types (imagined vs executed trials)

For this dataset, we have run the pre-processing evaluation script three times. One using only trials of imagined movement, one using only trials of executed movement and one using trials of both imagined and executed movement. The results reveal a very small improvement of 0.016 when using trials of executed instead of imagined movement. There is no further improvement if we combine trials of imagined and executed movement.

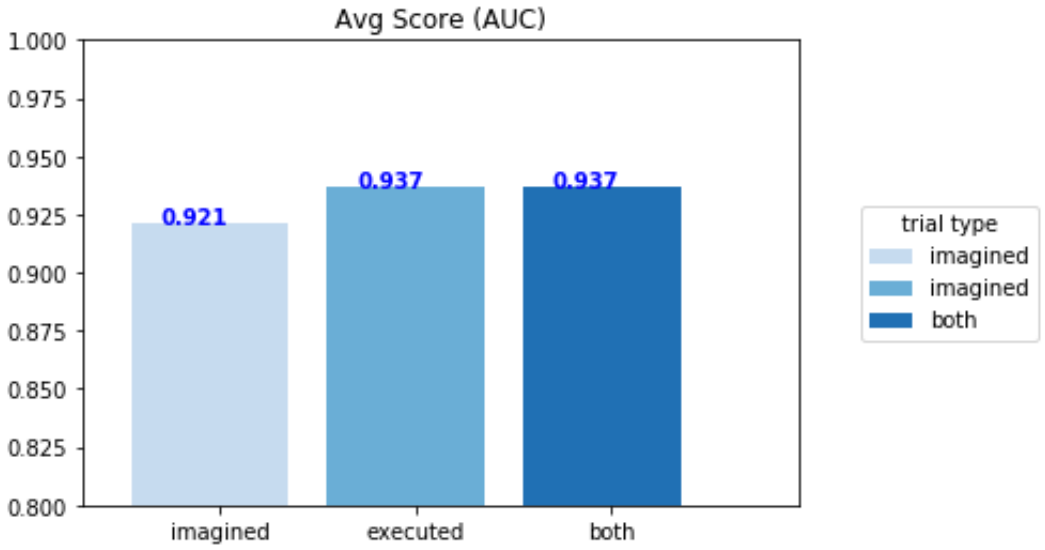


Figure 19: PhysionetMI - Average score per trial type for the best 5 pipelines.

Average cross-validation score per movement type, for the best 5 pipelines using the optimal pre-processing configuration (before subject selection and hyper-parameters tuning).

Test on completely unseen data

To test the performance of the optimal pipeline on completely unseen data, we fit a model using the optimal hyperparameters on the training subjects (subjects [1 – 9]) and used it to predict the probabilities on a new unseen subset of subjects (subjects [10 – 19]) of the same dataset. The mean classification score was 0.743, indicating that our optimal model generalizes well and is able to make 74% correct predictions on completely unseen data.

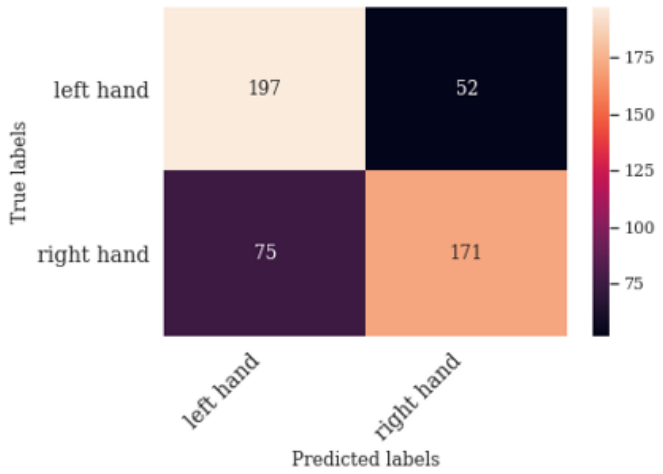


Figure 20: PhysionetMI - Confusion matrix for the optimal model on unseen data

Ensemble learning methods

In order to evaluate if an Auto Machine Learning framework could reach the performance of our approach, we used TPOT (Olson et al., 2016), a Python library that automatically creates and optimizes full machine learning pipelines using genetic programming. Automatic machine learning (AutoML) frameworks reduce the load on data scientists so they can spend less time on feature engineering and hyperparameter tuning. We tested TPOT on the PhysionetMI dataset, using 50 generations and a light configuration (a built-in configuration with only fast models and pre-processors).

The best result we managed to get was **0.763** (AUROC) which is significantly lower than the score we got from our best performing pipeline. The main reason for this lower performance is that TPOT was not able to use EEG-specific features extraction algorithms.

The exported pipeline from TPOT is the following:

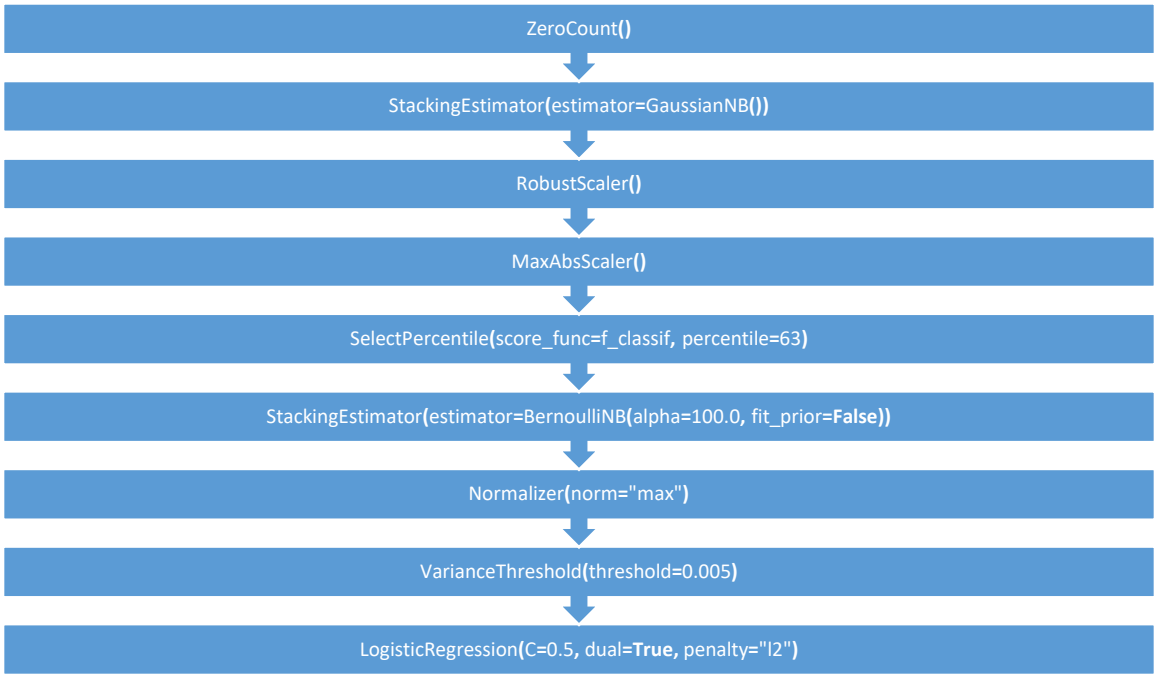


Figure 21: PhysionetMI - The TPOT generated pipeline

Deep learning methods

As a final experiment, we decided to test deep learning methods on the PhysionetMI dataset, in order to evaluate the performance of end-to-end learning, i.e. leaning from raw data. For this, we used a convolutional network with shallow architecture (one temporal convolution, one spatial convolution, squaring and mean pooling, a softmax layer) as proposed by (Schirrmeister et al., 2017). For training, we used the first 8 subjects of the PhysionetMI dataset and for validation the last 2 subjects (subject 9 & 10). We run the neural network on different time-windows and spectral filters and the best cross-validation result we got was 0.656 (accuracy).

Table 12: PhysionetMI - results using a shallow convolutional neural network

filtermin	filtermax	tmin	tmax	accuracy
1	40	0.6	3.6	0.656
1	30	0.6	3.6	0.644
1	35	0.4	3.4	0.644
1	35	0.6	3.6	0.644
1	40	0.2	3.2	0.644
1	40	0	3.5	0.633
1	40	0.4	3.4	0.633
1	45	0.2	3.2	0.633
1	50	0.2	3.2	0.633

Discussion

The results of cross-subject classification for both datasets are in the upper end of the scale and demonstrate that our method for automatic optimization of the pre-processing configuration works well. Although that we trained the pipelines with a relatively small number of subject's data (9 subjects) the classification algorithms generalized well, and we managed to get a 74% prediction score on completely unseen data using unseen subjects of the PhysionetMI dataset.

The primary reason for the lower classification results of BNCI compared to PhysionetMI is the fact that the BNCI EEG recording was conducted using only 3 electrodes while 64 electrodes were used for the PhysionetMI recording. Another possible reason for the lower performance is the high inter-subject and inter-session variability of this dataset. According to the description provided for the BNCI recording procedure, prior to the first motor imagery training, the subject executed and imagined different movements for each body part and selected the one, which they could imagine best (e. g., squeezing a ball or pulling a brake). Furthermore, the placement of the three electrodes was slightly different for each subject. Those design decisions probably made the decoding algorithms less robust to modifications of the EEG source spatial distributions that are typically observed across sessions and across subjects. In addition, the BNCI dataset exhibits greater variability because it contains 5 different sessions per subject, while on the other hand, the PhysionetMI dataset included a single session per subject.

In relation to the decoding pipeline, we demonstrated the importance of the signal pre-processing stage and the selection of the features extraction method for the final classification performance. Generally, regarding classification algorithms, we discovered that very high performances can be obtained using simple linear classifiers such as Logistic Regression or SVM. The most important factor seems to be the design and selection of the features that describe the EEG signal. Riemannian Geometry based pipelines projecting the data points to a tangent space followed by a standard linear classifier performed unexpectedly well in both datasets. CSP methods performed much better when applied to band-passed trials or in datasets having a limited number of channels.

Approaches that could not make use of EEG specific feature extraction methods did not perform well, compared to classical machine learning pipelines that used CSP or

Riemannian Geometry. The shallow Convolution Neural Network approach did not achieve good results compared to the traditional machine learning approaches.

Finally, we should emphasize that the performance we got in our approaches have been evaluated only offline, using data acquired in lab conditions. However, an actual BCI application is primarily online and in an unknown environment. In the next chapter, we will attempt to build an online BCI system using MUSE, in order to study and validate these classification methods online as well.

Chapter 7.

Experiments with MUSE headband

Introduction

In this chapter, we describe our approach to build a prototype BCI system using the MUSE headband. The requirements we set for our BCI system were as follows:

- **Complete**: In the sense that it shall encompass all three parts of a BCI system: the signal acquisition from the EEG hardware, the signal processing, and classification part, and finally the control – feedback mechanism.
- **Active**: Driven from brain activity which is directly consciously controlled by the user, namely based on the Motor Imagery paradigm.
- **Online**: Able to analyze signals coming in real-time from the EEG hardware.
- **Asynchronous**: able to continuously analyze the signals for command patterns and able to identify (and ignore) rest states.

We started with the development of the necessary software stack that allowed the acquisition of EEG signals from the MUSE device in real time. Once we managed to connect and get the EEG signal, we performed various offline experiments with data recorded from the MUSE device, in order to evaluate the performance of classification using EEG signal from the MUSE deice. The analysis revealed that while MUSE was able to detect and identify ERP components phase-locked to a specific stimulus, its performance for the Motor Imagery paradigm was very low. The number and the location of MUSE's electrodes are not suitable for the development of a reliable active BCI system based on the Motor Imagery paradigm. For this reason, we decided to 'soften' our initial requirements, excluding the Motor Imagery paradigm and allowing control via identification of eye-blinks or eye movements that are known to produce clearer signals with larger amplitudes.

The Muse Headband

We decided to use the MUSE headband for the data acquisition in our experiments because it is one of the easier bands to get started with, as it requires no head preparation and is relatively cheap. It allows for wireless signal transport via Bluetooth and most importantly it broadcasts the raw EEG signal.

Figure 22: The MUSE headband and the location of its electrodes

The MUSE has electrodes located analogous to AF7, AF8, TP9, and TP10 of the 10-20 International System and utilizes the electrode at FPz as the reference electrode. It has a 256 Hz EEG Sampling Rate and an A/DC resolution of 12 bits.

Software Tools & Frameworks

Signal capturing & decoding

The digital processing algorithms have been written in Python 3. The main Python modules used were:

- **pandas** for file operations
- **numpy** for matrix calculations
- **mne** for epoching and time-domain & spatial filtering
- **sklearn** for classification algorithms
- **pyriemann** for classification using Riemannian geometry of covariance matrices
- **matplotlib** and **seaborn** for plotting.

EEG data streaming

For real-time streaming of EEG time series data over the network, we used a Python interface to the Lab Streaming Layer (LSL). The Lab Streaming Layer is a system designed to unify the collection of time series data for research experiments originally developed at the Swartz Center for Computational Neuroscience. LSL has become standard in the field of EEG-based brain-computer interfaces for its ability to make separate streams of data available on a network with time synchronization and near real-time access.

For the need of our system, we used the Python implementation of the core transport library, in order to:

- Stream real-time EEG data from the MUSE headband and synchronize them with the event marker data from the stimulus targets that appeared on the screen during the training data collection phase.
- Stream real-time EEG data, during the actual use of the system.

Controlling Home Automation

For the control of lights, we have developed a Python module responsible to interact with the Alexa Voice Service (“Alexa Voice Service,” n.d.) Amazon's suite of

services built around its voice-controlled AI assistant for the home and other environments. We first had to create a developer account with Amazon and then register a new product and create a security profile in the AVS Portal. After that, we implemented the required http requests to interact with the AVS API.

Because the AVS API accepts only requests containing speech (audio) we used a text to speech Python package to convert each command of the system to an audio file, which then we send to the AVS endpoint.

Offline Analysis

Before start developing the necessary modules for our BCI system, we decided first to perform some offline tests using data acquired directly from the MUSE. The rationale behind this was to first verify that the Motor Imagery paradigm can be supported from the MUSE, before proceeding with the online implementation.

Methodology

For the offline tests, we first prepared a Python script responsible for EEG data acquisition from the MUSE along with stimuli markers according to an experimental protocol. We then saved the acquired EEG data and the markers (as trial labels) into CSV files and analyzed them on an interactive python environment using Jupyter Notebooks.

In order to capture training data for offline analysis, we first developed a Python script responsible for the Bluetooth communication with MUSE and the presentation of various stimuli to the screen. The Python script starts two background process, one for EEG signal acquisition and one for stimuli presentation.

- The first process connects to the MUSE, receives EEG data using Bluetooth and pushes them continuously to an LSL stream.
- The stimulus presentation process is based on PsychoPy, an open source Python package for the generation of experiments for neuroscience and experimental psychology. We designed different experiments for the Motor Imagery and ERP Paradigms, both of which, start by repeatedly displaying images representing the stimuli of the experiment (e.g. left or right hand for the Motor Imagery paradigm). When a stimulus is displayed on the screen, the process pushes a marker with the class of the stimulus (e.g. left-hand or right-hand) to the second LSL stream.

After a predefined duration, the Python script closes the two streams, combines them and saves the results to a csv file. Having collected enough training trials to csv files, we then used the Jupyter Notebook Python environment for the analysis, pre-processing and classification tasks.

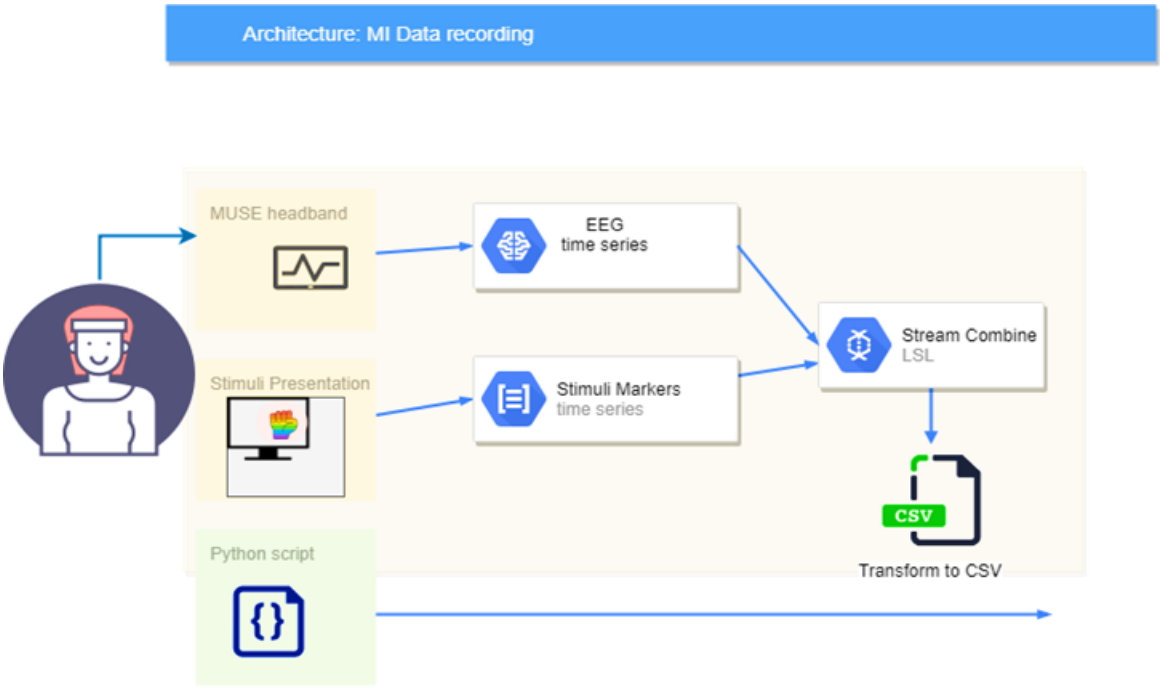


Figure 23: Data acquisition from MUSE (diagram)

Figure 24: Capturing data from the MUSE using Python

MI Paradigm Analysis

The purpose of our first experiment was to assess if the MUSE headband can detect the Motor Imagery phenomenon. For this, we first developed a Python script responsible for the collection of the training data. Then we analyzed the recorded data using the Python MNE library. Finally, we tried different feature extraction and classification pipelines on the captured data in order to evaluate the performance of a potential brain-computer interface using MUSE with the Motor Imagery Paradigm.

Data Recording Protocol

For the collection of the training data for the Motor Imagery Paradigm, we designed a training procedure in which each training session runs for a total duration of 2 minutes and contain multiples trials. In each trial, a fixation cross would first appear for a random time (less than a second), followed by the image of either a left or right fist on the screen. The user performed motor imagery while the image remained on screen for 4s. Each trial was followed by an inter-trial rest period of 2 s. With this procedure, each session contained approximately 20 trials.

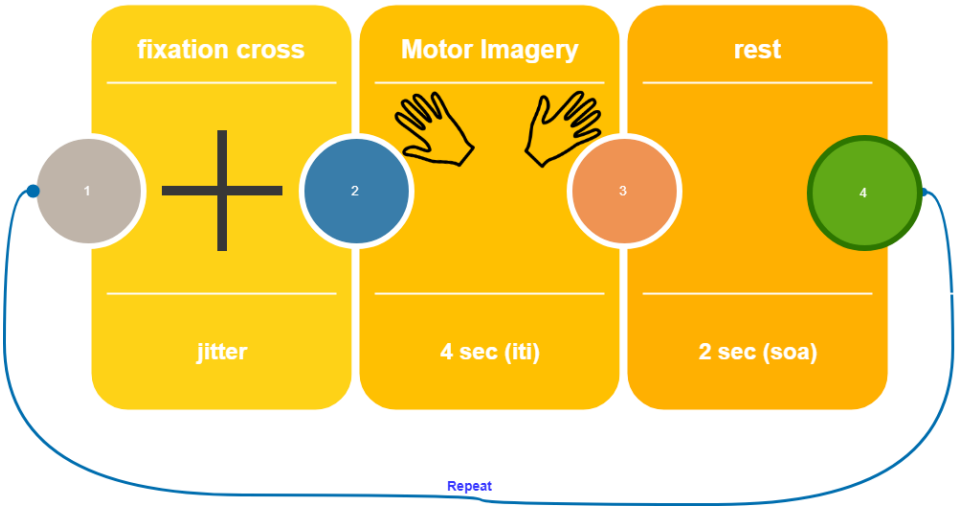


Figure 25: MUSE – Motor Imagery data recording protocol

Preprocessing and Analysis

Once we collected enough data for training (10 csv files of 2 minutes duration each) we proceeded with the preprocessing and the analysis of the data. For this part, we worked on Jupyter Notebooks in order to facilitate experimentations with various parameters and filters.

For the preprocessing, we used the pandas library to read and combine the data from the csv files and load them into a mne Raw object. Afterward, in order to check how noisy our data are, we plotted the power spectrum density (PSD). The PSD shows the signal power distribution along the range of frequencies. As can be seen from the PSD the data are noisy with a high peak on the upper frequencies. There is a large peak at 50 and 100 Hz, representing background electrical activity from the electric power grid.

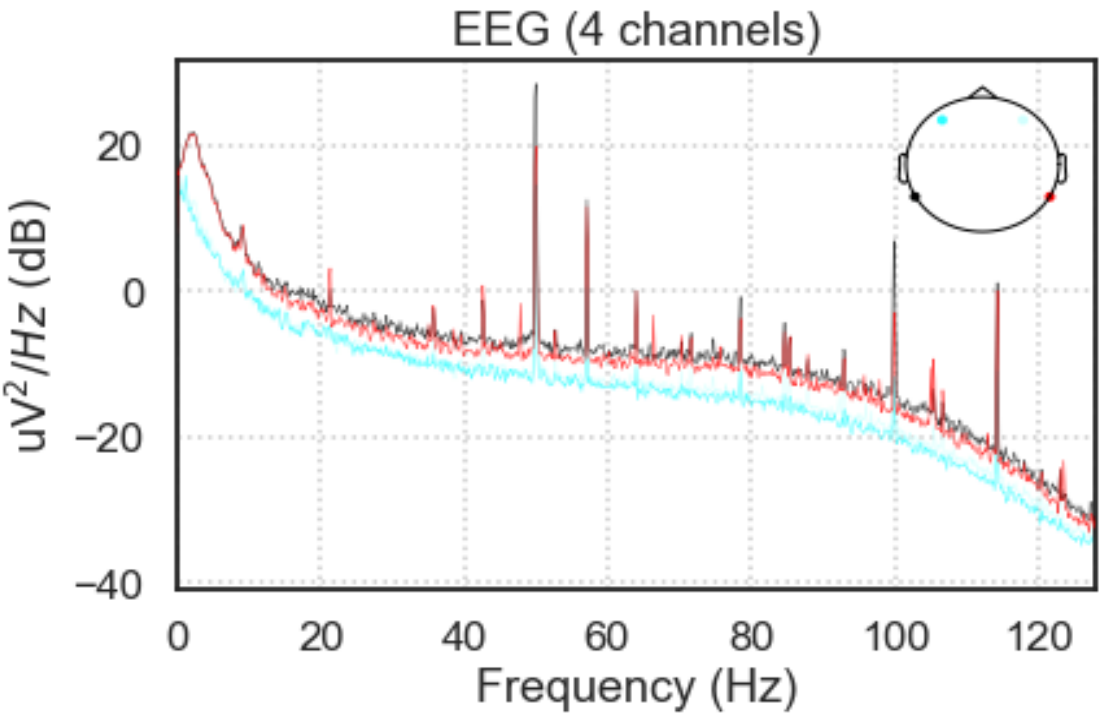


Figure 26: MUSE - The PSD of the Motor Imagery task

The peak at 50 and 100 Hz represents the line frequency from the electric power grid.

Because event-related desynchronization/synchronization is detected mainly in the μ (8- 12 Hz) and β (18 – 26 Hz) bands, we decided to filter the raw signal, in order to keep only frequencies between 1 and 30 Hz. As can be seen from the next Figure (the PSD of frequencies between 1 and 30 Hz), the filtered signal is less noisy. The huge peak from 1 to 3hz is largely due to the presence of eye blinks, which produce large amplitude, low-frequency events in the EEG.

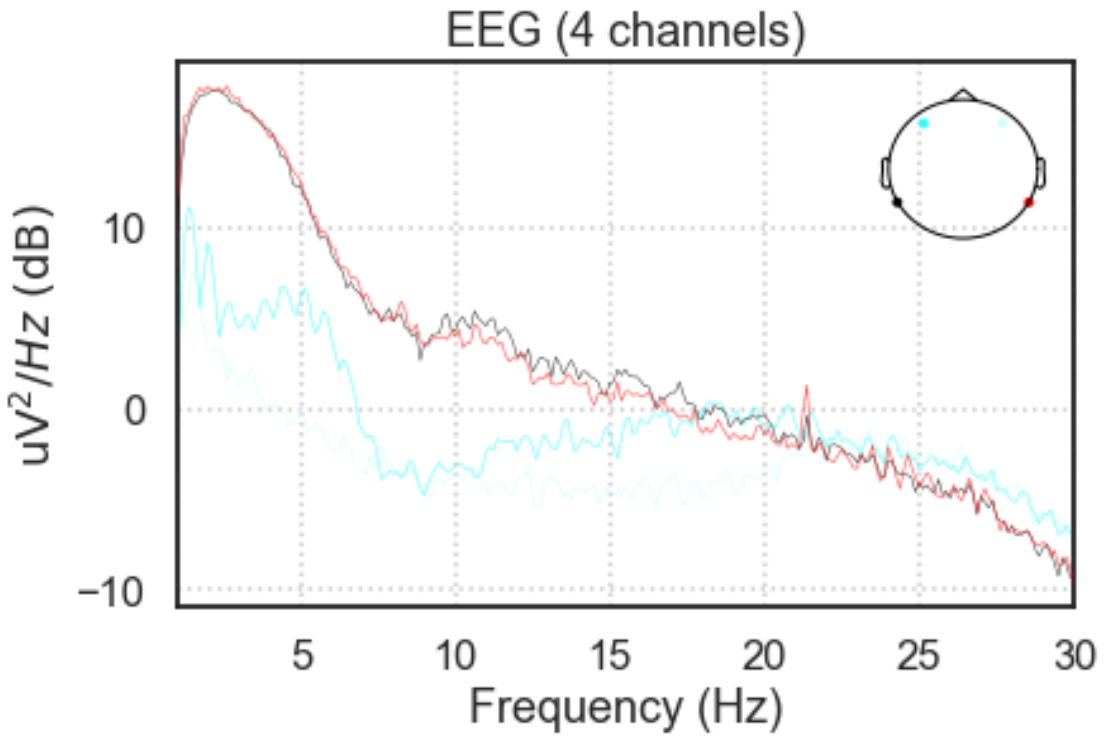


Figure 27: MUSE - The filtered PSD of Motor Imagery task

Next, we epoched the data into segments of 3 seconds representing the signal from 0.5 to 3.5 sec after each stimulus. We decided to use epochs that start 0.5 seconds after cue onset to avoid evoked potentials that can falsely contribute to BCI classification accuracy independent of user-driven modulation. We then rejected epochs where the amplitude of the signal exceeded 100 uV, which removed most trials with eye blinks. From a total of 120 epochs, we rejected about 1.7 % of them, remaining with 118 trials (71 for the left hand and 47 for right hand).

We used the CSP algorithm to detect components that maximize the difference between the 2 classes (left & right fist).

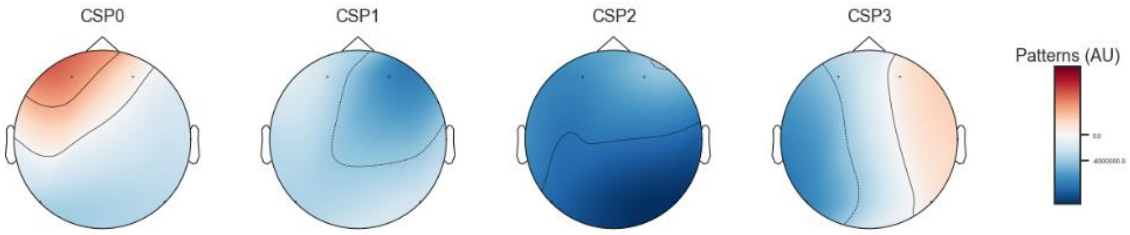


Figure 28: MUSE - The CSP components of the captured data

Features extraction and classification

For the features extraction step we applied the Common Spatial Patterns algorithm with 4 components, while for the classification step, we used both Logistic Regression and LDA. In order to be able to compare the results, we tested also 2 other pipelines based on vectorization of the data for features extraction with Logistic regression or LDA for the classification. In total we run the following four pipelines over the data:

- **Vect + LR**: Vectorization of the trial and Logistic Regression. This can be considered the standard decoding pipeline for MEG / EEG.
- **Vect + RegLDA**: Vectorization of the trial, Regularization and Linear Discrimination Analysis.
- **CSP4 + LR**: Spatial filtering using Common spatial Patterns algorithm (4 components) and Logistic Regression.
- **CSP4 + LDA**: Spatial filtering using Common spatial Patterns algorithm (4 components) and Linear Discrimination Analysis.

For the cross-validation, we used a Stratified Shuffle Split with 20 splits and a 75 – 25 ratio between training and test. The stratified approach ensures the preserving of the percentage of samples for each class. For each pipeline, we used the sklearn's **cross_val_score** method to evaluate a score by cross-validation. We used ROC AUC as

the scoring function and saved the results on a pandas data frame in order to display and visualize them.

Discussion of results

The best classification pipeline uses the CSP algorithm for the features extraction and has a mean AUROC score of **0.61**. This is very close to chance level and probably not enough to run a brain-computer interface.

Table 13: MUSE - Motor Imagery Classification results

Method	AUC
CSP4 + LDA	0.591
CSP4 + LR	**0.613**
Vect + LR	0.522
Vect + RegLDA	0.530

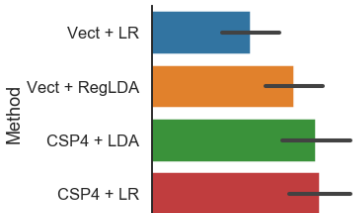


Figure 29: MUSE - Motor Imagery Classification results

The poor results of the MUSE for the Motor Imagery classification task can be explained by the number and the location of its electrodes. The sensorimotor rhythms (SMRs) of motor imagery are primarily linked to the central area of the brain. Most researches focused on the classification of EEG signals of left / right hand motor imagery are proposing the use of data captured from C3 and C4 electrodes which are located on the top of the head. The MUSE has the minimum number of sensors and its electrodes sites (TP9, AF7, AF8, TP10) are not well positioned for the motor imagery task.

Another factor that can explain the poor performance of the MUSE decoding compared to the decoding of the two public datasets, is the biological, environmental and instrumental artifacts that had probably contaminated our raw EEG signal. These contaminations can better be controlled and suppressed in a laboratory as compared to a home environment.

ERP Paradigm Experiments

The purpose of our second experiment was to assess if the muse headband is capable of detecting ERP components. We have chosen the N170 ERP component for this experiment because MUSE's temporal electrodes (TP9 and TP10) are well positioned to detect the N170 which is most easily detected at lateral posterior electrodes.

The N170 is a large negative event-related potential (ERP) component that occurs after the detection of faces, but not objects or other body parts such as hands. The N170 occurs around 170ms after face perception and is most easily detected at occipital-temporal electrode sites such as T5 and T6 (Eimer & Williams, 2000). Although there is no consensus on the specific source of the N170, researchers believe it is related to activity in the fusiform face area, an area of the brain specialized for facial recognition.

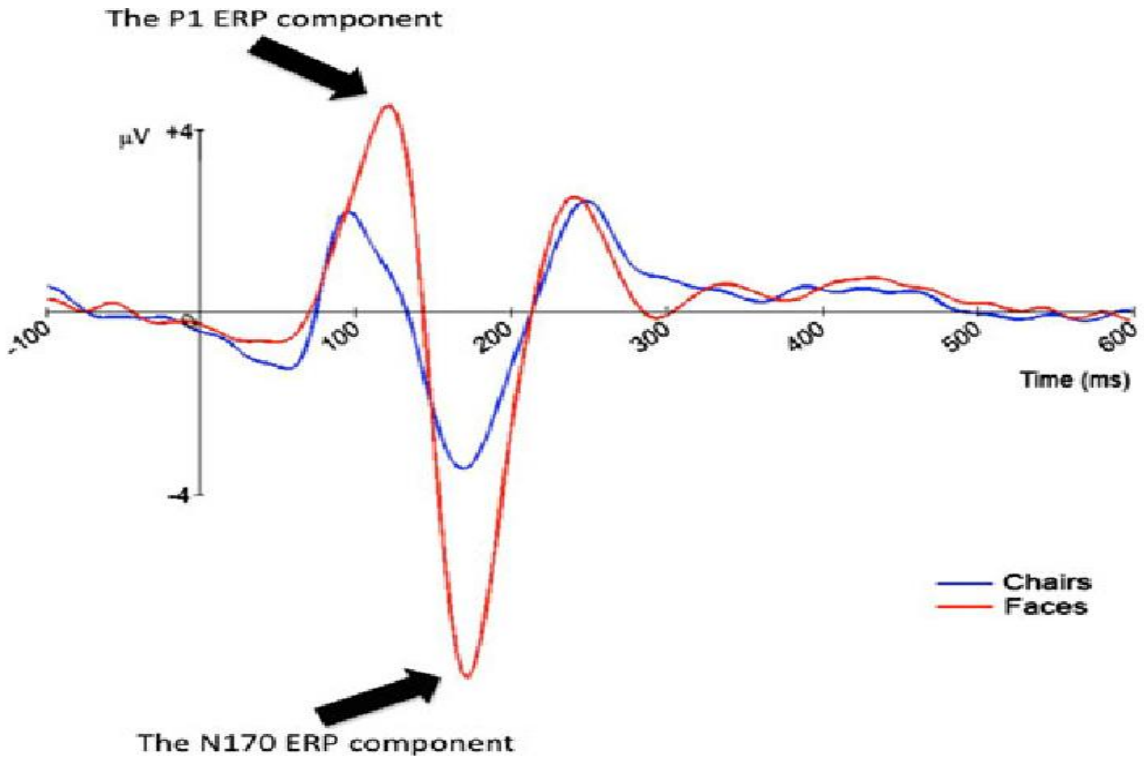


Figure 30: The N170 ERP component

In order to verify the N170 component, we developed a Python script responsible for the presentation of stimuli that include faces images and the recording of the corresponding brain activity. Using Jupyter Notebooks, we analyzed the recorded data and applied different classification pipelines in order to evaluate the performance of a potential brain-computer interface using the N170 ERP component.

Data Recording Protocol

For the collection of the training data for the N170 ERP Paradigm, we designed a training procedure in which each training session runs for a total duration of 2 minutes. When a session starts, a series of images are displayed on the screen, some of which are showing faces. Each image is displayed for an interval between 0.8 and 1 sec. (iti = 0.8sec + jitter). There is also a small stimulus onset asynchrony (soa) of 0.2 seconds between each trial. The user is looking at the screen and his EEG signal is recorded simultaneously with the stimulus markers representing the currently displayed image. With this procedure, each session contained approximately 120 trials.

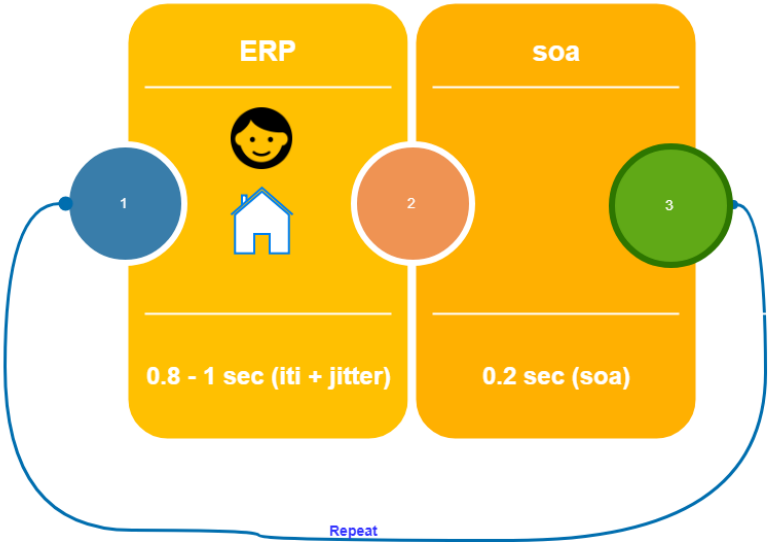


Figure 31: MUSE - ERP data recording protocol

Preprocessing and Analysis

Once we collected enough data for training (10 csv files of 2 minutes duration each) we proceeded with the preprocessing and the analysis of the data in order to see if we can

identify the N170. For the preprocessing, we first used the MNE library to read the data from the csv files and load them into an MNE Raw object.

Frequency domain analysis

In order to check how noisy our data are, we plotted the power spectrum density of the dataset. As can be seen from the next figure, the PSD looks good. There is a large peak at 50hz, representing background electrical activity from the electric power grid.

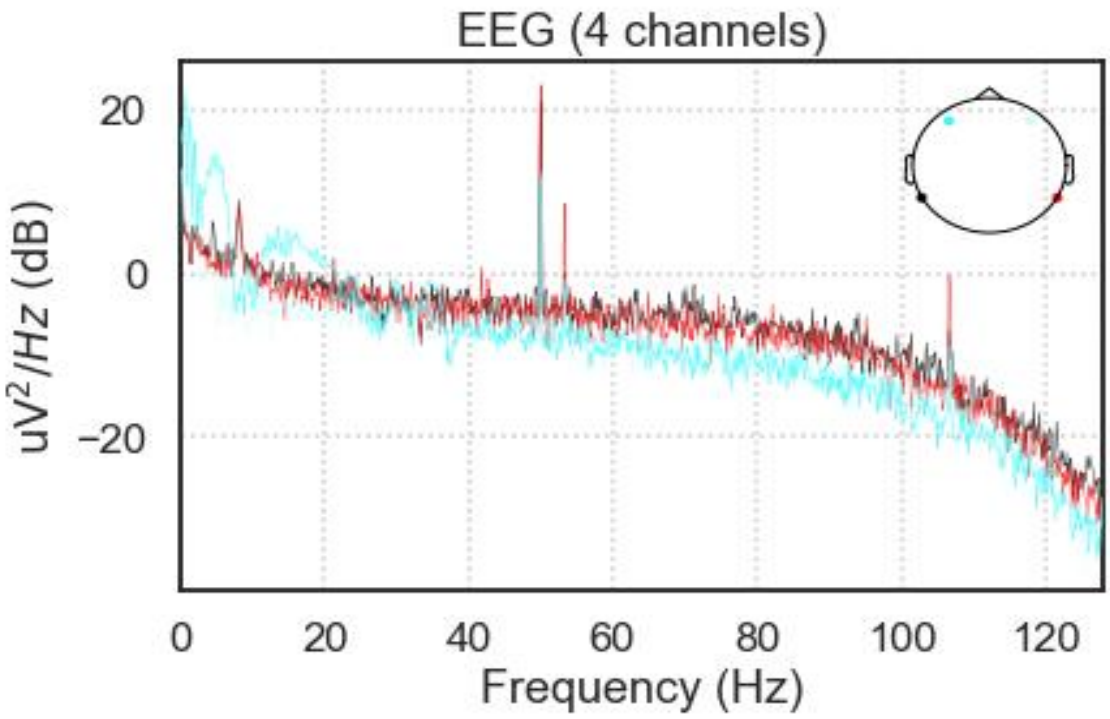


Figure 32: MUSE - The PSD of the ERP task

The peak at 50hz represents the line frequency from the electric power grid.

Because most ERP components are composed of lower frequency fluctuations, we decided to apply a band-pass filter in order to keep only frequencies between 1 and 30 Hz. This increased the ability to detect the underlying component in our data.

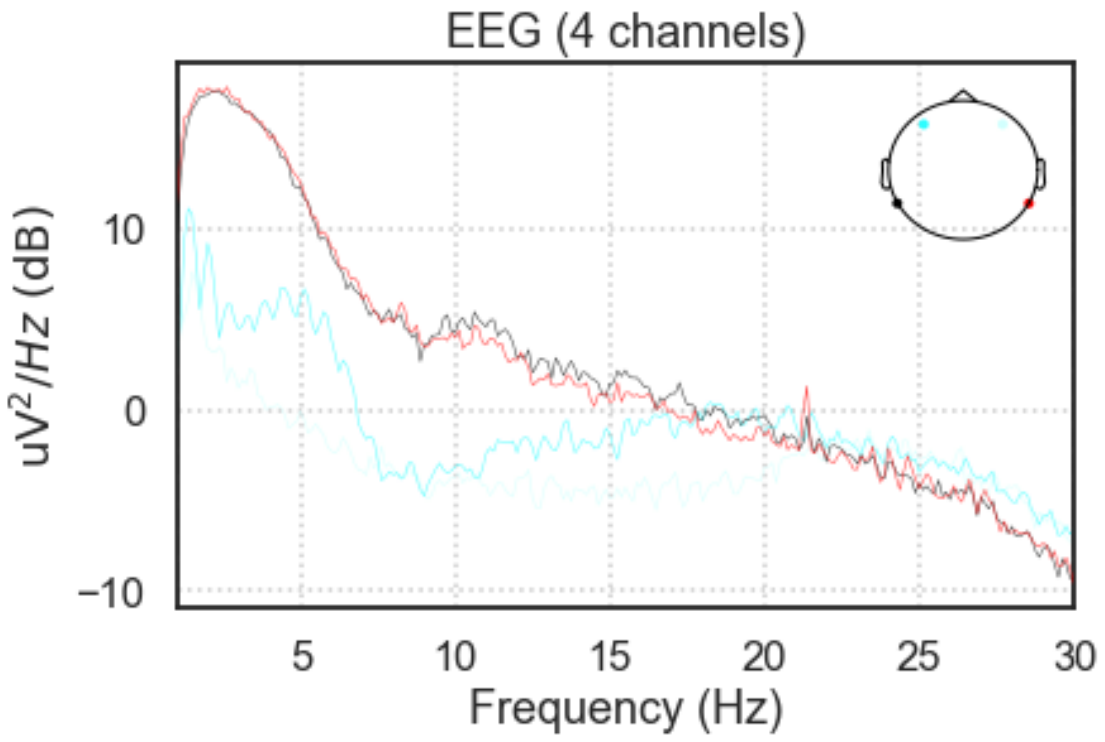


Figure 33: MUSE - The filtered PSD of the ERP task

As can be seen from Figure 33, (the PSD of frequencies between 1 and 30 Hz) the difference between the temporal channels (red and black) and the frontal channels (blue and green) is clearly evident. The huge peak from 1 to 3hz is largely due to the presence of eye blinks, which produce large amplitude, low-frequency events in the EEG.

Time domain Analysis

After the filtering, we epoched the data into segments of 900ms representing the data 100ms before to 800ms after each stimulus. We also rejected epochs where the amplitude of the signal exceeded 100 uV, which removed most trials with eye blinks. From a total of 1300 epochs we rejected about 4% of them and then we averaged the remaining epochs over each class and plotted the corresponding waveforms over each electrode.

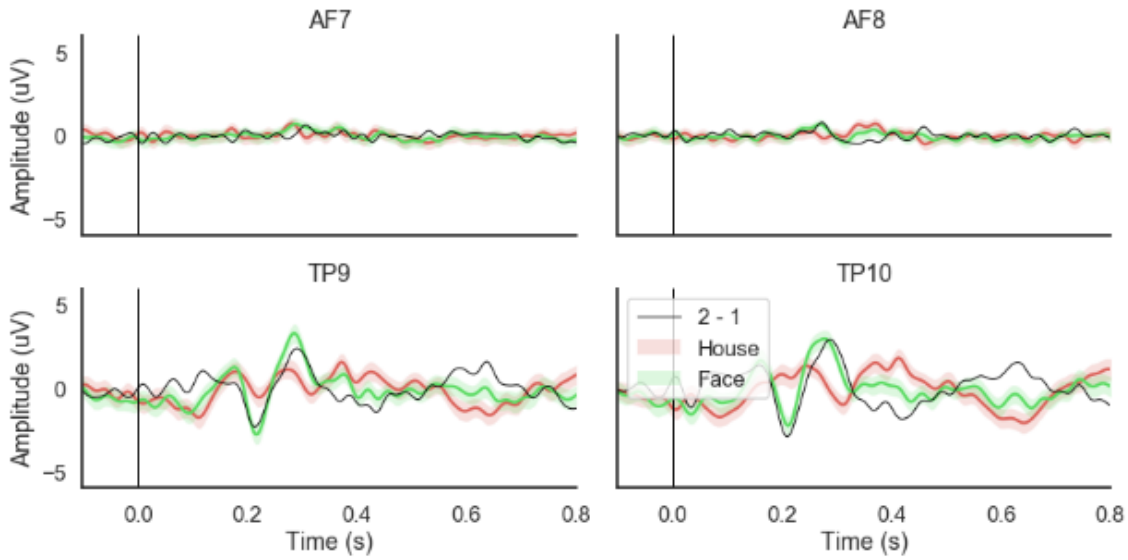


Figure 34: MUSE - averaged waveforms over each electrode for the N170

There is an evident deflection in the temporal channels, representing the N170 component.

In this plot, there is a noticeable deflection in the temporal channels around 200ms for face stimuli. This is likely the N170, although appearing slightly later due to delay in receiving the data over Bluetooth. There's not much to see in the frontal channels (AF7 and AF8), but that is expected, as the N170 is mostly a lateral posterior brain phenomenon.

Features extraction and classification

We used seven pipelines to evaluate different approaches for the features extraction and classification of the EEG signals. Most of them are commonly used in ERP-based BCIs:

- **Vect + LR**: Vectorization of the trial with Logistic Regression. This can be considered the standard decoding pipeline for MEG / EEG.
- **Vect + RegLDA**: Vectorization of the trial and Regularized Linear Discrimination Analysis (LDA). This adds an extra step of transforming the data such that its distribution will have a mean value 0 and standard deviation of 1.
- **ERPCov + TS + LR**: ERP Covariances with tangent space mapping. One of the most reliable Riemannian geometry-based pipelines.

- **ERPCov + TS +SVM Rbf**: The same with the previous pipeline, but using SVM with an Rbf kernel for the classification step.
- **ERPCov + MDM**: ERP Covariances with Riemann Distance to Geometric mean classifier. A very simple, yet effective (for low channel count), Riemannian geometry classifier.
- **XdawnCov + TS + LR**: data are dimensionally reduced using XDawn algorithm, the covariance matrices for each ERP epoch are calculated and projected to Tangent space. Logistic Regression is used for the classification step.
- **XdawnCov + MDM**: The xDAWN algorithm is used to reduce the dimensionality of the data and then Riemann Distance and Geometric mean are applied to classify the covariance matrices.

For the cross-validation, we used a Stratified Shuffle Split with 20 splits and a 75 / 25 ratio between training and test. The stratified approach ensures the preserving of the percentage of samples for each class. For each pipeline, we used the sklearn's **cross_val_score** method to evaluate a score by cross-validation. We used ROC AUC as the scoring function and saved the results on a pandas dataframe in order to display and visualize them.

Discussion of results

The best classifiers for this dataset appear to be the ERPCov and XdawnCov with tangent space projection pipelines. AUC is around .71, which is good, but on the low end for being able to run a brain-computer interface. The Muse's temporal electrodes (TP9 and TP10) are well positioned to detect the N170 and it allowed us to detect the N170 emerge from just a few dozen trials. The performance would be probably much better is we used an EEG device with more channels.

Table 14: MUSE - ERP Classification results

VECT + LR	0.673
VECT + REGLDA	0.685
ERPCOV + TS + LR	**0.712**
ERPCOV + MDM	0.698
XDAWNCOV + TS + LR	**0.712**
XDAWNCOV + MDM	0.693
ERPCOV + TS + SVM RBF	0.703

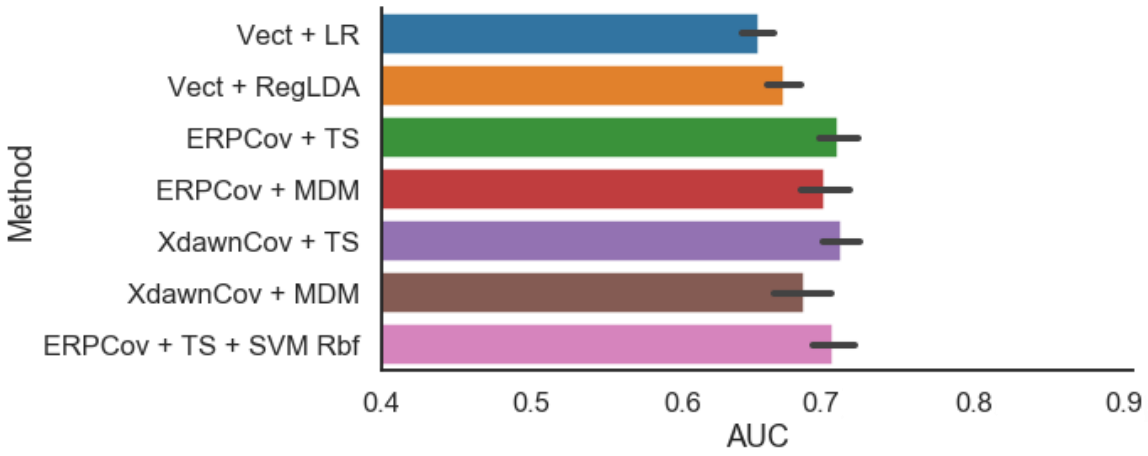


Figure 35: MUSE - ERP Classification results

Experimental Online BCI using MUSE

The accuracy level of Motor Imagery classification using MUSE is only a bit higher than chance. Considering that the performance we obtained using MUSE was evaluated only offline, we are not very optimistic about the ability of MUSE to support an online BCI based on the Motor Imagery paradigm. To be able to operate an active BCI using MUSE, we decided to 'soften' our requirements replacing the Motor Imagery paradigm with simpler paradigms that can be detected with higher accuracy. We tried two different mental paradigms for the control of the smart-home environment. The first distinguished between the rest state, where user relaxes without performing any mental task, and another state in which the user performs mental calculus. The second paradigm uses eye-blinks that are known to produce a stronger signal which can be detected with higher accuracy. The basic modules of our BCI are presented below.

Stream Receiver

The base module for acquiring signals from the MUSE. The underlying data communication is based on Lab Streaming Layer (LSL) which provides sub-millisecond time synchronization accuracy.

Stream Recorder

Connects to the Stream Receiver LSL stream and records signals into csv files. It also receives and records stimulus events for the epoching of the EEG signals.

Stimulus Presentation

This module is responsible for the presentation of the stimulus during the collection of training data. We used the Python pygame library to create various visual presentation scenarios. When a stimulus is presented to the user, this module sends a marker to the Stream Recorder module with the timing and the class of the event.

Offline Trainer

Reads the data from the saved files and trains a pipeline using a specific combination of feautures extraction methods and classifier for each mental paradigm. The

signal is filtered in the range of 8 to 30 Hz and epoched based on the events captured during the training. Subsequently, it runs a 10-Fold cross-validation on the epoched data in order to calculate the mean performance of the classifier. The best pipeline is serialized and saved for later use using the pickle Python module.

Online Decoder

This is the module that runs our BCI during actual use. The decoder first loads the best saved pipeline from the disk and then connects and acquires EEG raw data from the Stream Receiver module. The raw data are then buffered and send (in chunks) to the trained model to extract probabilities about each class. We use a 65% threshold filter on the output probability in order to decrease false positives. This module makes use of multi-processing parallel execution to achieve high-frequency decoding.

Amazon Alexa Agent

When a control signal is detected, the Online Decoder module forwards the predicted command to the Amazon Alexa Agent module. This module communicates with the Amazon Voice Service asking it to perform a specific action. Because the Alexa Voice Service API accepts only requests containing speech (audio) we used the Windows SAPI programming interface to convert each command of the system to an audio file, which then we upload to the AVS endpoint. The Amazon Voice Service interprets the command and communicates with the Amazon echo device which in turn sends the requested command to the smart light bulbs.

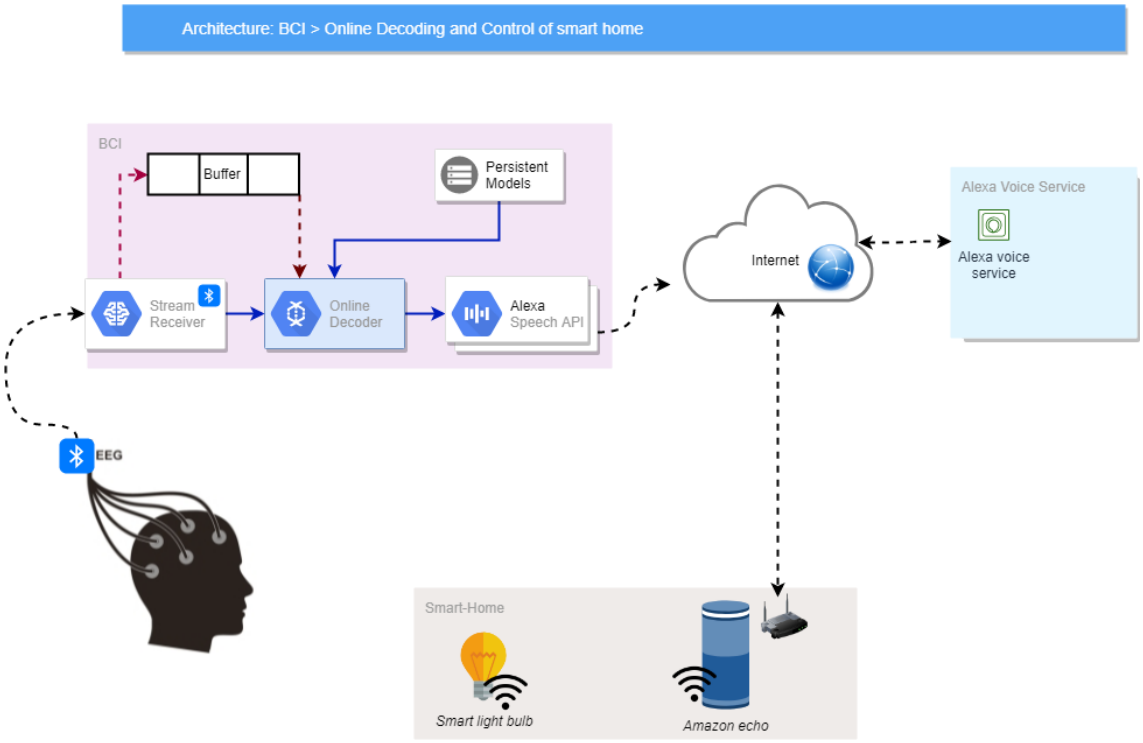


Figure 36: The basic architecture of the implemented BCI system

The user thoughts are translated to control signals

After multiple experiments with the online decoder, we trained two different families of pipelines. The first distinguishes 2 states; a mental rest state where the user stays calm without thinking and a mental intense calculus state during which, the user is performing internal arithmetic calculations. The second pipeline detects 3 states; rest, left eye blink and right eye blink.

The first pipeline uses feautures based on the signal power on the alpha, theta and delta bands, as well as, a constructed feature based on the Alpha/Theta Protocol, a popular neurofeedback metric for stress reduction. The second pipeline uses an averaging time-window and detects the eye blinks using the maximum voltage in this window. Because the eye blinking mostly affects the frontal electrodes, we decided to use data only from the electrodes AF7 and AF8 that are located on the forehead. After testing both pipelines, we concluded that the one based on eye-blinks is more suitable for our scenario because it can express two control commands and it allows better control and shorter feedback times.

Chapter 8.

Summary and Future Work

Summary

The purpose of this thesis was to study Brain-Computer Interfaces both theoretically and in practice. We started the thesis with an introduction into brain-computer interfaces and gave an overview of types and applications for BCI. We discussed the emerging commercialization of EEG equipment and the trends in this field. We then divided the rest of this thesis into three parts.

In the first part, we presented the workflow for brain signal classification and reviewed the current trends and algorithms for extracting features and classifying EEG signals. We also presented the main supervised classification approaches and made a brief introduction to deep learning and neural networks.

In the second part, we conducted a comparative empirical study of various EEG classification pipelines on two public Motor Imagery datasets. We presented the results of a series of offline experiments and pointed out the importance of the signal pre-processing stage. For this part, we developed and presented a methodology for improved classification results by automatic optimization of the pre-processing configuration, via the search in a space of possible pre-processing parameters. Using this approach, we got good results for cross-subject classification of Motor Imagery trials on both public datasets. Although the good results, we would like to emphasize that both datasets had been gathered from synchronized tasks, allowing to know the exact onset and offset time of imagery events. This bypasses many of the difficulties in signal cleaning and normalization. However, in real applications, the MI-based BCI needs to be asynchronous. This raises the need to explore other approaches for the signal classification.

Moving forward, in the third and final part, we experimented with the MUSE headband and built a prototype home-control BCI, supporting 1) real-time communication and acquisition of EEG signal from MUSE, 2) online decoding and classification of the captured signal, and 3) home-automation control via the Amazon Alexa Voice Service.

All three parts of our BCI system are written in Python, which allows our system to run independently on most modern operating systems.

Future Work

This research proposed a system that uses a commercial EEG headband (MUSE) to interact with a smart-home environment. The performance of the prototype BCI system was severely reduced, mainly because of the limitations of the EEG headset. Furthermore, many initial ideas, adaptations, tests, and experiments, were left for the future due to lack of time or resources (i.e. the experiments with deep learning methods was very time consuming, requiring even days to finish on our laptop).

In a future work, we would opt for a better (and more expensive) EEG device with more channels and improved signal acquisition capabilities. Having access to more raw data, we would then focus on developing more robust and consistently efficient algorithms that can be used easily and online and are able to work with small training samples, noisy signals, high-dimensional and non-stationary data. A promising approach in this direction would be the adoption of Reinforcement learning methods for the signal classification. Future work may also be done on multi-class classification methods, allowing the detection of more commands from the system and thus a better control experience for the user.

An initial idea we had little time to test and maybe a possible extension of this work is the use of the emerging technology of Web Bluetooth for the physical Web. Web Bluetooth is an API that enables interaction with Bluetooth devices through web browsers. Interaction through browsers is a key part of the physical web that adds many possibilities for device interaction. In the case of BCI systems, a possible use could be the crowdification of the training data acquisition. Users owning EEG headbands could simply open a URL and record their EEG signal under different experimental protocols. This would allow the collection of massive amounts of data that could be fed to train data 'hungry' deep learning models. Such a system could also allow for an automatic signal screening process, rejecting sessions where the recorded signal does not satisfy specific criteria. Web Bluetooth can also be used to simplify and decentralize the architecture of a BCI system that will control a smart-home. It will allow zero-configuration deployments and instant use of the system, eliminating the need for a dedicated computer acting as a middleware between the headset and the smart-home devices. Finally, cloud-based intelligence could be used to adapt and improve the system's performance, by exploiting reinforcement learning methods on new training data from multiple users.

A different possible future extension of the work presented in this thesis could be the combination of an EEG device with a Virtual Reality (VR) Headset in order to build a brain-controlled AR/ VR environment with increased channels of input. Such an environment could be both responsive and adaptive to user behavior combining an active and a passive BCI. An active BCI would explicitly allow users to issue commands or to enter text in order to interact with the VR environment, while the passive BCIs could monitor the user's state (e.g. workload level, attentional state) and adapt the VR/AR interface.

Connecting human minds to various technological devices and applications through brain-computer interfaces (BCIs), will allow intriguingly novel ways for humans to engage and interact with the world and with each other. In the following decades, BCI systems that only a few years ago might have been considered science fiction, will be realized and become part of our lives. The only limit is our imagination.

Bibliography

Alexa Voice Service. (n.d.). Retrieved December 16, 2018, from https://developer.amazon.com/alexa-voice-service

Alimardani, M., Nishio, S., & Ishiguro, H. (2016). The Importance of Visual Feedback Design in BCIs; from Embodiment to Motor Imagery Learning. *PLOS ONE*, *11*(9), e0161945. https://doi.org/10.1371/journal.pone.0161945

Alomari, M. H., Samaha, A., AlKamha, K., H., Mohammad, A., Samaha, A., & AlKamha, K. (2013). Automated Classification of L/R Hand Movement EEG Signals using Advanced Feature Extraction and Machine Learning. *International Journal of Advanced Computer Science and Applications*, *4*(6), 6. https://doi.org/10.14569/IJACSA.2013.040628

Ang, K. K., Chin, Z. Y., Wang, C., Guan, C., & Zhang, H. (2012). Filter Bank Common Spatial Pattern Algorithm on BCI Competition IV Datasets 2a and 2b. *Frontiers in Neuroscience*, *6*. https://doi.org/10.3389/fnins.2012.00039

Barachant, A. (2014). *MEG decoding using Riemannian Geometry and Unsupervised classification.* Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=A4E3625B2376C1E3310DA262731CB7F6?doi=10.1.1.713.5131&rep=rep1&type=pdf

Barachant, A. (2015). pyRiemann v0.2.2. https://doi.org/10.5281/ZENODO.18982

Bashivan, P., Rish, I., Yeasin, M., & Codella, N. (2015). LEARNING REPRESENTATIONS FROM EEG WITH DEEP RECURRENT-CONVOLUTIONAL NEURAL NETWORKS. In *ICLR 2016*. Retrieved from https://arxiv.org/pdf/1511.06448.pdf

BCI 2000. (n.d.). Retrieved from http://www.schalklab.org/research/bci2000

Blankertz, B., Tangermann, M., Vidaurre, C., Fazli, S., Sannelli, C., Haufe, S., … Müller, K. R. (2010). The Berlin brain-computer interface: Non-medical uses of BCI technology. *Frontiers in Neuroscience*. https://doi.org/10.3389/fnins.2010.00198

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (n.d.). *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. Retrieved from

https://arxiv.org/pdf/1406.1078v3.pdf

Cohen, M. X. (2014). *Analyzing Neural Time Series Data: Theory and Practice (Issues in Clinical and Cognitive Neuropsychology)*. Cambridge. https://doi.org/2008-2045

CREx. (n.d.). Filtering – The Basics [Image]. Retrieved from https://blricrex.hypotheses.org/filtering-introduction

CREx. (2018). Resampling. Retrieved October 28, 2018, from https://blricrex.hypotheses.org/ressources/eeg/pre-processing-for-erps/resampling

Doud, A. J., Lucas, J. P., Pisansky, M. T., & He, B. (2011). Continuous Three-Dimensional Control of a Virtual Helicopter Using a Motor Imagery Based Brain-Computer Interface. *PLoS ONE*, *6*(10), e26322. https://doi.org/10.1371/journal.pone.0026322

Eimer, M., & Williams, L. (2000). *The face-specific N170 component reflects late stages in the structural encoding of faces*. Retrieved from http://www.brainb.psyc.bbk.ac.uk/PDF/NR3.PDF

Faust, M. (2012). *The handbook of the neuropsychology of language*. Wiley-Blackwell. Retrieved from https://books.google.gr/books?id=_2bneq9tl9sC&dq=.+In+this+context,+the+cognitive+“events”+of+interest+may+include+a+particular+class+of+stimulus,+the+absence+of+an+expected+stimulus

Fedjaev, J. (2017). *Decoding EEG Brain Signals using Recurrent Neural Networks*. Technische Universität München. Retrieved from http://mediatum.ub.tum.de/doc/1422453/552605125571.pdf

Fergus, P., Hignett, D., Hussain, A., Al-Jumeily, D., & Abdel-Aziz, K. (2015). Automatic epileptic seizure detection using scalp EEG and advanced artificial intelligence techniques. *BioMed Research International*, *2015*. https://doi.org/10.1155/2015/986736

Goodfellow, I., Bengio, Y., & Courville, A. (n.d.). *Deep learning*. Retrieved from https://books.google.gr/books?id=Np9SDQAAQBAJ&printsec=frontcover&dq=deep+learning+book&hl=el&sa=X&ved=0ahUKEwju5czihtTeAhWRlosKHeKvA24Q6AEIMDAB#v=onepage&q=deep learning book&f=false

He, H., Wu, D., & Member, S. (2018). *Transfer Learning for Brain-Computer Interfaces: An Euclidean Space Data Alignment Approach*. Retrieved from https://arxiv.org/pdf/1808.05464.pdf

Heyden, M. (2016). Classification of EEG Data Using Machine Learning Techniques. Retrieved from http://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=8895013&fileOId=8895015

Hochreiter, S., & Urgen Schmidhuber, J. J. (1997). *LONG SHORT-TERM MEMORY*. *MEMORY Neural Computation* (Vol. 9). Retrieved from http://www7.informatik.tu-muenchen.de/~hochreithttp://www.idsia.ch/~juergen

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, *4*(2), 251–257. https://doi.org/10.1016/0893-6080(91)90009-T

Hülsmann, J., Jirku, M., Dyck, A. (2017). *EEG Signal Decoding and Classification*. Retrieved from https://www.ni.tu-berlin.de/fileadmin/fg215/teaching/nnproject/EEG_Signal_Decoding_and_Classification.pdf

Krigolson, O. E., Williams, C. C., & Colino, F. L. (2017). Using Portable EEG to Assess Human Visual Attention, 56–65. https://doi.org/10.1007/978-3-319-58628-1

Larson, E., Gramfort, A., Engemann, D. A., jaeilepp, Brodbeck, C., Jas, M., … Ruzich, E. (2018). mne-tools/mne-python: v0.17. https://doi.org/10.5281/ZENODO.1491843

Leeb, R., Brunner, C., Müller-Putz, G. R., Schlögl, A., & Pfurtscheller, G. (2008). BCI Competition 2008 – Graz data set B Experimental paradigm. *Knowledge Creation Diffusion Utilization*.

Ling, C. X., & Zhang, H. (2003). AUC: a Statistically Consistent and more Discriminating Measure than Accuracy. https://doi.org/https://doi.org/10.1007/3-540-44886-1_25

Lotte, F. (2014). *A Tutorial on EEG Signal Processing Techniques for Mental State Recognition in Brain-Computer Interfaces. Eduardo Reck Miranda; Julien Castet. Guide to Brain-Computer Music Interfacing*. Springer. Retrieved from https://hal.inria.fr/hal-01055103

Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F., Arnaldi, B., Lotte, F., … Arnaldi, B. (2007). *A review of classification algorithms for EEG-based brain-computer interfaces A review of classification algorithms for EEG-based brain A Review of Classification Algorithms for EEG-based Brain-Computer Interfaces*. *Journal of Neural Engineering* (Vol. 4). IOP Publishing. Retrieved from https://hal.inria.fr/inria-00134950

Lotze, M., Montoya, P., Erb, M., Hülsmann, E., Flor, H., Klose, U., … Grodd, W. (1999). Activation of Cortical and Cerebellar Motor Areas during Executed and Imagined Hand Movements: An fMRI Study. *Journal of Cognitive Neuroscience*, *11*(5), 491–501. https://doi.org/10.1162/089892999563553

Nasiri Ghosheh Bolagh, S., Bagher Shamsollahi, M., Jutten, C., & Congedo, M. (2016). Unsupervised Cross-Subject BCI Learning and Classification using Riemannian Geometry. In *ESANN 2016 CONFERENCE*. Bruges, Belgium. Retrieved from http://e-nns.org/2016/02/esann-2016-conference/

Nunez, P. L., & Williamson, S. J. (1996). Neocortical Dynamics and Human EEG Rhythms. *Physics Today*, *49*(4), 57–57. https://doi.org/10.1063/1.2807585

Olson, R. S., Urbanowicz, R. J., Andrews, P. C., Lavender, N. A., Kidd, L. C., & Moore, J. H. (2016). Automating biomedical data science through tree-based pipeline optimization. Retrieved from http://arxiv.org/abs/1601.07925

OMR. (2018). Global Brain Computer Interface Market Research and Forecast 2018-2023. Retrieved February 12, 2019, from https://www.omrglobal.com/industry-reports/bci-brain-computer-interface-market/

Ossmy, O., Tam, O., Puzis, R., Rokach, L., Inbar, O., & Elovici, Y. (2017). MindDesktop: a general purpose brain computer interface. Retrieved from http://arxiv.org/abs/1705.07490

Pires, G., Torres, M., Casaleiro, N., Nunes, U., & Castelo-Branco, M. (2011). Playing Tetris with non-invasive BCI. In *2011 IEEE 1st International Conference on Serious Games and Applications for Health, SeGAH 2011*. https://doi.org/10.1109/SeGAH.2011.6165454

Rao, R. P. N., Stocco, A., Bryan, M., Sarma, D., Youngquist, T. M., Wu, J., & Prat, C. S. (2014). A Direct Brain-to-Brain Interface in Humans. *PLoS ONE*, *9*(11), e111332. https://doi.org/10.1371/journal.pone.0111332

Rodrigues, P., Bouchard, F., Congedo, M., Jutten, C., Rodrigues, P., Bouchard, F., … Re-, C. J. D. (2017). Dimensionality Reduction for BCI classification using Riemannian geometry To cite this version : DIMENSIONALITY REDUCTION FOR BCI CLASSIFICATION USING.

Russell, S. J. (Stuart J., Norvig, P., & Davis, E. (2010). *Artificial intelligence : a modern approach*. Prentice Hall.

Schalk, G., McFarland, D. J., Hinterberger, T., Birbaumer, N., & Wolpaw, J. R. (2004). BCI2000: A General-Purpose Brain-Computer Interface (BCI) System. *IEEE Transactions on Biomedical Engineering*, *51*(6), 1034–1043. https://doi.org/10.1109/TBME.2004.827072

Schirrmeister, R. T., Springenberg, J. T., Dominique, L., Fiederer, J., Glasstetter, M., Eggensperger, K., … Ball, T. (n.d.). Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human.

Schirrmeister, R. T., Springenberg, J. T., Fiederer, L. D. J., Glasstetter, M., Eggensperger, K., Tangermann, M., … Ball, T. (2017). Deep learning with convolutional neural networks for EEG decoding and visualization. *Human Brain Mapping*, *38*(11), 5391–5420. https://doi.org/10.1002/hbm.23730

Sleight, J., Pillai, P., & Mohan, S. (2009). Classification of Executed and Imagined Motor

Movement EEG Signals. *Analysis*, 1–10.

Souloumiac, A., Souloumiac, A., Attina, V., & Gibert, G. (2009). xDAWN Algorithm to Enhance Evoked Potentials: Application to Brain Computer Interface. *Biomedical Engineering, IEEE Transactions On*, *56*(8), 2035–2043. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.700.1457

Strategien, R. L., Master-thesis, G. C. S., Sharma, D., Tag, M., Rueckert, E., Peters, J., & Grosse-wentrup, M. (n.d.). Adapting Brain Signals With Reinforcement Learning Strategies for Brain Computer Interfaces.

Van De Laar, B., Gurkok, H., Plass-Oude Bos, D., Poel, M., & Nijholt, A. (2013). Experiencing BCI control in a popular computer game. *IEEE Transactions on Computational Intelligence and AI in Games*. https://doi.org/10.1109/TCIAIG.2013.2253778

Widmann, A., Schröger, E., & Maess, B. (2015). Digital filter design for electrophysiological data – a practical approach. *Journal of Neuroscience Methods*, *250*, 34–46. https://doi.org/10.1016/J.JNEUMETH.2014.08.002

Yoo, S.-S., Kim, H., Filandrianos, E., Taghados, S. J., & Park, S. (2013). Non-Invasive Brain-to-Brain Interface (BBI): Establishing Functional Links between Two Brains. *PLoS ONE*, *8*(4), e60410. https://doi.org/10.1371/journal.pone.0060410