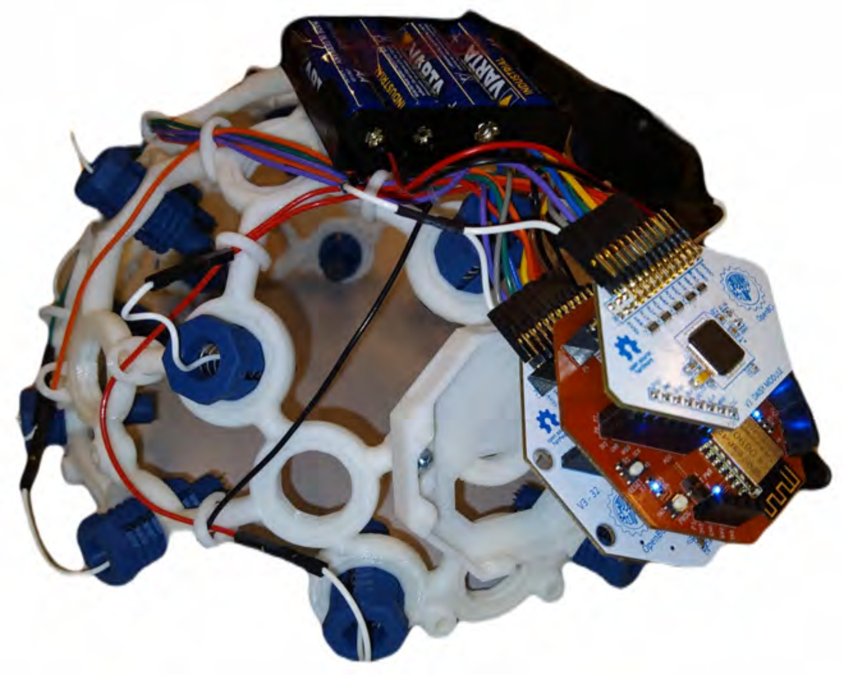
EEG Data Classification

Classifying Motor Imagery EEG data to Create a Brain Computer Interface For Gaming



Author:	A. van Suijdam BSc
Supervisor:	Dr. M. Hoogendoorn
Second Reader:	Dr. R. Bekker
External Supervisor:	W. de Nie MA MSc
Submission Date:	May 2022

EEG Data Classification

Classifying Motor Imagery EEG data to Create
a Brain Computer Interface For Gaming

Arnoud van Suijdam

Vrije Universiteit Amsterdam
Faculty of Science
Business Analytics
De Boelelaan 1081a
1081 HV Amsterdam

Sogeti
Lange Dreef 17
4131 NJ Vianen

April 2022

Preface

The thesis is written as a graduation project of the Master Business Analytics of the Vrije Universiteit of Amsterdam. The project is intended as a deepening of the study in which it is in line with the learning material of the courses. In addition, it should be of scientific value and useful to the host company. It describes a problem of the company Sogeti in the department of Cloud, Development & Data. I am grateful that I was allowed to do an internship at Sogeti and for their cooperation.

Furthermore, I would like to thank all my colleagues at Sogeti for their interest and enthusiasm. This thesis would not have been possible without the support and feedback from my supervisors. Therefore, I would like to thank Mark Hoogendoorn from the VU and Wouter de Nie from Sogeti. I would also like to thank my second reader René Bekker for the time spent on the thesis. Finally, I would like to thank my family for the emotional support during the writing of my thesis.

Abstract

The thesis conducts research as a first step of the project at Sogeti. This project has the practical goal of classifying thoughts for the purpose of moving the bar in the Breakout game. The research involves examining different *motor imagery* (MI) tasks measured with *electroencephalography* (EEG), different preprocessors and classifiers to achieve the highest accuracy to maximize the gameplay.

Common preprocessing techniques that have been tested in the literature are included in this study as preprocessing methods. These include *Fast Fourier Transforms* (FFT), *Common Spatial Patterns* (CSP), *Discrete Wavelet Transforms* (DWT), and statistical numbers of the data. Among all the tested features, the combination of CSP$\rightarrow$ DWT$\rightarrow$ statistical features turns out to contain the most predictive values to classify the data. Subsequently, different classifiers are tested as well. These are the *Logistic Regression* (LR), *Support Vector Machines* (SVM), and the *Temporal Convolutional Network* (TCN). These classifiers are compared to a state of the art *Long Short-Term Memory* (LSTM) neural network based on the description of Wang et al. (2018). This last model resulted in the highest accuracy, with a prediction score of 54.60%. This is 11.37% higher than the SVM, which was the second-best model. This model is presumably better because the changes over time is a more important feature for the classification than the statistical features. For further improvement, future research involving combinatorial models and adaptive learning may lead to better results to control the breakout game even better with a BCI.

Contents

Acronyms

BCI Brain-computer interface.

CSP Common spatial patterns.

DWT Discrete wavelet transforms.

EEG electroencephalography.

FFT Fast Fourier Transforms.

FIR Finite impulse response.

k-NN K-nearest neighbor.

LDA Linear discriminant analysis.

LR Logistic regression.

LSTM Long short-term memory.

MI Motor imagery.

NN Neural network.

RNN Recurrent neural network.

SVM Support vector machine.

TCN Temporal convolutional networks.

1. Introduction

Communication between people and computers have evolved significantly in the past decades. Currently, computers are most often used with a keyboard and mouse or a touchscreen. This research is focused on brain-activated controls also known as a Brain Computer Interface (BCI) (Vidal, 1973). The company Sogeti is entering this field with the BCI research project. The research in this thesis is a part of the whole BCI project. To make brain-activated controls possible, this research uses Electroencephalography (EEG) as main technology. This chapter introduces the EEG technology, describes the objective of this research and the structure of the report.

Electroencephalography

The BCI relies on EEG. This technology measures the electrical activity of the brain. With an EEG measurement, multiple sensors are placed on the head to measures the activity of a local area of each sensor (Britton JW, 2016c). Currently, this technology is used in research for applications in the medical sector as a replacement, rehabilitation (Padfield et al., 2019a) or diagnostic devices (Husain et al., 2003) and applications in the gaming industry (Portillo-Lara et al., 2021). Because the accuracies for controlling applications are not comparable to devices as a computer mouse, it still is an area of development (Padfield et al., 2019a). The reason that the accuracies are lower and the challenging part of the use of this technology is that the data always has a low signal-to-noise ratio. This can be improved by using more sensitive sensor types (Farnsworth, 2019), choosing the right brain areas (Garcia-Moreno et al., 2020a), the right frequency for the application (Britton JW, 2016b), and using a good machine learning algorithm. These points are dependent and limited by the objective of this study.

Objective

Most literature found, either tests one model or reviews multiple other papers which are compared while they work with different datasets. The uniqueness in this research is that it executes the whole process, using the same data to test multiple algorithms. Therefore, the research will include the collecting of EEG data and the classification of the thoughts with different kinds of machine learning techniques.

The research aim is to make the first steps in a BCI as a proof of concept. The goal is to classify human thoughts in order to play a game with your mind. To do this, the thoughts measured with electroencephalography (EEG) are converted to move a bar in the game "Break-out". The problem in this project is that the best setup for EEG measurements is unknown for this particular goal, as well as the machine learning techniques that will result in the highest accuracies for the best gameplay experience. For the research setup, there are requirements that apply for the future gaming application. One is that the EEG headset has to be consumer priced technology, which limits the quality of the data. Also, the quick changes in the game requires a quick reaction of the game. This requires the data window of the EEG on which a game action is based, to be of a shorter duration compared to the experiments found in the literature. The things that can be explored are the preprocessing and classifier of the machine learning. Therefore the main research question is:
"*What machine learning techniques will result in the highest accuracies for classifying the EEG data collected in this research?*"

Structure

Subsequent chapters provide background, describe the literature, and the study itself with the models and results. In the next chapter, the background of an EEG is described to get a better understanding of what exactly is measured and an understanding of the value of this technology with its possible applications. In the succeeding chapter Literature, studies are described with the types of headsets, current practices of recording and machine learning techniques. Based on this information, the research question is specified. In the following four chapters the methods are described with the data acquisition, the machine learning algorithms and the experimental setup. This is followed by the results of this research. At last, the research questions about obtaining the highest accuracy are answered in the Discussion.

2. Background

The previous chapter introduced the project. This section will provide a global understanding of the EEG technology. There is described for what purposes EEG is used, how it works and the different kinds of headsets are reviewed.

2.1 Brain-Computer Interface

A graphical user interface (GUI) is a screen that shows what it does to establish interaction between a person and a machine. The person can control the computer with a computer mouse and keyboard. Another way is by directly reading the brain activity to create a BCI. This has a multitude of prospective applications as this gives an additional way of directing machines. The current prospects of practical applications in the medical sector are as a replacement, rehabilitation or diagnostic devices and applications in the gaming industry. This section gives the examples and basic information of the applications of EEG technology.

Conventional BCI's are mostly used for prostatic limbs. Normally, the nerves always activate the muscles of the limbs. This is problematic for people who are missing a limb or cannot activate the nerves. Currently, the electrical signal of the nerves in muscles are used to make prosthetics move. This is called myoelectric and it assumes that the nerves are largely functional, while the technology is expensive (Padfield et al., 2019a). However, EEG can also be used for people who have damaged nerves or are completely paralyzed. This benefit of EEG makes it usable for wheelchair control, which is examined in Craig and Nguyen (2007). The study reached an accuracy of 82% with 3 classes. It can be imagined that head movement control might be preferable for most people that are only mostly paralyzed to have a more accurate control over the wheelchair.

Another application is for rehabilitation. This technology can be used by letting the patient imagine movements and give feedback by showing the movement being made by a robot or by walking in a virtual world (Padfield et al., 2019a). In diagnostics, an EEG can be used to find the abnormalities of brain functions to come

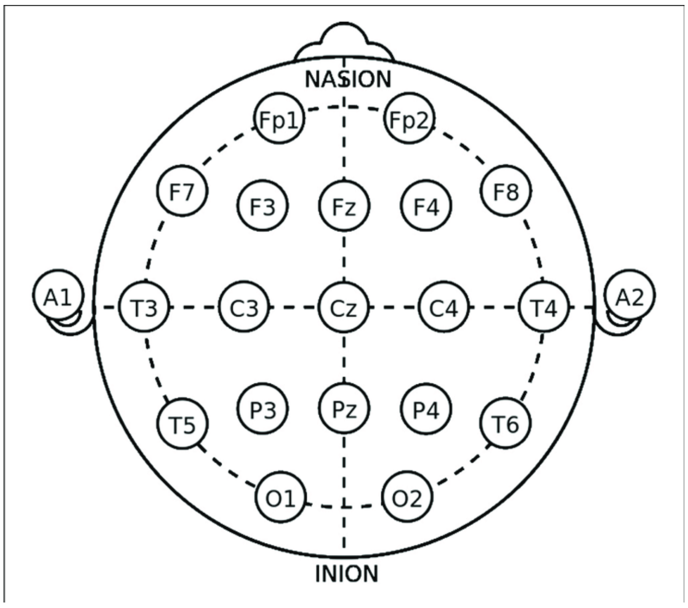
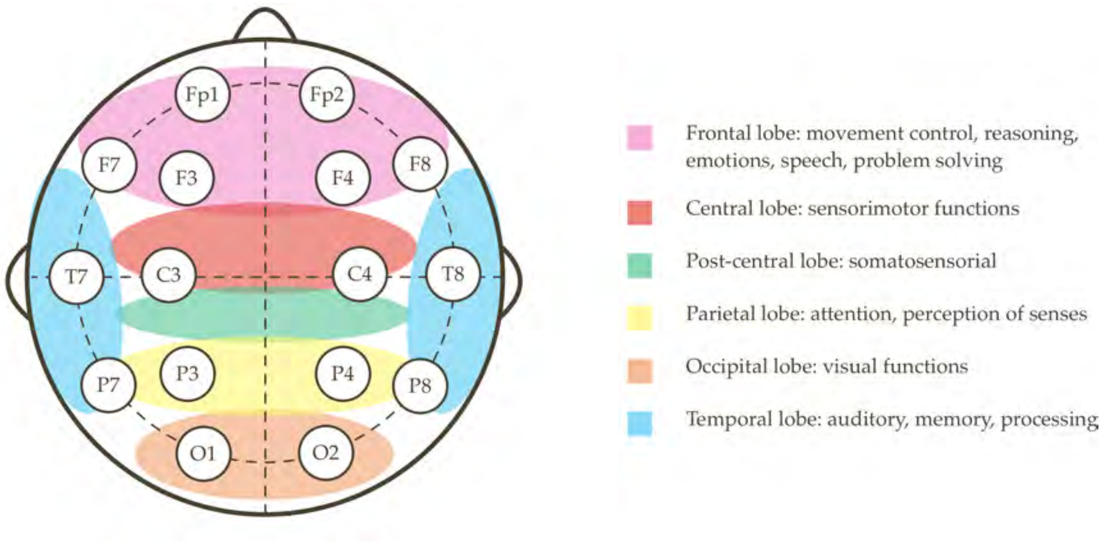


Figure 2.1: The international 10-20 sensor placement standard. Source: (Rojas et al., 2018)

to a medical diagnosis as is used in Husain et al. (2003) for epileptic patients. The EEG can also be used to detect emotions and that can be used for marketing or political agendas (Padfield et al., 2019a). The last application for this technology is the gaming industry as an addition or replacement of control. The current advances however report that it is "a poor replacement for traditional methods of controlling games" (Padfield et al., 2019a).

There are different methods to read brain signals to make a BCI. Electroencephalography (EEG) is typically used because it is cost-effective compared to functional magnetic resonance imaging (fMRI) and magnetoencephalography (MEG). EEG is a technique that measures the electrical activity of the brain. Due to the high sensitivity of an EEG, the signal can be different depending on external noise, and mood (Padfield et al., 2019a).

With an EEG measurement, multiple electrodes are placed on the head. For the correct placement there exists an international standard. A common EEG standard is the 10-20-system shown in Figure 2.1. The letters are used to refer to the names of the brain area. In this system Fp stands for fronto-polar, F for frontal, C for central, T for temporal, P for parietal, and O for occipital lobe (Alda & Torreblanca, 2020). In Figure 2.2 a the brain functions are shown to give an initial indication of the most important areas.



Fp1
Fp2
F7
F3
F4
F8
T7
C3
C4
T8
P7
P3
P4
P8
O1
O2
Frontal lobe: movement control, reasoning, emotions, speech, problem solving
Central lobe: sensorimotor functions
Post-central lobe: somatosensorial
Parietal lobe: attention, perception of senses
Occipital lobe: visual functions
Temporal lobe: auditory, memory, processing

Figure 2.2: In this figure, there is an illustrative top view of a head with electrode placement shown. The top is the front of the head and the dotted line is on the sides, while the rest of the sensors are placed on top of the head. Based on (Garcia-Moreno et al., 2020a)

2.2 Brain Activity

For further comprehension of the EEG technology, the electrical activity that is actually measured will be described. For this, it is needed to understand the concept of how the nerves work. The shortest explanation is that the brain consists of neurons that send an action potential to release a synaptic response to activate other neurons.

The complete understanding requires an explanation of the activation cycle of a neuron. Starting at rest, the neuron has an electrical charge of around -70mV. This is charged by a Sodium-potassium pump that pumps Sodium (Na^+) out of the neuron and potassium (K^+) inside, as shown in Figure 2.3. Then potassium can reenter the cell because of the concentration difference. The negative charge comes from proteins and chloride (Cl^-) inside the neuron. At the end of a neuron are synapses that can give an electrical signal towards the next neuron, as shown in Figure 2.4 with synapses E1, E2, E3 and I1. If the neuron gets a sufficiently strong signal and reaches -50mV, the potassium channels open. This is causing a rapid depolarization, continuing till a positive charge goes throughout the neuron and results in its own synaptic activation. Then it closes the Sodium channels, opens the potassium channels and restarts the Sodium-potassium pump to get back to the rest state. This is called the action potential (Kalat, 2004). The electric charge in this process changes as shown in the purple and again in green part of the graph of Figure 2.4.

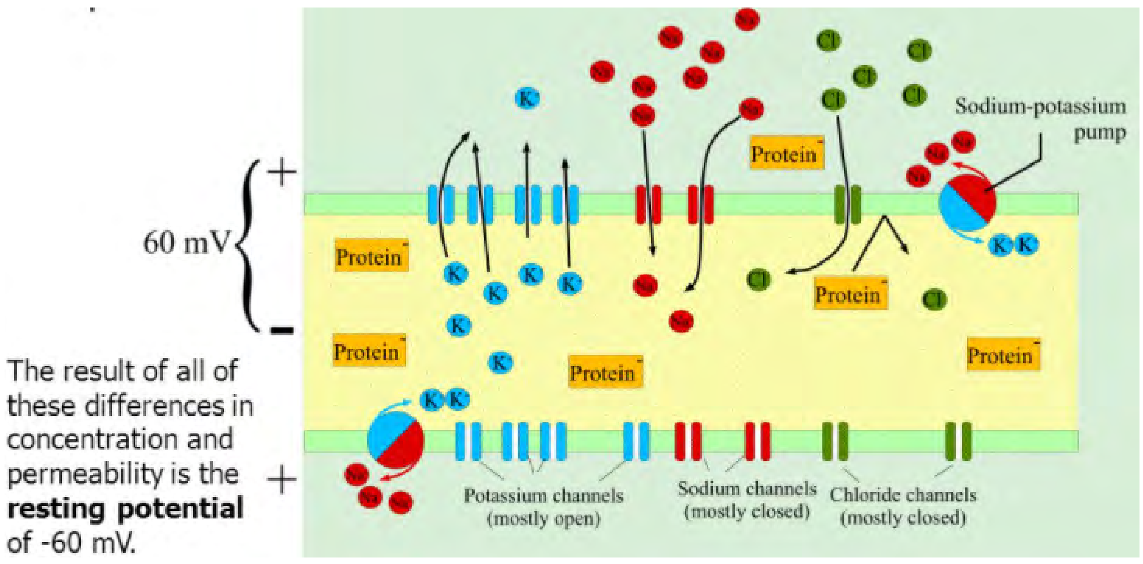

60 mV
+
−
The result of all of these differences in concentration and permeability is the **resting potential** of -60 mV.
Protein⁻
Sodium-potassium pump
Potassium channels (mostly open)
Sodium channels (mostly closed)
Chloride channels (mostly closed)
K
Na
Cl

Figure 2.3: The resting potential of a nerve. Source: (Mandira, 2016)

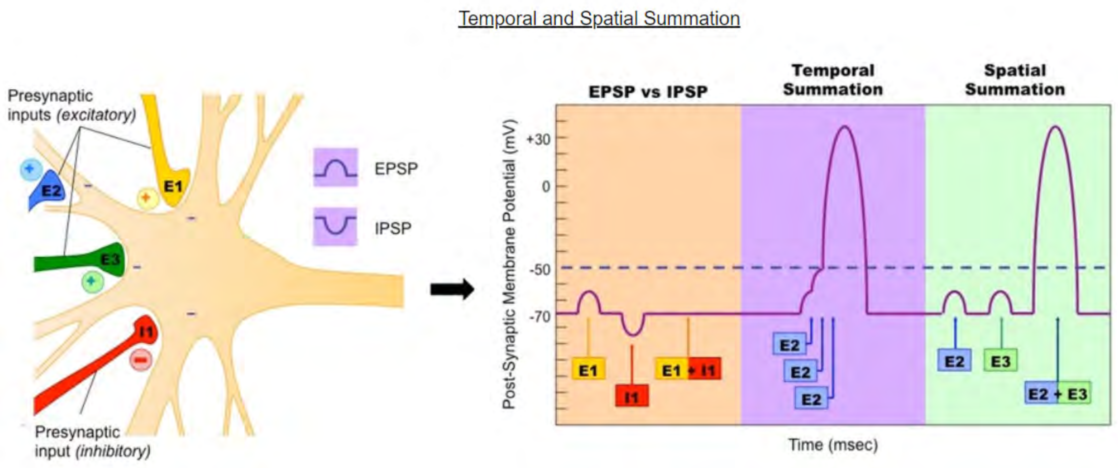

Temporal and Spatial Summation
Presynaptic inputs (excitatory)
E1
E2
E3
I1
Presynaptic input (inhibitory)
EPSP
IPSP
EPSP vs IPSP
Temporal Summation
Spatial Summation
Post-Synaptic Membrane Potential (mV)
+30
0
-50
-70
E1
I1
E1 + I1
E2
E2
E2
E2
E3
E2 + E3
Time (msec)

Figure 2.4: On the left the outside of a neuron with exhibitory and inhibitory synapses and on the right the electric potentials in the membrane of the exhibitory and inhibitory post synaptic potentials (EPSP & IPSP). Source: (BioNinja, 2016)

If the signal of the synapses is insufficient, we get a temporal charge that makes reaching the action potential easier or harder. This is called excitatory and inhibitory postsynaptic potentials (EPSPs and IPSPs) respectively, as shown in Figure 2.4. These are the electrical charges that we actually measure. The action potential happens too quickly to measure (Britton JW, 2016d). The recording cannot record a single neuronic charge, but the measurement requires at least a range of 6 cm^2 of synchronized cortical activity to produce a discharge that is measurable with an EEG sensor (Ramantani et al., 2016). What the EEG then measures are EPSPs and IPSPs of the neurons near the scalp (Britton JW, 2016a). This means that a sensor measures a multitude of neural activities. The outcome of the measurements is a summary of the activity of a local area of each sensor.

2.3 EEG Headsets

A benefit of the EEG compared to other BCI's is that the sensors are electrodes that can be tiny. This makes it possible to have EEG headsets that are portable and connected to wifi or Bluetooth. This increases the mobility of the wearer, increasing the possible applications. In addition to this, the electrodes can be different. They are mainly divided into wet or dry electrodes. The wet electrodes need electrogel and are considered the traditional method (Casson, 2019). Dry electrodes are more practical as it does not need electrogel to be applied (Padfield et al., 2019a). For electrogel there are multiple options available ranging from liquid to paste-like gels. The disadvantage of liquid gel is that it dries out quickly and is unusable for long sessions. The disadvantage of a more paste-like gel is that it is less user-friendly and takes more preparation time. The preparation time can take longer than 30 minutes as in Craig and Nguyen (2007), where they do not even take the cleanup time into account. Wet electrodes however are less prone to noise, as liquid penetrates well through hair and reaches the scalp easier (Shad et al., 2020) & (Padfield et al., 2019a). For conduction, the wet electrode itself is a solid metal electrode, where the dry electrode uses metal pins that can be uncomfortable depending on the design (Chi et al., 2010).

There are many EEG headsets in different price ranges. Of the different headsets, there is a variety in the amount of sensors and in the recording quality (Farnsworth, 2019). The choice for appropriate EEG headsets is a trade-off between the quality of the resulting data of the device and the price, which should be as low as possible to make it more affordable for consumers.

3. Literature

This section will review literature about EEG data. This will determine the scope of the research setup and machine learning techniques with the techniques that are shown to be promising in other research. After the experimental setting for the recording, the preprocessing and the classifiers are discussed.

3.1 Experimental setting

In the many pieces of research that are done with EEG headsets, all have a slightly different setting. There are a few groups in which these can be divided depending on the main differences. The main differences are the recording time, the mental task and executed movement of the test subject. Since the brainwaves react differently for each person while executing similar tasks, individual models have a higher accuracy compared to global models that use different test subjects for training and testing the prediction model. This is visible in multiple papers for example in Craig and Nguyen (2007), where the accuracy for an individual person was 82% and went down to 52.5% in a global model. This also suggests that each individual has different brain activity while having the same task and similar thoughts.

When we have a subject, there are different tasks that can be done and observed. It is commonly separated in control and monitor applications (Nijholt, 2016). The monitor applications can try to differentiate emotional state or mental state such as concentration. The test setup can be divided into four categories that are commonly used. These categories are: Visually Evoked Potentials, Slow Cortical Potentials, Sensori-Motor rhythms, and the Event-Related Potentials. The first occurs when the mind is activated by showing certain images to get brain activity known as Visually Evoked Potentials. The test setup can also be to continuously monitor the test subject. For example during sleep, to get Slow Cortical Potentials (Sreeja et al., 2017). This research is in the category of control applications. This is an application to control an environment, such as controlling a computer mouse without hands. The first test setup possibility to do this is by building on Sensori-Motor rhythms (Sreeja et al., 2017). These are the recorded waves measured from the sensorimotor cortex of the brain (Kübler & Mattia, 2016) that responsible for movements. To activate that part of the brain, motor imagery (MI) tasks are often executed by the test subject, where the subject imagines the movements instead of making real movements (Amarasinghe et al., 2016). Here MI movement of the legs are often taken as one class as it is hard to distinguish between left or right moving legs. Similarly, the movement of fingers cannot be seen. It is however possible to distinguish the movement between the right and left arm as is done in Amarasinghe et al. (2016) and take that as the recognition signal for the reaction of an application (Padfield et al., 2019a). The fourth and last research setup is to get Event-Related Potentials that are caused by other stimuli. In this category, P300 tasks are commonly given. These are mental tasks such as making calculations or imagining objects turning around (Amarasinghe et al., 2016). For all control applications research "it is recommended to instruct familiar mental tasks" as it leads to better results (Roc et al., 2021).

The last big difference in the research setup is the duration of each recording of each task. The shortest task recording found came from Padfield et al. (2019a). They state that each task took four seconds and they made different sized windows. It appeared that the windows from the first two seconds were better predictable compared to the windows containing data from the last two seconds. They point to concentration loss as the most likely cause. Most papers take longer task recordings and more data. Amarasinghe et al. (2016) took 20 seconds for each task and made 50700 records and Craig and Nguyen (2007) let the subject do each task 10 times 10 seconds long, with 10 seconds of rest in between for concentration restoration. This is also the forementioned paper that took more than 30 minutes to prepare the headset. One particular paper of Kostov and Polak (2000) had no time restriction and used an online research setting in which the test subject had to move the cursor on the screen towards a goal. Due to the extensive effort, costs of the headset, time to make their own data and assuring the quality of the data, multiple papers did not make their own recordings and referred to online available datasets (Sreeja et al., 2017) & (Lu et al., 2019).

There are a few takeaways from the research setting. These are as follows:

- An individual model is more accurate than a global model.
- For this research with a control application, the test subject can use P300 and MI movements to produce the right brain waves for the EEG, which can be classified. With MI movements there is proven that movements with legs, left and right arm, and tongue can be predicted to a certain degree.

- Most data windows used in research often lasts for several seconds. This can be taken into account since a game requires a relatively fast reaction to control the game and therefore long recordings are not possible. This can have an effect on the results.

3.2 Preprocessing

After the recording, the first transformations on the data can be performed. A big part of machine learning is the preprocessing where the data is prepared. Data preparation includes the selection of timeframes that contain the tasks and sorting the time frames that belong to each task class. Then the data is preprocessed by removing uninformative data, splitting the data, and transforming the data into features. Each of these last named points can be treated differently. The research describing the preprocessing strategies are reviewed in this section.

The removal of the recorded time in the tasks is explained in Craig and Nguyen (2007) that recorded each task 10 seconds. They removed the first and last two seconds. The first two seconds were to take the reaction time into account and the last two were removed because it was expected that the participant had already reached his goal or the participant lost concentration. After this, the data of each 10 second task was divided into 60 data windows to generate more data. It is noteworthy that most found offline research did not remove or split the data. In other research the reaction time was determined with MI motions and visual stimuli. The reaction time of the upper limbs is between 375-450 milliseconds while reaction time of the lower limbs was 450-600 milliseconds (Sirico et al., 2020). This speed can be used as reference for the minimum amount of time that should be removed at the beginning of each task.

When the data of each task of each class is determined, it can be transformed into more meaningful features. The raw data is recorded in the time domain. This means that each next data point is the next point in time. The features after transformation are mostly over the time- or frequency domain and sometimes a combination of the two. In the frequency domain only a small part of the frequency band is useful. The useful part consists of the frequencies that are related to the researched activated parts of the brain. What is known about the frequencies is that the delta (1-3 Hz) and theta (4-7 Hz) are normal occurring waves that become more obvious when a subject shows symptoms of drowsiness. In adults that are in an alert and wakeful state, measurements of many bursts in amplitude of these frequencies can be a characteristic of partial dysfunction of the brain and might be reason for further medical research. The alpha waves (8-12 Hz) are related to the somatosensory cortex that is responsible for the senses. In this frequency in 20-40 % of the adults a mu rhythm occurs. This is an arch shaped burst that is a reaction to movement, the thought of movement or the sense organs. The beta related frequencies (13-24 Hz) are related to the sensorimotor cortex that is responsible for movements. Frequencies bigger than 25 Hz are gamma related frequencies that are not commonly measured outside the scalp but can be seen with sensors that are planted behind the bone of the skull (Britton JW, 2016b). The frequency bands that are actually used for MI tasks in different papers are 7-30 Hz (Sreeja et al., 2017), 7-40 Hz (Lu et al., 2019) and comparable frequency ranges. The frequency filtering is done in the frequency domain. To get the data on frequency domain techniques as Fourier transforms, Wavelet Transforms and Finite Impulse Response (FIR) can be used. Here, Fourier transforms changes the raw data into frequencies that occur over the whole data, while Wavelet Transforms and Finite impulse response use a wave of a certain frequency and shifts that over the data to measure the overlap of the data points and de wave, to measure the occurrence of that frequency at each time point in the data. For the combination of the time-frequency domain the Short Fourier Transforms is also often used. This uses multiple small parts of raw data in the time domain and use Fourier Transforms on those parts. On the time domain the raw data, autoregressive methods and Common Spatial Patterns (CSP) are often used. These try to find a certain pattern in the data. The used techniques in this research will be explained in detail in section 5 Preprocessing.

In summary, the following points of preprocessing are important:

- Of the recorded task, the first two seconds are seemingly the most important. However, at the beginning there is about 500 milliseconds of unimportant reaction time, which does not provide useful data.
- Each frequency band is related to a different brain function. The most important is the MI measurable frequency band between 7-40 Hz.
- The following preprocessing methods are used in related research:
  (Short) Fourier Transforms, Wavelet Transforms, FIR, Autoregressive methods and CSP.

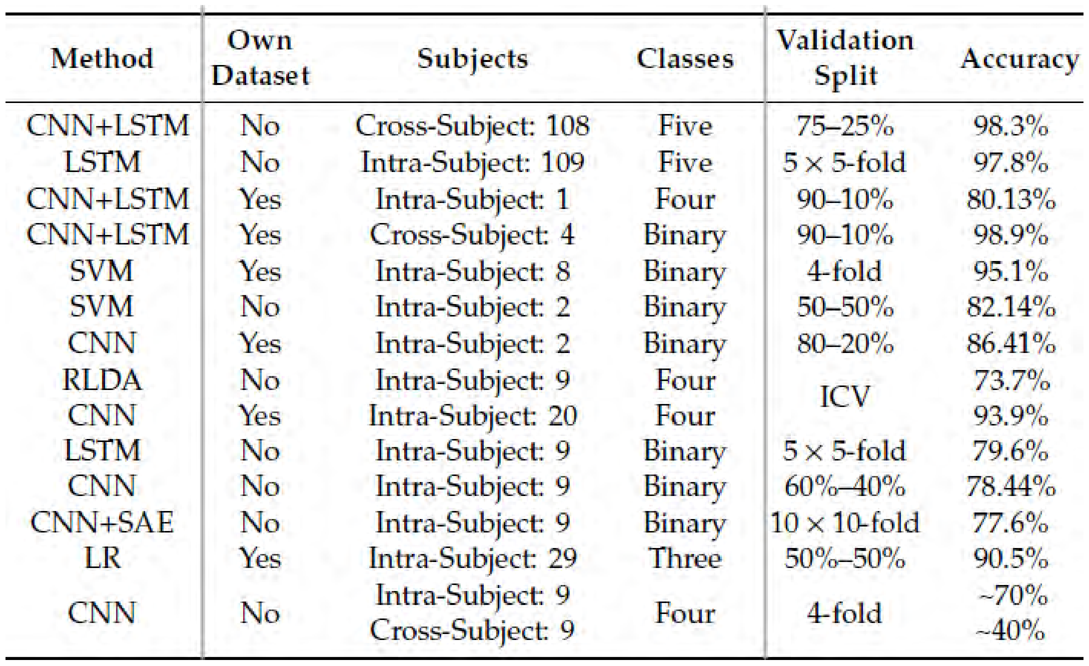
3.3 Classification

The classification is done with machine learning algorithms such that the label of unlabeled data can be predicted on a trained model. The results of machine learning are highly dependent on the combination with the input. The input is dependent on the quality of the recordings and the preprocessing as previously discussed. Below the most important algorithms and results of other research is shown.

Methods that are often used in literature are: support-vector machines (SVM), logistic regression (LR), k-nearest neighbor (k-NN), linear discriminant analysis (LDA) and neural networks (NN) (Padfield et al., 2019a). Of the NN variants, there are two modern variations that seem to work well with time dependent data. That is the long short-term memory (LSTM) as seen in table 3.1 and a temporal convolutional network (TCN) with an accuracy of 97% in the research of Lu et al. (2019). To get a complete overview, some studies of MI data are shown in Table 3.1, that comes from Garcia-Moreno et al. (2020b). In the table, the LSTM, SVM and the LR reach accuracies above 90%. The best scoring classifier from this, is the LSTM, with results that have multiple classes and still reaching accuracies above 90%.
There are a few takeaways from the classification:

- Commonly used classifiers are: SVM, LR, k-NN, LDA and NN variants.
- The highest accuracies in other research are reached with the LSTM and the TCN.
- The non-NN that have the highest accuracies are the SVM and LR.

Based on these accuracies and common use in other papers, the LSTM, TCN, SVM and LR are chosen to use for this research, which will be explained in depth in section Methods.



Method	Own Dataset	Subjects	Classes	Validation Split	Accuracy
CNN+LSTM	No	Cross-Subject: 108	Five	75–25%	98.3%
LSTM	No	Intra-Subject: 109	Five	5 × 5-fold	97.8%
CNN+LSTM	Yes	Intra-Subject: 1	Four	90–10%	80.13%
CNN+LSTM	Yes	Cross-Subject: 4	Binary	90–10%	98.9%
SVM	Yes	Intra-Subject: 8	Binary	4-fold	95.1%
SVM	No	Intra-Subject: 2	Binary	50–50%	82.14%
CNN	Yes	Intra-Subject: 2	Binary	80–20%	86.41%
RLDA	No	Intra-Subject: 9	Four	ICV	73.7%
CNN	Yes	Intra-Subject: 20	Four		93.9%
LSTM	No	Intra-Subject: 9	Binary	5 × 5-fold	79.6%
CNN	No	Intra-Subject: 9	Binary	60%–40%	78.44%
CNN+SAE	No	Intra-Subject: 9	Binary	10 × 10-fold	77.6%
LR	Yes	Intra-Subject: 29	Three	50%–50%	90.5%
CNN	No	Intra-Subject: 9 Cross-Subject: 9	Four	4-fold	~70% ~40%

Table 3.1: Table with machine learning algorithms that are used in other papers. Table comes from Garcia-Moreno et al. (2020b).

4. Data Acquisition

This chapter describes how the data acquisition is carried out. It includes a description about the subjects, environmental settings, the used interface, experimental setting, a description of the data and an initial data analysis.

4.1 Subjects

Part of the project is to collect the data. This ensures that the right consumer priced headset is used to create data and includes the measurement setup in the research. According to the literature in the previous chapter, individual models tend to lead to better accuracies (Craig & Nguyen, 2007) and the experiments in practice must be as close as possible to the real goal of the end task, which is playing the game breakout that is shown in Figure 4.1. For the individual models a lot of data is required. On top of that, multiple recordings should be made where the test subject executes different tasks. In order to ascertain that the tasks are executed in the right way, data was recorded by two test subjects that are familiar with the research. The subjects are 25 and 26 years of age.

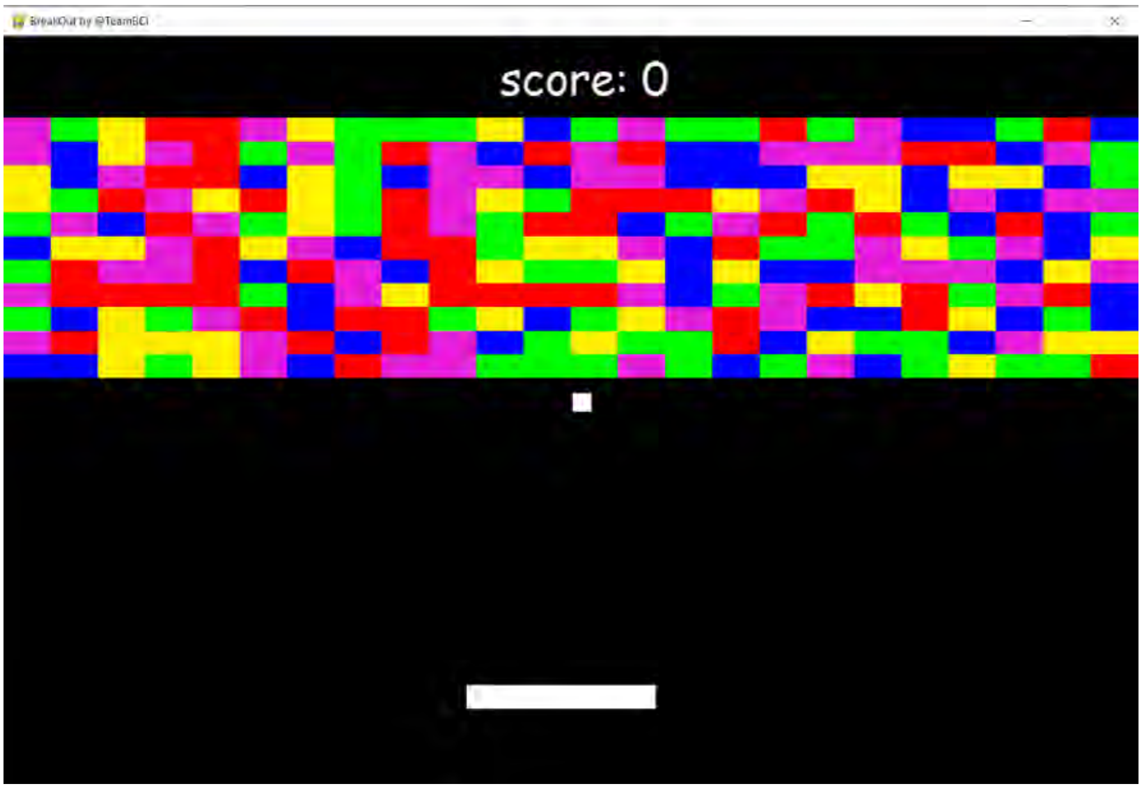


Figure 4.1: This figure illustrates the game Breakout. The player can only determine the movement of the bar and aims to break the colored bricks.

4.2 Environmental Setting

During the recordings it is important to keep the environmental factors constant. This is because changes in the environment might influence the recorded data or the calculations that are executed by the processing computer. For example, a faster computer might play the frames a bit slower or faster compared to the speed of the EEG recording signal. This might result in different data. Also, sounds that are coming from outside of the recording area or other distractions can influence the participant thought pattern, resulting in associated differences in the recorded EEG data. To keep data as consistent as possible, we used the same physical table in the same room during all our recording sessions. It is assumed that the interference signals are always the same when measuring at the same location. This would mean that by conducting all the recordings at the same place, the same disturbances would occur in each recording session. Hence, it would not influence the data differently in each recording.

The location is a lab environment at Sogeti Vianen in The Netherlands. The circumstances of the lab were as follows: There are no noticeable sounds, only two

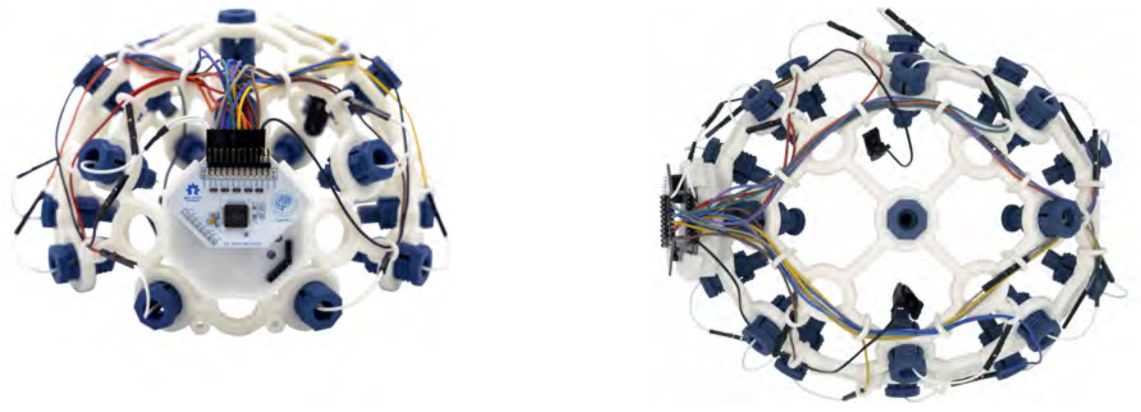

(a) Rear View (b) Top View

Figure 4.2: The Ultracortex Mark IV headset that is used for the EEG recordings. Source: (“Open BCI - All-in-One Biosensing R&D Bundle”, n.d.)

people were at the lab at the same time, where the test subject could not see the second person. The laptop had a Windows operating system, on which the code was played in Visual Studio Code. The hardware of the laptop consisted of an intel i5 processor and 16 GB of RAM, which both determine the computation speed. The EEG data is recorded with the Ultracortex Mark IV from OpenBCI as seen in Figure 4.2, with an additional Cyton and Daisy Biosensing Board (two silicon chips). These chips are needed for extra sensors and to get a higher sampling frequency that sends the signal over Wifi to the computer. This headset has 16 dry EEG sensors, with a sampling frequency with each sensor of 1000 Hz (“Open BCI - All-in-One Biosensing R&D Bundle”, n.d.). The headset is in the cheaper price range as shown in Table 4.1, that shows a short overview of EEG headsets. Therefore, the recordings are expected to have a lower signal-to-noise ratio compared to most literature research. Lastly, there is an interface which provides signals that tell the subject what action needs to be taken.

Company	Price range
NeuroSky Muse Emotiv **OpenBCI**	$99-$1000
Wearable Sensing ANT Neuro - eego rt Neuroelectrics G.tec Nautilus (PRO) wireless ABM B-Alert BioSemi Cognionics mBrainTrain Brain Products LiveAmp	$1000-$25000
ANT Neuro - eego rt 64 channels ANT Neuro -eego mylab Brain Product ActiCHamp BioSemi	$>$ $25000

Table 4.1: EEG headset prices from IMotions Farnsworth (2019)

4.3 Interface

The interface should have three different stimuli videos to indicate which task should be executed by the subject. This number of tasks represents the three moves that are used to play the breakout game. These three moves are the following: moving the bar to the left, moving the bar to the right, and keeping the bar in place. The corresponding tasks will be called the left, right and neither tasks in the rest of this report. The intention of the interface is to look similar to the real Breakout game. The real game itself would be unusable because it is often unclear in which direction the bar should move to catch the ball as it often changes direction. In theory, the real game will make it hard to let the participant make the right choices at every moment. This makes it difficult to determine the desired action as the game is interactive. The required action of the left, right and neither task will also be referred to as the y-label. To simplify the interface of the recordings, a video that is based on the game is used instead of the Breakout game itself. The interface clearly shows which task the participant should execute. There are three different videos used as stimuli for the different tasks where a ball falls vertically from the top left, right, or middle of the screen. It should be clear that the test subject needs to take the matching action to move the bar to the left, right or neither side. The ball falls down in 4 seconds in the original video and the bar is moving in the horizontal direction towards the ball. There are multiple interface videos that show the tasks in a different order. Every recording session uses one interface video that shows 36 tasks that consists of 12 stimuli videos of each task. There are different interface videos to decrease possible anticipations of the test subject. This is avoided because this might influence the data. In the initial trails it appeared that a direct transition of tasks is very stressing. Therefore, a one-second break between each stimuli video is played to restore concentration and to give the subject time for the transition to the next task. Figure 4.3 shows an example of a stimuli video. These videos should make it clear which task should be executed and serve as an interface.

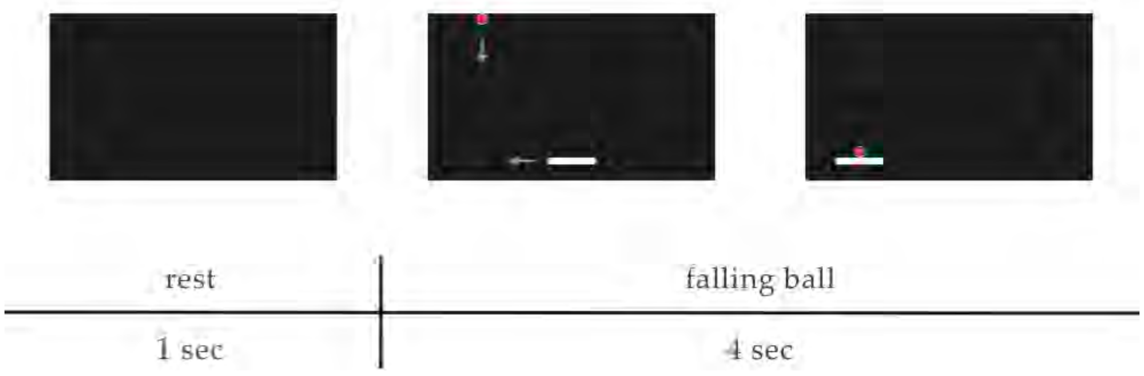


Figure 4.3: This figure illustrates the interface with the 'Left' task.

4.4 Experiments

To start the experiment a subject has to disconnect all wires of the sensors of the headset, before putting on the headset. The sensors have to be tightened by turning them until all sensors touch the head with some pressure. The pressure comes from the springs in the sensors. Then the wires must be connected again. The WiFi shield of the headset must then connect with the laptop. As a last step, the Python notebook that plays the video and starts the recording can run after filling in the right variables. The variables that come back in the name of the document with the data are the following: the code that stands for a specific test-subject that executes the experiment, one of the 3 videos of which each has a different order of the tasks, and how the tasks are executed. These variables are important to keep track of. The code of the subject is important because the aim is to make individual models as everyone's brain waves are different (Craig & Nguyen, 2007) as is discussed in the literature chapter. The video version is registered because it refers to the sequence of tasks in that version, so the right y-labels can be matched with the pieces of data that belong to those tasks.

The executed task is registered so they can be distinguished. The different kinds of tasks for each experiment are described in Table 4.2. In the experiments, the main executed tasks are the MI tasks that are proven to be distinguishable in Amarasinghe et al. (2016). The movement of lifting the left and right arms are used for the ”left” and ”right” tasks, and for the ”neither” tasks lifting the legs, moving the tong and not moving have been used. During these tasks the suggested movement is imagined by the subject. For thoughts to imagine this there is decided that the participant should depict the movement and think about the physical feeling of the activation of muscles. Both of these should be thought of simultaneously and repeatedly. With all registered recording variables and a specified research setting, the resulting datasets should be ready for the machine learning.

4.5 Data

During the recording all relevant data is stored on the computer. The data that is constant for each video is marked in the document name as mentioned in the previous paragraph. The relevant data that is made during the recording are the sensor measurements of the brain activity in microvolt and the placed markers for the tasks. The microvolt is the actual data that will be used to predict the executed tasks. The markers are placed in the data to keep track of when the data of each task starts and ends. This is used later to split the data in tasks in the preprocessing step. The markers are based on the video frames and exclude the frames during the resting moment that is between tasks. Simultaneously with the markers, the time points are tracked to determine the duration of each task. This is used in the data preparation to remove the reaction times in proportion to the duration. This is needed because the frames are not played at a constant rate. During the recording sessions, it appeared that some tasks had a deviation of a second in the duration, as the markers had a different amount of data points for some tasks. This means that the recording spreed was inconsistent. This problem could be in the speed of the frame rate or the EEG recording. To ensure the right amount of reaction time and time at the end is removed, there was an additional need to keep track of the time points at the beginning and end of each task. The resulting data was saved for the preprocessing.

Task name	**Meaning**
Thinking Motion 1	The test subject sits up straight and is making the MI of lifting the left or right hand, when the bar should move to the left or right respectively. Nothing is done for the neither-task.
Thinking Motion 2	In addition to “Thinking Motion”, the MI movement of lifting the knees should be done for the neither-task. The test subject assumes the posture of sitting slumped to imagine this movement.
Thinking Motion 3	In addition to “Thinking Motion”, the MI movement of pressing the tongue to the upper side of the mouth should be done for the neither-task.
Physical Motion 1/2/3	The test subject executes the physical motions instead of imagining the motions that are described in the corresponding Thinking Motions.

Table 4.2: The executed motions or tasks of the experiments.

4.6 Data Analysis

A data analysis was performed to get an initial understanding of the data. This was done with the raw data after it was divided into the three classes. After this division of the tasks, it was detected that the number of data points or length of each task that is measured with the time points, matched the distance of the markers in the data and the given sampling rate of 1000 Hz. From this can be deduced that the sampling rate is constant and that there are no missing values.

For the analysis, a plot of the data in the frequency domain can be seen in Figure 4.4. In this figure, there is mainly activity below 18 Hz and around 50 Hz. The activity around 50 Hz is possibly external noise coming from the electricity grid of the building as that also has a frequency of 50 Hz. This shows that there probably is a lot of external noise measured. Although lower frequencies are used in the data, there could also be interference at these bands.

In an additional analysis, one dataset of one person of measurements that are taken on the same day is examined. This is to examine the differences between the tasks as an initial indication of how well the classes can be distinguished with this data. A few statistics were determined for each sensor, of which three are shown in Figure 4.5. The statistics shown are the mean, standard deviation, minimum, and maximum. Only three sensors are shown as an example, as other sensors show similar similarities and differences. Most sensors only have small or no visible differences between the different tasks in the plots. An example of this is sensor P7 in the figure. However, larger differences do appear, as shown by the other sensors in the figure. For the final analysis, percentile plots of all sensors were made which also distinguished between tasks. The percentile plot of sensor T8 is demonstrated in Figure 4.6. It can be observed that there is a small difference in the classes. The percentile at which the largest difference occurs was noticeable differed from sensor to sensor. For outliers, it was observed that none of the percentile plots contained excessive high or low values. From this is derived that there are no outliers. By having visualized the differences in these statistics, the first insights were obtained. This is that the classes are mostly similar with small differences. The presence of the differences increases the likelihood of good classification. Therefore, this analysis indicates a positive expectation for the results of this study.

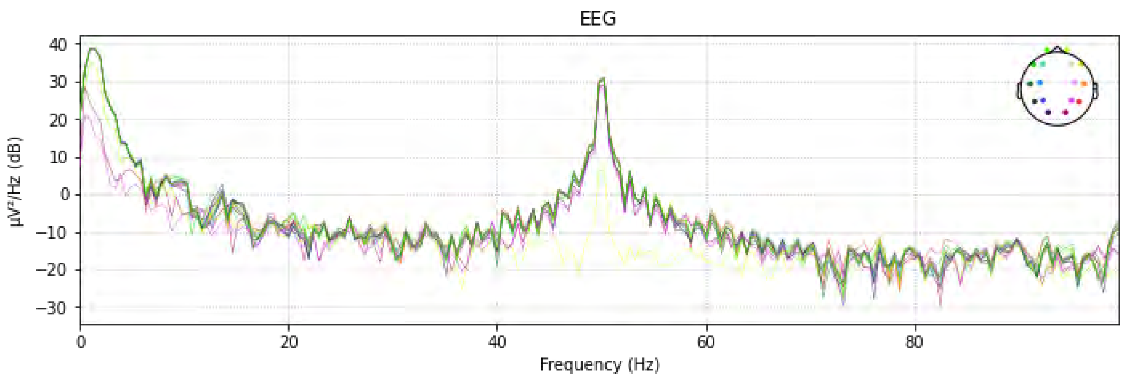


Figure 4.4: This figure shows the data in the frequency domain of a single task.

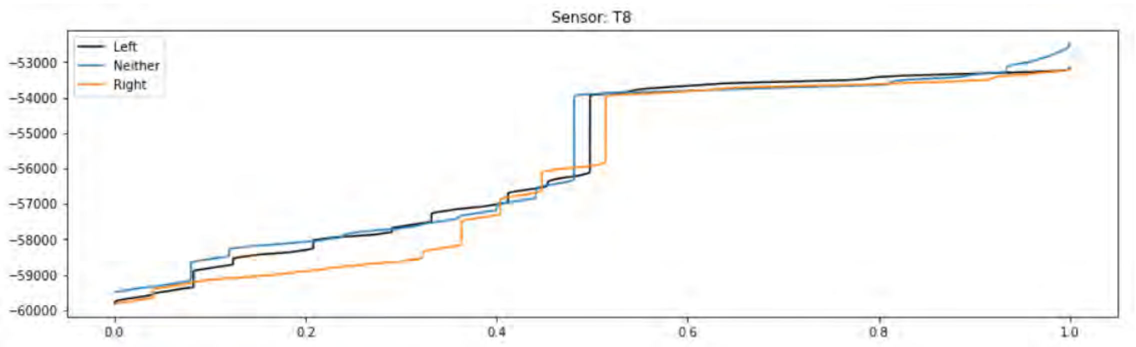


Figure 4.6: This figure shows a percentile plot of the data from sensor T8.

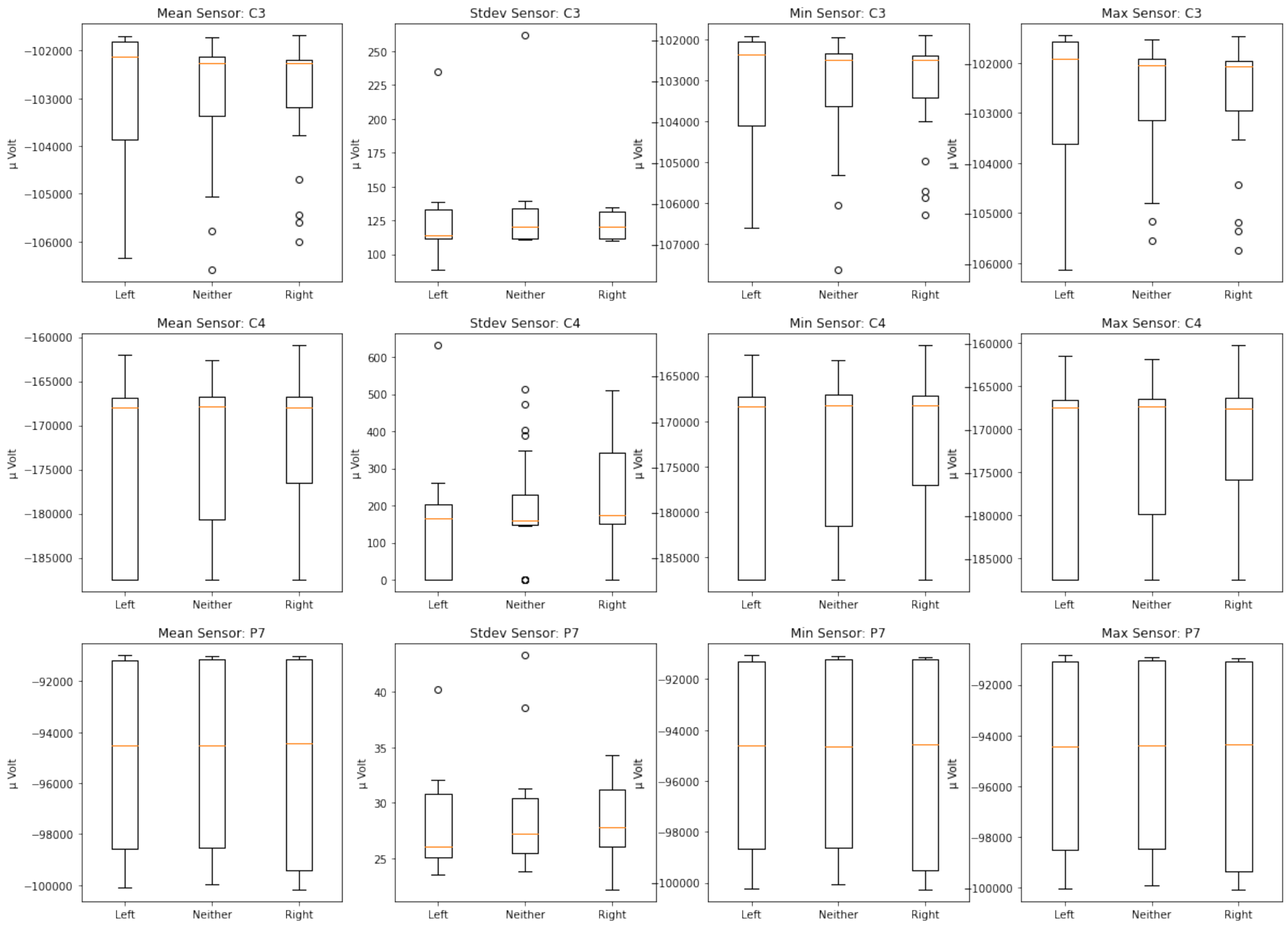


Figure 4.5: This figure shows box plots of the statistics, where each data point is a statistical value of a window. The classes 'Left', 'Neither' and 'Right' are serparated.

5. Preprocessing

After the data is collected, it has to be preprocessed. The method of preprocessing in this chapter is split into “creating usable raw data” and the preprocessing methods that are based on the reviewed literature. These techniques are “Fast Fourier Transforms” (FFT), “Discrete Wavelet Transforms” (DWT) and “Common Spatial Patterns” (CSP). The first part is about how all data is loaded and prepared before use, and the latter methods are follow-up steps to transform the data in a form that contains more meaningful information than the raw data. These preprocessing methods are used in different combinations with the classifying machine learning algorithms. However, the first step is creating usable raw data and will be discussed next.

5.1 Creating Usable Raw Data

The data that is loaded is specified in task, person and day, so multiple recordings are combined. When the recordings are loaded, they are split based on the markers in the data that indicate the beginning or end of a task and the tasks are given the label of the action that is executed. The 800 milliseconds in the beginning of each task are removed, in order to get rid of the reaction time and the time that is needed for the ball to become visible on the screen as in the interface video the ball starts outside the screen. Of each task, 200 milliseconds of the end is removed to get rid of the last few milliseconds where the ball is caught on the bar and already reached its target. To decrease the total amount of data only eight sensors are used. The specific sensors that will be used is determined by the first part of the research. To create balanced datasets, all tasks are separated in the group with the corresponding labels of: “Left”, “Right” and “Neither”. Then the three task groups with different labels are divided into a 60%/20%/20% split, for the training, validation and test set respectively. The tasks are not randomized in the same group so that it conserves the order in which it was recorded, such that the first 60% of the recorded tasks are in training set, and the last 20% of the recorded tasks in the test set. Then the groups with different labels that are meant for the same set are combined, so each set has an equal amount of tasks in each set. The datasets are separated before the rest of the preprocessing, so the datasets can get the same transformations as the first fitted training set. For example, the standard scalar uses the mean of the training set on all three datasets. In the last preparation step, the tasks are split in overlapping windows of 512 milliseconds. Each adjacent window has 50 uncommon and 462 overlapping data points for each sensor. The separation of tasks before the window split prevents that windows of the same task end up in different datasets. At this point, all relevant data is selected and the standard transformations that are relevant for all preprocessing techniques can be made.

5.2 Standard Preprocessing Transformations

The transformations that are used are a FIR filter and a standard scalar. The FIR filter is a band pass that lowers all values outside a given frequency range. The band pass is a combination of a low pass filter that passes all low frequencies and filters the high frequencies and a high pass that does the opposite of a low pass. As stated in the literature chapter, the useful range of EEG data is around 8-40 Hz (Lu et al., 2019). Therefore, this is the chosen range for the FIR filter. The other frequencies are considered as noise, which can make the prediction less accurate. An example of the result after the use of the filter is shown in Figure 5.1. The basic idea behind the filter is that the data is transformed into the frequency domain, the desired frequencies are kept, and the data is transformed back into the time domain. This is done in one transformation with the FIR filter. An example of how the domain can be changed to the frequency domain is done with FFT that is described in the next paragraph. The FIR filter works by calculating overlapping filtered data windows $h(n)$ with the n^{th} data point from a certain window with length M with Formula 5.1.

$$h(n) = \frac{sin(w_c(n - \frac{M-1}{2}))}{\pi(n - \frac{M-1}{2})} \tag{5.1}$$

Each $h(n)$ uses the Hamming window which is defined with the Hamming Formula 5.2 (Mynuddin, 2015).

$$w_c(n) = 0.54 - 0.46\, cos(\frac{2n\pi}{M-1}) \tag{5.2}$$

This gives the desired outcome for one window in the FIR filter. The new output data is made by using this with overlapping windows of size M of the original data.

The second transformation is a standard scalar as in Formula 5.3.

$$z_n = \frac{x_n - \mu}{s} \tag{5.3}$$

This scalar subtracts the mean μ of the n^{th} data point x_n and divides by the standard deviation s. This is done with all data points with the mean and standard deviation that belong to the training set for each individual sensor. Scaling the data is commonly recognized to improve almost all machine learning models as it gives the data of all the sensors the same scale (Géron, 2017). Because the model is trained on the training data, the mean and standard deviation of the standard scalar are determined by the training data. The measured values are reused for the standard scalar on the validation and test data. The result is the same transformation over the datasets. These transformations are used in all models. For the preprocessing to get the features only one or a combination of either FFT, CSP or DWT are used.

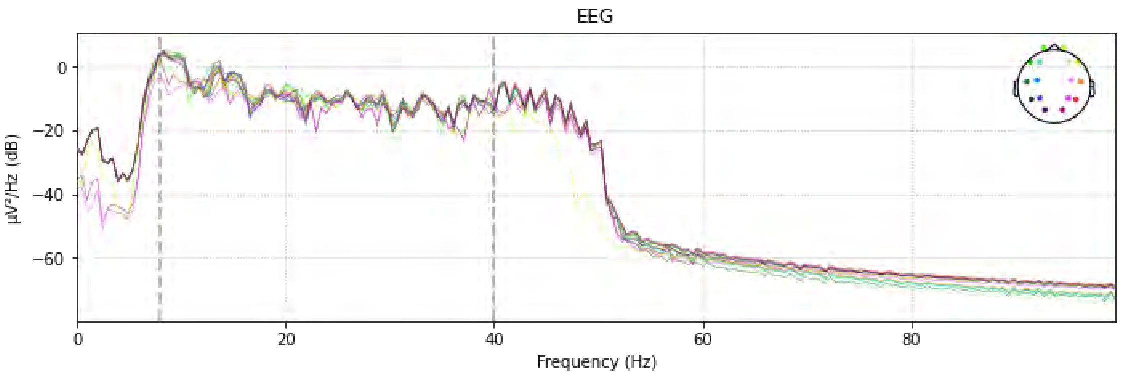


Figure 5.1: This figure shows the result after the FIR filter in the frequency domain of the same data as plotted in Figure 4.4

5.3 Fast Fourier Transforms

The recorded data contains a multitude of combined frequency waves, as is illustrated on the left side of Figure 5.2. The FFT dissects these frequencies into the individual frequency waves and measures of each frequency how much it occurs in the data in proportion to other frequencies. That is how the FFT transforms the data that is recorded from the time domain into the frequency domain. The disadvantage of this technique is that it loses all information on the time domain and it loses short occurrences of individual frequencies. The basis of FFT is a Discrete Fourier Transform (DFT) that is displayed in Formula 5.4.

$$X_k = \sum_{n=0}^{N-1} x_n * e^{-\frac{j2kn\pi}{N}} \tag{5.4}$$

Here the resulting value X_k is the amplitude of the proportional occurrence of the k^{th} frequency that is measured over N data points. The k^{th} frequency is an integer that is not equal to the measured frequency in Hz. The main measurement of the frequency is the sinus and cosine of the radial frequency that is rewritten as a power of e as shown in Formula 5.5 (Burros et al., 2012).

$$e^{-\frac{j2nk\pi}{N}} = cos(\frac{2nk\pi}{N}) + j\ sin(\frac{2nk\pi}{N}) \tag{5.5}$$

The frequency in Hz is related to the k^{th} frequency in the formula. How it is related dependents on the sample rate T and the given amount of data points N. In this research the $T = 0.001$ as the headset that is used takes 1000 samples each second. The frequency f in Hz in terms of input variable k can be determined with Formula 5.6, that is derived from Erickson (2021).

$$f = \frac{k}{NT} \tag{5.6}$$

This means that if the input window contains 500 data points $N = 500$ then the first amplitude X_k with $k = 1$ is the amplitude of frequency $f = k/NT = 1/(500 * 0.001) = 2\ Hz$. Since not all output frequencies are of importance, only the relevant frequency terms are used, which are around 8-40Hz. The calculations are decreased using the fast Fourier transforms. The FFT is derived by making the calculations of the DFT more efficient with the use of the symmetry and matrix calculations as demonstrated in Brigham (1988). The output of the FFT are the values with the frequencies from each sensor. These are used as input for the machine learning models.

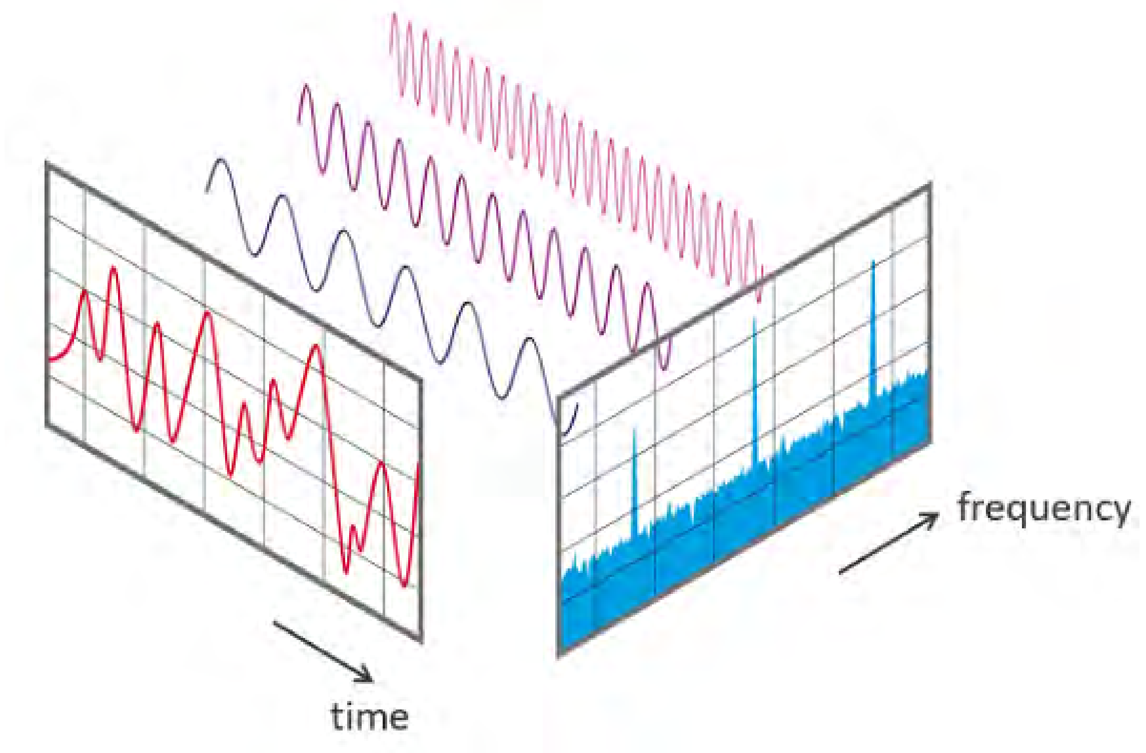


Figure 5.2: The concept of the FFT is displayed. The input in the time domain is transformed into the frequency domain. Source: (“Fast Fourier Transformation FFT - Basics”, n.d.)

5.4 Discrete Wavelet Transforms

The DWT is a different technique that uses the frequency domain in combination with the time domain. It tries to split the data into multiple frequency bands. It does this in the time domain, so the result is data in the time domain of a certain frequency band. The different frequency ranges, called levels, are maintaining short occurring waves that would be lost in the FFT (Taspinar, 2018). The DWT splits the data into multiple frequency bands by splitting the data with a low pass (LP) and a high pass (HP) filter in each level, as shown in Figure 5.3. For example, data with a sample frequency of 1000 Hz contains information in the frequency range 0-1000 Hz. The first level of the data after the LP is half the amount of data points and can contain information of 0-500 Hz. The HP in the first level contains the information of 500-1000 Hz (Taspinar, 2018).

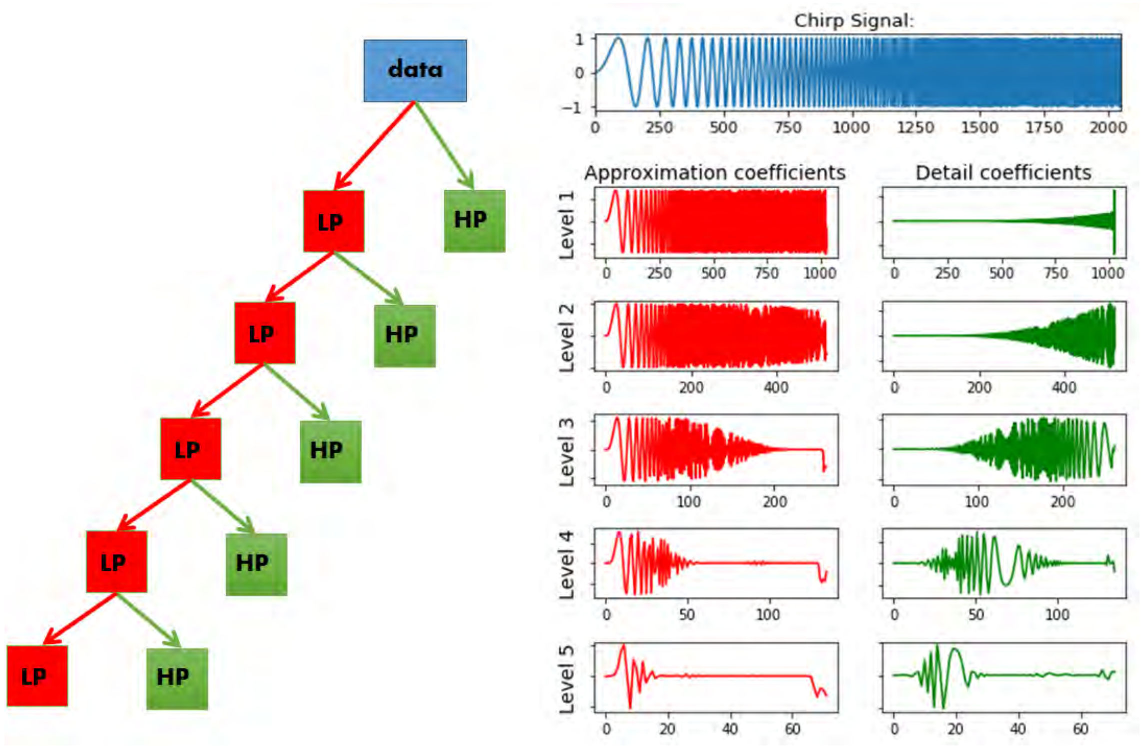


Figure 5.3: The concept of the different levels in wavelet transforms is displayed. Source: (Taspinar, 2018)

This is calculated with the discrete wavelet coefficient $T_{m,l}$ that is shown in Formula 5.7 with scale m and location l.

$$T_{m,l} = \int_{-\infty}^{\infty} x_n \psi_{m,l}(n)\, dt \tag{5.7}$$

Here $\psi_{m,l}(n)$ is the function that stands for the wavelet that is used (Addison, 2005). The wavelet is a standard wave that is used to analyze the data and can be seen as the filter. For each level the wave is scaled. In the scaling the wavelet is stretched to measure another frequency range. Each wavelet decomposes the signal in two subsignals with half the length of the original. A simple example is the Haar wavelet. This wavelet decomposes the signal into the trend or detail coefficients as HP, with Formula 5.8 with input signal f.

$$a_m = \frac{f2m - 1 + f2m}{\sqrt{2}} \tag{5.8}$$

Simultaneously, it obtains approximation coefficients by measuring the difference as LP in Formula 5.9, with the m^{th} data point from the signal.

$$b_m = \frac{f2m - 1 - f2m}{\sqrt{2}} \tag{5.9}$$

Using this one time on the original signal gets the result of level 1 (Walker, 2019).

Half of the data points remain in the trend which also halves the measured frequencies.

The trend has a frequency range that starts at 0 and has an upper bound frequency f as in Formula 5.10.

$$f = \frac{\text{sample rate}}{2^{\text{level}}} \tag{5.10}$$

The frequency range of the approximation coefficients are between the upper frequency of the trend according to the formula and the upper frequency of the input. In this research only the approximation coefficients of level 6 and 7 are used with the ranges of 15,63-31,25 Hz and 7,81-15,63 Hz because those are the earlier discussed important frequency ranges (Lu et al., 2019). There are other wavelets such as the Gaussian and the Mexican hat wavelet as shown in Figure 5.4. These waves measure different forms instead of taking the mean and difference as in the Haar wavelet (Addison, 2005). It depends on the data which wavelet is more informative (Taspinar, 2018).

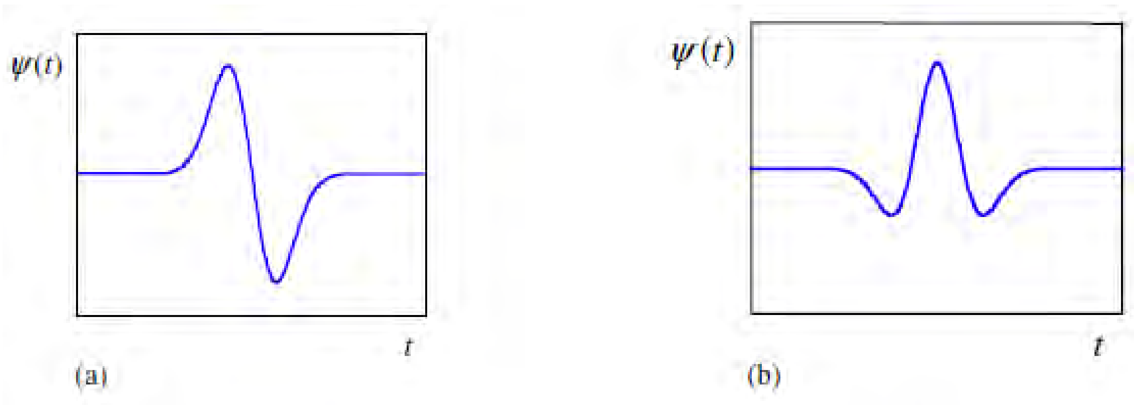


Figure 5.4: In this figure the Gaussian wavelet (a) and the Mexican Hat wavelet (b) are shown. Source: (Addison, 2005).

In this research, the Daubechies 4 (db4) wavelet is used as it was the best wavelet in the EEG research of Shukla and Kumar Chaurasiya (2018) and it is used in Verma et al. (2014).

5.5 Common Spatial Patterns

The CSP is another preprocessing method that is used to transform the data in such a way that it maximizes the variance of one class and minimizes that of another class, as seen in Figure 5.5. In the figure, the intuitive process is shown that CSP scales the data, and squeezes the data diagonally to get the data points of one class on the vertical axis and the data points of the other class on the horizontal axis to maximize the variance difference between the classes. By changing the differences in variance of the classes, the variance feature becomes more evident. The theoretical information of this technique is described by Blankertz et al. (2008). This technique is usually used to differentiate between two classes. In multiclass problems this is often solved by dividing it in multiple one versus the rest binary problems. In this research, an Approximate Joint Diagonalization is used to solve the multiclass problems (Trachel et al., 2021). The technique is based on the paper of Pham (2001), which describes Pham's algorithm. For the calculation of a two class CSP, the first step is to get the covariance matrixes $\sum^{(c)}$ of each class. The covariances are calculated over the sensors with the c class. With an example of the $+$ and $-$ classes, we can find the scaler w by solving the eigenvalue problem in Formula 5.11 (Blankertz et al., 2008).

$$\begin{aligned} & \Sigma^{(+)}w = \lambda\Sigma^{(-)}w \\ \text{s.t.}\quad & \lambda = \lambda_j^{(+)}/\lambda_j^{(-)}, \qquad j = 1, \ldots, J \\ & \lambda_j^{(+)} + \lambda_j^{(-)} = 1, \quad\ j = 1, \ldots, J \end{aligned} \tag{5.11}$$

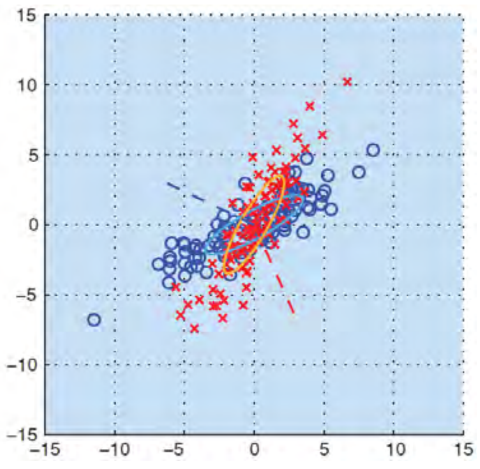
The $\lambda_j^{(c)}$ is defined as in Formula 5.12 with column vector j.

$$\lambda_j^{(c)} = w_j^T \Sigma^{(c)} w_j \tag{5.12}$$

Then the transformation of the data is a simple multiplication of the scaler w^T and all data points **x** as in Formula 5.13.

$$\mathbf{X}_{CSP}(t) = w^T * \mathbf{x} \tag{5.13}$$

In the output, the class variances are sorted such that in the first column the "+" class has the highest variance and the last column the lowest variance. The CSP preprocessing is then used for the DWT and as individual preprocessor for the classifiers.


(a) Before CSP

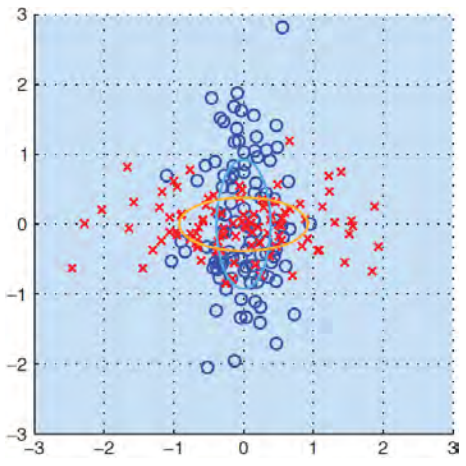

(b) After CSP

Figure 5.5: The effect of CSP. The data is transformed to increase the variance of one class and to minimize that of another class. In this figure, the blue circles and red crosses stand for two different classes. The axis of figure (a) are sensors values and on the axis of figure (b) are the transformed sensors values. Source: (Blankertz et al., 2008)

5.6 Statistical Numbers

From the output of the DWT and SVM there are statistical features calculated that replicate the feature extraction of Taspinar (2018). These are the variance, standard deviation, mean, median, the 5th, 25th, 75th and 95th percentile, the entropy and root mean square. An addition to these features are the skewness and area under the curve, as these proved to be good predictors in Zhang et al. (2020). For the DWT, the features are done over both subbands resulting in 36 features for each sensor as input for the machine learning models.

6. Machine learning

When the data is preprocessed, it can serve as input to the classifier algorithms. This chapter will explain the intuitive idea behind the algorithms of LR, SVM, TCN and LSTM.

6.1 Logistic Regression

The LR is one of the simplest models that is still used with success in classifying EEG data (Padfield et al., 2019b). LR has a comparable part with linear regression as shown in Formulas 6.1 and 6.2,

$$y = a + \mathbf{X}_t\mathbf{b} \tag{6.1}$$

$$p(\mathbf{X}) = \frac{1}{1 + \exp(-(\mathbf{w} \cdot \mathbf{X}_t - d))} \tag{6.2}$$

with feature vector $\mathbf{X}_t$ belonging to window t, and a and d as constants and weight vectors **b** and **w** (Flach, 2012a). The difference is that the LR has a sigmoid function over the linear regression, which makes it an s shaped function as shown in Figure 6.1. This difference causes linear regression to be used to fit numerical data while logical regression attempts to classify the data into two categories. To train the logistic model, the optimization problem in Formula 6.3 needs to be solved with the log-conditional likelihood (LCL), which is worked out in Flach (2012a).

$$\mathbf{w}*, t* = argmax(LCL) \tag{6.3}$$

Since this is a convex problem with only one solution there can be multiple iterations where the update rule in Formula 6.4 can be used with learning rate η and window t.

$$w_{new} = w + \eta(y_t - p_t)\mathbf{X}_t \tag{6.4}$$

This method is applied to distinguish between 2 classes. To make this a multiclass problem, this model is usually trained 3 times with a one class versus the rest scheme.

The output will be based on three calculated probabilities from the three models, where the class that has the highest probability is the output.

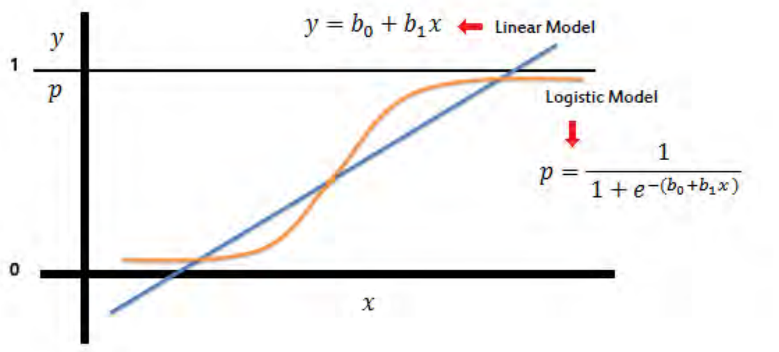


Figure 6.1: This figure shows the difference between a linear model and an LR. Source: (Sayad, 2022)

6.2 Support Vector Machines

The SVM is used as a classifier. To divide classes, the SVM attempts to find hyperplanes that maximize the difference between the two classes. The theoretical framework is from the book of Flach (2012b). There is a minimum of two support vectors as that is needed for the decision boundaries with the separating hyperplane in between. The SVM tries to make the margins between the vectors as big as possible. In Figure 6.2 a two-dimensional example with the classes “+” and “-” is shown. In this figure the data points are separated by the linear hyperplane and the support vectors are the data points on the decision boundary. The separating hyperplane should meet the conditions of Formula 6.5,

$$\mathbf{w} \cdot \mathbf{X}_t - d = 0 \tag{6.5}$$

with weight vector $\mathbf{w}$, vector $\mathbf{X}_t$ with window t, and constant d, with $\cdot$ as the inner product. So the distance of data point $\mathbf{X}_t$ to the hyperplane would be $\mathbf{w} \cdot \mathbf{X}_t - d$. SVM tries to maximize the margins which are expressed as $\frac{1}{||\mathbf{w}||}$. To maximize this, the problem in Formula 6.6 must be solved.

$$\begin{aligned} \mathbf{w}* &= \text{argmin} \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & y_t(\mathbf{w} \cdot \mathbf{X}_t - d) \geq 1, \;\; t = 1, \ldots, M \end{aligned} \tag{6.6}$$

Here y_i is equal to 1 or -1 depending on the class. A way to solve this is to transform this problem into the dual of the Lagrange multiplier and solve it with dedicated quadratic optimization solvers (Flach, 2012b). If the boundary cannot separate all points of the classes correctly, there is a need for an additional slack variable, which results in new formulas and optimization problems. These are further explained in detail by Flach (2012b). In addition to a linear SVM the kernel trick can be used. With this trick, the inner product "$\mathbf{X}_i \cdot \mathbf{X}_j$" can be replaced by a different kernel, such as a polynomial or a radial base kernel, which creates a non-linear hyperplane (Eaton, 2017). The kernels that are used are limited with the use of the Python package that is used for SVM. The used kernels in this research are described in the section 7 Experimental Setup.

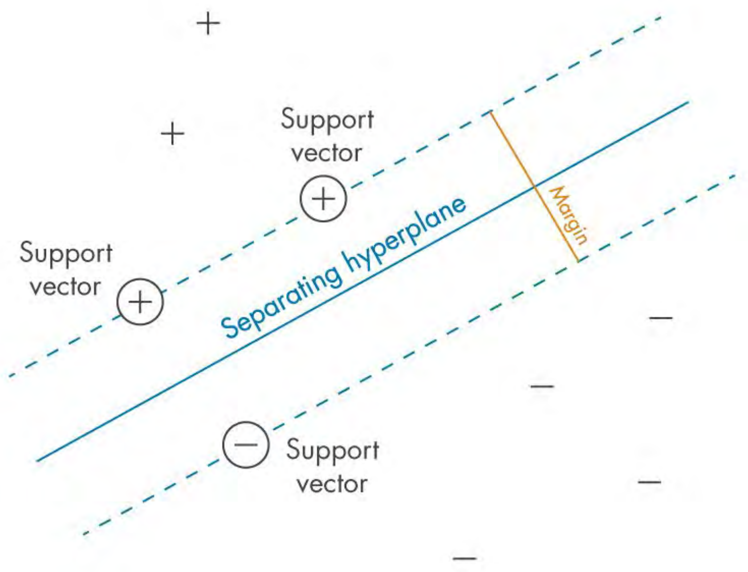


Figure 6.2: This figure shows the separating hyperplane of two dimensions of an SVM with the support vectors on the marginal distance of the plane. Source: ("Mathworks:Support Vector Machine (SVM)", n.d.)

6.3 Temporal Convolutional Network

A TCN is a variant of a neural network (NN). Therefore, it is needed to understand how a NN works. A NN is a machine learning algorithm that replicates how a human brain works with the nerves, as explained in 2.2 Brain Activity. The NN has an input layer, hidden layers and output layer as shown in Figure 6.3. Each layer has multiple notes. Each note calculates the sum of the feature input X_{ti} from window t and input i that is multiplied with weight w_i, as shown in Formula 6.7.

$$a_t = \sum_{i=1}^{N} w_i X_{ti} \tag{6.7}$$

Then the sum a_t goes through an activation function, which is usually a sigmoid function as shown in Formula 6.8, resulting in a value close to 0 or 1 (Gurney, 1997).

$$\sigma(a_t) = \frac{1}{1 + e^{a_t}} \tag{6.8}$$

This is comparable with the biological brain synapses that try to get an action potential of the next neuron. The number of notes that are connected to a note in the next layer is the kernel size. These calculations continue in each layer and is a trainable machine learning model, which has many variations. One variation is a recurrent neural network (RNN) in which outputs refer back to previous layers.

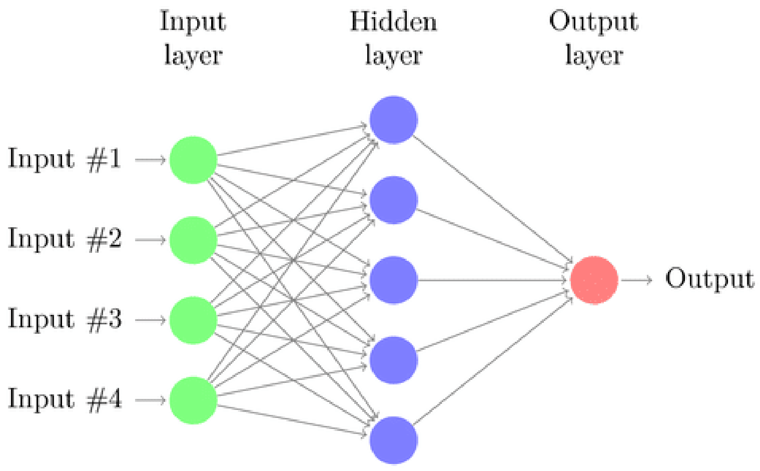


Figure 6.3: This figure shows the structure of a basic NN with a kernel size of 5. Source: (Dilmegani, 2021)

A TCN is a NN variant that can handle data in the time domain well, while it is not an RNN. It achieves similar and often better results compared to the Long Short-Term Memory (LSTM) network in the review of Bai et al. (2018). The TCN is based on a one-dimensional convolution or filter, as shown in Figure 6.4. This is similar to a normal NN but with a decreasing amount of notes in each layer to dilate the network and a smaller kernel size. With data in the time domain, the inputs of the current, past and future time points are used. In the figure, the blue node in the middle represents the input node with the current time. The TCN adjusts this by removing the future data points in the model, such that the current output only depends on the past and present data points, as shown in Figure 6.5. This is also called a causal convolution. To include dependencies from the distant past, nodes are skipped in the hidden layers. This is a dilation of the network. In each layer the dilation factor is doubled and half of the nodes remain (Romero, 2020). The total size of the receptive field R or input that is included can be computed with Formula 6.9, with vector d_i of dilation layers, kernel size k and N_{stack} is the number of TCN's on top of each other (Remy, 2020).

$$R = 1 + 2(k-1) * N_{stack} \sum d_i \tag{6.9}$$

The model has the advantage that it can be trained in parallel, therefore backpropagation is faster and does not suffer from vanishing gradients, in contrast to an RNN that does have these disadvantages (Bai et al., 2018).

The TCN model that is used in this research has an input layer, a TCN and a softmax output layer with three output notes corresponding to the three classes. The TCN code part is from Bai et al. (2018), and various hyperparameters determining the receptive field and training methods are tested. Since this model is made for sequential data, it is expected that the model performs better with the input of data points in the time domain in comparison with other models.

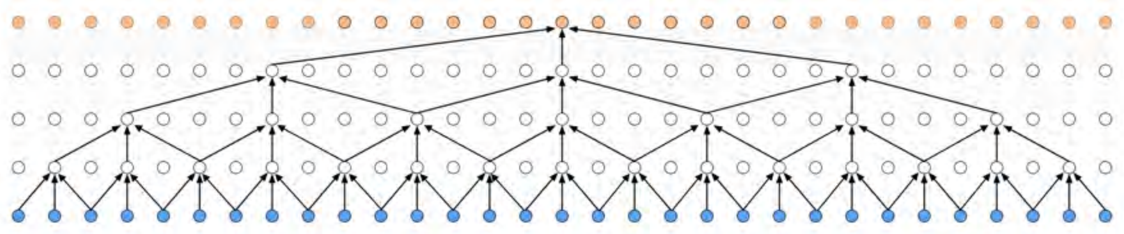


Figure 6.4: This figure shows a dense layer of a NN. Source: (Remy, 2020)

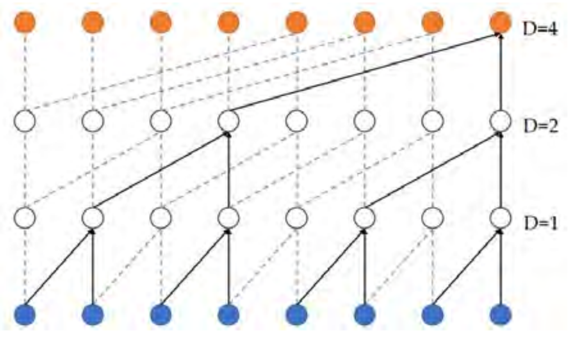


Figure 6.5: This figure shows the TCN with three different dilations. Source: (Bai et al., 2018)

6.4 Long Short-Term Memory

A LSTM model is an RNN that uses LSTM nodes. These are recurrent nodes that pass data from another time point to the next node. This allows the model to deal well with data in the time domain (Zhang et al., 2020). The LSTM node depicted in Figure 6.6 determines whether the current data should be forgotten or updated from the current input vector X_t at time t and the output of the previous LSTM node C_{t-1} with hidden layer h_{t-1}. This is calculated with multiple gates that have their individual function. The information state is C_t, the forget gate f_t, input gate i_t and output gate O_t. Each of these are calculated with Formulas 6.10 until 6.14, with weights W_c, W_f, W_i, W_o, and biases b_c, b_f, b_i, b_o from Zhang et al. (2020).

$$i_t = \sigma(W_i \cdot [h_{t-1}, \mathbf{X}_t] + b_i) \tag{6.10}$$

$$f_t = \sigma(W_f \cdot [h_{t-1}, \mathbf{X}_t] + b_f) \tag{6.11}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, \mathbf{X}_t] + b_o) \tag{6.12}$$

$$C_t = f_t * C_{t-1} + i_t * tanh(W_c \cdot [h_{t-1}, \mathbf{X}_t] + b_c) \tag{6.13}$$

$$h_t = o_t * tanh(C_t) \tag{6.14}$$

The formulas use the sigmoid function as shown in Formula 6.8. The weights and biases are trainable with normal back propagation to train the model (Zhang et al., 2020).

The LSTM model that is used is from the paper of Wang et al. (2018). Besides that, the preprocessing method described in the paper has been used as well. It is replicated from the paper to have a comparison of a realistic accuracy measure applied on the data made in this research. This is because the results of all studies from the literature chapter have very high accuracies that were not expected. For the preprocessing of the model, the window of each sensor is split into eight parts and a linear regression is made. The means and slopes are used as input for Input_A and Input_B, as shown in Figure 6.7. Then each input goes through a dense layer, where all notes are connected with the notes of the next layer. This layer theoretically finds the most important sensors (Wang et al., 2018). Then the data is normalized before it goes through an LSTM layer. After these nodes, the output is merged, and the prediction is extracted from the dense softmax layer with three outputs that stand for the three classes.

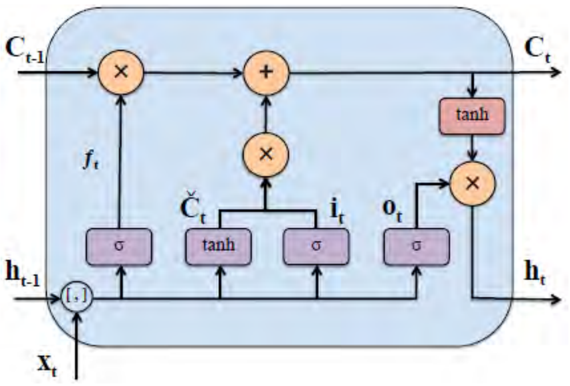


Figure 6.6: This figure shows an LSTM node of a NN. Source: (Zhang et al., 2020)

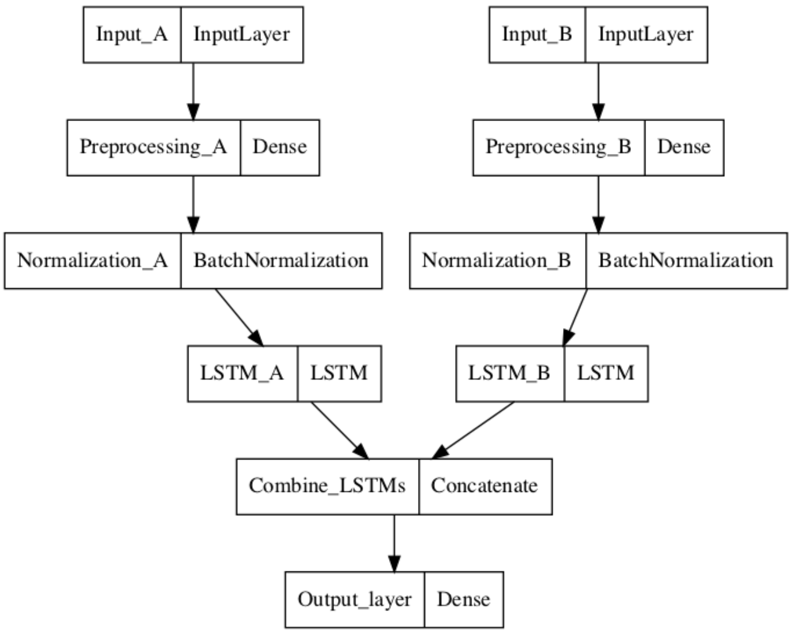


Figure 6.7: This figure shows the LSTM model that is used for this study.

7. Experimental Setup

All of the preprocessing and machine learning techniques described above are used in various combinations to find the best model to categorize the data. However, there are multiple unknowns which also needs to be tested. All experiments are divided into five main categories which are given in Figure 7.1. The most useful sensor set is determined because using the data of all sensors would take too long to train the model. The data selection is used to determine a good training set to train on, as there is a low signal-to-noise ratio. In the extreme case some datasets might have the same predictability as random data, which is not usable to train on. Then the hyperparameters are optimized in the model tuning. The hyperparameters are optimized stepwise. This means that only a few hyperparameters have multiple options to test, while the other hyperparameters will stay constant. In these first experiments, the best are determined by the highest accuracies on the validation set. The model comparison then compares the optimized models with the test set accuracies. Lastly, there is an error analysis made of the best model on multiple datasets. These experiments are described in more detail in this chapter.

$$\begin{tikzcd}
\boxed{\textbf{Sensors Set}} \arrow[r] & \boxed{\textbf{Data Selection}} \arrow[r] & \boxed{\textbf{Model Tuning}} \arrow[r] & \boxed{\textbf{Model Comparison}} \arrow[r] & \boxed{\textbf{Error Analysis}}
\end{tikzcd}$$

Figure 7.1: The flow diagram of the experiments.

7.1 Sensor set

The first experiment is to find the best sensor combination, because there are multiple setups used in the literature. The locations of the sensors on the helmet are both around the sensorimotor cortex, which is the most important area for MI tasks (Kübler & Mattia, 2016). The first group also has sensors distributed around the frontal lobe as has been done in Kübler and Mattia (2016). The second composition has the sensors around the parietal lobe as is done in Bousseta et al. (2018) and Garcia-Moreno et al. (2020b). In Figure 2.2, details of the sensor placement and brain functions are shown. From this is derived that the parietal lobe can be used to detect activation of the senses, and the frontal lobe can be used to detect movement control.

To examen the best sensor composition, the data is preprocessed by the CSP followed by the DWT and the statistical feature abstraction. For the data windows it was decided to use 512 data points, as 512 milliseconds was considered a sufficient reaction speed for playing the game. This is also a multitude of 2^7, that is needed as the seventh level of the DWT is used, and because of this the data needed to be separated seven times. The preprocessing sequence was chosen as it appeared to result in good accuracies during the initial trails. The sensor compositions are tested over three datasets with differently performed tasks, because the different tasks may produce different brain waves and influence the results. The tasks are ”ThinkingMotion2b” with lifting the knees, ”PhysicalMotion” with physical movements instead of MI tasks, and ”Thinkingmotion3” with pushing the tong to the roof of the mouth as stated in the tasks description of Table 4.2. Each dataset consists of multiple recordings in order to test over sufficient data and avoid high scores due to chance. To investigate this, the machine learning algorithms SVM and LR are used, as they are faster to train compared to the TCN and LSTM. The learning rates have been varied on a logarithmic scale from 1.0 to 0.001 as that is commonly used (Goodfellow et al., 2016) and the maximum iterations have been varied over 100, 500, and 1000 iterations. The maximum of 1000 iterations is decided upon because it appeared in the initial trails that the models did not improve with more iterations. This is varied to prevent possible under- or overfitting of the data. It is assumed that the classifying algorithm should have no influence on the differences between the two tested sensor groups. This is based on the trash in trash out principle, where a wrongly placed sensor provides no information and will result in bed predictions independent of which algorithm is used. Therefore, it is not needed to use all algorithms, as the goal of this experiment is to find which sensors contain the most useful information for an initial setup for the rest of the experiments. The best will be determined by comparing the validation accuracies of all datasets with the two different compositions.

7.2 Dataset selection

The second search is to find the best dataset to test the models on. The datasets have a different quality because the predictability can be different for each measurement and task (Zhang et al., 2020). It is assumed that the best dataset with a high accuracy gives a clearer view for comparing models, because the tasks are more likely to have a high predictability. The highest accuracies indicate the dataset with the best predictable windows of the tasks. The logic behind this is that less predictable datasets have more random data or suffer more from the low signal-to-noise ratio, that results in lower accuracies. Therefore, selecting a dataset prevents training on random data that has low predictability. This would result in a validation accuracy around the random probability of success, which is 33%. In this search for the best dataset, it is assumed that the eventual optimal hyperparameters are transferable to other datasets for the highest accuracies. Selecting a dataset instead of using all datasets also reduces the duration of the subsequent experiments. The measurements were performed with only the LR and the SVM, because the models have a shorter training duration. The models used the initial hyperparameters as shown in Table A2. A disadvantage of this is that the subsequent experiments will be optimized for only this dataset and that might be a local optimum or the assumption could be incorrect and the same hyperparameters not be transferable. This would however not be logical as it is still the same kind of data.

Classifier	Parameter	Constant Value
LR	Penalty	l2
	Solver	lbfgs, saga
	Learning Rate	1, 0.1, 0.01, 0.001
	Maximum Iterations	100, 500, 1000
SVM	Kernel	rbf
	Learning Rate	1, 0.1, 0.01, 0.001
	Maximum Iterations	100, 500, 1000
TCN	Epochs	50
	Learning Rate	1, 0.1, 0.01, 0.001
	Batch Size	100
	Optimizer	Adam
	Dilations	6
	Kernel Size	4
	Stacks	1
	Normalization	Layer
	Dropout Rate	0.001
	Filter	3
LSTM	Epochs	100
	Learning Rate	0.001
	Batch Size	64
	Optimizer	Adam
	Normalization	Batch
	LSTM-cell Dropout Rate	0.4
	Loss Function	Crossentropy

Table 7.1: The initial hyperparameters used for the first sub-study are shown. In the case of multiple values, they were all considered individually with only the model with the highest validation accuracy being included in the results tables.

7.3 Model Tuning

In the next experiment, the models are optimized. For this, the best preprocessing is first selected, followed by the selection of the best hyperparameters of every model based on the highest validation accuracy. Because the learning rate and the maximum iterations can influence the model, the same hyperparameter values for the SVM and the LR as before are varied in these measurements. For the TCN, the same learning rates are maintained and the model is initialized several times with different random weights. With this setup all possible preprocessing methods are measured with the three models. The preprocessing methods are CSP, DWT, CSP$\rightarrow$DWT, FFT and all these features are tried with and without statistical features as described in section 5.6 Statistical Numbers. The preprocessor CSP is the only technique that has multiple settings. The CSP can use different regularizations and can return a different amount of components. Therefore, these hyperparameters are investigated as well in two steps with all possible hyperparameters. The hyperparameters of the classifiers to test the preprocessing are shown in Table A2.

For the hyperparameter optimization of the classifiers, the SVM varied over all Kernel and Gamma options in a single step. The LR varies over all solver and penalty combinations in the first step as those hyperparameters influence each other. The second step is the investigation of the learning speed related hyperparameters: Learning rate and iterations, of the SVM and the LR. If the best values are either the highest or the lowest, a value was added to make sure the optimal value is obtained. The first optimization for the TCN is the receptive field for which values are taken such that all inputs can be taken into consideration with the biggest receptive field. This is followed by the normalization hyperparameter individually. The last step is the optimization of the learning related parameters: batch size, optimizer and learning rate. Because the TCN in heavily dependent on the initialization of the weights, each hyperparameter combination is tested multiple times. The exact hyperparameter settings of each model for determining the best hyperparameters initiated according to Table A.1 and the hyperparameters which were varied over stepwise are shown together with the results in table 8.4.

7.4 Model Comparison

In the last experiment, the final models are compared. The only variables left in the creation of the models are the randomly initialized weights of the TCN and LSTM. For this, the models are trained 20 times and the model with the highest validation accuracy is used as the final model. Then the models are compared with four datasets. The accuracy of the test set is used to determine the best model. This accuracy is considered the real accuracy of the model since the test set is completely independent of the model as it was not yet used in previous experiments. The predictions that are made by the models are tested with a binary statistical test to determine or the models statistically differ. It was made binary by considering a prediction either correct or wrong, and the test is done considering all predictions of the test sets of the four datasets. The test that is used is the binary prop test in R.

7.5 Error Analysis

A short error analysis is made with the best model and the previously used four datasets to acquire additional insights. There are confusion matrixes plotted to detect in which tasks the most occurring errors are. The average test accuracy over the tasks is plotted over time to acquire information about the predictability over time.

8. Results

The findings of the study are presented in this chapter. The study is divided into several experiments. In each experiment, the information obtained is used in the subsequent experiment. The values that are varied in each experiment are listed with the results to maintain an overview. The sensors and dataset were determined first. This is followed by multiple searches to find the best hyperparameters of the LR, SVM and TCN. Then all the models are compared in the final experiment.

8.1 Sensor set

The first experiment is to determine the best sensor set. The main results can be seen in Table 8.1. It shows that the sensor set with the sensors placed around the Parietal lobe achieves higher accuracies on the validation in five of the six sensor comparisons. It appears that this set of sensors therefore provides more or better information, and this set of sensors will therefore be used in the subsequent experiments. The eight sensors are then around the (post-)central lobe, that is responsible for the senses on top of the headset, the sensors around the parietal lobe for attention and perception of the senses, and the temporal lobe for the memory and processing on the sides of the headset.

Furthermore, the validation accuracies where the frontal lobe sensors are used, are often almost equal to the accuracy that would be acquired by the random chance that a class is correctly predicted, which is 33%. This can mean that the data from these sensors mostly consists of noise. It is also remarkable that the dataset with task ThinkingMotion2b and the parietal lobe has the highest validation accuracy of 45.54% with LR, but the lowest training accuracy with SVM. This can be explained by that the models do have a significant influence and the trend is not always detected by the algorithm.

SVM										
Dataset	**Frontal lobe sensor composition**					**Parietal lobe sensor composition**				
	Sensors: ”C3”, ”C4”, ”T7”, ”T8”, **”F3”, ”F4”, ”F7”** and **”F8”**					Sensors: ”C3”, ”C4”, ”T7”, ”T8”, **”P3”, ”P4”, ”P7”** and **”P8”**				
A 2022-01-11 ThinkingMotion2b		Train acc	Val acc	Avg	Parameters		Train acc	Val acc	Avg	Parameters
	1	35.30	33.26	34.28	1.0, rbf, 1000	1	38.24	35.02	36.63	0.001, rbf, 1000
A 2021-11-02 PhysicalMotion		Train acc	Val acc	Avg	Parameters		Train acc	Val acc	Avg	Parameters
	1	59.57	33.28	46.43	1.0, rbf, 1000	1	56.63	37.04	46.84	1.0, rbf, 1000
	2	45.91	33.36	39.64	0.1, rbf, 1000					
H 2021-12-08 ThinkingMotion3		Train acc	Val acc	Avg	Parameters		Train acc	Val acc	Avg	Parameters
	1	59.12	35.95	47.54	1.0, rbf, 1000	1	59.93	29.67	44.80	1.0, rbf, 1000
						3	42.22	38.04	40.13	0.01, rbf, 1000
LR										
Dataset	**Frontal lobe sensor composition**					**Parietal lobe sensor composition**				
	Sensors: ”C3”, ”C4”, ”T7”, ”T8”, **”F3”, ”F4”, ”F7”** and **”F8”**					Sensors: ”C3”, ”C4”, ”T7”, ”T8”, **”P3”, ”P4”, ”P7”** and **”P8”**				
A 2022-01-11 ThinkingMotion2b		Train acc	Val acc	Avg	Parameters		Train acc	Val acc	Avg	Parameters
	1	51.32	33.37	42.34	1.0, 1000, l2, lbfgs	1	55.38	45.54	50.46	1.0, 500, l2, lbfgs
	19	45.98	33.42	39.70	0.001, 500, l2, saga					
A 2021-11-02 PhysicalMotion		Train acc	Val acc	Avg	Parameters		Train acc	Val acc	Avg	Parameters
	1	62.02	31.73	46.87	1.0, 1000, l2, lbfgs	1	55.41	38.68	47.04	0.1, 1000, l2, lbfgs
	11	57.00	33.20	45.10	0.01, 100, l2, lbfgs					
H 2021-12-08 ThinkingMotion3		Train acc	Val acc	Avg	Parameters		Train acc	Val acc	Avg	Parameters
	1	58.99	35.03	47.01	0.1, 500, l2, lbfgs	1	62.43	33.59	48.01	1.0, 1000, l2, lbfgs
	10	56.57	36.34	46.46	1.0, 100, l2, saga	3	60.73	34.38	47.56	1.0, 100, l2, lbfgs

Table 8.1: A summary of the results from the two sensor sets are shown across two classifiers and three data sets. The results consist of the train accuracy, validation accuracy,and the average of the two. In addition, the parameters learning rate, maximum iterations and kernel or penalty and solver are given from the associated models. The model with the highest average and highest validation accuracy are listed in the table. Only one model result is shown if th model was had the highest accuracy on both. The number in the first column of the results is the ranking of the model with a differen parameter combination, that is ranked on the average accuracy.

8.2 Dataset selection

In the second experiment, it has been examined which dataset is most useful for the rest of the experiments. The results are shown in Table 8.2. The datasets with a validation accuracy above 40 on both classification models are highlighted. Of the five datasets where all accuracies score over 40, ThinkingMotion2b has the highest accuracies with 48.93% with the SVM and 47.29% with LR. Another advantage of this dataset is that the final goal should also be an MI motion for the applications. For this reason, this dataset was chosen. Furthermore, the datasets with tasks without number, that did not have a task for the ”Neither” label do not have any accuracy above 40%. The datasets with other tasks have good and poor results among them.

Recording data				**Support Vector Machines**			**Logistic Regression**		
Person	**Date**	**Task**	**Files**	**Train Accuracy**	**Validation Accuracy**	**Test Accuracy**	**Train Accuracy**	**Validation Accuracy**	**Test Accuracy**
A	2022-01-11	**ThinkingMotion2b**	2	38.24	35.02	34.40	55.40	45.23	37.85
		ThinkingMotion3	2	57.34	30.59	38.05	61.41	30.72	39.92
	2021-11-26	ThinkingMotion	18	34.12	33.02	33.38	37.34	33.01	33.17
	2021-11-02	ThinkingMotion	3	51.96	29.03	34.78	52.53	33.33	37.77
		PhysicalMotion	3	56.63	37.04	41.18	55.41	38.68	39.84
	2021-12-08	PhysicalMotion2	2	71.38	50.26	43.08	74.03	45.82	45.54
		ThinkingMotion2	3	CSP error			CSP error		
	2022-01-18	PhysicalMotion2	4	48.17	39.9	38.39	58.98	44.7	41.56
		ThinkingMotion2b	1	60.80	38.75	38.14	61.81	38.40	35.70
H	2022-01-11	ThinkingMotion2b	2	34.90	38.32	30.05	51.47	40.64	29.22
		ThinkingMotion3b	1	64.38	43.86	45.88	65.18	40.09	44.97
		ThinkingMotion2	2	58.68	43.99	35.46	59.22	42.30	36.51
	2021-11-10	PhysicalMotion	2	58.86	37.67	37.12	59.48	37.67	37.25
		ThinkingMotion	2	54.59	35.92	29.71	56.93	37.37	27.43
	2021-12-08	**ThinkingMotion3**	2	59.93	29.67	40.05	62.43	33.59	40.57
	2021-12-08	PhysicalMotion3	2	61.85	43.27	37.24	65.38	43.79	34.79
	2022-01-18	PhysicalMotion2	3	64.90	42.75	48.56	71.64	36.19	37.75
		ThinkingMotion2b	1	64.78	48.93	54.79	65.36	47.29	52.72

Table 8.2: This shows all the accuracies of the data sets from the SVM and the LR classifiers. In case both the accuracy of the validation and the test set are above 40, the score is highlighted green. If both classifiers are highlighted green, the task is also highlighted. The tasks in bold are the datasets used in the previous experiment. Lastly, the CSP error is a convergence error, caused by zero values.

8.3 Model Tuning

Classifier	**Preprocessing**	**Train Accuracy**	**Validation Accuracy**
SVM	CSP	90.12	45.14
	CSP $\rightarrow$ features	66.86	38.64
	DWT	79.84	35.68
	DWT $\rightarrow$ features	47.12	31.9
	CSP $\rightarrow$ DWT	85.49	38.18
	CSP $\rightarrow$ DWT $\rightarrow$ features	64.78	48.93
	FFT	65.69	35.25
LR	CSP	56.06	39.72
	CSP $\rightarrow$ features	68.52	36.81
	DWT	42.42	43.08
	DWT $\rightarrow$ features	54.43	33.62
	CSP $\rightarrow$ DWT	45.75	36.54
	CSP $\rightarrow$ DWT $\rightarrow$ features	57.33	47.29
	FFT	42.85	35.08
TCN	CSP	50.16	35.34
	CSP $\rightarrow$ features	43.90	34.39
	DWT	43.43	34.31
	DWT $\rightarrow$ features	32.28	35.51
	CSP $\rightarrow$ DWT	50.38	37.49
	CSP $\rightarrow$ DWT $\rightarrow$ features	56.39	47.12
	FFT	40.08	36.11

Table 8.3: The different possible preprocessing techniques have been applied to the classifiers. The preprocessing technique with the highest validation accuracy of each classifier is highlighted in green.

The results of the preprocessing are shown in Table 8.3. For all the models, the sequential preprocessing of the CSP, DWT followed by the features appears to generate the most informative data for the classifiers to achieve the highest validation accuracy. For SVM the bigger variance of the CSP seems to be especially benefitting, while summarizing it with features already loses its value. The rest of the preprocessing techniques for the SVM seems to be closer to random chance. The LR shows similar results with the exception of a better result on the DWT, which is probably a lucky guess results as it has a higher training accuracy which is not realistic. The TCN only has the CSP, DWT followed by the features as a good result with almost 10% higher accuracy than the second best preprocessing method. After this measurement, there are several regularizations and number of components investigated for the CSP. The results are elaborated in Appendix A.2 and show that empirical regularization with four components leads to the highest validation accuracy as more do not result in a higher accuracy and would be redundant.

The hyperparameter tuning of all models is done step by step, as described in Table 8.4. In this table, the final hyperparameters that are decided upon are shown in bold. The improvement of the validation accuracy exists, but is very minimalistic, with the biggest improvement of 3.27% from the receptive field experiment of the TCN. The results of the individual steps are described in Appendix A.3. Most results with LR only show marginal differences that are smaller than 2%. The TCN has similar differences, except for the normalization layer. Not having this layer results in a validation accuracy that is around 6% lower. The SVM only shows big differences in the experiment of the learning rate and iterations. These values show in Table that the two lowest tested values should not be used. Because these hyperparameters can cause a big difference in the predictability of the models, they are more important hyperparameters.

As a final result, the first three final models have the same preprocessing and have the following hyperparameter settings: The TCN has a receptive field of 61, a layer optimization, the SGD optimizer, a batch size of 100 and 100 iterations. The LR has a learning rate of 1 with 500 iterations, and uses the saga solver without penalty. As the last tuned model, the SVM uses the radial basis function as kernel with scaling as gamma value and learning rate 1 with 1250 iterations.
The final LSTM model used for the comparison of the models has the same settings as described in the paper of Wang et al. (2018).

Model	Step	Tested Parameter	Tested Values	Table	Validation Accuracy
SVM	0	Preprocessing	FFT, **CSP, DWT, feature** combinations	8.3	48.93
	1	Kernel Gamma	linear, poly, **rbf**, sigmoid **scale**, auto	A6	49.01
	2	Learning Rate Iterations	2, **1**, 0.1, 0.01, 0.001 250, 500, 750, 1000, **1250**, 1500	A7	49.01
LR	0	Preprocessing	FFT, **CSP, DWT, feature** combinations	8.3	47.29
	1	Solver Penalty	lbfgs, newton-cg, sag, **saga** **none**, l1, l2, elasticnet	A5	47.55
	2	Learning Rate Iterations	2, **1**, 0.1, 0.01, 0.001 250, **500**, 750, 1000, 1250, 1500	A7	47.55
TCN	0	Preprocessing Learning Rate	FFT, **CSP, DWT, feature** combinations 1, 0.1, 0.01, 0.001	8.3 A4	47.12
	1	Dilations Kernel Size Stacks	**4**, 5, 6 2, **3**, 4 1, **2**	A9 A10	50.39
	2	Normalization	none, batch, weight, **layer**	Figure A3	
	3	Batch Size Optimizer Learning Rate	**10**, 50, 100, 200, 300 Adam, **sgd** 1, 0.1, **0.01**, 0.001	A11	50.82

Table 8.4: The different possible hyperparameters were tested for the classifiers. The hyperparameters with the highest validation accuracy of each step are shown in bold. The more detailed results of each step can be found in the indicated tables.

8.4 Model Comparison

In the last experiment, the final models were compared. The models are tested over the four datasets that have the best data, as shown in section 8.2 Data selection. The results are shown in Figure 8.1. Only the test accuracy was used to determine which classifier has the highest accuracy on the same training data, as the validation was already used to determine part of the models. The results show that the LSTM classifier is the best model of the tested models. In Table 8.5 the results of a statistical test are shown that compares the consecutive scoring models. Only the accuracy of the LSTM scores significant different, while the other models show similar prediction accuracies.

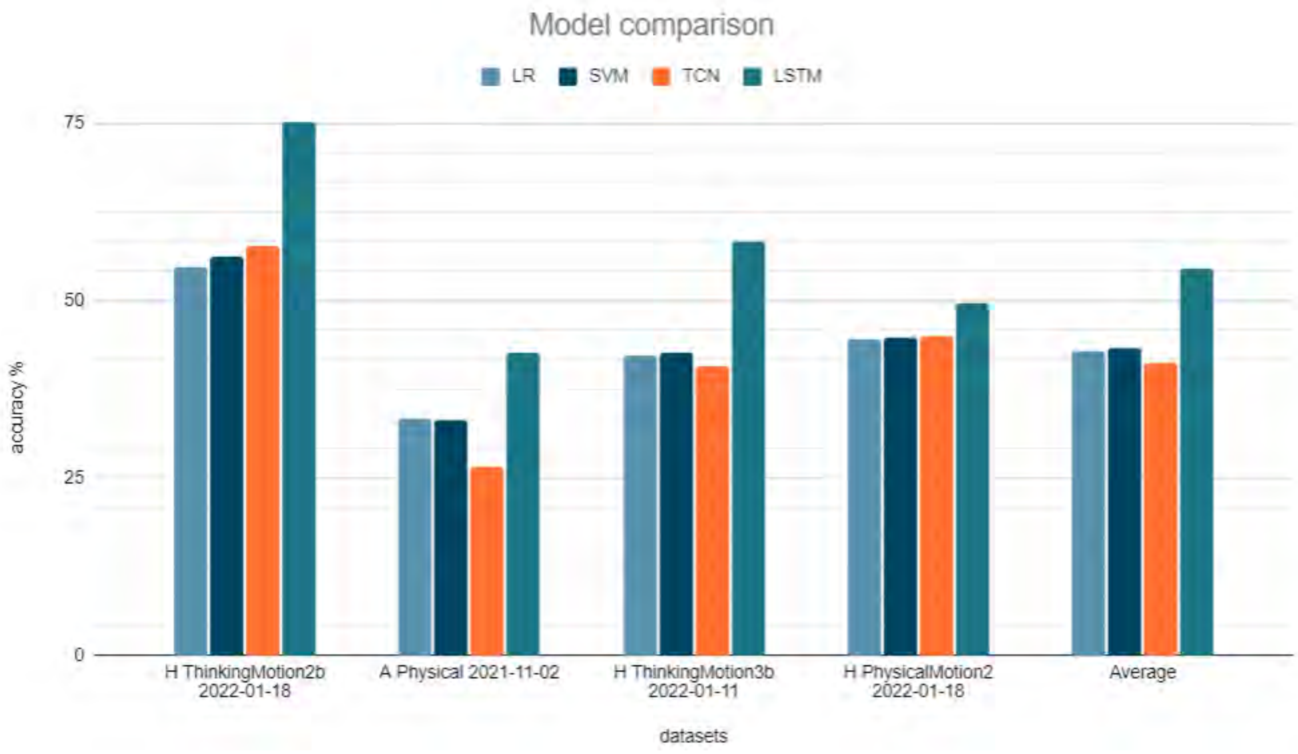


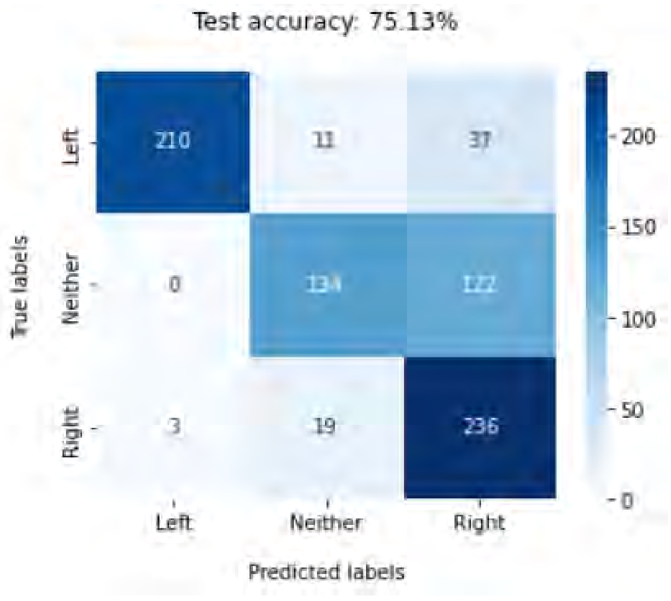
Figure 8.1: This figure shows the test accuracies of the models on four datasets.

P-value	LR	SVM	TCN
LSTM	>2.2e-16	>2.2e-16	>2.2e-16
TCN	0.1842	0.1571	-
SVM	0.5865	-	-

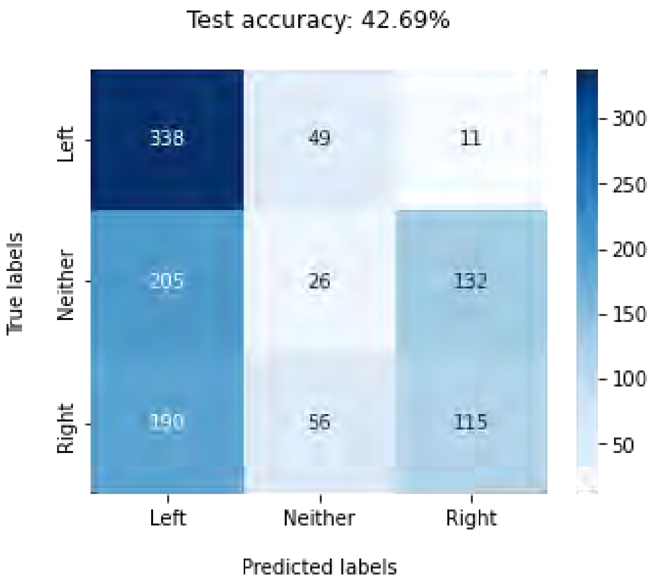
Table 8.5: The statistical differences of the models are displayed with the P-value from the prop-test from Rstudio.

8.5 Error Analysis

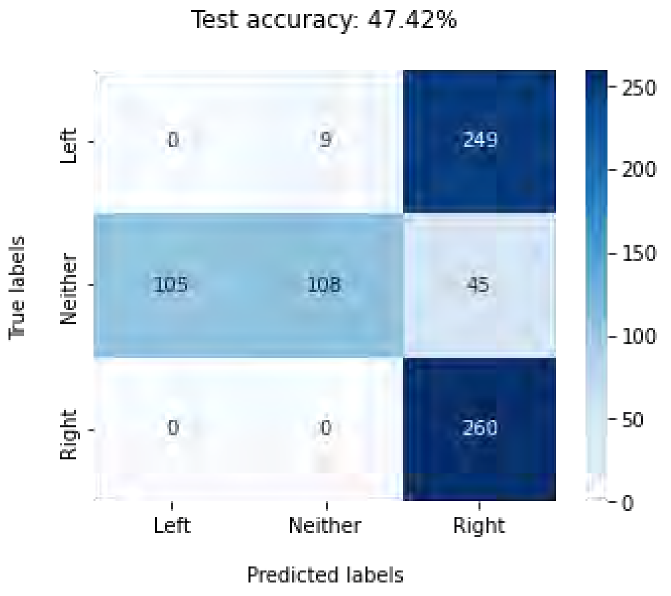
To get an insight in the model, a short error analysis is performed. Confusion matrices are shown in Figure 8.2. On each of the previously used datasets, the most common prediction error is different. There is especially confusion with the neither tasks, that are either predicted too often or few times. The third dataset, on the other hand, has the most difficulty predicting the ”Left” task, as shown in confusion matrix (c). The successful part of this result is that the model split the ”Neither” tasks from the others almost perfectly. The most occurring error differs, because the models are probably trained to find other aspects. This difference between models could have been caused by the use of datasets that are too small or the data collection tasks and person give totally different brain activity.



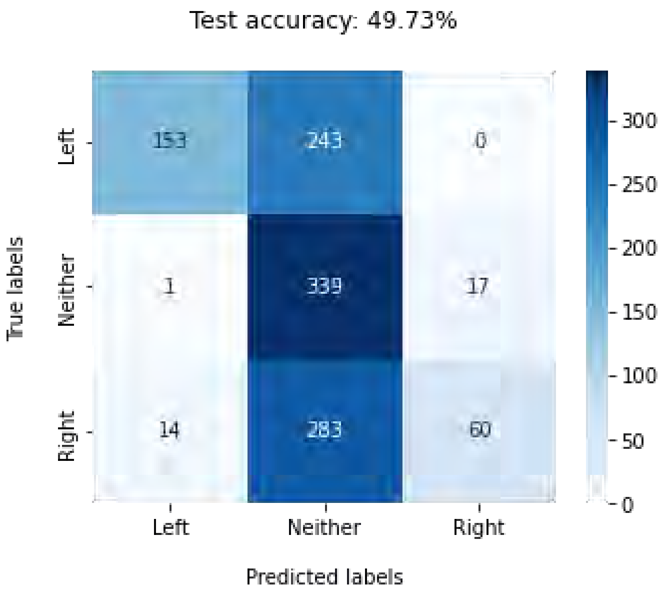
(a) H ThinkingMotion2b 2022-01-18



(b) A PhysicalMotion 2021-11-02



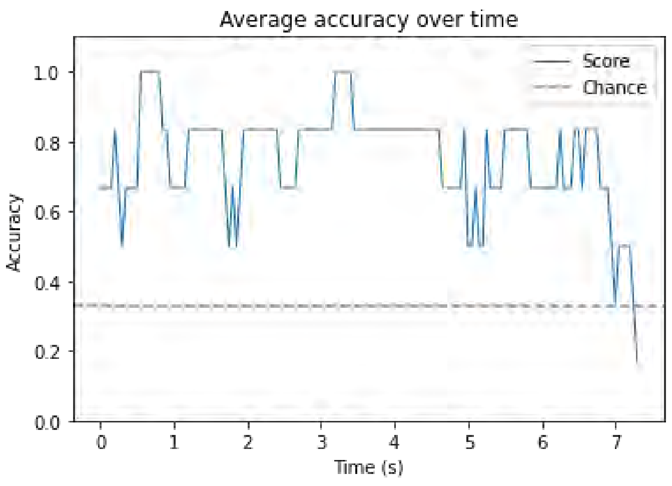
(c) H ThinkingMotion3b 2022-01-11



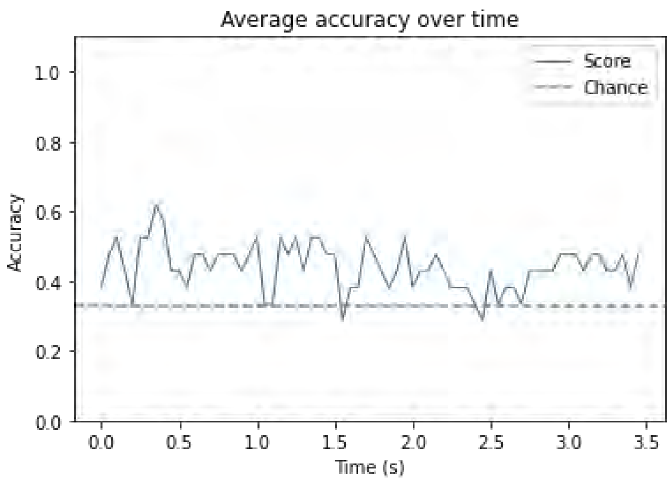
(d) H PhysicalMotion2 2022-01-18

Figure 8.2: The Confusion matrix with the same four datasets. The number represents the number of windows that is predicted.

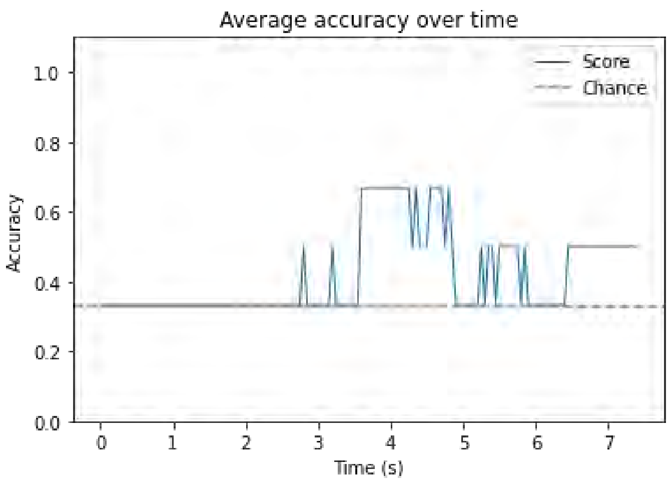
The accuracy over time of the tasks has been plotted with the LSTM as shown in Figure 8.3. The plots are made without the removal of the beginning and end of the data. Plot (c) is the only unstable one that shows an accuracy that is equal to the random chance at the first 3 seconds. The other models show a more consistent variability. Plot (a) and (d) also show that the predictions are poor at the first few windows. This is logical as there is a certain reaction time needed.



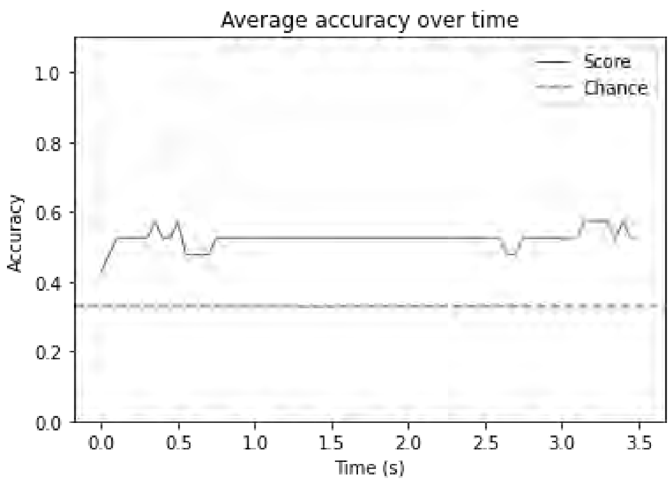
(a) H ThinkingMotion2b 2022-01-18



(b) A PhysicalMotion 2021-11-02



(c) H ThinkingMotion3b 2022-01-11



(d) H PhysicalMotion2 2022-01-18

Figure 8.3: The accuracy over time with the same four datasets. The x-axis shows the start time of the windows that are predicted. Each window has a length of 0.512 seconds.

9. Discussion

From the results, the LSTM appears to produce the most accurate predictions. In this chapter, the underlying aspects are discussed. This means that the models and limitations are discussed and some suggestions are made for future research.

9.1 Best model

To answer the main research question: "*What machine learning techniques will result in the highest accuracies for classifying the EEG data collected in this research?*", it appeared that splitting up the windows and fit the slope and mean of each part, in combination with an LSTM as described in Wang et al. (2018) is the best machine learning technique to acquire the highest accuracies. There are several aspects that can explain this result. A unique point with this model is that the preprocessing is reproduced from the paper and is therefore different from the other models. The preprocessing is to change the data, which has no intuitive or direct prediction value, into features with greater informative value where the classifier can make a better prediction. It may be the applied preprocessing of the LSTM that takes 8 times the slope and average of a portion of the window has more informative value than the other applied preprocessing with CSP$\rightarrow$ DWT$\rightarrow$ features. Furthermore, the LSTM is made for sequential data and has this as input at the preprocessing. The TCN was also made for sequential data. However, the non-sequential data features appeared to work better as the current preprocessor is the only one that was far from the gambling chance with 47% and scored at least 9% better than other preprocessors with the TCN. It may be that the time changes of the data windows are more important for classification and therefore the LSTM classifier worked better.

9.2 Features

To further elaborate on the effects of the features, it can be seen in Table A9 that the average of the best TCN models shows that a larger receptive field on average results in a higher accuracy. However, in the top 10 TCN models in Table A10 it shows that it mainly has smaller receptive fields. This may be because there are only a few key features, which lead to better results if those are included, but the use of all features results in overfitting. This theory can also explain that the CSP and DWT individually give better results compared to the same with features in the preprocessing experiment with Table 8.3. Future research can investigate what the most important features are, which can prevent overfitting by too many irrelevant features. New models can also be created by combining the preprocessing of the LSTM and the features. Since these both result in similar high accuracies, it is possible that these combined may lead to better results.

9.3 Assumptions

There are several assumptions made during this research that have not been specifically verified. One assumption is that the optimal untrained model in which the hyperparameters are determined for a specific dataset is also the optimal model for other datasets. However, it may be that only a local optimum is found due to limited and specific data. For example, other sensors may be better because the headset is positioned differently on the specific person, or the brainwaves are different. As a solution the data can be expanded to include multiple datasets. However, Table 8.2 shows that dataset 'A 2021-11-26 ThinkingMotion' performed poorly with 18 data files. It may be that this is due to poor data, which also causes a lot of training on noise. This theory is made knowing that there are good and poorly predictable tasks among them (Zhang et al., 2020). That there is a lot of noise is also visible in the comparison of the datasets, because a lot of results had a test accuracy around the random chance of 33%. This means that the individual windows are hard to predict. For future research, it is possible to try to extract well predictable tasks from each dataset and train on them instead to increase the accuracies.

The evaluation method used for the choices of the best dataset and parameters was the validation accuracy and for the best model the test set. Using the accuracy is also common in EEG data classification. However, this does not take the chance into account that a class is predicted correctly by coincidence. This is not a problem within this research since the probability is the same for all experiments within the research as they all needed to predict three classes. In this particular research the evaluation could also have been based on playability of the game. It can be considered to search how well two of the three classes can be differentiated, since the game can be played with two moves if the move of letting the bar stay in place (the Neither task) would be replaced by switching between left and right. For example, confusion matrix (c) of Figure 8.2 would then get a high score as it almost perfectly differentiate the neither tasks from the others. Therefore, these result would be different and that might be more appropriate for playing the game if switching left and right to stay in place works. Further assumptions of the data are: that the correct start and end of the data is removed, so that the well predictable data is preserved and that the correct window size is selected. The removal of data could have been smaller and around 300 milliseconds as that appeared out of the error analysis. However, the assumption is correct as the important data part is preserved. The window size is decided by the practical use, as a larger window size leads to more data per window and therefore theoretically leads to better predictions, but in practice this means that the delay time during playing becomes larger. Also, the number of samples (or windows) is reduced. Both used values are estimates that could possibly be investigated further.

9.4 Recording setup

A subject of discussion that could be improved for better results is the recording setup. It was known that the headset is in the cheaper price range which can indicate poorer quality material. It also uses dry sensors which both causes poorer quality data. This can be improved by using EEG headsets that has proven the quality of the generated data. For example, A medical grade EEG headset would be considered to generate high quality data. However, using consumer grade equipment was also a part of this study.

In the data analysis, it was noticed that there was very likely external noise measured at 50 Hz. This gives reason to think that there is also external noise on other frequencies that is measured and causes lower accuracies of the classifiers. In the ideal case, this could be avoided by using a Faraday cage that blocks some external signals.

Lastly, it was mentioned in the literature that the mood could influence the brain activity (Padfield et al., 2019a). This information was not used during the data collection. It could be that the test subject had different moods during each recording, causing different brain activity for the same dataset, as there can be multiple recordings in the same dataset. In the best case scenario, the same data is obtained for the same tasks. For future research, the mood of the subject should be stabilized by using the same surroundings and music before the start of the measurements.

9.5 Subjects

As a final point, the test subjects are not experienced EEG users and do not know which form of thinking is most effective. Therefore, in future research, an online setting with adaptive learning could work to get a better model, and could serve to teach the users which way of thinking provides EEG data that has good predictability (Portillo-Lara et al., 2021). In the ultimate case, an experienced user that knows how he should think to generate the most useful brain activity for the EEG data is used as subject. This could result in higher accuracies for a better gaming experience.

10. Conclusion

In conclusion, the LSTM model is the best studied model. In accordance with the literature, this study also shows that there is a low signal-to-noise ratio and that the data quality can vary from task to task. For further developments, several combinations are possible in data collection, the models, and features to obtain even higher accuracies in the future.

Bibliography

Addison, P. (2005). Wavelet transforms and the ecg: A review. *Physiological measurement*, *26*, R155–99. https://doi.org/10.1088/0967-3334/26/5/R01

Alda, A., & Torreblanca, N. (2020). *Eeg electrode placement: Fixed vs. variable*. https://www.bitbrain.com/blog/eeg-electrode-placement

Amarasinghe, K., Sivils, P., & Manic, M. (2016). Eeg feature selection for thought driven robots using evolutionary algorithms. *2016 9th International Conference on Human System Interactions (HSI)*, 355–361.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *CoRR*, *abs/1803.01271*. http://arxiv.org/abs/1803.01271

BioNinja. (2016). *E4 neurotransmitters and synapses*. Retrieved December 21, 2021, from http://www.old-ib.bioninja.com.au/options/option-e-neurobiology-and-2/e4-neurotransmitters-and.html

Blankertz, B., Tomioka, R., Lemm, S., Kawanabe, M., & Muller, K.-r. (2008). Optimizing spatial filters for robust eeg single-trial analysis. *IEEE Signal Processing Magazine*, *25*(1), 41–56. https://doi.org/10.1109/MSP.2008.4408441

Bousseta, R., El Ouakouak, I., Gharbi, M., & Regragui, F. (2018). Eeg based brain computer interface for controlling a robot arm movement through thought. *Irbm*, *39*(2), 129–135.

Brigham, E. O. (1988). The fast fourier transform (fft). *The fast fourier transform and its applications* (pp. 131–164). Prentice-Hall, Inc.

Britton JW, H. J., Frey LC. (2016a). Appendix 1. the scientific basis of eeg: Neurophysiology of eeg generation in the brain. *Electroencephalography (eeg): An introductory text and atlas of normal and abnormal findings in adults, children, and infants* (p. 81). American Epilepsy Society.

Britton JW, H. J., Frey LC. (2016b). The background. *Electroencephalography (eeg): An introductory text and atlas of normal and abnormal findings in adults, children, and infants* (pp. 9–12). American Epilepsy Society.

Britton JW, H. J., Frey LC. (2016c). *Electroencephalography (eeg): An introductory text and atlas of normal and abnormal findings in adults, children, and infants*. American Epilepsy Society.

Britton JW, H. J., Frey LC. (2016d). Introduction. *Electroencephalography (eeg): An introductory text and atlas of normal and abnormal findings in adults, children, and infants* (pp. 1–6). American Epilepsy Society.

Burros, C. S., Frigo, M., Johnson, S. G., Pueschel, M., & Selesnick, I. (2012). *Fast fourier transforms*. Connexions.

Casson, A. J. (2019). Wearable eeg and beyond. *Biomedical engineering letters*, *9*(1), 53–71.

Chi, Y. M., Jung, T.-P., & Cauwenberghs, G. (2010). Dry-contact and noncontact biopotential electrodes: Methodological review. *IEEE reviews in biomedical engineering*, *3*, 106–119.

Craig, D. A., & Nguyen, H. (2007). Adaptive eeg thought pattern classifier for advanced wheelchair control. *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2544–2547.

Dilmegani, C. (2021). *Dark side of neural networks explained*. Retrieved January 1, 2021, from https://research.aimultiple.com/how-neural-networks-work/

Eaton, E. (2017). *Introduction to machine learning*. Retrieved October 2, 2017, from https://www.seas.upenn.edu/~cis519/fall2017/ (accessed: 17-02-2022)

Erickson, J. (2021). *Math methods - discrete fourier transform: Intro theory*. https://erickson.academic.wlu.edu/teaching/mathmethods_w2021/ (accessed: 10-02-2022)

Farnsworth, B. (2019). *Eeg headset prices – an overview of 15+ eeg devices*. Retrieved July 17, 2019, from https://imotions.com/blog/eeg-headset-prices/

Fast fourier transformation fft - basics. (n.d.). https://www.nti-audio.com/en/support/know-how/fast-fourier-transform-fft (accessed: 10-01-2022)

Flach, P. (2012a). Discriminative learning by optimising conditional likelihood. *Machine learning: The art and science of algorithms that make sense of data* (pp. 282–286). Cambridge University Press.

Flach, P. (2012b). Support vector machines. *Machine learning: The art and science of algorithms that make sense of data* (pp. 211–). Cambridge University Press.

Garcia-Moreno, F. M., Bermudez-Edo, M., Garrido, J. L., & Rodrıguez-Fórtiz, M. J. (2020a). Reducing response time in motor imagery using a headband and deep learning. *Sensors*, *20*(23), 6730.

Garcia-Moreno, F. M., Bermudez-Edo, M., Garrido, J. L., & Rodrıguez-Fórtiz, M. J. (2020b). Reducing response time in motor imagery using a headband and deep learning. *Sensors*, *20*(23), 6730.

Géron, A. (2017). *Hands-on machine learning with scikit-learn and tensorflow : Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. https://books.google.co.in/books?id=Np9SDQAAQBAJ

Gurney, K. (1997). *An introduction to neural networks*. UCL Press.

Husain, A., Horn, G., & Jacobson, M. (2003). Non-convulsive status epilepticus: Usefulness of clinical features in selecting patients for urgent eeg. *Journal of Neurology, Neurosurgery & Psychiatry*, *74*(2), 189–191.

Kalat, J. W. (2004). The nerve impulse. *Biological psychology* (pp. 39–44). Thomson Wadsworth.

Kostov, A., & Polak, M. (2000). Parallel man-machine training in development of eeg-based cursor control. *IEEE Transactions on Rehabilitation Engineering*, *8*(2), 203–205.

Kübler, A., & Mattia, D. (2016). Brain–computer interface based solutions for end-users with severe communication disorders. *The neurology of conciousness* (pp. 217–240). Elsevier.

Lu, N., Yin, T., & Jing, X. (2019). A temporal convolution network solution for eeg motor imagery classification. *2019 IEEE 19th International Conference on Bioinformatics and Bioengineering (BIBE)*, 796–799.

Mandira, P. (2016). *Socratic qa logo: What is the resting potential for a neuron?* Retrieved September 18, 2016, from https://socratic.org/questions/what-is-the-resting-potential-for-a-neuron

Mathworks:support vector machine (svm). (n.d.). https://nl.mathworks.com/discovery/support-vector-machine.html (accessed: 17-02-2022)

Mynuddin, M. (2015). Designing a low- pass fir digital filter by using hamming window and blackman window technique. *Science Journal of Circuits, Systems and Signal Processing*, *4*, 9. https://doi.org/10.11648/j.cssp.20150402.11

Nijholt, A. (2016). The future of brain-computer interfacing (keynote paper). *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*, 156–161.

Open bci - all-in-one biosensing r&d bundle. (n.d.). https://shop.openbci.com/collections/frontpage/products/all-in-one-biosensing-r-d-bundle?variant=13043151994952 (accessed: 10-01-2022)

Padfield, N., Zabalza, J., Zhao, H., Masero, V., & Ren, J. (2019a). Eeg-based brain-computer interfaces using motor-imagery: Techniques and challenges. *Sensors*, *19*(6), 1423.

Padfield, N., Zabalza, J., Zhao, H., Masero, V., & Ren, J. (2019b). Eeg-based brain-computer interfaces using motor-imagery: Techniques and challenges. *Sensors*, *19*(6), 1423.

Pham, D. T. (2001). Joint approximate diagonalization of positive definite hermitian matrices. *SIAM Journal on Matrix Analysis and Applications*, *22*(4), 1136–1152.

Portillo-Lara, R., Tahirbegi, B., Chapman, C. A. R., Goding, J. A., & Green, R. A. (2021). Mind the gap: State-of-the-art technologies and applications for eeg-based brain–computer interfaces. *APL Bioengineering*, *5*(3), 031507. https://doi.org/10.1063/5.0047237

Ramantani, G., Maillard, L., & Koessler, L. (2016). Correlation of invasive eeg and scalp eeg. *Seizure*, *41*, 196–200. https://doi.org/https://doi.org/10.1016/j.seizure.2016.05.018

Remy, P. (2020). Temporal convolutional networks for keras.

Roc, A., Pillette, L., Mladenovic, J., Benaroch, C., N'Kaoua, B., Jeunet, C., & Lotte, F. (2021). A review of user training methods in brain computer interfaces based on mental tasks. *Journal of Neural Engineering*, *18*(1), 011002.

Rojas, G., Alvarez, C., Montoya Moya, C., de la Iglesia Vaya, M., Cisternas, J., & Gálvez, M. (2018). Study of resting-state functional connectivity networks using eeg electrodes position as seed. *Frontiers in Neuroscience*, *12*. https://doi.org/10.3389/fnins.2018.00235

Romero, D. (2020). Lecture 5.4 - cnns for sequential data.

Sayad, S. (2022). *Logistic regression*. https://www.saedsayad.com/logistic_regression.htm (accessed: 12-02-2022)

Shad, E. H. T., Molinas, M., & Ytterdal, T. (2020). Impedance and noise of passive and active dry eeg electrodes: A review. *IEEE Sensors Journal*, *20*(24), 14565–14577.

Shukla, P. K., & Kumar Chaurasiya, R. (2018). An experimental analysis of motor imagery eeg signals using feature extraction and classification methodologies. *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*, 796–799. https://doi.org/10.1109/GUCON.2018.8675032

Sirico, F., Romano, V., Sacco, A. M., Belviso, I., Didonna, V., Nurzynska, D., Castaldo, C., Palermi, S., Sannino, G., Della Valle, E., et al. (2020). Effect of video observation and motor imagery on simple reaction time in cadet pilots. *Journal of functional morphology and kinesiology*, *5*(4), 89.

Sreeja, S., Rabha, J., Nagarjuna, K., Samanta, D., Mitra, P., & Sarma, M. (2017). Motor imagery eeg signal processing and classification using machine learning approach. *2017 International Conference on New Trends in Computing Sciences (ICTCS)*, 61–66.

Taspinar, A. (2018). *A guide for using the wavelet transform in machine learning*. https://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/ (accessed: 8-02-2022)

Trachel, R., Gramfort, A., Barachant, A., Brunner, C., & King, J.-R. (2021). *Mne decoding csp*. https://mne.tools/stable/generated/mne.decoding.CSP.html (accessed: 10-02-2022)

Verma, N. K., Rao, L. S. V. S., & Sharma, S. K. (2014). Motor imagery eeg signal classification on dwt and crosscorrelated signal features. *2014 9th International Conference on Industrial and Information Systems (ICIIS)*, 1–6. https://doi.org/10.1109/ICIINFS.2014.7036473

Vidal, J. J. (1973). Toward direct brain-computer communication [PMID: 4583653]. *Annual Review of Biophysics and Bioengineering*, *2*(1), 157–180. https://doi.org/10.1146/annurev.bb.02.060173.001105

Walker, J. (2019). Haar wavelets. *A primer on wavelets and their scientific applications* (pp. 1–28). CRC Press. https://books.google.nl/books?id=F-9VGmsdbdYC

Wang, P., Jiang, A., Liu, X., Shang, J., & Zhang, L. (2018). Lstm-based eeg classification in motor imagery tasks. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *26*(11), 2086–2095. https://doi.org/10.1109/TNSRE.2018.2876129

Zhang, G., Davoodnia, V., Sepas-Moghaddam, A., Zhang, Y., & Etemad, A. (2020). Classification of hand movements from eeg using a deep attention-based lstm network. *IEEE Sensors Journal*, *20*(6), 3113–3122. https://doi.org/10.1109/JSEN.2019.2956998

A. Appendix

A.1 Initial hyperparameters

Variables	**Constant Values**
Window size	512 ms
Skiptime start	800 ms
Skiptime end	200 ms
Preprocessing	CSP + DWT + features

Table A1: The initial default values that are used for the first subtest are shown. The constant values in the table are defined in milliseconds worth of data.

Classifier	Parameter	Constant Value
LR	Penalty	l2
	Solver	lbfgs, saga
	Learning Rate	1, 0.1, 0.01, 0.001
	Maximum Iterations	100, 500, 1000
SVM	Kernel	rbf
	Learning Rate	1, 0.1, 0.01, 0.001
	Maximum Iterations	100, 500, 1000
TCN	Epochs	50
	Learning Rate	1, 0.1, 0.01, 0.001
	Batch Size	100
	Optimizer	Adam
	Dilations	6
	Kernel Size	4
	Stacks	1
	Normalization	Layer
	Dropout Rate	0.001
	Filter	3
LSTM	Epochs	100
	Learning Rate	0.001
	Batch Size	64
	Optimizer	Adam
	Normalization	Batch
	LSTM-cell Dropout Rate	0.4
	Loss Function	Crossentropy

Table A2: The initial hyperparameters used for the first sub-study are shown. In the case of multiple values, they were all considered individually with only the model with the highest validation accuracy being included in the results tables.

A.2 CSP settings

Classifier	Regularization	Train Accuracy	Validation Accuracy	Test Accuracy
SVM	None	59.83	49.70	54.27
	Empirical	55.59	51.50	55.96
	Diagonal_Fixed	45.67	41.36	36.53
	Ledoit_Wolf	55.05	50.30	54.79
	PCA	51.00	45.14	51.30
LR	None	61.53	49.53	54.53
	Empirical	61.53	49.53	54.53
	Diagonal_Fixed	58.41	38.87	50.78
	Ledoit_wolf	64.24	48.58	53.37
	PCA	58.96	43.59	53.24

Table A3: The regularization of CSP is tested with two classifiers. Since the LR gives the same result for two regularizations, there is decided to take the empirical regularization, as that is also the best regularization with the SVM. The logic behind this decision is that the preprocessing changes the data in a form that contains more informative data for the classifiers, and there should be some overlap in what is useful for the classifiers. This can also be seen in the results of the different preprocessing techniques in Table 8.3, where the same preprocessing resulted in the highest validation accuracy for all classifiers.

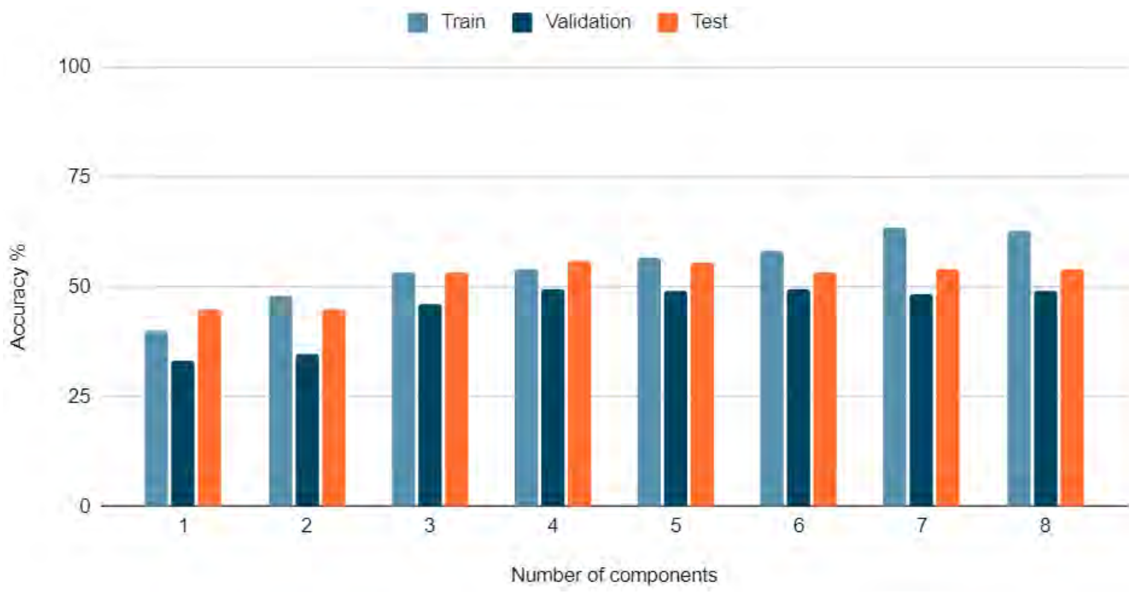


Figure A1: The accuracies of LR are shown to examine the effects of the amount of CSP components. It can be seen that the validation accuracy is optimal with four components.

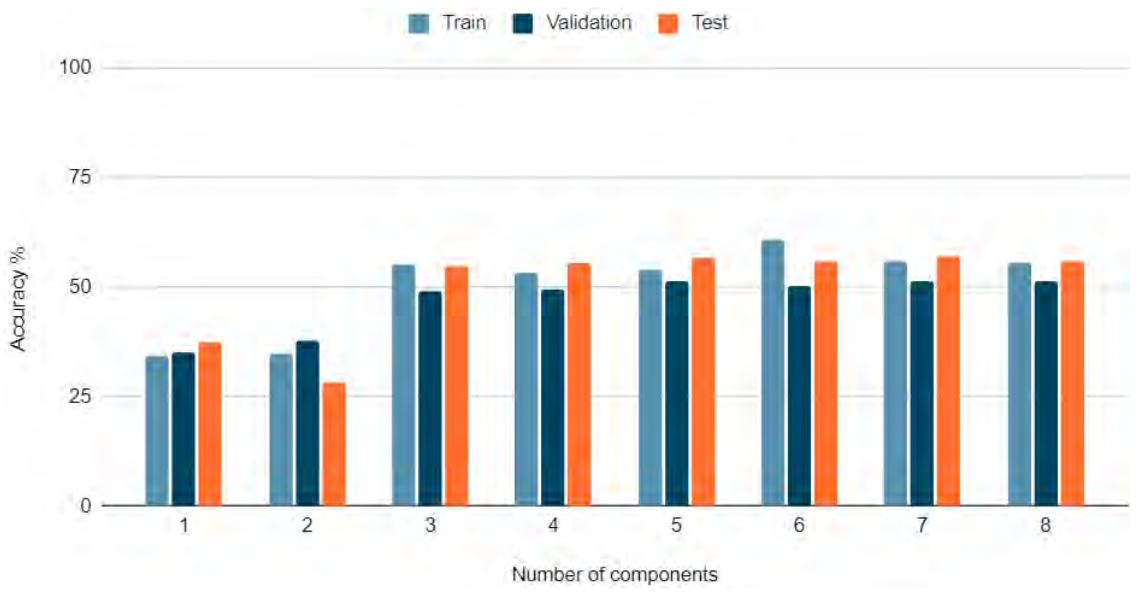


Figure A2: The accuracies of SVM are shown to examine the effects of the amount of CSP components. It can be seen that the validation accuracy is optimal with five components. However, the difference with four components is very small and therefore there is decided to use the simplest model, that also reached the highest accuracy with LR.

A.3 Hyperparameters of classifiers

Learning Rate	Train Accuracy	Validation Accuracy
0.01	56.39	47.12
0.01	56.53	46.52
0.01	56.79	46.26
0.01	49.55	46.00
0.01	56.71	44.97
0.01	53.53	44.63
0.01	55.63	44.54
0.01	55.59	44.45
0.01	52.01	43.77
0.01	54.72	43.68

Table A4: Here you can see the top 10 models of the TCN from the earlier preprocessor tuning. This is step 0 in Table 8.4 with the corresponding results from Table 8.3. It can be noticed that all learning rates are the same, while there were four different learning rates tested. Therefore, this learning rate is also used as a constant in the subsequent experiments.

Solver	Penalty	Learning Rate	Maximum Iterations	Train Accuracy	Validation Accuracy	Test Accuracy
lbfgs	none	0.01	500	58.31	46.78	54.27
	l2	1	1000	58.31	46.69	54.79
newton-cg	none	0.001	500	57.80	46.35	54.40
	l2	1	500	58.27	46.69	54.66
sag	none	0.001	500	57.40	46.69	53.89
	l2	0.1	1000	57.33	47.12	54.40
saga	none	0.1	500	56.89	47.55	54.66
	l2	1	500	56.82	47.55	54.53
	l1	1	1000	57.29	47.29	54.66
	elasticnet	1	1000	57.40	47.21	54.53

Table A5: The results for determining the best solver and penalty for the LR classifier are shown. There are two models that have the highest validation accuracy. The highlighted line without penalty is preferred, as it has a higher accuracy on the training set.

Kernel	Gamma	Learning Rate	Maximum Iterations	Train Accuracy	Validation Accuracy	Test Accuracy
linear	scale	0.01	1000	55.70	48.75	53.24
	auto	0.01	1000	55.70	48.75	53.24
poly	scale	1	1000	59.65	48.50	56.09
	auto	0.1	1000	65.47	46.00	51.81
rbf	scale	1	1000	58.89	49.01	56.09
	auto	1	1000	70.50	47.55	55.96
sigmoid	scale	0.1	1000	48.72	45.57	56.35
	auto	1	1000	39.16	40.93	34.84

Table A6: The results of the SVM classifier are shown with different values for the kernel and gamma. The model with the highest validation accuracy is highlighted.

Learning Rate	Iterations	Train Accuracy	Validation Accuracy	Test Accuracy
2	250	55.95	47.29	55.18
	500	56.89	47.55	54.66
	750	57.37	47.21	54.40
	1000	57.37	46.60	53.89
	1250	57.73	46.52	54.15
	1500	57.84	46.09	53.89
1	250	55.95	47.29	55.18
	500	56.89	47.55	54.66
	750	57.33	47.21	54.40
	1000	57.37	46.60	53.89
	1250	57.73	46.52	54.15
	1500	57.84	46.09	53.89
0.1	250	55.95	47.21	55.18
	500	56.89	47.55	54.66
	750	57.33	47.21	54.40
	1000	57.37	46.69	53.89
	1250	57.73	46.52	54.15
	1500	57.84	46.00	53.89
0.01	250	55.92	47.21	55.18
	500	56.89	47.55	54.66
	750	57.33	47.21	54.40
	1000	57.37	46.60	54.02
	1250	57.73	46.52	54.15
	1500	57.84	46.00	53.89
0.001	250	55.95	47.29	55.18
	500	56.89	47.55	54.66
	750	57.33	47.21	54.40
	1000	57.37	46.69	54.02
	1250	57.73	46.52	54.15
	1500	57.84	46.00	53.89

Table A7: The results of the LR can be viewed with the different learning rates and iterations. This shows that 500 iterations gives the highest and same results for each learning rate. To determine the learning rate, successive learning rates with the same number of iterations are compared. Light green indicates a higher accuracy and light red indicates a lower accuracy compared to any of the successive learning rates with the same number of iterations. With these small differences, a learning rate of ”1” is selected because it has two light green boxes and the accuracy decreases slightly after this point.

Learning Rate	**Iterations**	**Train Accuracy**	**Validation Accuracy**	**Test Accuracy**
2	250	32.9	33.36	33.42
	500	32.94	33.19	33.42
	750	61.13	48.07	54.4
	1000	61.24	48.93	55.31
	1250	61.38	49.10	55.31
	1500	61.35	49.10	55.57
1	250	32.9	27.17	37.31
	500	32.94	32.42	39.51
	750	61.13	48.93	56.09
	1000	61.24	49.01	56.09
	1250	61.38	49.01	56.22
	1500	61.35	49.01	56.22
0.1	250	35.47	27.34	39.38
	500	35.61	31.38	39.77
	750	50.85	45.66	50.65
	1000	52.12	48.24	58.68
	1250	52.12	48.24	58.68
	1500	52.12	48.24	58.68
0.01	250	35.5	28.89	38.21
	500	36.95	33.02	40.54
	750	38.00	34.74	37.05
	1000	34.46	33.28	33.16
	1250	34.46	33.28	33.16
	1500	34.46	33.28	33.16
0.001	250	35.03	28.29	37.95
	500	36.66	32.76	39.51
	750	37.42	34.57	36.79
	1000	34.46	33.28	33.16
	1250	34.46	33.28	33.16
	1500	34.46	33.28	33.16

Table A8: The results of the SVM are shown with the different learning rates and iterations. This shows that a learning rate of 2 with 1250 iterations results in the highest validation and training accuracy.

Kernel Size	Dilations	Stacks	Receptive Field	Validation Accuracy
3	6	2	253	48.50
4	5	1	94	47.82
3	4	2	61	46.95
4	4	1	46	46.84

Table A9: Four models are shown with a varied receptive field of the TCN. There is varied between 4, 5 or 6 dilations, 2, 3 or 4 kernels and 1 or 2 stacks. By grouping in a different order with the kernel size, dilations and stacks, the optimal models appear to have a different combination of hyperparameters. Those different models are shown. The validation accuracy here is the average of all TCN models with the hyperparameters described. It is notable that a larger receptive field, that takes more features into consideration, results in a higher average validation accuracy.

Dilations	Kernel Size	Stacks	Receptive Field	Train Accuracy	Validation Accuracy
4	3	2	61	52,70	50,39
5	2	2	63	53,24	50,13
5	3	2	125	52,08	49,44
6	3	2	253	53,75	49,36
4	3	2	61	49,19	49,27
5	4	1	94	55,05	49,18
5	3	2	125	53,09	49,18
5	2	2	63	52,15	49,10
5	2	1	32	52,44	49,10
5	2	2	63	51,97	49,10

Table A10: The top 10 models of the TCN receptive field experiment are shown. It can be noticed that most of the models do not consider all 144 features. The explanation for the contradiction with the results from Table A9 is that only a part of the features proves to be useful, which causes the average accuracy to be lower when not all features are considered. Furthermore, a possible explanation is that there is an overfit when all the features are considered. This explains that most of the models in the top 10 do not include all features. Therefore, it was decided to adopt the hyperparameters of the best model. It has been remembered for this decision that the training of the TCN must eventually run several times because of the initialization of the random weights.

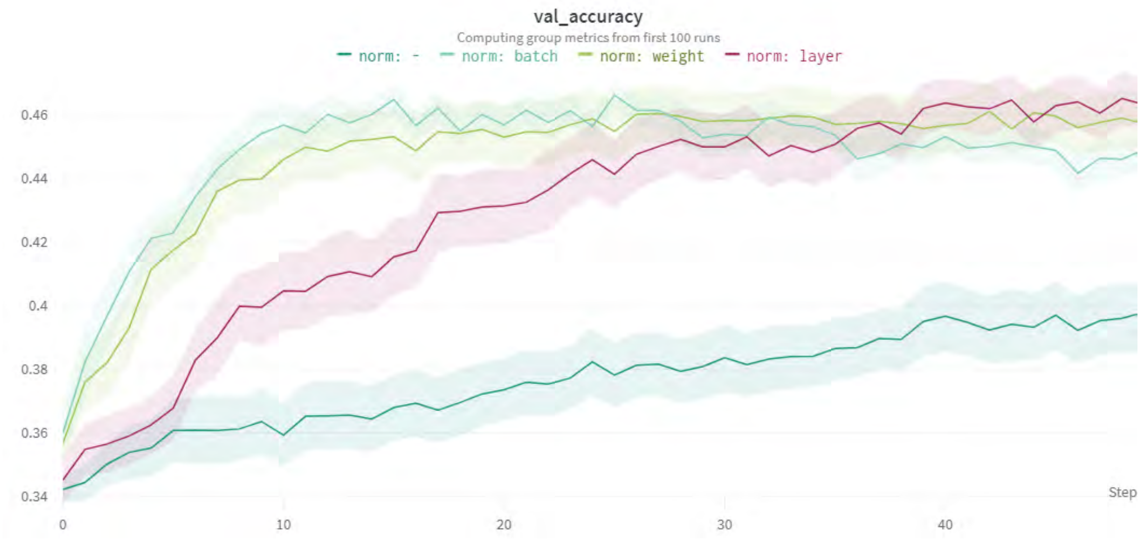


Figure A3: The results of the different normalizers of the TCN are shown. It was decided to use layer normalization as best normalizer, because it shows a greater improvement at the end and therefore has a higher potential for a better model if more training epochs are used.

Batch-size	**Learning Rate**	**Optimizer**	**Train Accuracy**	**Validation Accuracy**
10	0.01	sgd	51,32	50,82
10	0.01	sgd	50,34	50,04
300	0.1	sgd	51,14	49,87
10	0.01	sgd	51,07	49,79
100	0.1	sgd	45,02	49,61
50	0.01	sgd	51,32	49,27
50	0.1	sgd	49,37	49,10
10	0.001	sgd	51,39	49,01
50	0.01	Adam	50,63	49,01
100	0.1	sgd	51,25	48,93

Table A11: The top 10 models of the results of the TCN learning related setting are shown. It can be concluded that optimizer 'sgd', with learning rate 0.01 and batch size 10 are the most frequent models in the top 10 and are therefore the best hyperparameters.