VRIJE
UNIVERSITEIT
BRUSSEL

Master thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in de Ingenieurswetenschappen: Computerwetenschappen

# TRANSFER LEARNING IN BRAIN-COMPUTER INTERFACES: LANGUAGE-PRETRAINED TRANSFORMERS FOR CLASSIFYING ELECTROENCEPHALOGRAPHY

Wolf De Wulf

2021-2022

Promotor(s): Prof. Dr. Kevin De Pauw & Prof. Dr. Geraint Wiggins

Advisor(s): Arnau Dillen

**Science and Bio-Engineering Sciences**

VRIJE
UNIVERSITEIT
BRUSSEL

Proefschrift ingediend met het oog op het behalen van de graad van Master of Science in de Ingenieurswetenschappen: Computerwetenschappen

# KENNISOVERDRACHT IN BREIN-COMPUTER INTERFACES: TAAL-VOORGETRAINDE TRANSFORMERS VOOR HET CLASSIFICEREN VAN ELECTRO-ENCEFALOGRAFIE

Wolf De Wulf

2021-2022

Promotor(s): Prof. Dr. Kevin De Pauw & Prof. Dr. Geraint Wiggins

Advisor(s): Arnau Dillen

**Science and Bio-Engineering Sciences**

# Abstract

The capability of a language-pretrained transformer-like model to transfer to the classification of electroencephalography, i.e., electrical brain waves, is analysed. On first sight, these two domains seem unrelated. However, recent work has found language-pretrained transformers to be surprisingly transferable to a variety of non-language modalities, thereby, making this analysis worthwhile. Brain-computer interfaces would benefit greatly from a set of pretrained models that, similarly to how this is done in natural language processing, can easily be plugged in in a variety of contexts. The scarcity of labelled brain signal data and its nonstationary nature make the idea of using pretrained models from a different domain even more attractive. A study that verifies the possibility of transfer between these two domains might even result in insight regarding transferable characteristics within the domains. This would be particularly interesting for the domain of electroencephalography, as in that case, transferable characteristics are complex to understand and currently relatively unknown. This phenomenon is reflected in the design of the deep learning models that brain-computer interfaces currently implement. Through a background study, relevant foundational knowledge from the four research areas inherent to the research question is collected. The research areas in question are neuroimaging, brain-computer interfaces, transfer learning, and natural language processing. In addition, preceding work that is explicitly related to the research question is discussed. A methodology is designed specifically to be able to perform a set of empirical evaluations that can shine light on the research question. Such a set of empirical evaluations is established and performed, the results reported and discussed. Interestingly, the empirical evaluations suggest that positive transfer from the natural language domain to that of electroencephalography is indeed possible. A number of hypotheses regarding the language-pretraining and the depth with which it can transfer to electroencephalography are established. While not falsified, there is no statistical basis to claim that they hold. Additional experimental evaluations are required to verify them. Moreover, a multitude of new questions arise, making this thesis a starting point for a variety of successive work.

# Acknowledgments

Throughout my years as a computer science student I have received a great deal of guidance, advice, and support.

First and foremost, I would like to thank my supervisor, Professor Geraint Wiggins, whose approach at teaching and supporting students is one that I have come to appreciate deeply. While compassionate, relatable, and kind at certain times, you remain correct and strict at others. Apart from what you taught me in your courses, which I eagerly followed, you taught me the importance of completeness and correctness in science. Doing so, sometimes through something as simple as a remark about my pronunciation of the English language. You immediately saw what I needed in a supervisor and I am very grateful for the way you treated me as a person interested in science rather than as a student that had to be strictly monitored.

Secondly, I would like to thank Arnau Dillen. Arnau is working on his PhD on brain-computer interfaces and supervised my thesis as an assistant to Professor Wiggins. Although I pride myself on my independence when working on projects like this, I found that Arnau was always available to help. He taught me a lot about the area in which he is so passionately interested and thus transferred this interest to me.

Thirdly, Professor Bart Bogaerts, who supervised my bachelor's thesis, merits thanks as well. It was he who first introduced me to the academical and who taught me what scientific contribution is and how it is formed, all the while also treating me not as a mere student, but as an individual.

In addition, I want to thank my parents and my partner for their never-ending encouragement towards my search for knowledge and growth. From my parents, I inherited my interest and lust for science as well as a sceptical view on life. In my opinion, these two make a very good combination which has brought me to where I am now. I am therefore also very grateful to have a partner who shares these qualities and who is willing - or at least pretends - to listen to my ramblings.

Lastly, I would like to thank the Vrije Universiteit Brussel for educating me in its beautiful and open-minded way. While I will be leaving Brussels next year, the ideas and behaviour that you have instilled in me will never be lost.

The resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

A Brain-Computer Interface (BCI) is a device that digitises the neurophysiological. By interpreting brain-generated signals, BCIs are to perform a variety of tasks in both a clinical and a daily-life context. Recent advances in BCI research have allowed for clinical applications such as early epileptic seizures prediction (Liang et al., 2010; Young et al., 2011; S.-K. Lin et al., 2018), spellers for the paralysed (Sellers & Donchin, 2006; Nijboer et al., 2008; Townsend et al., 2010), post-stroke neurorehabilitation (Buch et al., 2008; Ramos-Murguialday et al., 2013; Mrachacz-Kersting et al., 2016), and control of prosthetics (Müller-Putz & Pfurtscheller, 2008; Gao et al., 2017; AL-Quraishi et al., 2018), as well as daily-life applications such as intent-recognition for smart-living (X. Zhang et al., 2017, 2019; Belkacem et al., 2020), control systems for gaming and virtual reality (Lalor et al., 2005; van de Laar et al., 2013; Marshall et al., 2013), and vehicle and robot control (Redrovan & Kim, 2018; Aznan et al., 2019; W. Lu et al., 2020). Nevertheless, aside from a number of very specific cases, efficient and reliable BCIs are still far away. Specifically, the machine learning methods that BCIs implement to interpret neurological activity have yet to face numerous challenges. Inter- and intra-person variability as well as data scarcity make decoding neuroimaging modalities a complex matter. Luckily, there are other research areas that deal with their respective instances of these issues such that inspiration can be found in the most successful solutions. Particularly interesting and novel in the research area of BCIs is the self-supervised learning paradigm, originally cross-pollinated from research areas that have access to vast quantities of unlabelled data such as Computer Vision (CV; Chen et al., 2015; Grill et al., 2020; Henaff, 2020; van den Oord et al., 2019) and Natural Language Processing (NLP; Peters et al., 2018; Radford & Narasimhan, 2018; Radford et al., 2019; Devlin et al., 2019; Raffel et al., 2020). Self-supervised learning is an unsupervised learning paradigm that consists of training models using large unlabelled datasets, usually to afterwards finetune them to specific tasks using smaller labelled datasets. The nontransferable nature of the current BCI machine learning state-of-the-art has resulted in variable performances (Ahn & Jun, 2015; Lotte et al., 2018; Sannelli et al., 2019), suggesting that research into self-supervised learning approaches could be fruitful. Various recent works have started laying the foundation for such research (Banville et al., 2019, 2021; Kostas et al., 2021). The work that is presented in this thesis is part of such endeavours.

Transfer learning in BCIs has mostly consisted of supervised pretraining followed by supervised finetuning. Pretraining would then consist of initialising the weights of a deep learning model by training it on the data of a number of participants (Dose et al., 2018; Fahimi et al., 2019) or even the runs of a single participant (Schwemmer et al., 2018). However, in such a transfer learning paradigm, the pretraining phase aims to adjust weights to fit only a small

subset of the target domain. In contrast, the self-supervised learning paradigm is meant to generalise, adjusting weights towards fitting the whole of the target domain. In some cases, pretraining happens in a completely different domain, for example, image recognition (G. Xu et al., 2019). Such research can be interesting as the success or failure of such knowledge transfer can provide insight and inspiration with regards to BCI transfer learning. In that light, the work that is presented in this thesis consists of a study of BCI transfer learning, investigating the applicability of a typical language-pretrained NLP model for the classification of a specific neuroimaging modality that has been successful in BCIs, i.e., electroencephalography (EEG). Recent work by K. Lu et al. (2021) has found language-pretrained transformers (Vaswani et al., 2017) to be surprisingly transferable to a variety of non-language modalities such as the MNIST handwritten digit benchmark (L. Deng, 2012), and protein sequence tasks such as remote homology detection (Rao et al., 2019). Therefore, the work that is presented in this thesis is to result in two types of insight. On the one hand, inspiration can be gained towards BCI transfer learning. On the other hand, K. Lu et al.'s claim that language-pretrained transformers can be seen as universal computation engines is tested towards the complex task of classifying EEG.

Explicitly, the research question studied in this thesis reads as follows: "How and to what degree can a language-pretrained transformer transfer to the classification of EEG and does the language-pretraining influence performance?" Insight regarding the answer to this question is gained through a range of empirical evaluations. To stay inside the scope of the encompassing research, an attempt is made to finetune the second generation Generative Pretrained Transformer (GPT2; Radford et al., 2019) to an EEG motor imagery classification task, namely, the Graz dataset A (Brunner et al., 2008). The empirical evaluations are split into two groups. While the first group uses raw EEG time-series data, the second group uses time-frequency features extracted from the time-series data. To investigate whether or not and to what degree language-pretraining is beneficial, the performance of a frozen language-pretrained instance of GPT2 is compared to that of an unfrozen randomly initialised instance. Additionally, more nuanced insight regarding the role of the language-pretraining is gained by gradually unfreezing the layers of the language-pretrained instance of GPT2. The results suggest that positive transfer is indeed possible. This means that the unsupervised pretraining phase of GPT2 manages to capture structure in language that is applicable to the classification of EEG. Additionally, a number of hypotheses regarding the language-pretraining and the depth with which it can transfer to EEG are established. While the results of the empirical evaluations do not falsify them, there is no statistical basis to claim that they hold. Verifying them will require additional experimental evaluations.

The structure of this thesis is as follows. Chapter 2 gives background knowledge regarding the four topics that are inherent to the research question: neuroimaging, brain-computer interfaces, transfer learning, and natural language processing. Chapter 3 discusses related work, relating the research question to existing work by discussing similarities and differences. Chapter 4 goes over the methodology. It explains the used dataset, the model, its training, and its evaluation. Subsequently, in Chapter 5, a set of empirical evaluations is established and its results are reported and discussed. Lastly, Chapter 6 concludes the thesis with a discussion on the conclusions that can be drawn as well as possible future work.

# Chapter 2

# Background

## 2.1 Introduction

The work that is presented in this thesis relates to four major areas of research. Firstly, an understanding of the different neuroimaging modalities that are currently used in machine learning research is necessary. Specifically, their advantages and disadvantages in that context should be discussed. Secondly, a thorough understanding of the history of BCI research and its current state-of-the-art is required. The end-goal of the research that encompasses this thesis consists of creating a BCI and thus understanding what these are and how they work is crucial. Thirdly, the essence of this thesis is to try to transfer knowledge between two different domains. Therefore, the concept of transfer learning needs to be introduced. Lastly, readers should have extensive knowledge of natural language processing. Understanding what might be the cause of the final results will only be possible if the ideas and reasoning behind the state-of-the-art NLP models are known.

The following sections lay a foundational background for all four of these areas of research. It is important to note that throughout this background chapter, elementary deep learning concepts are assumed to be known. For an introduction to deep learning, the reader is referred to the work of Goodfellow et al. (2015).

## 2.2 Neuroimaging

### 2.2.1 Introduction

While there exist a range of neuroimaging modalities, the focus of this section lies on neuroimaging modalities that are used in the context of machine learning and BCIs. A distinction is made between invasive and noninvasive neuroimaging modalities. The main trade-off between these two is that of risk and ease-of-use versus information-to-noise ratio (Steyrl et al., 2016; Waldert, 2016). Depending on the use case, invasive or noninvasive modalities can be preferred. The research that encompasses this thesis consists of designing a BCI that is mobile, user-friendly, and requires little to no installation time. In such a context, noninvasive modalities are the obvious preference. While the research question of this thesis does not innately exclude invasive modalities, the focus lies, as a start, only on noninvasive modalities.

The following sections give a general overview of noninvasive neuroimaging modalities that are used in BCIs. The focus lies on EEG. It is compared to MEG, fMRI, and NIRS. There

Table 2.1: EEG frequency bands. Data from Kawala-Sterniuk et al. (2021).

| Brainwave | Frequency band | Mental condition |
|---|---|---|
| Delta | 0 Hz to 4 Hz | State of deep sleep, when there is no focus, the person is totally absent, unconscious. |
| Theta | 4 Hz to 8 Hz | Deep relaxation, internal focus, meditation, intuition access to unconscious material such as imaging, fantasy, dreaming. |
| Low Alpha | 8 Hz to 10 Hz | Wakeful relaxation, consciousness, awareness without attention or concentration, good mood, calmness |
| High Alpha | 10 Hz to 12 Hz | Increased self-awareness and focus, learning of new information. |
| Low Beta | 12 Hz to 18 Hz | Active thinking, active attention, focus towards problem solving, judgement and decision making. |
| High Beta | 18 Hz to 30 Hz | Engagement in mental activity, also alertness and agitation. |
| Low Gamma | 30 Hz to 50 Hz | Cognitive processing, senses, intelligence, compassion, self-control. |
| High Gamma | 50 Hz to 70 Hz | Cognitive tasks: memory, hearing, reading and speaking |

exist a range of other neuroimaging modalities that are relevant for BCIs but that are not mentioned here. Examples are positron emission tomography (PET; Bailey et al., 2005) and functional ultrasound imaging (fUS; Macé et al., 2011). However, none of these are as successful or prevalent as the above-mentioned, which is why they are not discussed here. For a more detailed review, the reader is referred to the work of Ramadan and Vasilakos (2017).

### 2.2.2 Electroencephalography

In the context of BCI research, EEG is the most commonly implemented neuroimaging modality. Its success is due to its desirable traits such as its relatively low-cost and noninvasiveness. The brain's electrical activity consists of the sum of discharges that originate from neurons firing. The potential of a single neuron firing would be measurable in the millivolt range. However, the activity of a large population of neurons results in potentials that cancel each other out. Therefore, the voltage fluctuations that can be observed in brain tissue are measured in the microvolt range. Luckily, large groups of neurons often work together, resulting in synchronised oscillatory fluctuations that spread over the brain and even penetrate the skull. EEG measures exactly these fluctuations using electrodes spread out over the scalp. For a biological signal to be relevant with respect to the site at which it is recorded, it must be expressed as the difference between recordings at two sites, indicating nonzero amplitudes. Additionally, recording any electrical signal always requires some type of baseline, i.e., a ground. When recording EEG, a ground electrode is used to record a 0-amplitude ground signal (its amplitude is zero because it is recorded against itself). All other recordings are always recorded against the ground signal, which are thereby guaranteed to be nonzero. Hence, when recording EEG at a certain electrode site, it is recorded against the ground signal. Moreover, to make that recording relevant with respect to where it is recorded, another EEG signal that is also recorded against the ground signal but on a different nearby site, must be subtracted. Doing so is always done using a reference electrode, which is subtracted from the recordings from all other electrodes. EEG amplitudes generally range from -100 to +100 microvolt. Research has shown that EEG signals can be categorised into frequency bands, each of which can be linked to particular functions. Furthermore, irregularities in these frequencies can be used for the diagnosis of certain disorders (Fisch, 2003). Table 2.1 contains the frequency bands of EEG signals and brief descriptions.

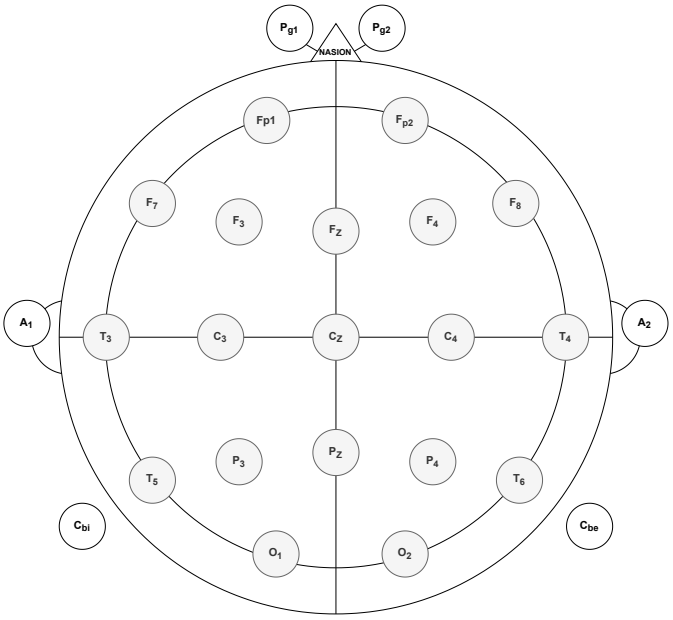
To record EEG, a number of electrodes are attached to the scalp. Typically, the number of electrodes ranges from 1 to 256, however, recent apparatus can go up to 1024 electrodes[1]. Numerous types of electrodes are used. Firstly, a distinction is made between *wet* and *dry* electrodes. The former of which is the most common. However, since they require applying

[1] `https://www.gtec.at/product/g-pangolin-electrodes/`

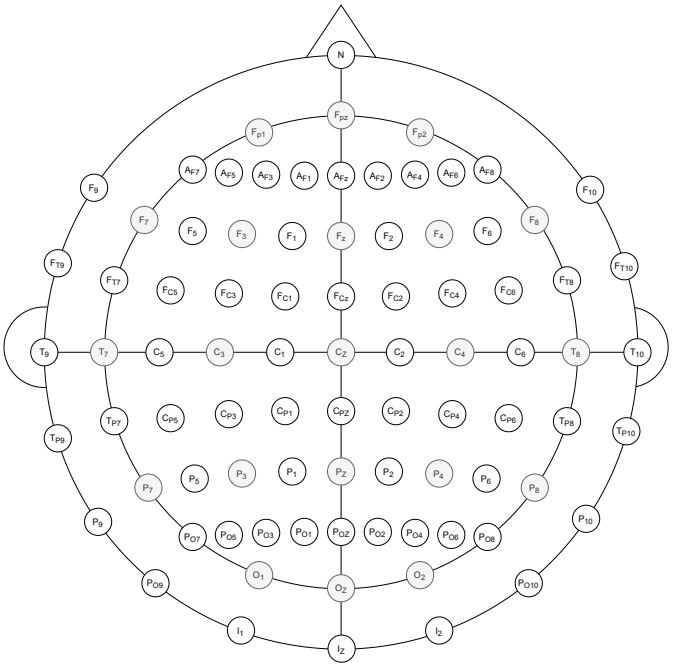
an electrolytic gel on the scalp, which can take a long time and must be done correctly and regularly, they are most often only used in research labs. Conversely, *dry* electrodes do not require the application of any gel, which comes at the cost of signal quality. Nevertheless, these dry electrodes are very desirable for the final product of BCI research, hence, their development is very important for the BCI community (Di Flumeri et al., 2019; Hinrichs et al., 2020). Secondly, a distinction is made between active and passive electrodes. Passive electrodes simply transmit the voltage fluctuation to the BCI's amplifier through a conductive wire. On the other hand, active electrodes consist of two components, i.e., the electrode itself and a preamplifier circuit. Before the recorded signal is transmitted through a conductive wire, it is immediately amplified by the preamplifier, thereby, improving the signal-to-noise ratio as the signal becomes less susceptible to electromagnetic interference (J. Xu et al., 2017). A disadvantage of the active electrode over the passive electrode is that it requires more parts and more wires and is thus more expensive to produce. Aside from the type of the electrode, its placement on the scalp is an important factor as well. The mastoids are often used as ground and reference sites. To ensure consistent electrode placement it is often the case that the standardised '10/20' system is implemented (Jasper, 1958). The values 10 and 20 indicate that the distances between adjacent electrodes are either 10 % or 20 % of the total front-back or right-left size of the skull. Figure 2.1a gives a graphical representation of the standardised '10/20' system. The labels of the electrodes relate to the lobes of the brain for which they record. With the advancement of EEG electrodes, various extensions of the standardised '10/20' system have emerged. Figure 2.1b gives a graphical representation of the '10/10' system (Oostenveld & Praamstra, 2001). For a detailed review of a range of electrode placement schemes, the reader is referred to the work of Jurcak et al. (2007).

Much work has been done with regard to decoding EEG signals, most often in the context of enabling a BCI to understand a user's intentions (Lotte et al., 2007; Roy et al., 2019; Craik et al., 2019; Rashid et al., 2020). Since these signals can capture neural activity related to sensory and cognitive processes, they are called Event-Related Potentials (ERP). Examples of ERPs are Slow Cortical Potentials (SCP; Kleber & Birbaumer, 2005), which are very slow fluctuations in cortical activity that can be made positive or negative through operant conditioning, ERror-Related Potentials (ErrP; Abiri et al., 2019), which occur when a BCIs action does not correspond with the user's intention, Steady-State Evoked Potentials (SSEP; Djamal & Lodaya, 2017), which appear when a person perceives a periodic stimulus, and Movement Related Cortical Potentials (MRCP; Shakeel et al., 2015), which are low-frequency negative shifts that take place about 2 seconds prior to voluntary movement. An overview of ERPs and their usage in BCIs is described by Rashid et al. (2020). When considering EEG signals from a machine-learning perspective, it is important to note that a person's intention can not only be decoded from invoked potentials such as ERPs but also from phenomena such as Event-Related Synchronisation (ERS) and Event-Related Desynchronisation (ERD). These denote, respectively, the increase and decrease of oscillatory activity within frequency bands (Pfurtscheller & da Silva, 2017). Such oscillations are known to occur in preparation for movement or imagining movement, i.e., motor imagery. Certain body parts, like the hands and tongue, are associated with a large and distinguishable cortical area, which has allowed BCIs to control them via motor imagery (Schlögl et al., 2005).

Since EEG is a noninvasive signal that records electrical activity on the surface of the scalp, it will naturally have a lower signal quality when compared to invasive signals. The electrical potentials that EEG measures have passed through the skull and can be distorted. In addition, the small amplitude of EEG makes it sensitive to noise. When other potentials, from both external and internal sources, interfere, the noise that this creates results in *artefacts*. A straightforward method for improving on this consists of recording the signal from within the skull, which has the disadvantage of being more risky and expensive. An example of a recording technique that




(a) The '10-20' system. Figure after Jasper (1958).




(b) The '10-10' system. Figure after Oostenveld and Praamstra (2001).

Figure 2.1: Standardised EEG electrode placement schemes.

does this is the electrocorticogram (ECoG; Wyler, 1987; Graimann et al., 2010; Miller et al., 2020). However, as mentioned, since this is an invasive technique, it is not discussed here. Lastly, a crucial property of EEG in the context of machine learning, is its nonstationarity (Klonowski, 2009). A stationary signal is one of which the properties do not change with time. On the other hand, a nonstationary signal is inconsistent with time. EEG's nonstationarity is a result of it capturing the activity of a multitude of different latent variables in the brain, each of which are transmitting with different and variable intensity. In a fraction of a second, the latent variables whose activities are currently dominant can change. Nonstationarity makes interpreting EEG a nontrivial task. If not properly addressed, machine learning models are certain to struggle to learn anything valuable from EEG. This is reflected by the number of preprocessing and feature extraction methods that exist for EEG.

### 2.2.3 Magnetoencephalography

Another noninvasive manner to measure the electrical activity of the brain is to measure the magnetic field that results from it using magnetoencephalography (MEG). The sensors that are used to record MEG are called magnetometers. Electrical currents are always associated with a magnetic field and it is exactly this field that is measured through MEG. A significant advantage of MEG over EEG is that magnetic fields pass through the skull without any distortion, resulting in a higher signal-to-noise ratio (Singh, 2014). MEG also provides better spatial resolution than EEG, with a difference of almost 5mm. Nevertheless, MEG recording equipment, which comes in the form of Superconducting Quantum Interference Devices (SQUID) or Spin Exchange Relaxation-Free magnetometers (SERF), is much more expensive as well as larger than EEG recording equipment. Hence, a portable MEG BCI would not be possible. Additionally, MEG BCIs require extensive magnetic shielding to reduce interference from external magnetic fields (Iivanainen et al., 2017; Boto et al., 2018; Iivanainen et al., 2019). Generally, MEG is less popular than EEG in BCI research.

### 2.2.4 Functional Magnetic Resonance Imaging

Functional magnetic resonance imaging (fMRI) is a neuroimaging modality that is based on the relation between neural activity and blood flow in the brain. When humans use a part of their brain, the blood flow in that part increases, which in its turn results in increased oxygen levels. These localised increases in oxygen can change the magnetic properties of tissue, which can then be detected by scanners with high magnetic fields. This technique is called fMRI (Glover, 2011; Menon & Crottaz-Herbette, 2005). It must be noted that it takes some time for blood to reach the brain and for it to increase oxygen levels, which means that fMRI has a bad temporal resolution when compared to EEG and MEG. On the other hand, the spatial resolution of fMRI is an order of magnitude better than that of EEG and MEG (Menon & Crottaz-Herbette, 2005). In the context of BCIs, fMRI is not a practical neuroimaging modality to use as it seriously lacks in temporal resolution. In addition, the apparatus needed to record fMRI is, again, too costly and non-portable.

### 2.2.5 Near infrared spectroscopy

Near infrared spectroscopy (NIRS) is a neuroimaging modality that works similar to fMRI as it also records the change of oxygen levels in the brain. However, the method used to record differs. NIRS uses near-infrared light to measure the oxygen level changes by looking at how deep the light penetrates the tissue or, in other words, how much the tissue absorbs the light. The

tissue's absorption rate changes when its oxygen level changes. NIRS suffers the same temporal resolution issues as fMRI, which is the main reason why it is not the optimal choice for BCIs. However, compared to fMRI, the apparatus needed for NIRS is relatively mobile and cheap. Moreover, it can be designed in such a manner that it does not produce electrical noise, which is why there have been various attempts at combining EEG with NIRS to create hybrid BCIs (Shin et al., 2018; Khan & Hong, 2017). This is not the case for MEG and fMRI, as the electrical circuits of the EEG electrodes interfere with the magnetic fields that these modalities measure.

## 2.3 Brain-computer interfaces

### 2.3.1 Introduction

In the context of this thesis, a brain-computer interface is defined as a device that allows for a connection between a human brain and some form of computer. Figure 2.2 gives a graphical representation of the general BCI pipeline. The BCI pipeline starts with signal acquisition, which implies that a thorough understanding of brain signals is required. BCIs are to extract these signals and classify them as device commands such that in the end, the BCI application can be steered. BCI frameworks are often divided into a number of branches (Rashid et al., 2020), which are categorised according to three schemes: recording technique, dependability, and method of operation. A BCI's recording technique, or signal modality, can either be synchronous, indicating that the interaction between the user and the system is time-dependent and thus must happen within a certain time interval after a cue imposed by the system, or it can be asynchronous, which means that the interaction is independent of time and that the user can interact with the system whenever they feel like it. The (in)dependability of a BCI relates to the requirement of any additional motor control by the user. Lastly, the method of operation of a BCI can be either invasive, indicating that to acquire the biological signals, sensors are implanted inside the skull, or it can be noninvasive, which only requires sensors on the scalp. The idea of BCIs originated from the more general Human Machine Interface (HMI; Kawala-Sterniuk et al., 2021), to which BCIs are a sub-class. A sibling to BCIs in this hierarchy are, for example, a form of neural prostheses that can replace a limb but are not controlled via brain signals but via nerves that are connected to the muscles (Yildiz et al., 2020). In the last 30 years, technological development has advanced in such a way that it is now increasingly possible to use biological signals to create devices that can help people with communication, movement control, environment control, and many other aspects of our daily life (Kübler, 2020; Shortliffe & Barnett, 2006; van Dokkum et al., 2015; Meng et al., 2016; Gao et al., 2017). Users with physical and mental disabilities also benefit from the more user-friendly and intuitive interface that BCIs bring (Kawala-Janik, 2013).

In what follows, Section 2.3.2 gives an overview of the history of BCIs, from their inception to the present. Subsequently, the BCI pipeline, i.e., the one presented by Figure 2.2, is discussed in Section 2.3.3. This section is meant to give an overview of the state-of-the-art machine learning methods that are currently used in BCIs.


Figure 2.2: The BCI pipeline. Figure after Rashid et al. (2020).

### 2.3.2 History

The history of BCIs starts in 1924 when Hans Berger is the first to record electrical brain activity during a neurosurgery on a 17-year-old boy (İnce et al., 2021). Berger calls the recordings electroencephalograms. Being neurosurgery, the first EEG recording was an *invasive* one. However, later on Berger also develops the first ever *noninvasive* recording technique. In 1929, Berger publishes "Über das Elektrenkephalogramm des Menschen", in which he discusses the EEG recordings of his patients and introduces his alpha and beta waves, which are certain frequency ranges in the EEG recordings. While Berger's discovery of EEG was very important to the development of modern BCI systems, research of the electrical signals of animals already existed in 1875, when Richard Caton was the first to record an animal's electrical signals (Haas, 2003). In the following years, between Caton's publication and Berger's discovery of the EEG, many others worked on similar endeavours (Coenen & Zayachkivska, 2013; Grigoriev & Grigorian, 2007). It is important to note that Berger considered his alpha and beta frequencies to be *artefacts*. Pioneers such as Gibbs et al. (1935, 1936) were the first to start taking the alpha and beta frequencies of the EEG recordings into consideration. Later on, it would become clear that certain frequency intervals, like the alpha and beta frequencies, in the EEG signal are closely related to certain functions, as mentioned earlier and shown by Table 2.1.

BCI research as it is now known only began in the 70's in California. Animals were experimented with in the context of direct communication between the brain and an electrical device (Vidal, 1973). The very first human tests happened only later, in the 90's. Controlling a cursor on a computer using brain waves was already possible in 1991 (J. R. Wolpaw et al., 1991). Around the same time, Pfurtscheller and Lopes da Silva (1999) worked with ERD and ERS as input for a BCI. In the year 2000, the first international conference regarding BCIs was held as a part of the IEEE conference on Rehabilitation Engineering. J. Wolpaw et al. (2000) established the first formal definition of a BCI in a review of this conference: "A brain–computer interface is a communication system that does not depend on the brain's normal output pathways of peripheral nerves and muscles." This definition attracted a great many researchers from all over the world to the BCI field. In the years that followed, up until now, many advances would be booked in the BCI field.

At first, invasive BCIs had great success. Leuthardt et al. (2006) describe an ECoG based BCI system that is able to achieve accuracies above 73%. In 2012, two pioneering papers are published in Nature (Kingwell, 2012; Ethier et al., 2012). Both showing the possibilities of BCIs for restoring movement after paralysis. While invasive BCIs had existed for a while and their successes started even before the 90's, noninvasive BCIs, while already known and being researched, experienced more difficulties. Their successes only started from and after the 90's. An important milestone for noninvasive BCIs was when the Graz group (Pfurtscheller et al., 2003) proposed their cue-based system that allowed for controlling a virtual keyboard or a hand orthosis[2]. This system made use of a method that the Graz group called the Common Spatial Pattern algorithm (CSP; Koles et al., 1990). The CSP algorithm is a feature extraction method that uses spatial filters to maximise the contrast between two classes. CSP and its variants are extensively used to this day. In the present, BCI research is very popular and commercialised. Big tech companies have joined the hunt for BCI systems for a variety of tasks. A number of known examples are Elon Musk's Neuralink (Musk & Neuralink, 2019), which is a general invasive BCI, OpenBCI's 'Galea', a BCI aimed towards virtual and augmented reality which was co-developed by the game development company Valve, and Facebook's 'Building 8', which worked towards a headset or headband BCI that would allow people to send text messages by thinking. However, it

[2] *noun*: a surgical appliance that exerts external forces on part of the body to support joints or correct deformity. Definition by Oxford Concise Medical Dictionary.

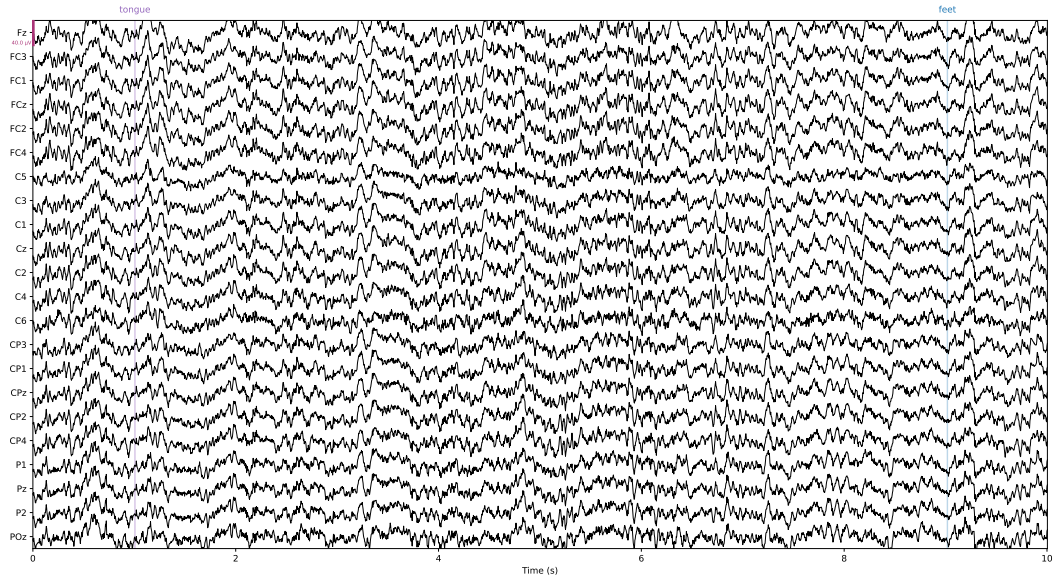



Figure 2.3: Raw EEG data at 250 Hz sampling rate. The y-axis shows 22 channels, 1 per electrode. The coloured lines indicate motor imagery events. Data from Brunner et al.'s (2008) 'The Graz dataset A', the first training run of the first participant.

is important to note that BCI research is very complex and big tech companies often require fast and visible progress. Recently, Neuralink and Facebook have experienced a number of setbacks. In the case of Facebook, this has lead to 'Building 8' being disbanded.

### 2.3.3 The BCI pipeline

This section discusses the state-of-the-art machine learning methods that are currently used in BCIs. It is important to note that the BCI research area is still active and that there is no universally adopted pipeline. Therefore, there is some freedom in choosing which methods to use. In the context of this thesis and its encompassing research, the focus lies on methods that are applicable in the context of noninvasive, mobile, online, EEG BCIs.

In what follows, sections 2.3.3.1 to 2.3.3.4 discuss the steps of the general BCI pipeline from Figure 2.2. Afterwards, Section 2.3.3.5 explains how the machine learning methods used in BCIs can be evaluated.

#### 2.3.3.1 Signal acquisition

The BCI pipeline starts with signal acquisition. A distinction is made between offline and online BCIs. An offline BCI is one that can separate processing from recording. Such BCIs are most prevalent in research and in clinical contexts such as, for example, diagnosing (Fisch, 2003). On the other hand, online BCIs are those that have to work in real-time. They perform every step of the BCI pipeline without breaks. Typical examples of online BCIs are brain-controlled prosthetics (Müller-Putz & Pfurtscheller, 2008; Meng et al., 2016; Bright et al., 2016; Gao et al., 2017; AL-Quraishi et al., 2018; Yildiz et al., 2020). While there is a lot of overlap in the methods

that are used for these two types of BCIs, online BCIs have the added requirement that they should be fast and should be able to run on limited resources.

A variety of EEG recording apparatus exists, directed towards both offline and online BCIs. The most prominent companies that make research-grade EEG recording apparatus are g.tec[3], EMOTIV[4], Braint Producs[5], and OpenBCI[6].

There is no single universally adopted data format for EEG data. Different coding libraries and frameworks all use different file formats. Nevertheless, in theory, they will always contain the same information: the number of channels, the reference scheme used for the placing of the electrodes, sampling frequency, power-line frequency, recording duration, and the raw time-series signal. Examples of well-established formats are: the BioSemi data format[7] (.bdf), the European Data Format[8] (.edf) and the FieldTrip Matlab toolbox[9] format (.mat). There is no single adopted data format for EEG. However, there exist successful overarching frameworks that can be used to convert between formats and to process, analyse, and visualise EEG data. The most prominently used frameworks are the *Python MNE* library (Gramfort et al., 2013) and the *EEGLab* Matlab toolbox (Delorme & Makeig, 2004).

The most important part of EEG data is the raw time series signal. While the metadata surrounding this signal is useful for learning, it is the raw signal that contains the neurological information. A crucial property of EEG is that it is a signal and thus it is sequential. Section 2.3.3.4 explains that for EEG classification, it is desirable to use machine learning models that can capture sequential information in the data. Figure 2.3 shows 10 seconds of the Graz data set A (Brunner et al., 2008) which was used as one of the datasets for the fourth BCI competition (Tangermann et al., 2012). The dataset is a 22-channel motor imagery ERP dataset, as can be seen from the 22 different channels and the motor imagery events that are marked in colour.

#### 2.3.3.2 Preprocessing

It has been established that EEG has a low SNR. Artefacts and nonstationarity make it that, to improve machine learning performance, preprocessing is crucial. Roy et al. (2019) reviewed a set of EEG classification articles published between 2010 and 2018 and found that 72% of the articles employed at least one preprocessing step. Artefacts can be divided into two source classes, they can be of external source or of internal source. External artefacts are a result of external factors such as power-lines or electronic devices. On the other hand, internal artefacts are a result of internal activities that are irrelevant for the study at hand. For example, when recording EEG from a person, if that person is tired, it will often occur that they generate large alpha wave spikes. If tiredness is not relevant for what is being researched, these alpha wave spikes could be seen as artefacts.

In what follows, the most widely used EEG signal preprocessing methods are discussed. It is important to note that discussing all of them falls outside of the scope of this thesis. Over the years, many novel methods, some completely new and some consisting of adaptations of others, have emerged. There is no optimal preprocessing pipeline. Every preprocessing method has advantages and disadvantages and thus composing a preprocessing pipeline is a process that depends on the context in which the pipeline is used (Bashashati et al., 2007). Another nontrivial

[3] https://www.gtec.at/
[4] https://www.emotiv.com/
[5] https://www.brainproducts.com/
[6] https://openbci.com/
[7] http://www.biosemi.com/faq/file_format.htm
[8] https://www.edfplus.info/
[9] https://www.fieldtriptoolbox.org/getting_started/

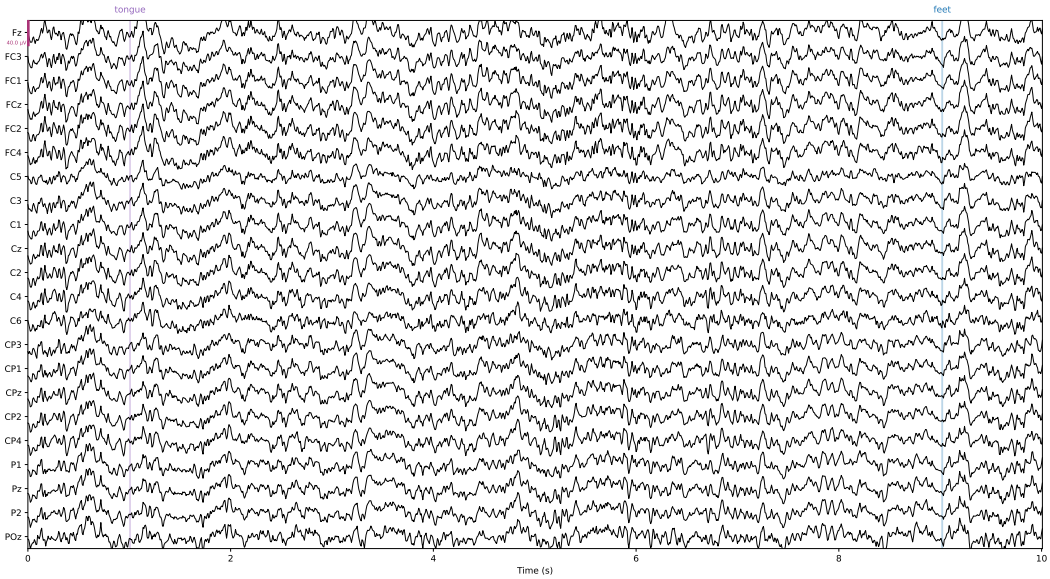



Figure 2.4: Downsampled EEG data. The original sampling rate was 250 Hz, here downsampled to 45 Hz. The y-axis shows 22 channels, 1 per electrode. The coloured lines indicate motor imagery events. Data from Brunner et al.'s (2008) 'The Graz dataset A', the first training run of the first participant.

aspect of such a pipeline is the ordering in which the methods are applied. Whether or not certain steps need to be applied before others depends on whether or not that step involves a linear or a nonlinear operation (Luck, 2014). While a sequence of linear operations can be applied in arbitrary order, the moment a nonlinear operation comes into play, the ordering matters. Hence, the (non)linearity of a method must be considered when composing a preprocessing pipeline. It is often the case that nonlinear operations are applied as early as possible in the pipeline such that linear operations can proceed thereafter with as little modification as possible.

**Downsampling** Downsampling is an EEG preprocessing method that is not intended to remove artefacts from the signal but is rather a way of reducing data size. This can be needed as, for example, when EEG recording apparatus with 32 channels and a sample rate of 250 Hz, i.e., 250 samples per second, represents each sample as a 32-bit float, the EEG data would consist of $32{*}250{*}32$ bits per second of recording, which is approximately 32 $\frac{\text{kB}}{\text{s}}$. Reducing this can be done through downsampling, which consists of simply selecting every nth sample and dropping the rest. Downsampling must be done with extreme care as, through a phenomenon called aliasing, it can introduce new artefacts to the data. Important to keep in mind when downsampling is the Nyquist-Shannon sampling theorem, which states that if the sampling rate is $R_1$, then any signal with a frequency $R_2 \geq R_1 + \frac{R_1}{2}$ will be perceived as a signal with a lower frequency (Shannon, 1949). This means that, for example, when trying to detect certain frequency bands in an EEG signal, the sampling rate must at least be double the higher bound of the frequency band and thus downsampling can have a significant effect on the information captured by EEG data. Since downsampling completely removes data points, it should always be applied as the first step in

the preprocessing pipeline. Figure 2.4 shows the dataset from Figure 2.3 but downsampled to 45 Hz, the signals are the same but there are simply fewer points in time.

**Channel cleaning and interpolation** In EEG data, channels can be excluded for various reasons: the corresponding electrode was not placed properly, the electrode was malfunctioning for some reason, the channel contains an excessive amount of noise, etc. Detecting bad channels often requires domain knowledge. Once the bad channels have been detected, they can be compensated for using interpolation with the data from the other channels. A universally adopted method for this is interpolation by spherical lines, first introduced to EEG by Perrin et al. (1989). This method consists of three steps: projecting the channel locations onto a unit sphere, computing a matrix that represents the relationship between the good and bad electrodes, and using this matrix to interpolate data for the bad channels. The MNE Python library supports methods that implement this and various other types of interpolation. Most of these interpolations consist of nonlinear operations and therefore the position at which this preprocessing step is placed in the pipeline is important. Intuitively, one can see that removing channels and interpolating to compensate for them belongs at the beginning of the preprocessing pipeline.

**Rereferencing** Section 2.2.2 mentions that EEG corresponds to the voltage difference between a certain electrode and a reference electrode. Any EEG recording apparatus comes with a predefined reference electrode. However, through rereferencing, it is possible to change the reference electrode after recording. In the optimal scenario, each electrode only records the signal from the location at which it is placed. However, in reality, this is not the case. Due to different types of conduction, signals that originate from different parts from the brain are interweaved. This phenomenon is called cross-talk. Hence, to be able to correctly interpret these signals, it is best to separate them in the spatial domain. In signal processing, this is usually done through spatial filters. The most simple of such filters consists of applying rereferencing. Examples are the average reference, which consists of taking as reference the average of the EEG signal over all electrodes, and the surface Laplacian (Fitzgibbon et al., 2013), which consists of averaging over certain neighbourhoods of electrodes instead of all electrodes. Generally, rereferencing methods are linear and thus they can be applied in any order in a sequence of linear operations.

**Windowing** Windowing, or epoching, EEG data consists of slicing the continuous signal in the time domain to get overlapping or nonoverlapping segments. Remember from Section 2.2.2 that signals that try to capture a person's intentions are called ERPs. When trying to decode ERPs, windows are created by slicing the continuous signal at the start and end points of the events that caused the ERPs. Since windowing does not actually change the data, it is a linear operation and thus it can be applied in any order in a sequence of linear operations. Figure 2.5 shows the first channel from the dataset from Figure 2.3 but cut into windows that start 2 s before the event and end 4 s after. For ERP signals such windowing is typical when the data is to be passed to machine learning algorithms.

**Baseline correction** In practice, because of the nonstationary nature of EEG signals, the start points of windows are chosen earlier in the time domain than the actual event offsets to incorporate a baseline during which no event occurred and to which the whole window can be corrected. The average ERP voltage differs between occurrences even for the same person and a baseline correction tries to transform the windows such that it seems as if they all oscillate around the same voltage. The windows in Figure 2.5 are baseline corrected with respect to the 2 s before the event. Since a baseline correction consists only of averaging and subtracting, it is a

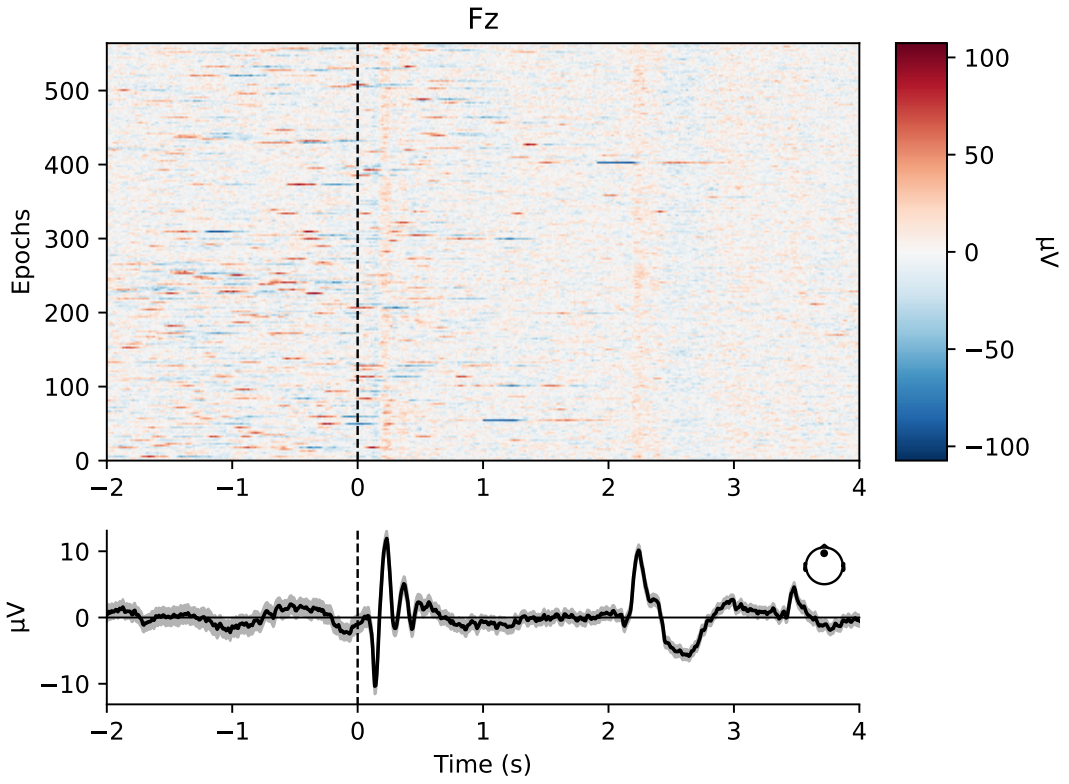



Figure 2.5: Windowed EEG data. The windows are created by cutting the signal 2 s before an event and 4 s after. The heatmap shows electrical spikes in the windows. The underneath plot gives the average window. Data from Brunner et al.'s (2008) 'The Graz dataset A', the 'Fz' channel of the first training run of the first participant.

linear operation and thus it can be applied in any order in a sequence of linear operations. In the context of classifying EEG, the baseline correction is almost always applied after the windowing step.

**Frequency filtering** Another common, if not the most common, EEG preprocessing step is frequency filtering. EEG often contains electrical noise with frequencies depending on the power-line. In Europe, power-lines have frequencies around 50 Hz. Filtering these frequencies from EEG data can result in cleaner data. Moreover, only certain frequency ranges can be of interest, for example, in EEG data, the frequency bands that are mentioned in Section 2.3. In these cases, frequency filtering can be used to filter out frequencies that are not part of the chosen frequency bands. Frequency filtering can be done in multiple ways: low-pass filters, which filter out high frequencies above a certain threshold, high-pass filters, which do the opposite, band-pass filters, which do both, and notch filters, which remove a single frequency. Figure 2.6 shows the (nondownsampled) dataset from Figure 2.3 but with a 1 Hz to 45 Hz band-pass filter applied to it. When frequency filtering EEG data, filtering can happen both digitally and analogly. Many EEG BCIs have built-in filters and can thus filter the EEG signal at the time of recording. Once EEG

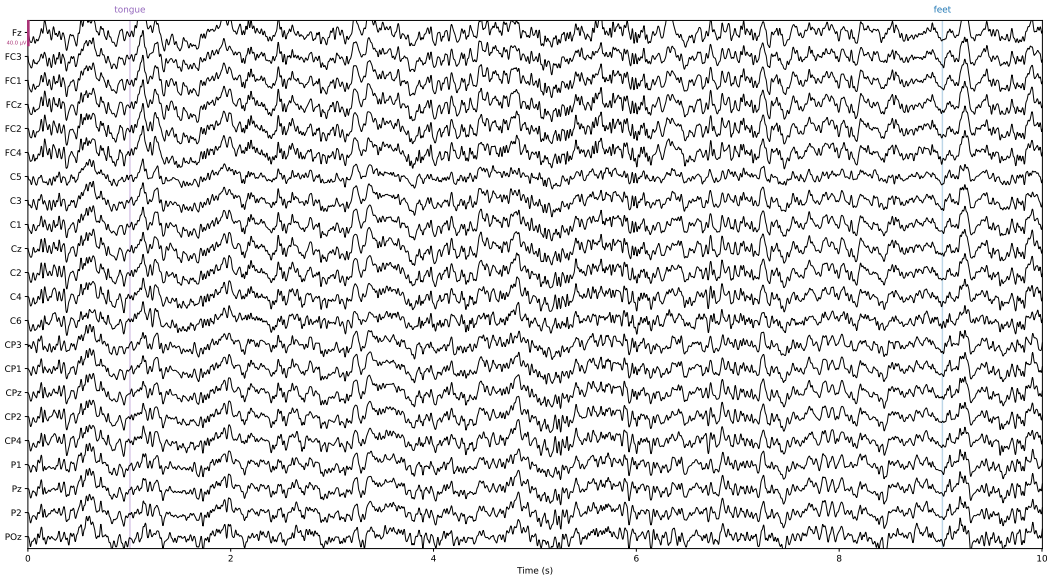



Figure 2.6: Frequency filtered EEG data. The original frequency range was 0.5 Hz to 100 Hz, here it is 1 Hz to 45 Hz. The y-axis shows 22 channels, 1 per electrode. The coloured lines indicate motor imagery events. Data from Brunner et al.'s (2008) 'The Graz dataset A', the first training run of the first participant.

data has been recorded, it can still be filtered. Through a series of additions, subtractions, and multiplications, filtering can be simulated as if an analog filter was present on the EEG recording device. The exact mathematics are not explained here, but in general they consist of applying some transformation function to each of the data points of a signal, with the possibility of having recursive transformations (Burgess, 2003). The complexity of the transformation often results in hardware filters being much faster than digital filters, which is why real-time decoding BCIs are often equipped with them. Luck (2014) states that the most common frequency filters for EEG are based on mathematical convolution, which is a linear process. However, theoretically, frequency filters are applied on signals of infinite duration, while in practice, they are applied to finite signals, requiring some additional nonlinear steps to deal with the edges at the beginning and the ending of the finite signal, resulting in *edge artefacts*. Therefore, Luck (2014) advises to apply frequency filtering on the continuous EEG signal, before it has been windowed, as to limit the number of edges to the beginning and ending of each run and not of each window. Additionally, to avoid aliasing issues, frequency filtering should always be applied before downsampling.

**Artefact rejection and correction** When artefacts only span a single frequency or when they span a frequency range that does not contain relevant information, frequency filtering can be used to filter out these frequencies and with that remove the artefacts. However, when artefacts span multiple frequencies and these frequencies can not just be filtered out, other methods must be used to get cleaner data. Upon reasoning about this problem, two intuitive approaches arise. Either the artefacts can be *rejected*, by selecting the segments of the time-series data in which they occur and not using them, or they can be *corrected*, by transforming the signal in such a way

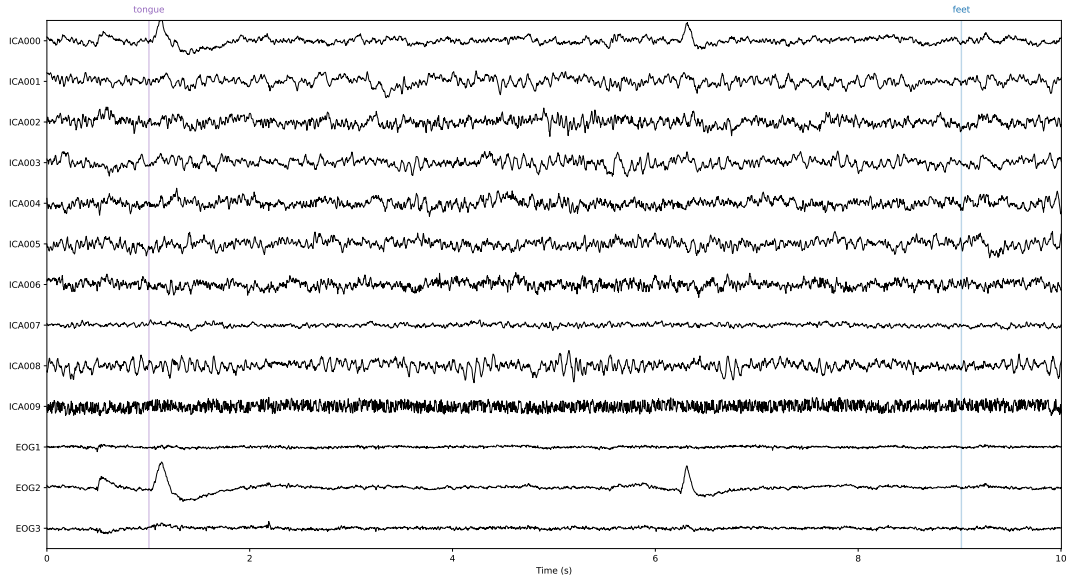



Figure 2.7: ICA components of EEG data. The y-axis shows 10 components as well as 3 EOG channels. The coloured lines indicate motor imagery events. Data from Brunner et al.'s (2008) 'The Graz dataset A', the first training run of the first participant.

that the part of the signal that is the artefact is removed and the relevant information is kept. When performing artefact *rejection*, complete segments of signals are flagged as noisy when there is some form of justification for doing so. Biological artefacts often have recognisable patterns and thus visualising signals and looking for such patterns can be an artefact rejection approach. It goes without saying that such an approach is inefficient. Other justifications for rejecting certain segments of signals are based on analysing the variance of the signal or the magnitude of voltage increases. Additionally, it is often the case that EEG data is accompanied by other biological signals. Comparing the two is another way of identifying artefact segments. The methods that are often used to do artefact *correction* have some overlap with the feature extraction methods that are discussed in Section 2.3.3.3. Nevertheless, in this section their application for removing artefacts from a signal is discussed. A general assumption that is made when correcting signals is that if two signals are statistically independent and they are combined into a single signal, this signal can also be separated again into its statistically independent components. This process of separation is called *source decomposition* and methods that are used to do this are often based on Independent Component Analysis (ICA; Sun et al., 2005; Turnip & Junaidi, 2014), Principal Component Analysis (PCA; Turnip & Junaidi, 2014), Signal Space Projection (SSP; Uusitalo & Ilmoniemi, 1997), or Artefact Subspace Reconstruction (ASR; Chang et al., 2018). ICA and PCA are both methods that separate independent signals that have been combined. However, since it does not assume orthogonality between the individual signals, ICA is generally assumed to be the better one. Both PCA and SSP do make this assumption, which is unrealistic for EEG data. While all three of these are relatively expensive in terms of running time and computational resources, ASR is an online and real-time capable component-based method. Once signals have been separated into components, selecting which ones are artefacts and which ones

are not can be done in two ways. Some artefacts, like those that originate from eye movements or heartbeats, have known patterns and can be recognised manually. Figure 2.7 shows 10 ICA components derived from the EEG data from 2.3. Aside from the 22 EEG channels, this dataset contains 3 electrooculography (EOG) channels that record the potential that exists between the front and the back of the human eye. By comparing these channels to the found components, the components that correspond to eye blinks or movements can be found and nullified, thereby, correcting artefacts. In this case, the zeroth component seems to match the second EOG channel. Components that are not easily recognised can be linked to the area in the brain that they originate from. This method is called *source localisation*. The most used approach for source localisation is dipole fitting (Cuffin, 1998). If the origin of the components is known, irrelevant components can be dropped. Both artefact rejection and correction often work with thresholds. In artefact rejection, a threshold is used to determine whether or not an artefact is to be rejected. In artefact correction, a threshold is used to determine whether or not a component should be nullified if it matches certain artefact patterns. The fact that these methods use thresholds make them nonlinear processes and therefore, their position in the preprocessing pipeline should be reasoned about. Luck (2014) states that the general rule is to apply other preprocessing steps before artefact rejection or artefact correction only if it makes these work better.

#### 2.3.3.3 Feature extraction

Even after the original EEG signal is preprocessed and noise has been reduced as much as possible, the resulting signal is still complex and hard to interpret. To make the signal more understandable for both humans and the classification methods that are discussed in Section 2.3.3.4, feature extraction methods are applied. Once more, the nonstationary nature of EEG is the main issue for these feature extraction methods to address. For EEG feature extraction methods to be successful, they are to extract shift-invariant features, i.e., features that are insensitive to changes in the input's time axis. The concept of shift-invariance is well-established in the computer vision field, where the shifts are two-dimensional (R. Zhang, 2019). As mentioned, there is some form of semantic overlap between feature extraction and preprocessing to remove artefacts. While it seems as if the two are equal, they differ in their intention. The former tries to *extract* useful information while the latter tries to *remove* useless information. Feature extraction can be done both unsupervised as supervised and, in the context of time-series, the most popular can be categorised under the following domains: the time domain, the frequency domain, the time-frequency domain, and the spatial domain. In the following paragraphs, each of these domains are explained and a number of associated feature extraction methods are discussed. Just as for preprocessing, there is no optimal feature extraction method and selecting methods is heavily dependent on the context in which they are used. For a more detailed review on feature extraction for EEG, the reader is referred to the works of Al-Fahoum and Al-Fraihat (2014) and Geethanjali et al. (2012).

**Time domain** When extracting features in the time domain, the data is analysed as a voltage-over-time signal. This is the data format in which the raw signal is acquired and thus no transformations are needed. A recent time domain feature extraction method is autoregressive modelling (AR; Lawhern et al., 2012; Chai et al., 2017). AR consists of a linear regression of an observation in the signal against one or more earlier observations. The AR model for a single channel can be written as:

$$y(t) = \sum_{i=1}^{p} \alpha_i y(t-i) + \epsilon_t \tag{2.1}$$

with $p$ the number of points in the past used to model the current point in time and $\epsilon$ a zero-mean noise process with variance $\sigma^2$. The value for $p$ is often chosen by looking at the auto-correlation function of the signal, which shows the correlation of each data point of a signal to each other data point. The coefficients of the AR model $\alpha_i$ are its parameters and they can be estimated using Maximum Likelihood Estimation (MLE; Weisberg, 1980). In the context of feature extraction for EEG, the parameters of the AR models for each channel in the data are collected in a vector and used as a feature to describe a signal. Other, less complex, time domain feature extraction methods are the Mean Absolute Value (MAV), which consists of using the mean absolute value of a signal as a feature, Zero Crossing (ZC), which takes the number of signal sign changes as a feature, Slope Sign Changes (SSC), which takes the number of times that the slope of the signal changes sign as a feature, and Waveform Length (WL), which takes the cumulative length of the signal's waveform as a feature. For details regarding these time-domain feature extraction methods, the reader is referred to the works of Lotte (2012), Geethanjali et al. (2012), and Sharmila and Geethanjali (2020).

**Frequency domain** To be able to extract features in the frequency domain, the original signal needs to be transformed from the time domain to the frequency domain. Such a transformation can be done through a Fourier Transform (FT; Bracewell, 1986):

$$\hat{g}(f) = \int_{-\infty}^{+\infty} g(t) \cdot e^{-2\pi i f t} dt \tag{2.2}$$

with $g$ the time domain signal and $\hat{g}$ the transformed signal. A FT is reversible, through an Inverse Fourier Transform (IFT), which is why the FT is said to be a frequency domain representation of the original signal. An IFT is defined as follows:

$$g(t) = \int_{-\infty}^{+\infty} \hat{g}(f) \cdot e^{2\pi i f t} df \tag{2.3}$$

However, both of these equations are purely theoretical. In practice, a Discrete Fourier Transform (DFT) is used to estimate the full integral solution for a finite window of samples $x_0, \ldots, x_{N-1}$ of a signal:

$$x_k = \sum_{n=0}^{N-1} x_t \cdot e^{\frac{-2\pi i k n}{N}}, 0 \leq k \leq N-1 \tag{2.4}$$

where $x_k$ is a complex number which encodes the amplitude and phase of a sinusoidal wave with frequency $\frac{k}{N}$ cycles per time unit. Symmetrically, an Inverse Discrete Fourier Transform (IDFT) is defined as follows:

$$x_k = \frac{1}{N} \sum_{k=0}^{N-1} x_k \cdot e^{\frac{2\pi i k n}{N}}, 0 \leq k \leq N-1 \tag{2.5}$$

In practice, DFTs are almost exclusively performed using the Fast Fourier Transform algorithm (FFT; Cooley & Tukey, 1965). For a detailed explanation of the FFT algorithm the user is referred to the work of Oppenheim et al. (2001). The frequency domain representation of a signal consists of complex numbers that capture the amplitude and phase of each of its frequency components. While these are interesting, a well-established practice in signal processing is to consider the squared amplitude, which is called the power. The collection of squared amplitudes of the DFT representation of a signal is called a periodogram. It must be noted that when a periodogram is computed, the variance of the power estimates does not decrease as the number of samples of the signal increases. To mitigate this, Power-Spectral Density (PSD) estimation

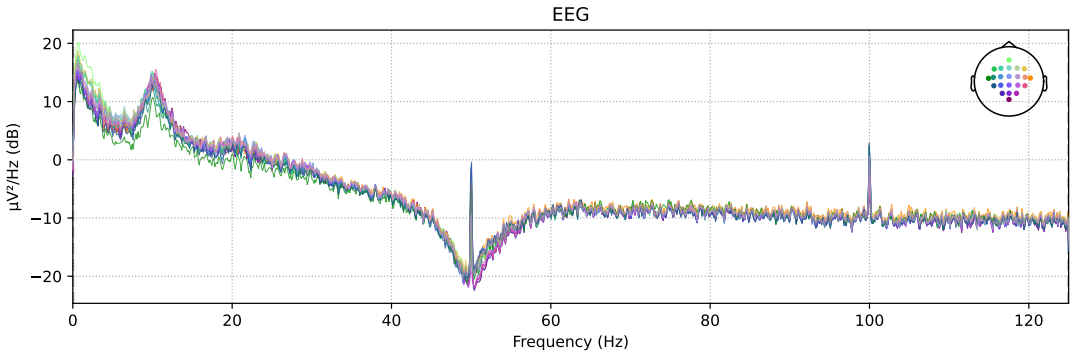



Figure 2.8: Power spectral density estimation of EEG data. The peaks at 50 Hz and 100 Hz are caused by electrical noise. Data from Brunner et al.'s (2008) 'The Graz dataset A', the first training run of the first participant.

algorithms often average periodograms of multiple possibly overlapping segments of the signal. An example of a well-known PSD estimation algorithm is Welch's (1967) method. For detailed explanations of a variety of PSD estimation algorithms the user is once more referred to the work of Oppenheim et al. (2001).

Once the signal is in the frequency domain, frequency domain features can be extracted. A number of examples of frequency domain features that are used in EEG BCIs are PSD (Bascil et al., 2016; Chakladar & Chakraborty, 2018), band power (Serdar Bascil et al., 2015; Kreilinger et al., 2016), which uses PSD to calculate the average frequency band power as a feature, and the spectral centroid (Murugappan et al., 2014), which is a measure of spectral position and shape that represents the centre of mass of the frequency spectrum. Figure 2.8 gives a PSD estimation for the data from Figure 2.3. The peaks at 50 Hz and 100 Hz are caused by electrical noise originating from power-lines, as mentioned in Section 2.3.3.2.

**Time-frequency domain** An issue with feature extraction methods for the time domain and the frequency domain is that they lack information about each other. Time-frequency domain feature extraction methods address this issue. These feature extraction methods are thought to be very promising for BCI research (Rashid et al., 2020). An intuitive time-frequency domain feature is a Short Time Fourier Transform (STFT; Zabidi et al., 2012). A STFT consists of dividing an input signal into multiple short time windows and applying a FT to these separately to get frequency domain data segments. To each of these, frequency domain feature extraction methods can be applied, resulting in features that are parameterised by both frequency and time. Therefore, a STFT is similar to Welch's method, however, at the end of Welch's method, the PSDs of the segments are averaged to get the PSD of the complete signal and thus the temporal information is lost. A limitation of a STFT is the trade-off between the time and frequency resolutions due to the finiteness of the chosen time window. If a narrow time window is chosen, time resolution will be good, but frequency resolution will suffer, and vice versa. An alternative that tries to improve on this is Wavelet Transformation (WT; H. K. Lee & Choi, 2019; Guo et al., 2015). Instead of moving a time window over the original signal, WT uses a wavelet to do this. A wavelet is a wave-like function that is not only parameterised by time but also by scale, effectively allowing WT to also change the size of its version of the time window. Instead of doing a FT per time window, WT consists of calculating the product of the signal and the wavelet at

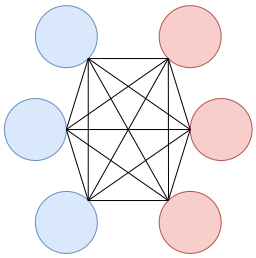
each time step. By repeating this process for a range of wavelet scales, both time and frequency information is captured in the resulting signals, which in contrast to those of a STFT, each have a different frequency resolution. A distinction is made between continuous and discrete wavelet transformation. The former applies every possible wavelet in a range of scales and time steps. The latter applies only a finite number of wavelets, defined for a specific set of scales and time steps.

**Spatial domain** Extracting features in the spatial domain means extracting, from the EEG signal, information that captures where in the brain the signal was produced. Section 2.3.3.2 explains that due to conduction, the signals that are recorded by the electrodes are an interweaving of signals that originate from different sources in the brain. Through spatial filtering, it is possible to remove these spatial artefacts from the signals. However, the preprocessing methods that can be used for this purpose all assume that their input is stationary. When the input is a nonstationary signal such as EEG, it is better to use adaptive spatial filters. This type of filters derives its weights directly from the spatial properties of the signal and thus is not affected by the nonstationarity of its input. A supervised approach consists of giving the raw EEG signal to a neural network and trying to learn a good spatial filter through backpropagation and gradient descent (Robbins & Monro, 1951). However, there are two disadvantages to this approach: the fact that it is supervised and thus training examples are needed as well as the possibility of gradient descent ending up in a local minimum. Because of the lack of labelled data in the EEG classification field, unsupervised methods are a better fit. Examples are methods such as ICA and PCA, which try to learn spatial filters that extract dominant features in their input without considering the labels. The disadvantage of these methods is that there is no guarantee that the found features capture the desired spatial information. As explained in Section 2.3.3.2, they are used in multiple contexts and thus they are not tailored to spatial filtering. Lastly, an approach that has gained attention recently tries to make the spatial filtering problem solvable by introducing additional assumptions. A successful example of such an approach is the CSP algorithm (Koles et al., 1990). CSP assumes that the frequency band and time window are known and that the signal is jointly Gaussian[10] within the time window. Through these assumptions, the CSP algorithm tries to learn a set of linear transformations that can separate the input signal into components that are maximally different in variance, between the time windows. Due to the dependence of CSP on the person-specific frequency band, multiple extensions have been applied to CSP. An example is the Filter Bank Common Spatial Pattern (FBCSP; Ang et al., 2008), which consists of applying CSP to a range of frequency bands and combining the found features in one large feature vector.

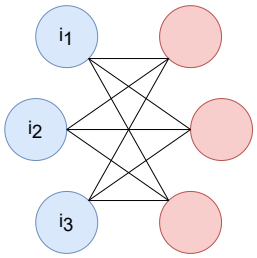
#### 2.3.3.4 Classification

The last step in the BCI pipeline is the classification of the preprocessed and possibly feature extracted data. While it is possible to classify EEG data using regression, the current trends favour pure classification methods. Most conventional classification methods have been applied to EEG: k-Nearest Neighbours (Bablani et al., 2018), support vector machines (Rakotomamonjy et al., 2005), linear discriminant analysis (Long et al., 2012), feed-forward neural networks (Haselsteiner & Pfurtscheller, 2000), etc. For a detailed review, the reader is referred to the work of Lotte et al. (2007). In this section, the focus lies on deep learning approaches. While the conventional methods have had great successes, the nature and shortage of EEG data in combination with the expert knowledge that is needed to extract features from it have made researchers try new approaches. Due to its ability to extract features from data while learning and the recent increase

[10] A signal is jointly Gaussian when it is a linear combination of Gaussian components.



(a) Boltzmann Machine




(b) Restricted Boltzmann Machine

Figure 2.9: Example structures of Boltzmann Machines.

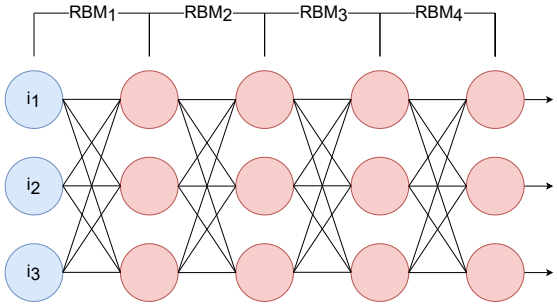



Figure 2.10: Example structure of a Deep Belief Network.

in EEG data, deep learning has become the most recent addition to the assortment of machine learning methods that are researched in the context of EEG classification.

In what follows, three of the most successful deep learning approaches to EEG classification are discussed. These approaches have been chosen because of their relevance to this thesis. For a more detailed review of deep learning approaches for EEG classification, the reader is referred to the works of Roy et al. (2019) and Craik et al. (2019).

**Deep Belief Networks** A Deep Belief Network (DBN), when used for classification, is a deep learning model that has a structure identical to that of a Multi-Layer Perceptron (MLP) but differs in the type of its connections. Both DBNs and MLPs consist of fully connected layers. However, while connections in MLPs can only go in one direction, a DBN uses undirected connections. Another difference is that MLPs are only trained using backpropagation, while DBNs also go through a pretraining phase. The layers in a DBN can be seen as Restricted Boltzmann Machines (RBM), which are an adaption of traditional Boltzmann Machines (Ackley et al., 1985) which are undirected and fully connected graphs. Figure 2.9 gives graphical examples of traditional and restricted Boltzmann Machines. The restriction that RBMs have over traditional BMs is that they require their neurons to form a bipartite graph. Such a restriction makes a clear distinction between two types of nodes: input nodes and hidden nodes. Figure 2.10 gives a graphical representation of an example of a DBN. In a DBN, the hidden nodes of each RBM are the input nodes of the next RBM. Before a DBN is trained as a whole, each of its RBMs is separately trained to reconstruct its input as accurately as possible using an unsupervised algorithm called Contrastive Divergence (Hinton, 2002). By first pretraining each RBM separately, they can already learn to detect patterns without needing a labelled dataset. Once each RBM has had its pretraining, the complete DBN can be finetuned on a classification task by using back-

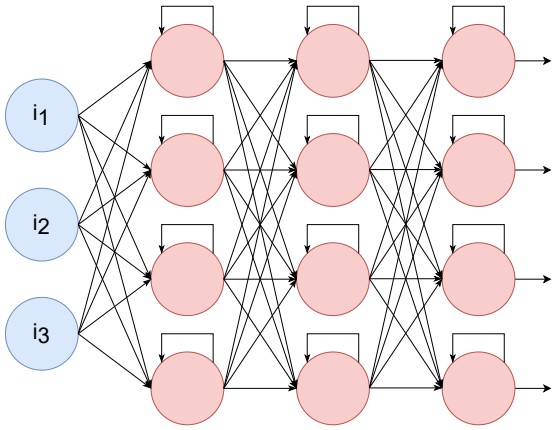



Figure 2.11: Example structure of a Recurrent Neural Network.

propagation and a relatively small labelled dataset (Hinton et al., 2006). Due to only needing a small labelled dataset and the ability to learn positional information through the undirected connections, DBNs have been used for classifying EEG (Z. Lu et al., 2018; T.-J. Lee & Sim, 2015).

**Recurrent Neural Networks** A Recurrent Neural Network (RNN) is a deep learning architecture that is specifically designed for learning from sequential data. Figure 2.11 gives a graphical representation of an example of a RNN. RNNs use directed connections between layers and additional recurrence connections within layers. When a data sequence is given to the input layer of a RNN, each of its tokens is processed sequentially and each time a neuron receives information from the previous layer, it also passes itself its own learned information. By doing this, RNNs can capture the positional information in its input. Due to the vanishing/exploding gradients problem in RNNs (Hochreiter, 1998), various adaptions of RNNs were created. An example that addresses the issue uses Long Short-Term Memory units (LSTM; Hochreiter & Schmidhuber, 1997) instead of traditional neurons. Such LSTM units use combinations of input, forget, and output gates, to be able to process data sequentially while keeping their state over time. Nevertheless, LSTM networks only suppress the vanishing/exploding of gradients, they do not exclude it. Moreover, while training RNNs can be very slow due to the fact that each input token needs to be passed separately and thus parallelisation is not possible, due to the added complexity, training RNNs with LSTM units is even slower. Despite all this, RNNs have been used for EEG classification to some extent (Michielli et al., 2019; Li et al., 2016).

**Convolutional Neural Networks** Convolutional Neural Networks (CNN; LeCun et al., 1989) are a type of deep learning architecture that have had great success in classifying images (Krizhevsky et al., 2012; Simonyan & Zisserman, 2015). Just like RNNs, CNNs have the ability to capture positional information. However, RNNs do this in a sequential and thus one-dimensional way, while CNNs have the ability to capture multidimensional information. In the context of classifying EEG, it could be said that RNNs are a better match since EEG is a time-series and thus sequential. However, through the correct choice of feature extraction, CNNs have recently been able to achieve significant success (Olivas-Padilla & Chacon-Murguia, 2019; Amin et al., 2019; Tayeb et al., 2019). Figure 2.12 gives a graphical representation of an example of a CNN. Typically, CNNs implement three main concepts: convolution operations, local connectivity, and

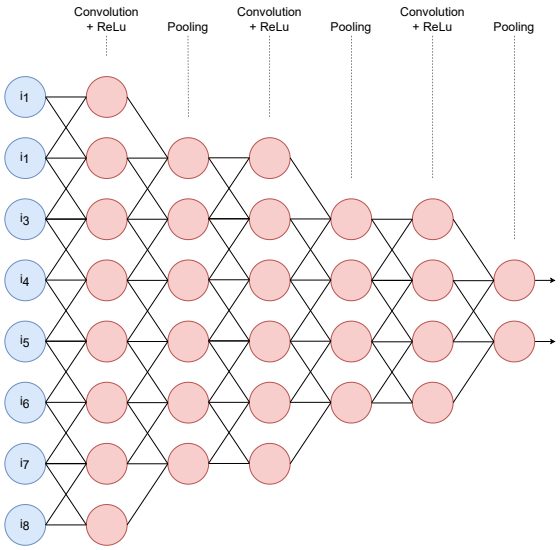



Figure 2.12: Example structure of a Convolutional Neural Network with a kernel of size 3.

pooling.

In the one-dimensional case, convolution consists of an operation on two functions $x(t)$ and $w(a)$:

$$s(t) = \int x(a)w(t-a)da = (x \times w)(t) \tag{2.6}$$

with $x(t)$ often referred to as the input, $w(a)$ as the kernel, and $s(t)$ as the output. Theoretically, convolution is defined for any two functions for which the above integral is defined. As an example, imagine an EEG time-series signal $x(t)$ and a weighted average kernel $w(a)$ resulting in a convolution output $s(t)$ which gives a smoothed version of the EEG signal. In practice, time is discretised and $x(t)$ consists of samples $x_0, \ldots, x_n$ conforming to the recording sampling rate. Therefore, in practice, discrete convolutions are used:

$$s(t) = \sum_{a=0}^{n} x(a)w(t-a)da = (x \times w)(t) \tag{2.7}$$

Finally, in CNN models, both the input and the kernel are usually multidimensional arrays. The kernel array then consists of parameters that are learned. In the two-dimensional case, the formula becomes:

$$S(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) = (I \times K)(i,j) \tag{2.8}$$

In neural networks, convolution can be seen as matrix multiplication where several matrix entries are constrained to be the same as others. This practice is often denoted parameter sharing or tied weights.

Typically, CNNs only connect the neurons of a layer to neurons in the next layer when they are spatially close to them. This is called local or sparse connectivity and it corresponds to using kernels that are smaller than the input. Consequently, CNNs requires less parameter storage

as well as fewer operations when compared to fully-connected neural networks. In addition, the tied weights in the kernel matrix reduce parameter storage requirements even more. Most importantly, the use of kernels that are smaller than the input allows for finding small, meaningful, and shift-invariant features in very large inputs. Shift-invariant features are features that are minimally influenced by shifts in the input, a very desirable property in the context of image recognition and EEG or any other time-series.

As can be seen in Figure 2.12, a typical CNN combines convolution with a nonlinear activation function, for example, the Rectified Linear Unit (ReLU) activation function. ReLU maps an input value to the maximum of that input value and zero. Convolution is inherently linear and thus, to increase the CNNs expressive power, nonlinear activation functions are crucial. Additionally, convolution is usually followed by a pooling layer. A pooling layer receives the activation values from a previous layer and transforms these into summary statistics that describe groups of inputs. Examples of pooling functions are max pooling, which outputs the maximum activation value within a rectangular neighbourhood, average pooling, which outputs the average activation value of a rectangular neighbourhood, and weighted average pooling, which outputs a weighted average based on the distance to the centre of some form of neighbourhood. Just as using kernels that are smaller than the input, pooling layers aim to reduce the influence of shifts in the input.

#### 2.3.3.5 Evaluation

In a research context, classification accuracy is the most used evaluation metric for EEG classification. A BCI that classifies its input correctly is seen as a performant BCI. In some cases, BCI researchers also look at metrics that are calculated from the confusion matrix: sensitivity, specificity, or precision. Receiver Operating Characteristics (ROC) and Area Under the Curve (AUC) are often utilised as well (Sameer et al., 2020). In other cases, task-specific metrics are established (H. Wang et al., 2014; Chung et al., 2019). However, when deploying BCIs in a real-world scenario, many other factors come into play. Examples are user experience and acceptance. It is a general issue in machine learning research that such deployment factors are not accounted for (D'Amour et al., 2020), BCI research included (Dillen et al., 2022). Nevertheless, when introducing new methods, it remains the standard to start with an initial evaluation phase where metrics such as classification accuracy and the other aforementioned are implemented. The work that is presented in this thesis is an example of such work. Finally, it must be noted that in an offline research setting such as the setting of this thesis, metrics are calculated on prerecorded datasets using strategies such as train-test splits and cross-validation. When doing so, splits can be made on different levels: the trial level, the run level, the session level, or the participant level. Depending on the research question, the most fitting level should be chosen.

## 2.4 Transfer learning

### 2.4.1 Introduction

Transfer learning is the concept of storing learned knowledge for one problem and trying to transfer it to the solving of another related problem. Pan and Yang (2010) state that the assumption that training and testing data must come from the same domain and must have the same distribution is very common in machine learning but does not hold in various real-world applications. By considering data with different features or that is distributed differently, the performance of machine learning algorithms could be greatly improved and data-labelling

issues reduced. Transfer learning is supposed to address this. However, it comes with its own difficulties.

In what follows, firstly, a brief history of transfer learning is given. Afterwards, a number of notations and definitions are established. Subsequently, a number of related paradigms are briefly explained and compared to transfer learning. Thereafter, a taxonomy of transfer learning is established and discussed. Lastly, transfer learning is briefly framed in the context of BCIs. It is important to note that transfer learning is a fairly novel research area and no single agreed taxonomy exists. Various works name and group methods in various different ways. The following sections use the well-established works of Pan and Yang (2010) and Zhuang et al. (2021) as guidelines.

### 2.4.2 History

Contrary to popular belief, transfer learning was not first introduced in the early 1990's (Pratt et al., 1991; Pratt, 1992) but much earlier in 1976 by Bozinovski (2020). The work of Bozinovski (2020) presents a mathematical model and geometric interpretation of transfer learning as well as a measure of transfer learning indicating positive, negative, and no transfer learning. According to Tan et al. (2018), this measure has been very relevant for current transfer learning attempts. Later work such as that of Pratt et al. (1991) and Pratt (1992) builds further on transfer learning, coming up with new types of algorithms such as the discriminability-based transfer (DBT) algorithm. Research evolved to investigate transfer learning for a variety of fields such as natural language processing (Blitzer et al., 2006, 2007; Arnold et al., 2007), image recognition (Wu & Dietterich, 2004), and cognitive science (Pratt & Jennings, 1996). Presently, transfer learning is believed to be very promising for the commercial success of machine learning.

### 2.4.3 Formal definition and notation

The notations and definitions that are established in this section are mostly adopted from the work of Pan and Yang (2010).

**Definition 1** *A domain $\mathcal{D}$ consists of two components: a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$, with $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$.*

For example, if the learning task consists of classifying EEG and the samples of an event window are seen as features, then $x_i$ is a single time-point and $X$ is a particular learning sample. Generally, for two domains to be different, either their feature spaces, their marginal probability distributions, or both are different.

**Definition 2** *Given a domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$, a task consists of two components: a label space $\mathcal{Y}$ and an objective predictive function $f(\cdot)$, such that $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$.*

The predictive function $f(\cdot)$ is expected to be learned from the training data, which consists of pairs $\{X_i, y_i\}$, with $X_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. Once learned, $f(\cdot)$ can be used to predict the label of a new unseen instance $X$, $f(\cdot)$ can be rewritten as $P(Y \mid X)$. In the EEG classification example, $\mathcal{Y}$ is the set of all possible event labels, for example, 'left hand' and 'right hand'.

For simplicity, Pan and Yang (2010) only consider the transfer learning case where there is exactly one source domain $\mathcal{D}_s$ and exactly one target domain $\mathcal{D}_T$. Pan and Yang (2010) claim that this is the most popular case in literature and that the choice is therefore justified. The source domain data is then denoted as $D_S = \{(X_{S_1}, y_{S_1}), \ldots, (X_{S_{n_S}}, y_{S_{n_S}})\}$, the target domain data is denoted $D_T = \{(X_{T_1}, y_{T_1}), \ldots, (X_{T_{n_T}}, y_{T_{n_T}})\}$, with $X_{T_i} \in \mathcal{X}_T$ and $y_{T_i} \in \mathcal{Y}_T$. In typical cases, $0 \leq n_T \ll n_S$.

**Definition 3 (Transfer Learning)** *Consider $\mathcal{D}_S$ a source domain and $\mathcal{T}_S$ a learning task, and $\mathcal{D}_T$ a target domain and $\mathcal{T}_T$ a learning task. Transfer learning consists of trying to improve the learning of the target predictive unction $f_T(\cdot)$ in $\mathcal{D}_T$, using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, with $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$.*

The condition $\mathcal{D}_S \neq \mathcal{D}_T$ refers to $\mathcal{X}_S \neq \mathcal{X}_T$ or $\mathcal{P}_S(X) \neq \mathcal{P}_T(X)$, hence, the source and target domains have different feature spaces or marginal probability distributions. For example, in the EEG classification example, between the source event window set and the target event window set, either the features are different (e.g., longer event windows), or their marginal distributions are different (e.g., the event windows come from a different run, session, trial, or even participant).

Similarly, the condition $\mathcal{T}_S \neq \mathcal{T}_T$ refers to $\mathcal{Y}_S \neq \mathcal{Y}_T$ or $P(Y_S \mid X_S) \neq P(Y_T \mid X_T)$, hence, the source and target domains have different label spaces or conditional probability distributions. For example, in the EEG classification example, between the source task and the target task, either the label spaces are different (e.g., more possible events: 'left foot' and 'right foot'), or their conditional probability distributions are different (e.g., the source and target event windows are unbalanced in terms of classes).

When the source and target domains are equivalent, i.e., $\mathcal{D}_S = \mathcal{T}_T$, and the source and target tasks are equivalent as well, i.e., $\mathcal{T}_S = \mathcal{T}_T$, then the problem is a traditional machine learning problem. Transfer learning is said to be positive if it improves the performance compared to when only $\mathcal{T}_T$ and $\mathcal{T}_T$ are used, otherwise it is said to be negative. Depending on which of the inequalities apply, transfer learning can take on a variety of different forms. Lastly, if there exists some form of relation, explicit or implicit, between the feature spaces of two domains, it is said that these domains are related.

### 2.4.4 Related paradigms

A number of paradigms that are closely related to transfer learning exist. The following sections briefly explain them, clarifying their differences and similarities when compared to transfer learning.

#### 2.4.4.1 Supervised learning

Supervised learning is the traditional case of machine learning where $\mathcal{D}_S = \mathcal{T}_T$ and $\mathcal{T}_S = \mathcal{T}_T$. If no labelled data is available, this becomes unsupervised learning, which is mostly concerned with clustering, grouping, or dimensionality reduction.

#### 2.4.4.2 Semisupervised learning

Semisupervised learning lies in between unsupervised and supervised learning. The goal of semisupervised learning is to reduce the need for labelled data by also training a model using a, typically, large set of unlabelled data (Chapelle et al., 2006). When practising semisupervised learning, it is still the case that $\mathcal{D}_S = \mathcal{T}_T$ and $\mathcal{T}_S = \mathcal{T}_T$ and thus both labelled and unlabelled instances are drawn from the exact same distribution. On the other hand, when practising transfer learning, source and target data distributions are usually different. Pan and Yang (2010) state that in the context of transfer learning, the term 'semisupervised' is controversial due to its ambiguity. In the context of transfer learning, it is not clear whether the label information of the source or target domain is unavailable.

#### 2.4.4.3 Self-supervised learning

Self-supervised learning is an unsupervised learning paradigm that, instead of focusing on clustering, grouping, or dimensionality reduction, aims to draw conclusions for regression and classification tasks (Jing & Tian, 2021). Through domain knowledge, models can be made to understand the underlying properties of the data and in that way learn some form of supervisory signal, which is not exactly a label but does contain information regarding the data. In itself, self-supervised learning has no aspect of transfer learning. However, typically, self-supervised learning is used in a semisupervised or pretraining context. In such cases, the representations that the model learns when training on the supervisory signal, this task is often called the pretext task, are finetuned to a downstream task using a labelled dataset. As an example, imagine a large unlabelled dataset of images $U$. From this dataset $U$, a new dataset $V$ is generated by lowering the resolution of the images $u_i$ from the original dataset, resulting in images $v_i$. A possible self-supervised learning approach would then consist of training a model using the labelled dataset $U \times V$ which consists of input-label pairs $(v_i, u_i)$, with $u_i$ the label. Such labels are semantically different from class labels, as in self-supervised learning the goal is not to perform classification but rather to train a model to learn generalised and meaningful representations from unlabelled data. At this point, no transfer learning has been performed. If afterwards, the trained model that has learned meaningful representations in the pretext task is finetuned to classify a dataset of labelled images, one can speak of semisupervised learning if the images come from the exact same distribution as those in $U$ and from transfer learning otherwise.

#### 2.4.4.4 Multitask learning

Multitask learning consists of simultaneously learning a group of related tasks (Caruana, 1997). The goal of multitask learning is to improve performance on each task by taking advantage of the relations between them and thus generalising over them. The crucial difference between multitask and transfer learning is that the former treats each task as equivalently important, while the latter focuses on the target task. Nevertheless, both of these paradigms try to improve performance by transferring information.

### 2.4.5 Taxonomy

Pan and Yang (2010) establish a taxonomy of transfer learning based on their formal definition. Figure 2.13 gives a graphical reconstruction of said taxonomy. Three settings that relate to different source and target task situations are established: inductive transfer learning, transductive transfer learning, and unsupervised transfer learning.

The following sections briefly explain the three situations of transfer learning. For a more detailed review, the reader is referred to the works of Pan and Yang (2010) and Zhuang et al. (2021).

#### 2.4.5.1 Inductive transfer learning

**Definition 4 (Inductive Transfer Learning)** *Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, inductive transfer learning aims to improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, with $\mathcal{T}_S \neq \mathcal{T}_T$.*

In this transfer learning setting, the target task is different from the source task, regardless of the equivalence of the source and target domains. Inductive transfer learning paradigms require labelled data from the target domain to *induce* a target predictive function. Depending

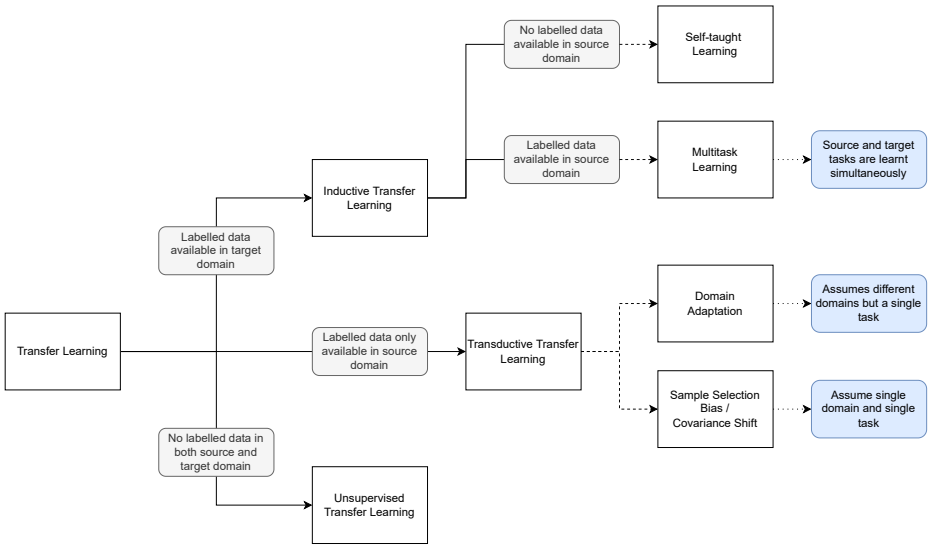



Figure 2.13: A taxonomy of transfer learning. Figure after Pan and Yang (2010)

.

on the availability of labelled data from the source domain, inductive transfer learning can be further divided into two groups. Either there is labelled data from the source domain or there is none. When labelled data is available from the source domain, Pan and Yang (2010) state that, aside from the mentioned differences, inductive transfer learning is similar to the multitask learning paradigm. When no labelled data is available from the source domain, depending on what is transferred, inductive transfer learning can be very similar to the self-taught learning paradigm (Raina et al., 2007). In the self-taught learning paradigm, the label spaces between the source and target domains may differ. Therefore, the information that is learned in the source domain can not directly be transferred to the target domain.

#### 2.4.5.2 Transductive transfer learning

**Definition 5 (Transductive Transfer Learning)** *Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, transductive transfer learning aims to improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, with $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$.*

With transductive transfer learning, the source and target tasks are equivalent, while the domains are different. With no labelled data from the target domain, transductive transfer learning paradigms require labelled data from the source domain. Pan and Yang (2010) further divide this paradigm into two cases: either the future spaces in the source and target domains are different, or they are the same but the marginal probability distributions are different. The second case is the most prevalent and Pan and Yang (2010) compare it to domain adaptation (Daumé & Marcu, 2006) or to situations where there is a sample selection bias (Zadrozny, 2004) or a covariate shift (Shimodaira, 2000).

#### 2.4.5.3 Unsupervised transfer learning

**Definition 6 (Unsupervised Transfer Learning)** *Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, unsupervised transfer learning aims to improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, with $\mathcal{T}_S = \mathcal{T}_T$ and $\mathcal{Y}_S = \mathcal{Y}_T$ are not observable.*

In unsupervised transfer learning there is no labelled data available from any domain. The target predictive function predicts latent variables such as clusters or reduced dimensions. Ample research regarding unsupervised transfer learning exists. Pan and Yang (2010) mention the transfer of clusters (W. Dai et al., 2008) and dimensionality reductions (Z. Wang et al., 2008).

### 2.4.6 Transfer learning in BCIs

Due to inter- and intra-person variability and the nonstationary nature of many neuroimaging modalities, every trial, run, session, or participant, can be seen as belonging to a slightly different domain. Coincidentally, transfer learning in BCI research has almost exclusively happened as some form of domain adaptation. Such transfer learning paradigms are mostly concerned with adapting a certain set of source domains to one or more specific target domains, where both sides can consist of trials, runs, sessions, or participants. A prevalent example of transfer learning in BCIs is domain adaptation of CSPs (Krauledat et al., 2008; Fazli et al., 2009). Additionally, domain adaptation paradigms that work through pretraining and finetuning have been successful, cross-participant (Dose et al., 2018; Fahimi et al., 2019) and within-participant (Schwemmer et al., 2018). Recently, however, efforts have been going towards a transfer learning paradigm very similar to domain adaptation, i.e., domain generalisation. Domain generalisation considers no specific target domains, but rather attempts to transform the data from the source domains in such a manner as to make them most useful for any new relevant target domain (Kostas & Rudzicz, 2020). Relevant examples of domain generalisation methods for BCIs are Riemannian alignment (Yger et al., 2017; Zanini et al., 2018; He & Wu, 2020) and Euclidian Space alignment (He & Wu, 2020; Kostas & Rudzicz, 2020). Additionally, variational autoencoders have been researched (Özdenizci et al., 2019) and, recently, self-supervision (Banville et al., 2019; Kostas et al., 2021). Given the complexity that inter- and intra-person variability and nonstationarity introduce to BCIs, transfer learning paradigms such as domain generalisation can be very interesting. In that light, the work that is presented in this thesis aims to get insight towards domain generalisation for EEG classification.

## 2.5 Natural language processing

### 2.5.1 Introduction

Processing natural language using computers is a practice that has existed since the second World war (Jones, 1994). The conversion between human languages or between human language and computer language is a topic that has been researched for a long time. Just as EEG classification, NLP has benefited a great deal from the recent advances in deep learning (Vaswani et al., 2017; Devlin et al., 2019; Z. Dai et al., 2019; Radford et al., 2019). Additionally, just as for all machine learning tasks, NLP suffers from the lack of labelled data. Because of this, various attempts have been made that try to create deep learning architectures that are able to learn with a minimum of labelled data (Devlin et al., 2019; Lan et al., 2020; Liu et al., 2019; Radford & Narasimhan, 2018; Yang et al., 2019). A common pattern in these attempts is self-supervised learning. NLP models

are often categorised by the structure of their input and output: sequence-to-vector models take as input a sequence of vectors and they output a single vector, vector-to-sequence models do the opposite, and sequence-to-sequence models take a sequence of vectors as input and they output a sequence of vectors. It is not difficult to imagine the NLP tasks that each of these model categories can be used for. The focus here lies on a specific type of sequence-to-sequence model: the transformer (Vaswani et al., 2017). This type of model has recently gained a lot of traction in the NLP field and various variants of the model exist, each with their own advantages and disadvantages.

The following two sections discuss, respectively, a brief history of NLP and the transformer model.

### 2.5.2 History

As mentioned, early forms of NLP have existed since the second world war. Patents for 'translation machines' were applied for even earlier (Daumas, 1965). The NLP machines that were devised at this time were very simplistic and would consist of translating input using a dictionary and a number of combination rules (Hutchins, 2004). It is only after Chomsky (1957) published 'Syntactic Structures' that NLP research started its evolution towards what is now known as NLP. At first, NLP systems mostly consisted of complex sets of hand-written rules (Winograd, 1971; Bundy & Wallen, 1984). However, in the 1980's, due to the introduction of machine learning algorithms that could be used for NLP, the systems drastically changed. Important factors for this change were the continuous increase in computational power and the fact that researchers became increasingly sceptical towards Chomsky's theories (Wermter et al., 1996). Chomsky claimed that language is generative in nature and thus that it can be generated by mathematical rules. In contrast, researchers that believed in the machine learning approach thought that linguistic rules are of no use to NLP and a purely statistical approach would be best. The linguistic approach to NLP saw a decline in produced results while the statistical approach gained more and more traction. The introduction of deep learning was crucial for NLP as nearly all present state-of-the-art NLP models are deep learning models. Presently, to cope with the deficit of labelled NLP datasets, various NLP research goes towards self-supervised and semisupervised learning algorithms (Qiu et al., 2020).

### 2.5.3 Transformers

Transformers are deep learning NLP models that were originally designed to combat the long-range dependency problems faced by RNN and LSTM models, which were previously used for sequence-to-sequence NLP tasks (Vaswani et al., 2017). Unlike RNNs, Transformers do not have recurrent connections to capture long-range dependencies. However, to tackle the issue of memory, transformers perceive entire sequences simultaneously and they make use of the concept of self-attention.

In what follows, the different techniques that are combined in the original transformer are discussed in a bottom-up manner. Afterwards, the architecture of the transformer is explained as a whole. The section ends with a discussion on a specific adaptation of the transformer, i.e., the one that is used in this thesis.

#### 2.5.3.1 Tokenisation

For a computer to be able to understand natural language, a transformation into numbers is required. A very simple example of such a transformation consists of assigning every possible

natural language word a different number and transforming input sentences using these assignments. More complex methods of tokenisation exist. For example, the transformer makes use of a bytepair encoding (BPE; Suarjaya, 2012). BPE consists of iteratively replacing the most frequent pair of bytes in a sequence with a single new byte. Sennrich et al. (2016) adapted this algorithm to allow for encoding words instead of bytes. While the original BPE algorithm merges the most frequent pair of bytes, the adapted algorithm merges characters.

#### 2.5.3.2 Text embeddings

While the transformation of natural language into numbers is enough to allow for computers to be able to handle it, learning from it is another question. For learning algorithms to be able to learn from natural language, the vectors of numbers that are given to them should contain meaning. This is what text embeddings are for. Text embeddings transform the variable-length token vectors into dense fixed-length real-valued vectors that encode the meaning of the word such that words that are closer in the vector space are expected to be similar in meaning. While text embeddings can be calculated using external pretrained models, the original transformer makes use of an embedding layer that learns word embeddings along with the complete model. The embedding layer consists of a large lookup table that stores an embedding for every possible token. These embeddings are adapted during the training of the model (Vaswani et al., 2017).

#### 2.5.3.3 Positional embeddings

As mentioned, the transformer makes up for the lack of recurrence using self-attention. By not using recurrence, transformers can be trained much faster than RNNs, as complete sentences can be handled at once. However, self-attention does not completely make up for losing recurrence. While transformers get complete sentences as input, each input word is treated separately. Therefore, the information that is contained in the position of the word in the sentence is lost. To solve this, a positional embedding vector, which captures the positions of the words in the input sequence, is added to each text embedding vector. Positional embeddings can be learned as well as fixed (Gehring et al., 2017). The original transformer makes use of fixed sinusoidal positional embeddings (Vaswani et al., 2017). The authors state that similar results were achieved using learned positional embeddings.

#### 2.5.3.4 Attention

The transformer substitutes recurrence in its layers with self-attention layers. Said layers try to capture the information that inputs contain about each other. The most simple version of self-attention works as follows. During training, three large matrices are learned: the query $W^q$, key $W^k$, and value $W^v$ matrices. When a sequence of embedded inputs enters a self-attention layer, each of its parts is multiplied by these matrices, resulting in three new vectors for each of the inputs, i.e., the query $\vec{q}$, key $\vec{k}$, and value $\vec{v}$ vectors:

$$\vec{q_i} = \vec{x_i} \cdot W^q \tag{2.9}$$

$$\vec{k_i} = \vec{x_i} \cdot W^k \tag{2.10}$$

$$\vec{v_i} = \vec{x_i} \cdot W^v \tag{2.11}$$

Subsequently, score vectors $\vec{z_i}$ are calculated for each combination of inputs by multiplying (dot product) their query vectors $\vec{q_i}$ and key vectors $\vec{k_i}$. The outputs are then calculated by summing

each of these products after multiplying them with their respective value vectors $\vec{v_i}$:

$$\vec{y_i} = \sum_{j=1}^{n} (\vec{q_i} \cdot \vec{k_j}) \cdot \vec{v_j} \tag{2.12}$$

The output vectors $\vec{y_i}$ are then supposed to contain contextual information. The above transformation process describes the simplest version of self-attention. To make self-attention more scalable and powerful, Vaswani et al. (2017) implement multi-heading. Splitting into attention heads or multi-heading consists of dividing an embedded input $\vec{x_i}$ into $n$ subvectors and applying self-attention to each of these separately. The original transformer uses an embedding size of 512. Therefore, it divides every embedded input vector into vectors of size $\frac{512}{n}$, where n signifies the number of heads. Each of these subvectors goes through their own self-attention process. Afterwards the $\frac{512}{n}$ resulting output vectors are concatenated into 1 and transformed such that they are suitable as input for the next layer. This projection transformation is implemented through a matrix product where the transformation matrix is learned by training the model. A good graphical representation of the multi-headed self-attention process is given by Alammar's (2018) "The Illustrated Transformer".

#### 2.5.3.5 Layer normalisation

To reduce training time as well as optimise learning, the transformer normalises the activities of its neurons in a layer-wise manner (Ba et al., 2016). Layer normalisation consists of normalising neuron activations using the mean and standard deviation of a single layer. Suppose $l$ is a hidden layer in a feed-forward network and $h_i^l$ are its inputs. Traditionally, the outputs $h_i^{l+1}$ of $l$ are calculated through a linear projection using weight matrix $W^l$:

$$h_i^{l+1} = f(a_i^l + b_i^l) \tag{2.13}$$

and:

$$a_i^l = w_i^l h_i^l \tag{2.14}$$

with $f(\cdot)$ a non-linear activation function and $b_i^l$ a scalar bias parameter. Ba et al. (2016) state that there is a strong dependency between the gradients of the weights $w_i^l$ in a layer and the outputs $h_i^{l-1}$ of the previous layer and that this can result in deep learning networks suffering from the 'covariate shift problem'. Such networks are so fixated on the patterns that they learned during training that when their input distribution changes (shifts), they fail to understand new patterns. To reduce this dependency, Ba et al. (2016) suggest to normalise a layer's activation values $a_i^l$:

$$a'^l_i = \frac{a_i^l - \mu^l}{\sigma^l + \epsilon}$$

where $\epsilon$ ensures numerical stability for when the denominator becomes zero by chance and $\mu^l$ and $\sigma^l$ denote, respectively, the layer-wise mean and standard deviation:

$$\mu^l = \frac{1}{H} \sum_{i=1}^{H} a_i^l, \qquad \sigma^l = \sqrt{\frac{1}{H} \sum_{i=1}^{H} (a_i^l - \mu^l)^2}$$

Additionally, the resulting normalised activation values $a'^l_i$ are to be scaled and shifted:

$$a''^l_i = \gamma a'^l_i + \beta$$

where $\gamma$ and $\beta$ are learnable scaling and shifting parameters, respectively.

#### 2.5.3.6 Architecture

The original transformer first introduced by Vaswani et al. (2017) consists of an encoder-decoder architecture. Figure 2.14 gives a graphical representation of the transformer architecture.

Once the words in an input sequence are tokenised and embedded, they are passed on to a set of encoder blocks. Each encoder block consists of a multi-headed attention layer followed by a feed-forward layer. The transformer's feed-forward layers consist of fully connected feed-forward neural networks that are its processing cores. Once self-attention has added the relevant context into the inputs, these layers will process them. The same feed-forward network is independently applied to each input. In the original transformer, the feed-forward networks consist of two layers.

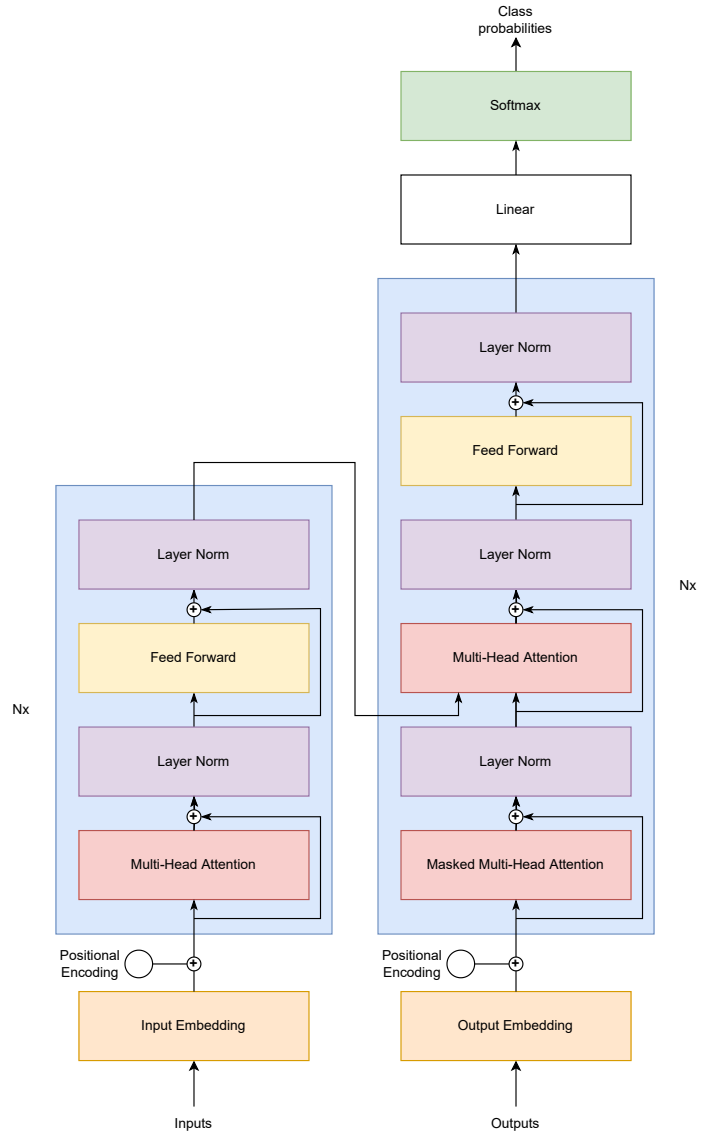



Figure 2.14: Transformer architecture. Figure after Vaswani et al. (2017).

The shared hidden layer between them has four times the size of the embedding size. Vaswani et al. find a factor of 4 to be enough to provide the transformer with sufficient representational capacity to be able to handle a variety of tasks. After every self-attention layer and feed-forward layer, the transformer applies layer normalisation in combination with a residual connection. Figure 2.14 shows that before every layer normalisation, an addition operation is performed, adding the inputs of the previous layer to its outputs. Residual connections are often implemented to allow gradients to flow through the network directly, counteracting the vanishing gradients problem.

The decoder part of the transformer is autoregressive, meaning that it begins with a start-of-sequence token and iteratively generates the next token. Every iteration, its generated output is added to its inputs. The decoder part stops once it generates an end-of-sequence token. Just as for the encoder part, inputs are tokenised and embedded. The embedded vectors are then passed on to a set of decoder blocks, as many as there are encoder blocks. Each decoder block consists of a masked multi-headed attention layer, a multi-headed attention layer, and a feed-forward neural network. The decoders also follow each of these up with a combination of layer normalisation and a residual connection. Masked self-attention consists of setting to infinity the scores of inputs with respect to inputs that have not yet been generated. Doing this is required as the future inputs are not yet accessible. The second multi-headed attention layer in the decoder blocks uses the outputs of the encoders as key and query vectors and the outputs of the masked attention layer as value vectors. By doing this, the encoder part's input is compared to the decoder part's input and thereby the decoders can learn which encoder inputs are to be focused on.

The set of decoder blocks is followed by a linear layer and a softmax layer, allowing to calculate probabilities for each possible output token. The token that corresponds to the highest probability is the output of the model.

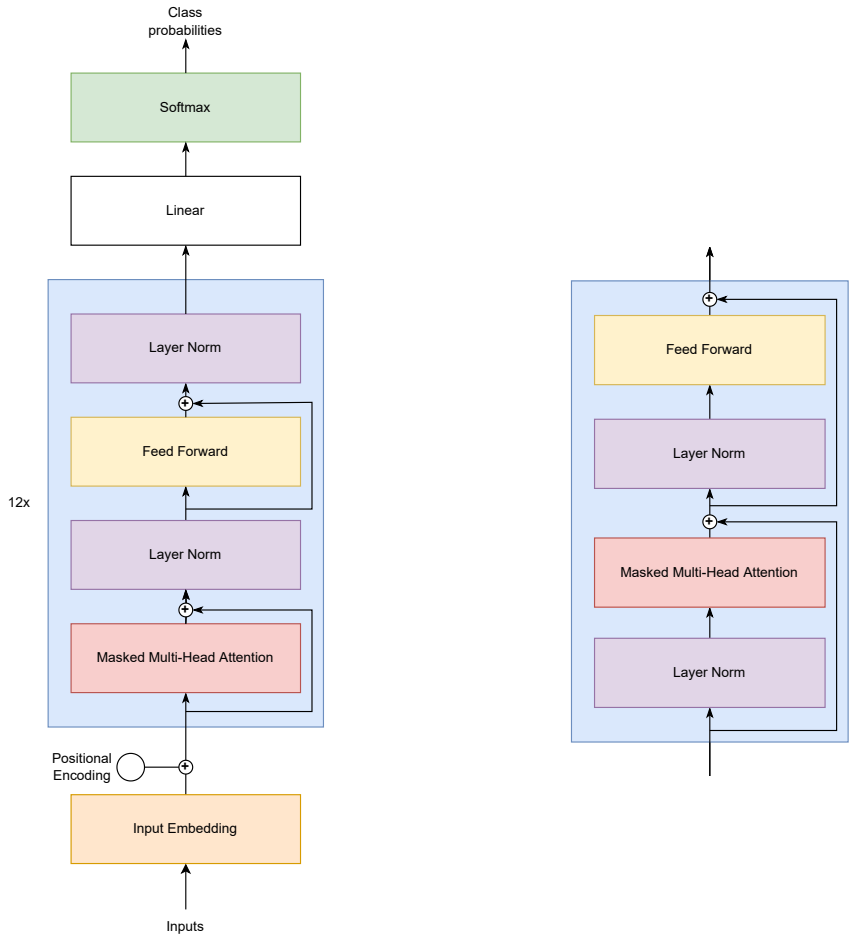
#### 2.5.3.7 The Generative Pretrained Transformer

Many successful adaptions of the transformer exist, for example BERT (Devlin et al., 2019), which consists of only encoders, and GPT (Radford & Narasimhan, 2018), which consists of only decoders. In the context of this thesis, the latter is discussed.

The Generative Pretrained Transformer (GPT; Radford & Narasimhan, 2018) and its successors GPT2 (Radford et al., 2019) and GPT3 (Brown et al., 2020) are adaptations of the transformer that consist of only the decoder part. Figure 2.15a gives a graphical representation of the architecture of GPT. The model consists of 12 decoder blocks, each of which consists of a masked multi-headed attention layer and a feed-forward layer. Just as the transformer, GPT follows each of these up with a combination of layer normalisation and a residual connection. The decoder blocks are preceded by a text and position embedding process. While transformers use a learned text embedding and a fixed position embedding, for GPT, the decision was made to learn both. Similarly to how the transformer generates the next token, i.e., through a linear layer and a softmax layer, GPT can be used to generate following tokens.

It is important to note that there is a significant difference between GPT and its successor GPT2, which is used in this thesis. Figure 2.15b gives a graphical representation of a GPT2 decoder unit. While GPT applies layer normalisation after each attention layer and feed-forward layer, GPT2 applies it before. Furthermore, GPT2 implements an additional layer normalisation after the final attention layer. GPT2 still applies residual connections after the attention and feed-forward layers. However, at initialisation, the weights of the residual connections are scaled by a factor of $\frac{1}{\sqrt{N}}$, where $N$ is the number of residual connections (Radford et al., 2019).

While GPT3 is the most recent version of the Generative Pretrained Transformer, in this thesis, the choice is made to use GPT2. The second version of GPT has had time to be extensively




(a) GPT (b) GPT2 decoder unit

Figure 2.15: GPT architectures. Figures after Radford and Narasimhan (2018).

analysed in literature. The work of K. Lu et al. (2021) on which this thesis builds makes use of GPT2.

Particularly interesting, in the context of this thesis, is the manner with which GPT is trained. GPT and its successors are trained in two phases, a self-supervised pretraining phase and a supervised finetuning phase (Radford & Narasimhan, 2018; Radford et al., 2019; Brown et al., 2020). In the self-supervised pretraining phase, given a large unlabelled corpus of language tokens $\mathcal{U} = \{u_1, \ldots, u_n\}$, a standard language modelling objective is used to maximise the following likelihood:

$$L(\mathcal{U}) = \sum_i logP(u_i \mid u_{i-k}, \ldots, u_{i-1}; \Theta) \tag{2.15}$$

with $k$ the size of the context window, and the conditional probability $P$ modelled using the GPT architecture with parameters $\Theta$. In other words, given a variable length sequence of language tokens, GPT is trained to learn to predict the next token. This is a self-supervised learning

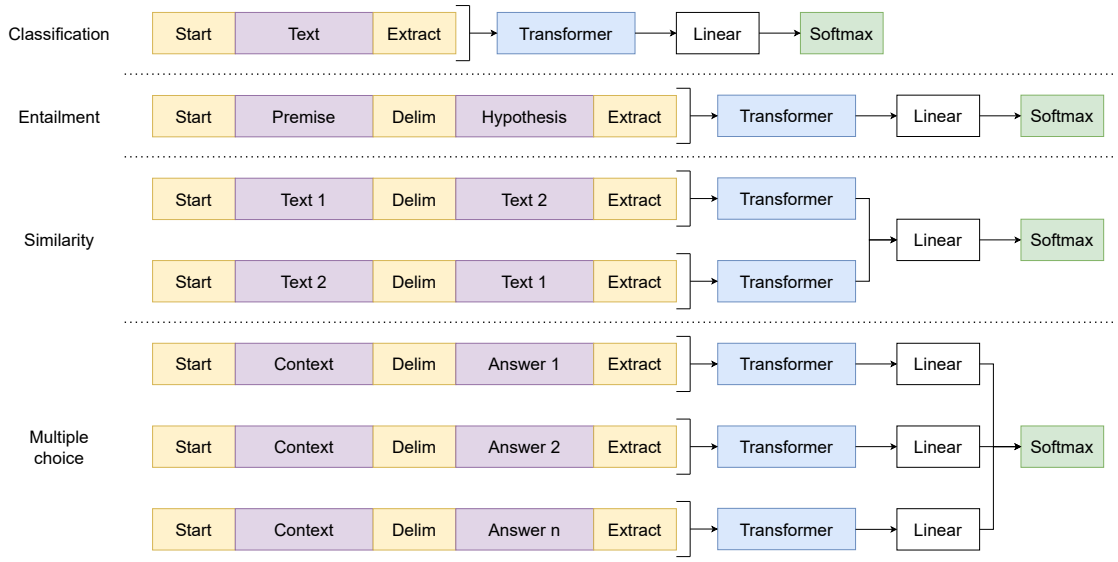



Figure 2.16: GPT input formats for finetuning on different tasks. Figure after Radford and Narasimhan (2018)

paradigm, as the training data is unlabelled and the learned information is intended to be used for regression or classification. Interestingly, due to the flexibility of natural language, GPT when trained this way, can be used for a variety of tasks. For example, a translation task could be performed by giving GPT the sequence: (translate, text in language 1, text in language 2). In fact, the pretrained model can even be further finetuned on such tasks in a supervised manner. Figure 2.16 gives a graphical representation of a number of input formats that Radford and Narasimhan use to finetune GPT. In doing so, they find state-of-the-art performances for a number of example tasks that conform to these input formats. These are all improved upon by the model's successors. In the article on GPT2, Radford et al. reason that the prevalence of single task training is the cause of a lack of generalisation that they observe in NLP models. They claim that multitask learning should result in a general performance improvement (McCann et al., 2018; Bowman et al., 2018; Yogatama et al., 2019). However, multitask training requires large labelled datasets, which is why Radford et al. implement self-supervised pretraining. Moreover, they state that by doing so, GPT and its successors are actually performing multitask learning.

# Chapter 3

# Related work

## 3.1 Introduction

While the contribution that this thesis presents is novel, it is preceded by a substantial corpus of research into related foundational topics such as deep learning and transfer learning, as well as much more specific related topics such as transfer learning in BCIs and transfer between significantly different domains. The sections in this chapter discuss the research that came before the research question that this thesis asks, thereby, clarifying its relevance. Initially, the more foundational topics are mentioned, the state-of-the-art is explained through examples of successes and issues. Gradually, the topics become more specific.

## 3.2 Deep learning and transfer learning

The reintroduction and growth of deep learning in recent years has meant great advancement for many fields of research (LeCun et al., 2015). Said growth powered by yet another process of evolution, i.e., the endlessly increasing number of transistors that can be placed on an integrated circuit, as dictated by Moore's (2006) law. Computational power increasing every year has allowed for successful applications of deep learning such as in computer vision (Cohen et al., 2018; Elsayed et al., 2018; Zamir et al., 2018; H. Zhang et al., 2019) and more specifically, image recognition (Krizhevsky et al., 2012; Zeiler & Fergus, 2014; Szegedy et al., 2015; F. Wang et al., 2017; Sankaranarayanan et al., 2018), NLP (Vaswani et al., 2017; Devlin et al., 2019; Radford & Narasimhan, 2018), computational neuroscience (DiCarlo et al., 2012; Yamins et al., 2014; Kriegeskorte, 2015, 2015), and many more. Another contribution to the success of deep learning is more data becoming available. An example is the world wide web that grows every day (Aghaei, 2012), allowing crawlers to collect a variety of data such as text for NLP (Elbaz, 2012; Ortiz Suárez et al., 2019). In the context of BCIs, the recent development in microelectrode array technology has made it possible to record neural activity of large populations of neurons (Jun et al., 2017; Raducanu et al., 2017; Saxena & Cunningham, 2019). In this case, the computational neuroscience community has even had to come up with novel methods that can learn from the new form of high-dimensional data (Stevenson & Kording, 2011).

Nevertheless, while more data becomes available, the need for it remains high and the increasing complexity of models makes it that data scarcity is still an issue. Moreover, handling immense datasets is hard (Oussous et al., 2018) and data can be biased or unrepresentative of real-world situations (D'Amour et al., 2020). The promise of deep learning algorithms is that

feature extraction can be done almost entirely by the model itself, i.e., end-to-end learning. However, this means that both features and classification (or regression) need to be learned from the scarce supply of data. Particularly suffering from data scarcity are supervised learning algorithms. They require labelled data, which, in most domains, is very costly to acquire. A variety of attempts have been made at reducing this issue. Data augmentation is implemented in NLP (Wei & Zou, 2019; S. Y. Feng et al., 2021) and in computer vision (Shorten & Khoshgoftaar, 2019). While such an approach can be very successful (Perez & Wang, 2017; Mikołajczyk & Grochowski, 2018), it is labour intensive. More desirable approaches consist of using existing data from related domains through transfer learning or using unlabelled data through unsupervised learning. In computer vision, state-of-the-art results have been achieved through supervised pretraining (Girshick et al., 2014; Girshick, 2015; Simonyan & Zisserman, 2014; Shelhamer et al., 2017), using large labelled datasets such as the ImageNet dataset (J. Deng et al., 2009). This transfer learning paradigm consists of initialising a deep learning network's parameters through a supervised learning task on a different but related labelled dataset, usually much larger than the one associated with the downstream task. It is important to note that these datasets are all labelled datasets, which, as said, are often very costly or even infeasible to acquire. Additionally, recent research has shown that the supervised pretraining paradigm does not necessarily improve performance on the downstream task even for tasks where it is assumed that it should (Zoph et al., 2020). In trying to address both of these issues, the NLP field has presented a different approach that has recently resulted in state-of-the-art performance, namely, self-supervised learning (Peters et al., 2018; Radford & Narasimhan, 2018; Radford et al., 2019; Devlin et al., 2019; Raffel et al., 2020). Instead of initialising a deep learning network's parameters using a supervised task, an unsupervised task can be used, using an unlabelled dataset. The advantage of this approach is that it is often the case that large unlabelled datasets are more easily acquired. Furthermore, unsupervised pretraining is thought to direct the parameters of a deep learning model towards basins of attraction of minima that are better with respect to the underlying data distribution (Erhan et al., 2010). A disadvantage of unsupervised pretraining is that it requires finding deep learning architectures and training objectives that can capture information in unlabelled data such that that information can be transferred to the solving of the downstream task. A successful example from the natural language processing field is the combination of a transformer-like model and a language modelling objective function (Radford & Narasimhan, 2018). While for natural language such pretraining objectives are intuitively easier to devise, for other fields of research, they can be more complex to come by. It is important to note that unsupervised pretraining has existed for a long time (Hinton et al., 2006; Ranzato et al., 2006; Bengio et al., 2006) and the natural language processing field is not the only research field that has been investigating its benefits. However, since the successes in the natural language processing field, it has become more popular. In the computer vision field, for example, unsupervised pretraining has been gaining traction as an alternative to its supervised counterpart (Chen et al., 2015; Grill et al., 2020; Henaff, 2020; van den Oord et al., 2019).

## 3.3 Neuroimaging and BCIs

One domain for which it is particularly difficult to obtain labelled data is that of brain signals. Signals such as EEG and MEG are very difficult to establish labelled datasets for. First of all, to get a substantially large dataset, a large number of appropriate participants need to be found and a large number of trials need to be ran for each of them. In addition, each trial needs to be carefully executed as to minimise the noise in the recordings. In the case of datasets where certain events are to be decoded from the recordings, it can be very difficult to isolate the events

from the other activities that are happening in the human body. Secondly, while technology has advanced significantly, recording devices for these types of signals are still far from perfect and often very costly. Aside from purely clinical applications, neuroimaging modalities are used in BCIs. Per definition, a BCI is a device that allows for a bridge between a brain and some form of computer. The research that encompasses this thesis consists of designing a noninvasive, mobile, online BCI. Applications of such a device are numerous. Examples are neural prosthetics (Bright et al., 2016; AL-Quraishi et al., 2018; Yildiz et al., 2020; Vilela & Hochberg, 2020), computer interfaces for people with handicaps (McCane et al., 2014; Gao et al., 2017; X. Zhang et al., 2018; Belkacem et al., 2020), and faster and better diagnosing (Fadzal et al., 2011; Mikołajewska & Mikołajewski, 2014; S. Raghu et al., 2020; Song et al., 2021). In addition to data scarcity, brain signals have other properties that make it hard for deep learning algorithms to learn from them, i.e., inter- and intra-person variability and nonstationarity. Because of these properties, the performance of models that are applied to brain signals varies a lot (Ahn & Jun, 2015; Lotte et al., 2018; Sannelli et al., 2019). Fundamentally, this means that the models lack in generalisation as they rely on characteristics particular to certain tasks, individuals, or recording sessions. In the computer vision field there is an understanding that almost all models learn low-level features in their more shallow layers (Krizhevsky et al., 2012; Yosinski et al., 2015; M. Raghu et al., 2019). Sadly, for most brain signals, there is no such understanding as there do not exist methods that are easily transferred to any task, individual, or recording session. This and the issue of labelled data scarcity has directed research concerning the decoding of neuroimaging modalities, and hence also that regarding BCIs, towards investigating the pretraining paradigm, both supervised (Y.-P. Lin & Jung, 2017; Dose et al., 2018; Schwemmer et al., 2018; G. Xu et al., 2019; Fahimi et al., 2019; Kostas & Rudzicz, 2020) and unsupervised (Banville et al., 2019, 2021; Kostas et al., 2021). Moreover, as computer vision research is shifting towards research into the latter, the same, and for the same reasons, seems to hold in the context of decoding neuroimaging modalities (Banville et al., 2021; Kostas et al., 2021). The work that is presented in this thesis contributes to such endeavours.

## 3.4 Transfer learning in BCIs

A particularly interesting attempt at transfer learning for BCIs is the work of Kostas et al. (2021). They reason that devising transferable deep learning models for EEG classification would not only present a new collection of models for EEG classification but would also allow for validating and even improving the existing models, as these are known to struggle with generalising. The approach they take mirrors the methods of the natural language processing field almost completely, driven by the great successes that these methods have been able to achieve. A transformer is pretrained on EEG using a self-supervised learning task. Afterwards, the transformer is fine-tuned to a variety of EEG classification tasks in a supervised manner. Just as for GPT, the self-supervised pretext task is designed in such a manner as to not focus on transferring to any specific downstream task, but rather to learn generalised and meaningful representations of EEG, i.e., domain generalisation. Two important contrasts with respect to this thesis are to be noted. Firstly, while Kostas et al. implement pretraining objectives and model architectures similar to those used in the natural language processing field, they do not use natural language pretraining but rather pretrain their model themselves using EEG data. Secondly, while it would be great if transformer-like models (Vaswani et al., 2017) could be given brain signals data directly, prior research has shown that reconstructing time-series data is very difficult and transformer-like models can therefore struggle to do so when given raw brain signals (Rivest & Kohar, 2020; Kostas et al., 2021; Jiang et al., 2021). Additionally, Kostas et

al. note that when using a BERT-like transformer (Devlin et al., 2019), which tries to learn to reproduce input tokens when parts of them are masked, with raw EEG data, instead of learning to reproduce the EEG data, an interpolation function might be learned (Jiang et al., 2021). Therefore, Kostas et al. claim to need a transformation from EEG to a representation that their model can handle and hence, they adapt an existing contrastive predictive coding (CPC; van den Oord et al., 2019) model called wav2vec (Baevski et al., 2020), to allow it to learn CPC representations of EEG data. The adapted model is called BENDR and it is pretrained on a large EEG dataset. While the BENDR model is applicable in the context of this thesis, there are two reasons for which it is not used. Firstly, the model is fairly new and would benefit from further analysis. Secondly, this thesis does not implement a *masked* language-modelling transformer, but rather GPT2, which is a *generative* language modelling model. Due to its generative nature, GPT2 is less likely to learn an interpolation function. The empirical evaluations presented in this thesis are designed to investigate to what degree GPT2 can be fine-tuned to EEG time-series data as well as to features extracted from this time-series data through a, when compared to BENDR, less complex method, namely, a windowed power spectral density analysis. In light of the work of Kostas et al., the results of these evaluations should provide insight into whether or not generative transformers can handle EEG time-series data and whether or not a complex feature extraction method such as BENDR is really necessary.

## 3.5 Transfer between significantly different domains

Conversely to pretraining an EEG model on EEG, another approach consists of pretraining a model using a dataset from another domain. As mentioned, the domain of brain signals is one for which it is hard to gather data. In that aspect, it might be interesting to pretrain a model using data from a domain that is known to have very large datasets, for example, natural language or images. K. Lu et al. claim that large language-pretrained NLP models, due to the immense datasets with which they are trained, can be seen as universal computation engines. According to K. Lu et al., what these models learn during training goes beyond natural language and steers towards general computation. To investigate their claims, they import GPT2 and fine-tune it to a variety of downstream tasks distinct from NLP. Examples are bit and list operations, the MNIST handwritten digit benchmark (L. Deng, 2012), and protein sequence tasks such as remote homology detection (Rao et al., 2019). It is important to note that these tasks all have a sequential structure, meaning that they are still somehow related to natural language. The same can be said of EEG. During fine-tuning, K. Lu et al. update only certain parts of GPT2 while others are kept fixed, namely, the self-attention layers and the feed-forward layers are kept fixed, while the linear input and output layers, the positional embeddings, and the layer-norm parameters, are fine-tuned. Such flexibility allows for devising experiments whose results can provide insight into the relationship between, on the one hand, the natural language pretraining and model architecture, and, on the other hand, performance for the downstream task. The results indicate that language-pretrained transformers can obtain strong performance on a variety of non-language tasks. In addition, K. Lu et al. state that a limitation of their analysis is that a specific model is used on a restricted set of tasks and that therefore, their work can serve as the foundation for future work investigating transfer between modalities. In that aspect, the work that is presented in this thesis acts on this statement.

A similar instance of transfer learning, but in the context of BCIs, is the work of G. Xu et al. (2019). They emphasise on the scarcity of labelled EEG data, as well as on the considerable time and resources required to train deep learning models from scratch. They look towards pretrained models as a solution. However, instead of looking at language models, G. Xu et al.

focus on pretrained image recognition models. Just as K. Lu et al., G. Xu et al. freeze certain layers of their pretrained model to get insights into how the knowledge transfer happens. In some way, the work that is presented in this thesis consists of a combination of the research questions from K. Lu et al. and G. Xu et al. where the source domain is that of K. Lu et al., while the target domain is that of G. Xu et al. It is important to note that K. Lu et al. reason that image-pretrained CNNs, such as those used by G. Xu et al., are arguably less likely to be universal computation engines. Their convolutional nature introduces an inductive prior that K. Lu et al. want to avoid. Nevertheless, computer vision is a data-rich research field and therefore, transfer to domains for which the CNN inductive prior holds can be interesting. In the context of classifying EEG, G. Xu et al. use time-frequency spectrum images of EEG to provide the image-pretrained CNNs with an input format that conforms to what they were pretrained with. A similar approach with similar features, i.e., windowed power spectral density analysis, is attempted in this thesis, albeit with language-pretrained transformers.

# Chapter 4

# Methodology

## 4.1 Introduction

This chapter discusses the methods implemented to get insight into how and to what degree a language-pretrained instance of GPT2 can transfer to the classification of EEG and whether or not the language-pretraining influences the performance. The software for data processing, model training, and evaluation are all implemented in Python. The *Python MNE* library is used for preprocessing and feature extraction (Gramfort et al., 2013). The *PyTorch* deep learning library is used for model training and evaluation (Paszke et al., 2019). The source code accompanying this thesis can be found on GitHub[1].

In what follows, Section 4.2 explains the EEG dataset that is studied. The model will be evaluated on two different representations of this dataset. Section 4.3 discusses preprocessing, which results in one data representation. Section 4.4 discusses feature extraction, which results in another data representation. Section 4.5 explains the model architecture. Section 4.6 explains how the model is trained. Lastly, Section 4.7 discusses how the model is evaluated and how different configurations are compared.

## 4.2 Data

The studied dataset is the Graz dataset A (Brunner et al., 2008) which is well-established (Naeem et al., 2006; Brunner et al., 2007; Dornhege et al., 2007; Barachant et al., 2012; Y. Zhang et al., 2019; Guan et al., 2019; Kostas et al., 2021) and known under multiple names: BNCI0012014, BCI competition IV 2a, and Graz dataset A. It was first published at the fourth BCI competition (Tangermann et al., 2012), however, it was created by Brunner et al. (2008) at the Graz university of technology with as original name: 'Graz dataset A'. Therefore, in what follows, the dataset is always referred to using said original name. The data used in the examples in Section 2.3 come from the Graz dataset A.

The Graz dataset A is a 4-class motor imagery dataset that consists of EEG data recorded from nine participants ($n_{\text{participants}} = 9$). The dataset was recorded using a cue-based paradigm that consisted of four possible motor imagery tasks ($n_{\text{classes}} = 4$), namely, the imagination of movement of the left hand (class 1), right hand (class 2), both feet (class 3), and tongue (class 4). The dataset consists of two sessions per participant, each consisting of 6 runs. In the context of the competition the two separate sessions were originally meant as training and evaluation split,

[1] `https://github.com/wulfdewolf/fpt-for-eeg`

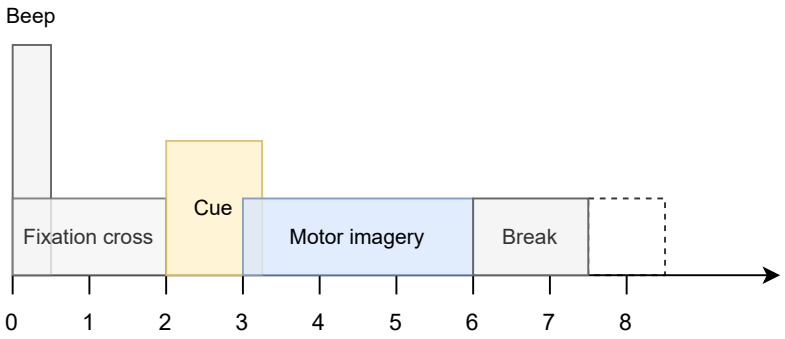



Figure 4.1: Graz dataset A trial timing scheme.

however, mirroring existing research, in this thesis, the two sessions are combined into a single large dataset. A single run consists of 48 trials, 12 per class. The timing scheme of the trials is depicted by Figure 4.1. A trial starts at $t = 0\,\text{s}$ when a fixation cross appears on a screen in front of the participant. After 2 seconds, a cue in the form of an arrow pointing in one of four directions, corresponding to the four classes, appears and stays on the screen for 1.25 s. When participants see this cue, they start performing the corresponding motor imagery task until the fixation cross disappears at $t = 6\,\text{s}$. Afterwards, a short break follows during which a black screen is shown.

The dataset consists of 25 channels of which 22 are EEG channels and 3 are EOG channels. The 22 EEG channels ($n_{\text{channels}} = 22$) were recorded using dry electrodes according to the standardised 10/10 system (see Figure 2.1b). The standardised electrode names of the used channels are: $F_Z$, $F_{C3}$, $F_{C1}$, $F_{CZ}$, $F_{C2}$, $F_{C4}$, $C_5$, $C_3$, $C_1$, $C_Z$, $C_2$, $C_4$, $C_6$, $C_{P3}$, $C_{P1}$, $C_{PZ}$, $C_{P2}$, $C_{P4}$, $P_1$, $P_Z$, $P_2$, $P_{OZ}$. The left mastoid electrode $T_9$ is used as reference while the right mastoid electrode $T_{10}$ is used as ground. The EEG signals were sampled at a sampling frequency of 250 Hz and band-passed filtered between 0.5 Hz and 100 Hz, complying with the Nyquist-Shannon sampling theorem (Shannon, 1949) and retaining all frequency bands relevant for motor imagery (see Table 2.1). Power-line noise is suppressed using a notch filter at 50 Hz. The three EOG channels record the electrical potentials that exist between the front and the back of the participant's eyes. They are provided to allow for certain artefact correction methods such as the ICA technique discussed in Section 2.3.3.2. However, in the context of the encompassing research which intends to create a mobile BCI with limited resources, such resource-intensive artefact correction techniques are not appropriate. Therefore, this thesis does not use them and thus the EOG channels are ignored.

## 4.3 Preprocessing

The signals of the 22 EEG channels are preprocessed as follows. Firstly, they are frequency filtered using a band-pass filter between 1 Hz and 45 Hz ($n_{\text{freqs}} = 45$). Motor imagery is thought to happen in the alpha and beta frequency bands (see Table 2.1) and thus the higher frequencies can be filtered out. Afterwards, the signals are downsampled to 125 Hz, complying with the Nyquist-Shannon sampling theorem (Shannon, 1949). Thirdly, they are cut into windows around the event cues, starting 2 s before the event cue and ending 4 s after. Every window is baseline corrected with respect to the first 2 seconds of the window. This means that the mean of the

first 2 seconds of the window is subtracted from the entire window. Finally, the windows are all standardised by channel. This means that for each channel, the mean and standard deviation is calculated across all windows and the channels' values are then transformed to their channel-wise z-score. For each of the 9 participants, preprocessing results in a dataset that consists of 564 windows (2 sessions of 6 trials each with 48 events, the first event is always dropped) each consisting of 22 channels by $6\,\mathrm{s} \cdot 125\,\mathrm{Hz} = 750$ samples ($n_{\text{samples}} = 750$).

## 4.4 Feature extraction

In addition to the preprocessed data, a dataset of features is established. In this case, the signals are again frequency filtered using a band-pass filter between 1 Hz and 45 Hz. However, they are not downsampled and also not standardised. The 6 s frequency filtered windows are divided into 10 subwindows, each of 0.6 s. For each of these 150 sample subwindows PSDs are estimated using a multitaper method (Thomson, 1982). If the windows were to be downsampled, the subwindows would contain much less samples, making the PSD estimation unrealistic. Additionally, the PSD estimation is in itself a form of normalisation, thereby, making the standardisation step obsolete. For each of the 9 participants, feature extraction results in a dataset that consists of 564 times a set of 10 PSDs across 22 channels by 45 frequencies. To eliminate the fourth dimension of this dataset, the channels and frequency bins are vectorised, resulting in vectors of size $22 \cdot 45 = 990$.

## 4.5 Model architecture

The model architecture mirrors that of K. Lu et al. (2021). It consists of GPT2, as explained in Section 2.5.3.7. The used GPT2 implementation and pretraining is that of the *Huggingface Transformers* library (Wolf et al., 2020). Both correspond to that of the original work by Radford et al. (2019), who created and investigated various sizes of the GPT2 model. In the work of K. Lu et al., and in this thesis as well, the smallest of the models is imported. This version of GPT2 has an embedding size of $d_{\text{GPT2}} = 768$, a maximum sequence length of $l = 1024$, $n_{\text{heads}} = 12$ attention heads, and consists of $n_{\text{decoders}} = 12$ decoder units, each as depicted by Figure 2.15b in Section 2.5.3.7.

Since GPT2 is used to embedded language tokens, initialising a separate input layer is crucial. This fully connected input layer is to learn how to query GPT2. The weight matrix of the linear input layer is initialised as an orthogonal matrix as described by Saxe et al. (2014), such an initialisation involves a scaling hyperparameter. Additionally, the input layer is extended with a dropout probability representing the probability of the layer zeroing some of the elements of its input, using samples from a Bernoulli distribution, following Pytorch defaults (Paszke et al., 2019). Doing this has proven to be effective for regularisation, as described by Hinton et al. (2012). Since GPT2 is a language model, it expects two-dimensional input representing a sentence of maximally $l$ language tokens. In the context of this thesis, this sequentiality is enforced by correctly formatting the data. When the model is fine-tuned using the time-series data, the $n_{\text{channels}} \times n_{\text{samples}}$ windows are transposed into a $n_{\text{samples}} \times n_{\text{channels}}$ format. Hence, the model gets as input a sequence of $d_{\text{in}} = n_{\text{samples}}$ ($\leq l$) tokens that consist of a single value for each of the channels. For further references, these type of tokens are denoted channel tokens. On the other hand, when the model is fine-tuned using the PSD features, the model gets as input a sequence of 10 vectorised $d_{\text{in}} = n_{\text{channels}} \cdot n_{\text{freqs}}$ representations, as explained in Section 4.4.

Initialising a separate output layer is important as well. The fully connected output layer is to learn to interpret the transformed output. It is important to note that only the last token of the transformed sequence goes through the output layer, mirroring K. Lu et al., who make this

choice to highlight that almost all the computation is being performed by GPT2. The linear output layer always results in vectors of size $d_{\text{out}} = n_{\text{classes}}$, i.e., one for each class. A softmax operation is applied to find the model's class prediction.

It is important to note that more complex input and outputs networks could be used, for example, a multilayer perceptron. However, mirroring K. Lu et al. once more, to emphasise on the fact that GPT2 is doing most of the computation, this is not done here.

As a whole, the model consists of the following parameters:

- **Input layer:** The fully connected linear input layer has $d_{\text{in}} \cdot d_{\text{GPT2}} + d_{\text{GPT2}}$ trainable parameters, where the additional $d_{\text{GPT2}}$ parameters correspond to the bias terms. In the case where the version of GPT2 that is used in this thesis is trained using the time-series EEG data, this results in $22 \cdot 768 + 768 = 17.664$ parameters.

- **Output layer**: The fully connected linear output layer has the same number of parameters as the input layer.

- **Layer normalisation parameters:** GPT2 applies layer normalisation twice per decoder unit. As explained in Section 2.5.3.5, layer normalisation consists of two learnable scale and shift parameters. When fine-tuning a model that implements layer-normalisation, it is standard practice to also fine-tune the scale and shift parameters as they are to model the statistics of the downstream task, whether this task belongs to a completely different domain or not (Rebuffi et al., 2017; Houlsby et al., 2019; K. Lu et al., 2021). In GPT2, layer normalisation accounts for $2 \cdot 2 \cdot d_{\text{GPT2}} \cdot n_{\text{decoders}}$ parameters. In the case of the version of GPT2 that is used in this thesis, this results in $2 \cdot 2 \cdot 768 \cdot 12 = 36.864$ parameters.

- **Positional embeddings:** The positional embeddings consist of $l \cdot d_{\text{GPT2}}$ learnable parameters. K. Lu et al. find that positional embeddings are similar across domains, however, fine-tuning them often results in slightly better performance. In the case where the version of GPT2 that is used in this thesis is trained using the time-series data, the positional embeddings account for $750 \cdot 768 = 576.000$ parameters. In the case where the PSD features are used, this results in significantly less parameters, i.e., $25 \cdot 768 = 19.200$.

- **Feed-forward neural networks:** The fully connected feed-forward neural networks in GPT2 all have a hidden layer of size $4 \cdot d_{\text{GPT2}}$. This means that the feed-forward neural networks in GPT2 account for $n_{\text{decoders}} \cdot (2 \cdot 4 \cdot d_{\text{GPT2}}^2 + 5 \cdot d_{\text{GPT2}})$ parameters. In the case of the version of GPT2 that is used in this thesis, this results in $12 \cdot (2 \cdot 4 \cdot 768^2 + 5 \cdot 768) = 56.669.184$ parameters.

- **Self-attention layers:** The masked multi-headed self-attention layers multiply input tokens by three learnable matrices (i.e., the query $W^q$, key $W^k$, and value $W^v$ matrices, see Section 2.5.3.4) $n_{\text{heads}}$ times, resulting in $3 \cdot n_{\text{heads}}$ vectors of length $\frac{d_{\text{GPT2}}}{n_{\text{heads}}}$. Including the shared biased terms, the $3 \cdot n_{\text{heads}}$ matrices account for $d_{\text{GPT2}} \cdot \frac{d_{\text{GPT2}}}{n_{\text{heads}}} \cdot 3 \cdot n_{\text{heads}} + 3 \cdot d_{\text{GPT2}}$ parameters. Subsequently, the $3 \cdot n_{\text{heads}}$ vectors of length $\frac{d_{\text{GPT2}}}{n_{\text{heads}}}$ are used to calculate $n_{\text{heads}}$ score vectors of length $\frac{d_{\text{GPT2}}}{n_{\text{heads}}}$. These are then concatenated into one large vector of length $d_{\text{GPT2}}$. Finally, this vector is transformed such that it becomes suitable as input for the feed-forward layer that follows after the self-attention layer. The transformation happens by passing the concatenated vector through a fully connected linear layer of size $d_{\text{GPT2}}$, which thus accounts for another $d_{\text{GPT2}} \cdot d_{\text{GPT2}} + d_{\text{GPT2}}$ parameters. Hence, the masked multi-headed self-attention layers in GPT2 account for $n_{\text{decoders}} \cdot [(d_{\text{GPT2}} \cdot \frac{d_{\text{GPT2}}}{n_{\text{heads}}} \cdot 3 \cdot n_{\text{heads}} + 3 \cdot d_{\text{GPT2}}) + (d_{\text{GPT2}} \cdot d_{\text{GPT2}} + d_{\text{GPT2}})]$ parameters. In the case of the version of GPT2 that is

used in this thesis, this results in $12 \cdot [(768 \cdot \frac{768}{12} \cdot 3 \cdot 12 + 3 \cdot 768) + (768 \cdot 768 + 768)] = 28.351.656$ parameters.

Clearly, the number of parameters is dominated by the feed-forward and self-attention layers. Luckily, these two comprise most of GPT2's computational power, which means that during fine-tuning, they are the layers that will often be frozen, thereby, significantly decreasing the time and memory needed to train the model.

## 4.6 Training

All training happens using the Adam optimiser (Kingma & Ba, 2017) following Pytorch defaults (Paszke et al., 2019), in combination with cross-entropy loss (Dayan et al., 1995)[2]. In the training of the model and in the model itself, five hyperparameters are involved:

- **Number of epochs:** the number of times to train the model on the training data and validate the model on the validation data.
- **Batch size:** the number of dataset entries, windows in the case of this thesis, to use in one forward-backward pass of gradient descent.
- **Learning rate:** a hyperparameter that represents the size of the steps that are taken by gradient descent.
- **Learning rate decay:** a hyperparameter that represents the factor with which the learning rate decays after each epoch. Gradually decaying the learning rate is often implemented to prevent gradient descent from overstepping and moving away from local minima. Unless otherwise stated, the learning rate decay is kept fixed at 1, meaning that the learning rate does not decay.
- **Dropout probability of the input layer:** the probability of the input layer zeroing some of the elements of its input, as described by Hinton et al. (2012). K. Lu et al. keep the dropout probability of their input layers fixed at 0.1 for most of their experiments, matching the dropout probability that is inherent in GPT2. Unless otherwise stated, the dropout probability is kept fixed at 0.1 here as well.
- **Orthogonal gain of the input layer:** the scaling hyperparameter involved in the initialisation of the input layer's weight matrix, as described by Saxe et al. (2014). K. Lu et al. keep the input layer's orthogonal gain fixed at 1.43 for most of their experiments, this is done here is as well.

To find good values for the hyperparameters, a 50-trial optimisation run is performed using the *Python Hyperopt* library's (Bergstra et al., 2013) implementation of a Tree-structured Parzen Estimator (TPE; Bergstra et al., 2011). A TPE models $P$(hyperparams|score) and $P$(score) by transforming the generative process of hyperparameters, replacing the distributions of the configuration prior with non-parametric densities (Bergstra et al., 2011, 2013). The hyperparameters that are optimised are the number of epochs, the batch size, and the learning rate. The search domains for these three hyperparameters are, respectively, $\{2^i | i \in \mathbb{N}, 1 \leq i \leq 6\}$, $\{2^i | i \in \mathbb{N}, 1 \leq i \leq 6\}$, and $[0.00005, 0.1]$. The TPE starts with 20 trials using random samples from these domains. After 20 trials, it starts its directed search. Due to time and resource constraints, not all model configurations are ran with hyperparameter optimisation.

[2] `https://scikit-learn.org/stable/modules/model_evaluation.html#log-loss`

## 4.7 Evaluation

The model is evaluated through a participant-wise cross-validation similar to those of Kostas and Rudzicz (2020) and Kostas et al. (2021). Classification accuracy is used as performance metric. Participant-wise cross-validation consists of setting aside the data from a single participant as test set. The remaining data is once more split, setting aside the data from another single participant as validation set and using the leftover participants as training set. Each of the $n_{\text{participants}}$ participants is individually chosen as validation participant, while the next participant is always chosen as test participant. The first participant is the test participant when the validation participant is the last participant. Every epoch, the model is trained with the data from the $n_{\text{participants}} - 2$ training participants and afterwards evaluated on the data from the validation participant. Training happens using batches. During the epochs, the values of the parameters that correspond to the current highest validation score are always retained. After the epochs, the model is evaluated once on the data from the test participant using the parameter values that correspond to the highest found validation score. The final cross-validation score is the average of the test participant scores.

Comparing the performance of two models always happens through a paired t-test. Such a test assumes that the paired differences of the test scores are normally distributed. Therefore, all paired t-tests are always preceded by a Lilliefors (1967) test to verify that the null hypothesis that states that the paired differences come from a normal distribution cannot be rejected. When more than two models are compared, a one-way ANOVA test is used (Girden, 1992). This test is always preceded by a number of separate tests that verify the ANOVA test's assumptions. The tests in question are a separate Lilliefors (1967) test to verify the normality of each of the groups of test scores as well as a single Levene's (1961) test to verify if the groups of test scores share a common variance. A significance level of $\alpha = 5\,\%$ is used for all tests. The tests that verify assumptions are only mentioned if they reject their null hypothesis, as in those cases, other tests need to be applied.

# Chapter 5

# Empirical Evaluations

## 5.1 Introduction

The following sections establish two sets of empirical evaluations designed to result in insight regarding this thesis's research question. In Section 5.2, the first set tries to grasp how and to what degree a frozen language-pretrained instance of GPT2 can transfer to the classification of EEG. In Section 5.3, the second set tries to get insight into the influence of the language pretraining. A summary of all the results can be found in Appendix A and on Weights & Biases[1].

## 5.2 How and to what degree can frozen language-pretrained GPT2 transfer to EEG classification?

### 5.2.1 Reasoning

To get insight into the answer to this question, the capability of a frozen language-pretrained instance of GPT2 to classify EEG is studied using two different representations of the EEG data. It is important to remember that K. Lu et al. (2021) investigate the ability of a language-pretrained GPT2 to transfer to various downstream tasks by freezing all of GPT2's feed-forward and self-attention layers and only fine-tuning the input layer, the output layer, the positional embeddings, and the layer normalisation layers. By freezing all of GPT2's feed-forward and self-attention layers, insight can be gained into the type of transfer that happens. If the model is able to achieve above-random classification scores, it can be concluded that positive transfer is possible. In this thesis, K. Lu et al.'s evaluation is repeated, but with the downstream task of classifying EEG, using both the EEG time series and the EEG PSD features as data. The results are compared to each other and to literature.

It is important to note that due to time and resource constraints, this is the only model configuration that is ran with hyperparameter optimisation. The values for the hyperparameters that are found in this optimisation run are used for all the following evaluations.

[1] `https://wandb.ai/wulfdewolf/lpt-for-eeg/reports/Transfer-learning-in-BCI-s-language-pretrained-transformers-for-EEG-classification--VmlldzoxOTIxNDU2?accessToken=r4hzxv3i86ovxcf01fdzcebnnpy79nc57stoew4gasvoboual6f2c93131ra4u1z`

Table 5.1: Hyperparameter values that produced the highest participant-wise average test classification accuracy after 50 runs using a TPE. The instance of GPT2 is language-pretrained and all of its feed-forward and self-attention layers are frozen.

(a) Time-series

| **Hyperparameter** | **Value** |
|---|---|
| number of epochs | 0.0004 |
| batch size | 32 |
| learning rate | 64 |

(b) Features

| **Hyperparameter** | **Value** |
|---|---|
| number of epochs | 0.00014 |
| batch size | 32 |
| learning rate | 64 |

### 5.2.2 Results

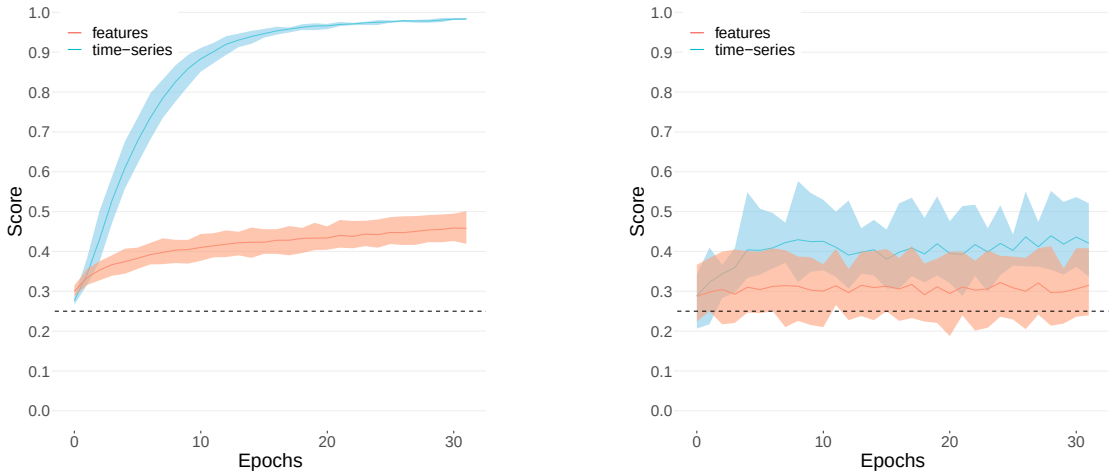
Table 5.1 gives the hyperparameter values that resulted in the highest participant-wise average test scores. Said test scores are presented by Figure 5.3 in the form of boxplots. For the same runs, Figure 5.1 depicts the training and validation classification accuracy of the model in function of the epochs. Figure 5.2 depicts the cross-entropy loss of the model in function of the epochs. All of these tables and figures always show results for the time-series data as well as for the PSD features.

Figures 5.1a and 5.2a show that the model learns best from the time-series data, as in that case the training accuracy grows up to 98 % and the cross-entropy loss drops to a value of 0.05. On the other hand, when learning from the PSD features, the model can not seem to reach the 50 % mark. This means that, even after training, the model is not able to correctly classify 50 % of its training set samples. Moreover, cross-entropy loss remains above 1. Figure 5.1b shows that both models have a hard time generalising. Both can not reach a participant-wise average validation classification accuracy of 50 %. However, the model that is trained using the time-series data seems to generalise better than the model that is trained using the PSD features. All of this is reflected by Figure 5.3, which shows that, on average, the model that is trained using the time-series data performs best, with a participant-wise average test classification accuracy of 41.6 %. A paired t-test indicates that this score is significantly higher than the 30.9 % the model achieves when it is trained using the PSD features ($p = 0.0011$).

It is important to note that when the model is trained using the time-series data, it clearly suffers from overfitting. Firstly, during training, the model reaches 98 % classification accuracy and near-zero cross-entropy loss, while its validation accuracy remains under 50 %. Secondly, while the validation score in Figure 5.1b slowly increases, validation cross-entropy loss, as shown by Figure 5.2b, instead of decreasing, rapidly increases. Both of these phenomena are universally known indications of overfitting (Ghojogh & Crowley, 2019; Ying, 2019).

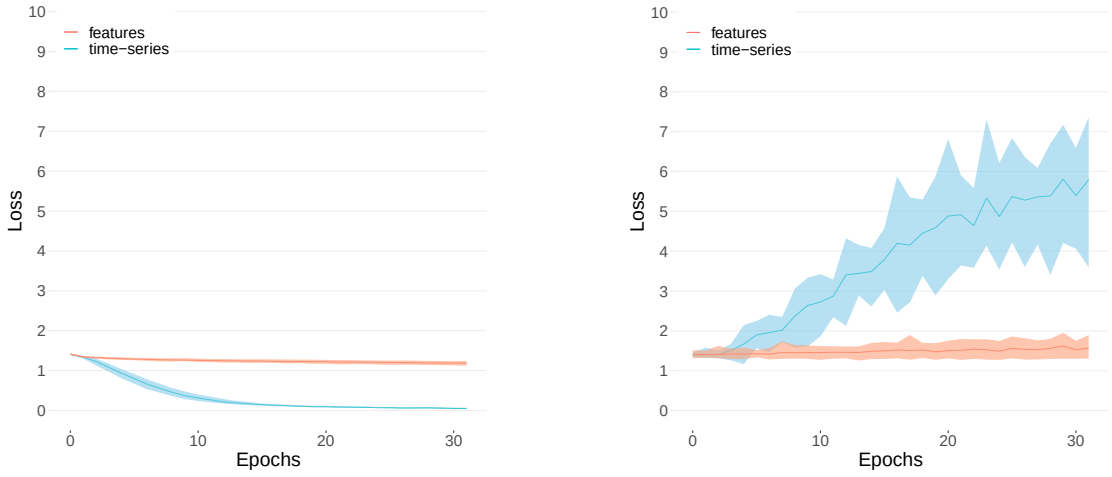
### 5.2.3 Addressing overfitting

A model that suffers from overfitting learns too much detail from the data, resulting in a high generalisation error. In the case of large deep learning models that are trained using relatively small datasets, overfitting can be seen as memorisation. The model has enough weights to be able to memorise almost every training input it gets, linking it to the correct output. However, when the model is later used to classify an unseen input, it often fails. The issue of overfitting/memorisation is omnipresent in deep learning and various works that try to address it exist, for example, regularisation (Krogh & Hertz, 1991; Kukacka et al., 2017) and dropout (Hinton et al., 2012). A preliminary attempt at reducing overfitting is made by training the model with decay and dropout rates of, respectively, 0.9 % and 0.2 %. This evaluation is only performed for




(a) Training (b) Validation

Figure 5.1: Participant-wise average training and validation classification accuracy of the model during fine-tuning, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and all of its feed-forward and self-attention layers are frozen. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.




(a) Training (b) Validation

Figure 5.2: Participant-wise average training and validation cross-entropy loss of the model during fine-tuning, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The instance of GPT2 is language-pretrained and all of its feed-forward and self-attention layers are frozen. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.

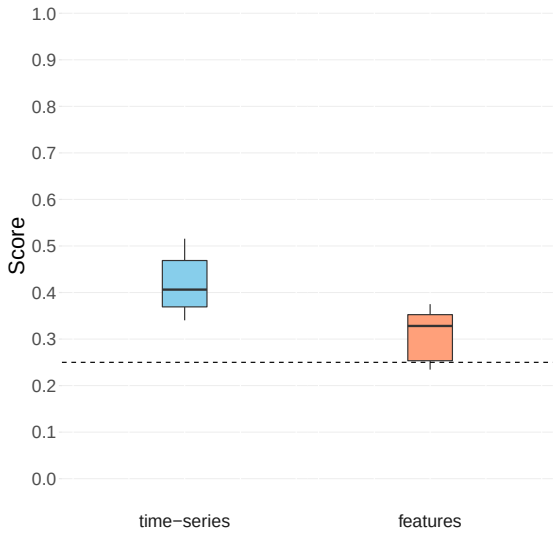


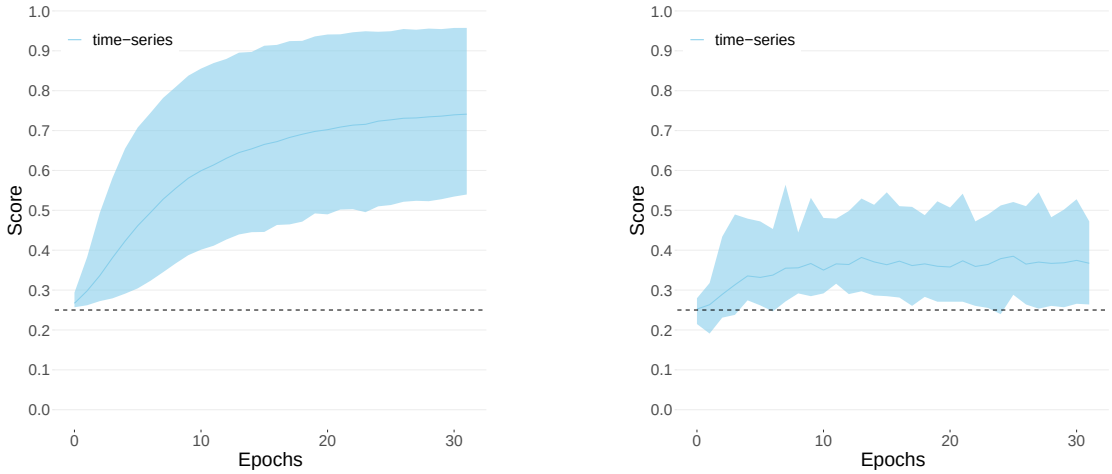

Figure 5.3: Participant-wise average test classification accuracy of the model after fine-tuning, for both the time-series data (blue) and the PSD features (red). The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and all of its feed-forward and self-attention layers are frozen. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.

the time-series data, as overfitting is not really an issue when the model is trained using the PSD features.

Figure 5.4 shows that the introduction of decay and an increased dropout rate succeeds in lowering the model's training scores, in some cases. For certain groups of training participants the model still reaches high training scores. Additionally, while the model's validation scores remain slowly increasing, Figure 5.5 shows that the model's validation cross-entropy loss still increases but much less steeply than before. Figure 5.6 shows that while overfitting might have been decreased, the generalisation error became higher. When trained in this configuration, the model achieves a participant-wise average test classification accuracy of 35.9 %, which is significantly lower than the 41.6 % that the model achieved without decay and with a lower dropout rate of 0.1 % when a significance level of 0.07 % is chosen (according to a paired t-test, $p = 0.061$). However, the obvious outlier in Figure 5.6 that is not present in Figure 5.3 makes it reasonable to assume that it is indeed lower.

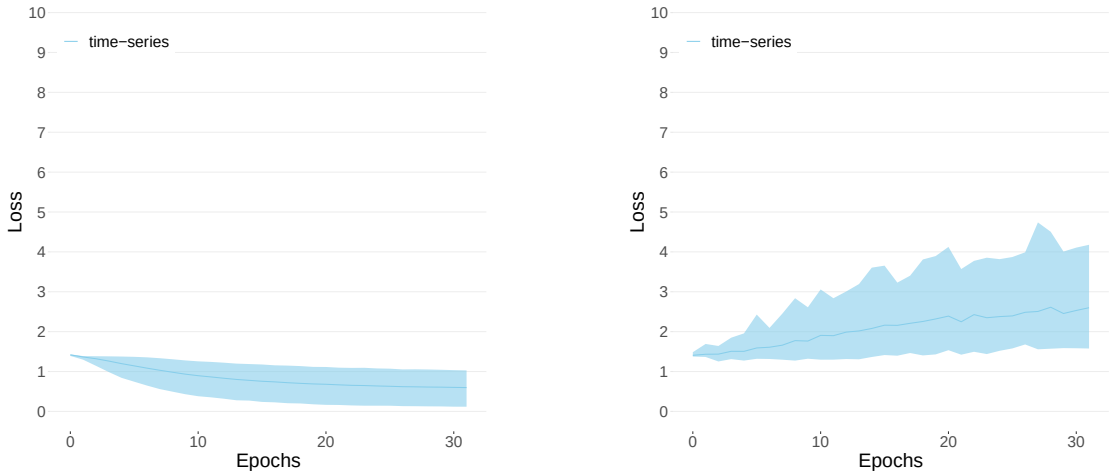
### 5.2.4 Discussion

While the achieved classification scores are not spectacular, it seems safe to state that GPT2, with all of its feed-forward and self-attention layers frozen, is able to transfer to classifying EEG data. Moreover, it seems to be able to perform better when trained using the time-series data compared to when trained using the PSD features. In fact, the participant-wise average test classification accuracy of the model when trained using the time-series data (41.6 %) is similar to the 42.6 % achieved by Kostas et al. (2021), on the exact same dataset, by classifying their BENDR representations using a linear layer followed by a softmax layer. Kostas et al. also pass their BENDR representations to an EEG-pretrained BERT-like transformer and in that case




(a) Training (b) Validation

Figure 5.4: Participant-wise average training and validation classification accuracy of the model during fine-tuning, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and all of its feed-forward and self-attention layers are frozen. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1. Additionally, the decay and dropout rates are, respectively, 0.9 % and 0.2 %.




(a) Training (b) Validation

Figure 5.5: Participant-wise average training and validation cross-entropy loss of the model during fine-tuning, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The instance of GPT2 is language-pretrained and all of its feed-forward and self-attention layers are frozen. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1. Additionally, the decay and dropout rates are, respectively, 0.9 % and 0.2 %.

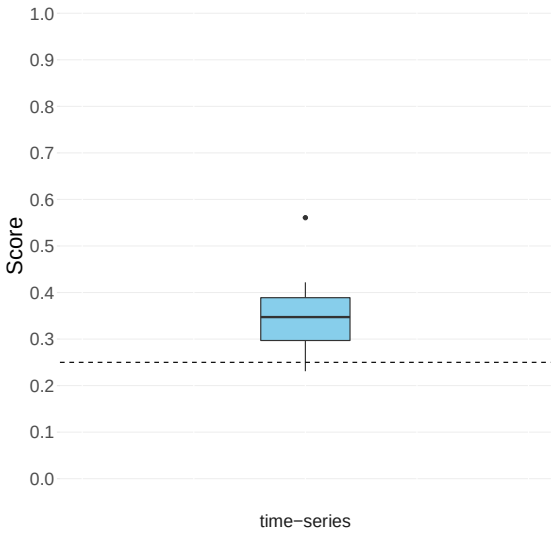



Figure 5.6: Participant-wise average test classification accuracy of the model after fine-tuning, for both the time-series data (blue) and the PSD features (red). The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and all of its feed-forward and self-attention layers are frozen. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1. The learning rate decays with a factor of 0.95.

the test scores are lower, i.e., around 30 % and thus similar to what the model here achieves when trained using the PSD features. It is important to note that, when compared to the EEG time-series data, the PSD features have a considerably lower temporal resolution. The spatial resolution of the PSD features is supposed to be equivalent to that of the time-series data. However, the vectorisation technique that is used to encode the PSDs of multiple channels into a single token might obstruct GPT2's ability to extract the spatial information that these contain. Given the sequential nature of transformer-like models and the fact that the spatial resolution of the PSD features is not explicitly reduced, temporal resolution seems to be more of a problem. In any case, the higher average test score of the model when trained using the temporally untouched time-series data suggests that establishing a type of features that minimally reduces temporal resolution could be favourable when training a transformer-like model using EEG data.

When comparing the model's performance to the state-of-the-art, two notes must be made. Firstly, the goal of this thesis is not to outperform the state-of-the-art but to investigate potential knowledge transfer and the feasibility of training a language-pretrained transformer on EEG data. In that light, the achieved classification scores are expected to be much lower than those of the state-of-the-art. Secondly, for a correct comparison, the evaluation strategy needs to be taken into account. On the same dataset, a variety of works report average test scores around 60 %, achieved through a multiclass CSP method, the reference method at the time of the competition in 2008 (Ang et al., 2012). Barachant et al. (2012) report average test scores up to 70 % achieved using models based on Riemannian geometry. However, all of these works implement intra-person training and evaluation, i.e., models are trained and evaluated using EEG data from a single participant. On the other hand, in this thesis, participant-wise cross-validation is used, i.e., models are trained using data from multiple participants and evaluated on data from an unseen

participant. There are two reasons behind the choice of this approach. Firstly, if an enormous model such as GPT2 were to be finetuned using only the data from a single participant, overfitting would become even more of an issue. Secondly, the focus of this thesis lies on transfer learning for BCIs, with the aim of creating pretrained models that be can easily and quickly be plugged into mobile BCIs. In such a context, the use of participant-wise cross-validation is interesting as it allows for evaluating the model's transfer capabilities. Remember that the participant-wise cross-validation approach was adopted from the work of Kostas and Rudzicz (2020), who in their turn adopted it from the work of He and Wu (2020). The focus of these works also lies on transfer learning in BCIs and thus it is these works that the performance of the model here should be compared to. He and Wu try to address inter- and intra-person variability by aligning the different participants' EEG data in Euclidian space (EA) before using a traditional combination of CSP and Linear Discriminant Analysis (LDA; Fisher, 1936) for classification. The best reported average test score lies around 73 %. Kostas and Rudzicz emphasise on the shallowness of deep learning models for EEG classification and implement a relatively deep model to investigate its cross-participant transfer capabilities when trained using EA processed EEG data. The model achieves average test scores up to 76 %. Both of these works report performances remarkably better than that achieved here. Nevertheless, the most relevant comparison remains that to the EEG-pretrained BERT-like transformer approach of Kostas et al. (2021), which, as mentioned, performs worse than language-pretrained GPT2. Moreover, properly addressing the overfitting issue of the model and EA processing the EEG data could further widen the gap between these two approaches.

## 5.3 What is the influence of the language pretraining?

### 5.3.1 Reasoning

A frozen instance of GPT2 is able to transfer to classifying EEG, to some extent. The question remains to what degree its language pretraining is beneficial towards its performance. In that regard, a variety of configurations are evaluated in order to gain insight into the influence of the language pretraining. It is important to note that, to get a complete image of the capabilities of these configurations, each would require individual hyperparameter optimisation. However, due to time and resource constraints they are ran with the hyperparameter values from Section 5.2, which should still allow for getting a general idea. The considered configurations are the following:

- **Randomly initialised:** an instance of the model where GPT2 is randomly initialised, as opposed to language-pretrained. For this configuration, none of GPT2's layers are frozen. The performance of this configuration should provide insight into the overall capabilities of the GPT2 architecture for classifying EEG. Moreover, it should confirm whether or not the language-pretraining is beneficial. The above-random test scores that the model achieves when its feed-forward and self-attention layers are frozen could be the result of the other unfrozen parameters learning from the EEG data. If the test score of this configuration is significantly lower than that which was achieved in Section 5.2, it can be concluded that the language-pretraining is beneficial.

- **Completely unfrozen:** an instance of the model where GPT2 is language-pretrained and none of its layers are frozen. Generally, this configuration would be expected to achieve higher test scores than those achieved in Section 5.2, i.e., when all of GPT2's feed-forward and self-attention layers are frozen. However, when overfitting is already an issue when all of these are frozen, it is safe to assume that unfreezing will increase overfitting. Therefore,

the performance of this configuration when trained using the time-series data is expected to be worse than that achieved in Section 5.2.

- **Shallow-to-deep unfrozen:** 11 configurations that are the result of unfreezing GPT2's decoder units one by one, starting from the shallowest. By gradually unfreezing GPT2 in this manner, insight can be gained into the depth with which GPT2's language pretraining can transfer to EEG classification. Additionally, in the case where the model is trained using the time-series data, insight can be gained into the overfitting issue observed in Section 5.2. A hypothesis is that, by unfreezing the more shallow layers and freezing deeper layers, overfitting can be reduced. Memorisation often occurs in the deeper layers of deep learning models. In those layers, models tend to learn the smallest details of their input. By freezing these deeper layers but allowing the more shallow layers to adapt, such memorisation could be reduced. Naturally, this all depends on whether or not GPT2's language modelling knowledge that is in the frozen decoder units can transfer to EEG data as well as whether or not the unfrozen decoder units, which start from a language-modelling background, can learn to handle the representations they receive. Hence, the hypothesis does not only relate to overfitting, it also relates to the depth with which the language-pretraining can be ported to EEG classification.

- **Deep-to-shallow unfrozen:** 11 configurations that are the result of unfreezing GPT2's decoder units one by one, starting from the deepest. By gradually unfreezing GPT2 in this manner, once more, insight can be gained into the depth with which GPT2's language pretraining can transfer to EEG classification as well as into overfitting. A hypothesis is that, by unfreezing the deeper layers and freezing the more shallow layers, overfitting will increase. Unfreezing the deeper layers could result in the model learning too much details and, thereby, overfitting its input data.

Each of the above configurations is always evaluated for both the EEG time series data and the EEG PSD features.

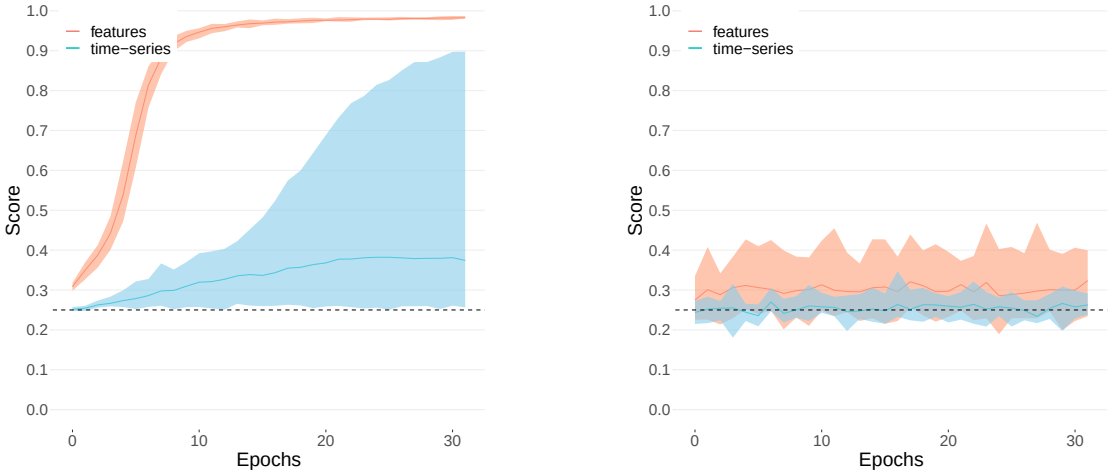
### 5.3.2 Results

#### 5.3.2.1 Randomly initialised

Figures 5.7 and 5.8 depict, respectively, the training and validation classification accuracy and cross-entropy loss of the model in the randomly initialised configuration.

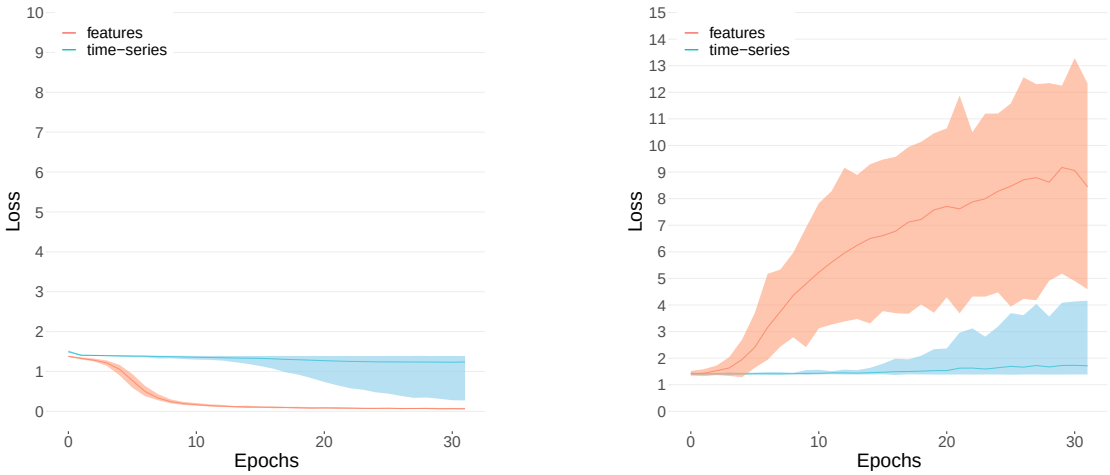
When this configuration is trained using the PSD features, the model is able to learn to the point that its accuracy during training, in Figure 5.7a, almost reaches 100 %, and its cross-entropy loss during training, in Figure 5.8a, goes down towards 0. Nevertheless, the model's validation accuracy and cross-entropy loss during training, are similar to when the model is language-pretrained. In fact, a paired student t-test finds that there is no statistically significant difference between the average test score of the model when randomly initialised, 31.3 % depicted in Figure 5.9, and the average test score of the model when it is language-pretrained ($p = 0.79$).

When this configuration is trained using the time-series data, contrarily, there does seem to be a notable difference. Firstly, the model's classification accuracy and cross-entropy loss during training suggest that now that the model is not language-pretrained, it seems to only be able to learn from some participants, while it struggles to learn from others. Secondly, the model's classification accuracy and cross-entropy loss during validation shows that, even though the model can learn from some participants, it struggles to generalise. In fact, the model's average test score of 26.0 % is close to random assignment and is significantly lower than the average test




(a) Training (b) Validation

Figure 5.7: Participant-wise average training and validation classification accuracy of the model during training, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is randomly initialised and all of its decoder units are trained. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.




(a) Training (b) Validation

Figure 5.8: Participant-wise average training and validation cross-entropy loss of the model during training, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The instance of GPT2 is randomly initialised and all of its decoder units are trained. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.

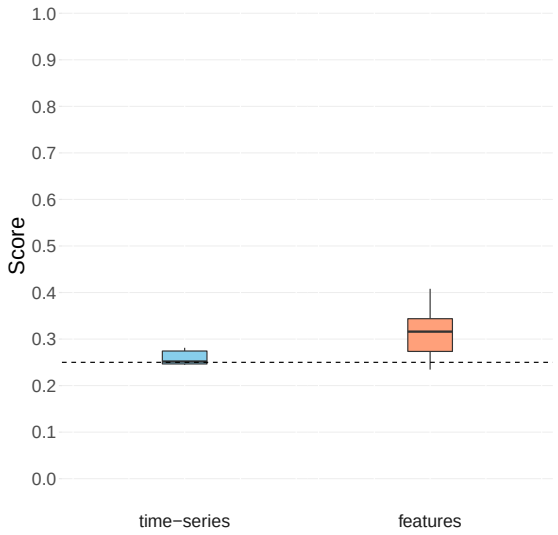



Figure 5.9: Participant-wise average test classification accuracy of the model after training, for both the time-series data (blue) and the PSD features (red). The dashed black line denotes random assignment at 25 %. The instance of GPT2 is randomly initialised and all of its decoder units are trained. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.

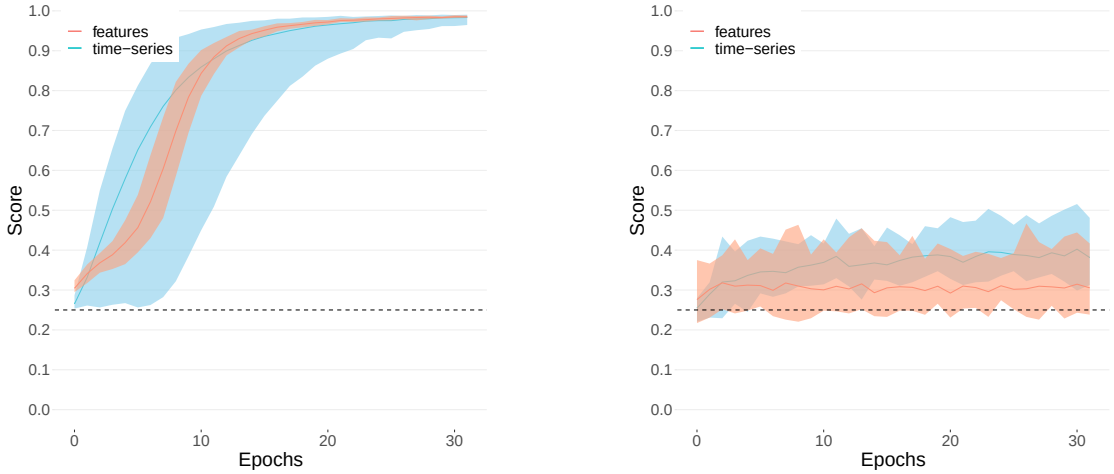
score of 41.6 % the model achieves when it is language-pretrained (according to a paired t-test, $p < 0.0001$).

#### 5.3.2.2 Completely unfrozen

Figures 5.7 and 5.8 depict, respectively, the training and validation classification accuracy and cross-entropy loss of the model in the completely unfrozen configuration. What can be seen is that in this case, the model seems to suffer from overfitting when trained using the time-series data as well as when trained using the PSD features.

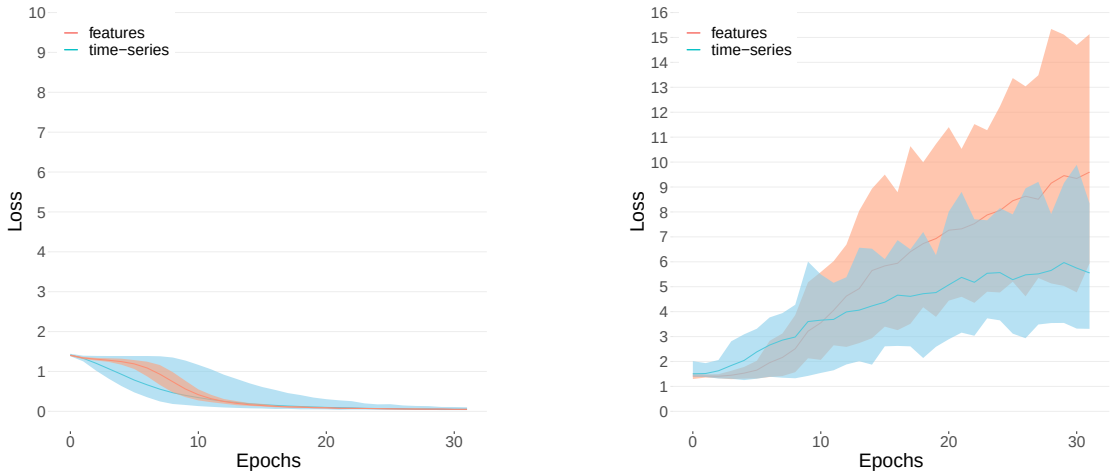
When this configuration is trained using the PSD features, the results are very similar to those of the randomly initialised configuration. The only noticeable difference is that the classification accuracy and cross-entropy loss during training seem to reach extremes slightly less quickly when the model is language-pretrained. Remember that in this case, when the model was language-pretrained but all of its feed-forward layers and self-attention layers were frozen, its classification accuracy and cross-entropy loss during training never reached these extremes. Nevertheless, a one-way ANOVA test finds no statistically significant difference between the average test score of this configuration, 30.5 % depicted in Figure 5.12, and those of the other two configurations ($p = 0.951$).

When this configuration is trained using the time-series data, the results are similar to when all of GPT2's feed-forward and self-attention layers are frozen, i.e., the configuration from Section 5.2. There seems to be slightly more variability in the training classification and cross-entropy loss curves. On first sight, this does not seem to be the case for the validation curves. A student t-test confirms that there is no statistically significant difference between the mean average test score of this configuration, 38.4 % depicted in Figure 5.12, and that of the configuration




(a) Training (b) Validation

Figure 5.10: Participant-wise average training and validation classification accuracy of the model during training, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and all of its are trained. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.




(a) Training (b) Validation

Figure 5.11: Participant-wise average training and validation cross-entropy loss of the model during training, for both the time-series data (blue) and the PSD features (red). The shade around the average curve depicts the minimum and maximum values. The instance of GPT2 is language-pretrained and all of its decoder units are trained. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.

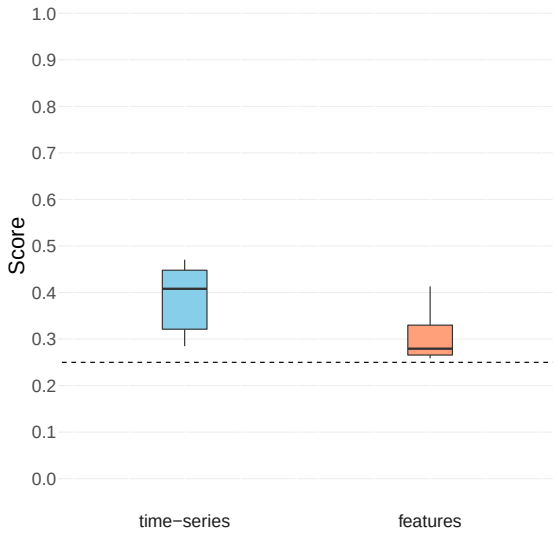



Figure 5.12: Participant-wise average test classification accuracy of the model after training, for both the time-series data (blue) and the PSD features (red). The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and all of its decoder units are trained. The values of the hyperparameters of this run correspond to the best found hyperparameter values given by Table 5.1.

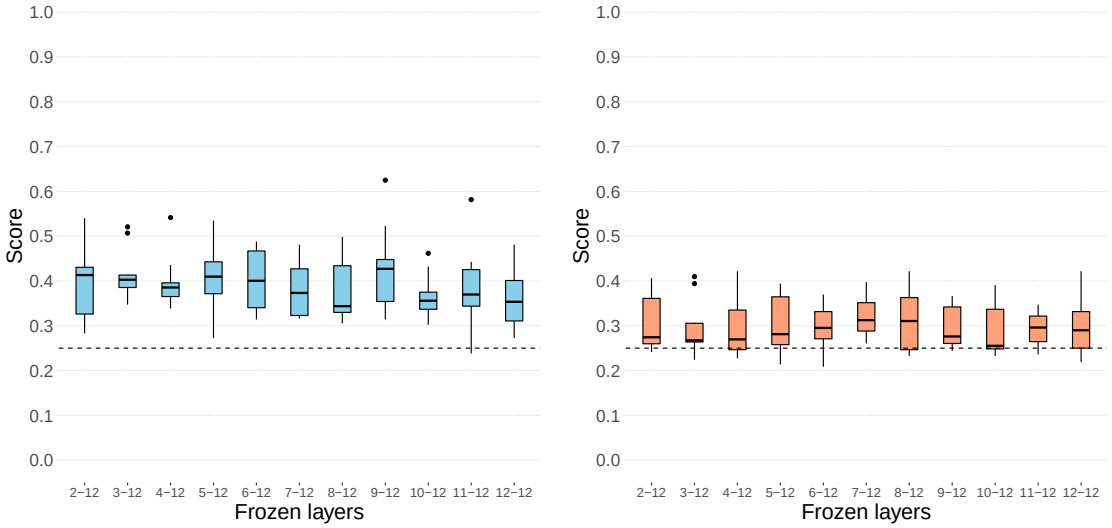
from Section 5.2 ($p = 0.18$).

#### 5.3.2.3 Shallow-to-deep unfrozen

Figure 5.13 depicts the average test classification accuracy for the configurations of the model where GPT2's decoder units are unfrozen from shallow to deep. The corresponding training and validation classification accuracy and cross-entropy loss curves can be found in Appendix B.

When the model is trained using the PSD features, there does not seem to be a difference between the average test scores of any of these configurations as well as when compared to the previous configurations. In fact, a one-way ANOVA test finds that there is no statistically significant difference between the average test scores of this and all previous configurations ($p = 0.99$). This is reflected by the validation classification accuracy curves in Figure B.3b. The training classification accuracy and cross-entropy loss curves in, respectively, Figures B.3a and B.4a, as well as the validation cross-entropy loss curve in Figure B.4b, suggest that the more decoder units are unfrozen, the more of an issue overfitting becomes.

When the model is trained using the time-series data there do seem to be differences, even though a one-way ANOVA test finds that there are none ($p = 0.67$). It is important to state that the average test classification accuracy of 43.1 % that is achieved when only the last four decoder units of GPT2 are frozen (i.e., configuration '9-12' in Figure 5.13a), is the highest average test score that is achieved throughout this whole thesis. Even though not significantly different to the 41.6 % that was achieved in the completely frozen configuration of Section 5.2 (according to a paired t-test, $p = 0.49$), this average test score is higher than the 42.6 % that Kostas et al. (2021) achieve by classifying their BENDR representations using a combination of a linear and a softmax layer. The training and validation classification accuracy and cross-entropy loss curves




(a) Time-series

(b) Features

Figure 5.13: Participant-wise average test classification accuracy of the model after training, for both the time-series data (left, blue) and the PSD features (right, red). The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from shallow to deep. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.

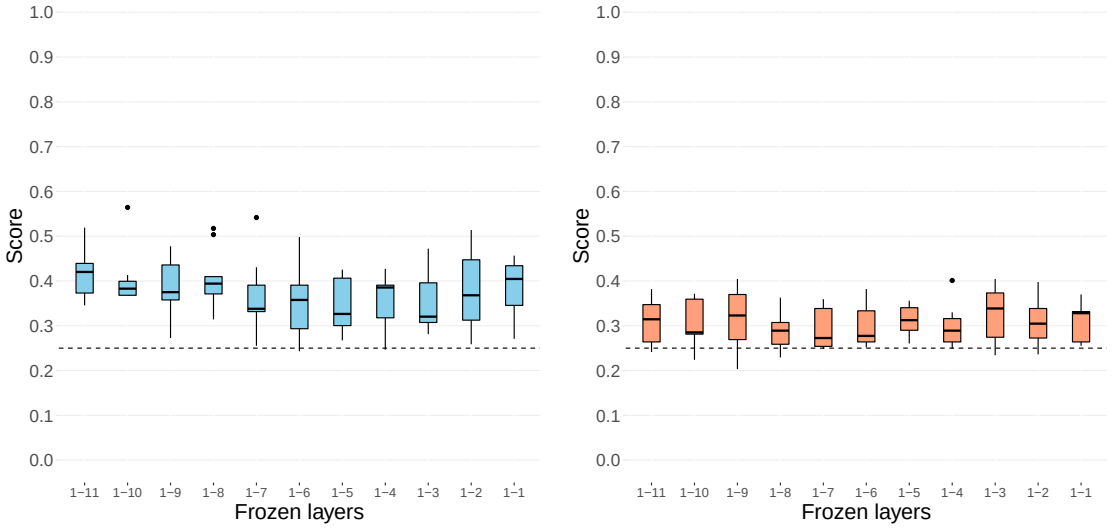
in, respectively, Figures B.1 and B.2, depict no clear relation between the unfreezing of decoder units and overfitting.

#### 5.3.2.4 Deep-to-shallow unfrozen

Figure 5.14 depicts the average test classification accuracy for the configurations of the model where GPT2's decoder units are unfrozen from deep to shallow. The corresponding training and validation classification accuracy and cross-entropy loss curves can be found in Appendix B.

When the model is trained using the PSD features, there once more does not seem to be a difference between the average test scores of any of these configurations as well as when compared to all of the previous configurations. A one-way ANOVA test, again, confirms this ($p = 0.99$). The validation classification accuracy curves in Figure B.3b also reflect this. Here as well, the training classification accuracy and cross-entropy loss curves in, respectively, Figures B.7a and B.8a, as well as the validation cross-entropy loss curve in Figure B.8b, suggest that the more decoder units are unfrozen, the more of an issue overfitting becomes. Overfitting does not seem to be influenced by the order in which the decoder units are unfrozen. It is rather the total number of unfrozen decoder units that dictates how big of an issue overfitting becomes.

When the model is trained using the time-series data, once more, an ANOVA test finds there to not be any statistically significant differences ($p = 0.37$). Nevertheless, just as before, a number of outliers and small differences suggest that unfreezing the decoder units does have some influence on the model's generalisation capabilities. In this case, the configuration where only the deepest decoder unit is unfrozen (i.e., configuration '1-11' in Figure 5.14a) achieves an average test score of 42.0 %, which is higher than what the completely frozen configuration of Section 5.2




(a) Time-series

(b) Features

Figure 5.14: Participant-wise average test classification accuracy of the model after training, for both the time-series data (left, blue) and the PSD features (right, red). The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from deep to shallow. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.

achieves. A paired t-test does, however, not find these two scores to be significantly different ($p = 0.91$). When more decoder units are gradually unfrozen, the average test scores tend to decrease. Here as well, the training and validation classification accuracy and cross-entropy loss curves in, respectively, Figures B.5 and B.6, depict no clear relation between the unfreezing of decoder units and overfitting.

### 5.3.3 Discussion

The performance of the model when trained using the PSD features is constant throughout all configurations. Interestingly, when the model is completely unfrozen, whether it is randomly initialised or language-pretrained, training scores tend to go up towards 100 % while training cross-entropy loss tends to go down towards 0. Contrarily, when all of GPT2's decoder units are frozen, i.e., the configuration from Section 5.2, this is not the case. From this, it can be concluded that for the model to learn to correctly classify its PSD features training data, GPT2's feed-forward and self-attention layers' gradients must be adapted to them. This is not the case for the time-series data. Nevertheless, when trained using the PSD features, the model's validation and test scores are always ±30 %. Clearly, the PSD features do not capture sufficient information regarding the participants' motor imagery for the model to be able to generalise. Because of this, nothing can be said about the influence of the language-pretraining for this type of training data. The hypothesis that, when designing EEG features for training transformer-like models, temporal resolution should be reduced as minimally as possible stands. If such features were to be devised and were to be found to capture more motor imagery information, empirical evaluations designed to verify the influence of language-pretraining should prove to

be interesting. Lastly, it is interesting to see the clear relation between the number of frozen decoder units and the speed at which the model's training classification accuracy curve, training cross-entropy loss curve, and validation cross-entropy loss curve, reach extremes. The order in which the decoder units are unfrozen does not seem to matter. Clearly, the model is too large for the small PSD features dataset and freezing decoder units directly translates to the degree of overfitting. Nevertheless, the conclusion remains that the PSD features do not capture sufficient motor imagery information for the model to be able to generalise and hence, for the PSD features, nothing can be said about the unfreezing hypotheses.

When the model is trained using the time-series data, the language-pretraining seems to be crucial. A randomly initialised model can learn to accurately predict its training data but is not able to generalise, as its average test score nears random assignment. Unfreezing all of GPT2's decoder units results in more variability in the training classification accuracy and cross-entropy loss curves. Aside from this minor phenomenon, in contrast to when the model is trained using the PSD features, there does not seem to be much difference when compared to the completely frozen configuration. For both the completely unfrozen and completely frozen configurations, the training classification accuracy and cross-entropy loss curves, as well as the validation cross-entropy loss curves, reach extremes. From this, it can be concluded that, unlike for the features, the language-pretraining that is frozen in GPT2's decoder units in combination with the unfrozen layer normalisation layers, are sufficient for the model to be able to learn to classify its time-series training data to perfection. While there is no statistically significant difference between the average test scores of the completely frozen and completely unfrozen models, the actual found average test score of the completely frozen model is higher than that of the completely unfrozen model. In any case, it is a fact that GPT2's language-pretraining is beneficial. A one-way ANOVA test finds that there is no statistically significant difference between the average test scores that are found by the configurations that gradually unfreeze their decoder units. Nevertheless, the hypothesis that a model whose shallow decoder units are unfrozen while its deeper decoder units are unfrozen could learn to generalise better due to diminished overfitting/memorisation in the deeper decoder units still stands. In the case of the 9-participants Graz dataset A, the differences might just be too small for the ANOVA test to capture. This claim is justified by the fact that the highest average test score that is achieved throughout this whole thesis is a model that conforms to the hypothesis, i.e., one of which the first eight decoder units are unfrozen while its last four are frozen. Nevertheless, due to the fact that a one-way ANOVA test does not find statistically significant differences between the average test scores, it is not claimed that the hypothesis is true. However, it has not necessarily been falsified. The unclear relation between gradually unfreezing the decoder units and its effect on the training and validation classification accuracy and cross-entropy loss curves does not help the hypothesis' cause. The same can be said for the hypothesis that states that unfreezing the deeper decoder units but keeping the more shallow decoder units frozen, should result in more overfitting/memorisation. To be able to prove or falsify these hypotheses, it seems that a larger EEG dataset or a less complex language-pretrained transformer is required. In comparison to the size of the dataset, the model is so complex that overfitting is an issue even when all the decoder units are frozen. Hence, the disproportion between the size of the Graz dataset A and the model as well as the questionable power of a one-way ANOVA test in the case of a 9-participant dataset might be clouding the effect of unfreezing the decoder units.

# Chapter 6

# Conclusions and Future Work

## 6.1 Introduction

Crucial for this chapter is a recapitulation of the research question that is asked in this thesis: "How and to what degree can a language-pretrained transformer transfer to the classification of EEG and does the language-pretraining influence performance?" With the goal of finding an answer to the research question, the thesis starts by giving a foundational background for the four topics inherent to it, i.e., neuroimaging, BCIs, transfer learning, and NLP. In a separate chapter, the research question is related to existing work, discussing similarities and differences. Using the collected knowledge, a methodology is designed specifically to be able to perform a set of empirical evaluations that should shine light on the answer to the research question. The empirical evaluations are established and performed. The results are reported and analysed.

With all of this in mind, this chapter is outlined as follows. Section 6.2 discusses the answers to the research question that are found to be indisputably correct. Subsequently, Section 6.3 concludes the thesis with a discussion on the parts of the research question that were not answered as well as new questions that have arisen. In addition, this section explains possible approaches for finding answers.

## 6.2 Conclusions

The main conclusion that can be drawn is that it is in fact possible for a language-pretrained transformer, GPT2 in this case, to transfer to the setting of EEG classification. Unlike what K. Lu et al. (2021) find for a variety of other tasks, the achieved classification scores are not state-of-the-art for the field of EEG classification – not unexpectedly so, when comparing the inherent complexity of EEG classification to the tasks that K. Lu et al. evaluate. Nevertheless, through evaluations similar to those of K. Lu et al., it is found that it is indisputable that positive transfer occurs. It seems that the language-modelling objective with which GPT2 is pretrained on a large language corpus allows for capturing structure that is relevant when classifying EEG. Considering the manner with which the EEG data is presented to GPT2, this can be explained as follows. The channels-by-samples matrix of every event window is seen as a sequence of channel tokens. GPT2 is pretrained to predict the next language token given an input sequence. From such a perspective, it seems reasonable that this could also help GPT2 to predict channel tokens, even though these are drastically different from the language tokens that GPT2 is used to and the order that these tokens occur in seems incomparable to that of natural language sentences.

Interestingly, it can be concluded that in GPT2's pretraining phase structure is learned in natural language that can be reapplied to EEG motor imagery signals.

An important detail of this conclusion, is that it is only found to hold when the language-pretrained model is fine-tuned to preprocessed EEG time-series data. When GPT2 is trained using PSD features calculated from the event windows, transfer is not observed. However, the experimental evaluations indicate that this can not be accredited to GPT2 and its language pretraining, but is rather caused by the features themselves. Regardless of whether it is pretrained or not, when trained on the PSD features, the model can not achieve the accuracy that it achieves when trained on the time-series data. In fact, the model's performance for this type of data is close to random assignment. From this, it can be concluded that the combination of this type of features and a language-pretrained transformer does not allow the transformer to capture sufficient motor imagery information to be able to generalise. Since the PSD features are focused on the frequency domain, the lack of information is most likely in the time and spatial domains. The temporal resolution of the PSD features is substantially reduced when compared to the raw EEG time-series data. Seeing how the model is able to perform better when trained using the time-series data, the PSD features can be concluded to lack temporal resolution. This is logical when considering the sequential nature of natural language and the deep learning models that are used to process it. The spatial resolution of the PSD features should be similar to that of the time-series data. However, the vectorisation technique that is used to encode the PSDs of multiple channels into a single vector might obstruct GPT2's ability to extract the spatial information. Coincidentally, both the reduction in temporal resolution and a complexified representation of spatial information are also the case for Kostas et al.'s (2021) BENDR representations. Fundamentally, devising good features from signal data comes down to balancing the three-way trade-off between the time, frequency, and spatial domains. In the context of training sequential natural language processing models for EEG classification, it can be concluded that the time domain must be sufficiently present for the features to be successful.

In light of Kostas et al.'s (2021) claim that masked language-modelling transformers are inclined to learn interpolation methods when trained on time-series data, it can be concluded that this is not the case for generative language-modelling transformers. The difference between these two types of models was expected to result in this conclusion. Nevertheless, the relatively low test scores of GPT2 when trained on the time-series data clearly show the complexity of reconstructing time-series data, which is inherently happening in the transformer-like model.

A final conclusion that can be drawn pertains to the complexity of the model. The empirical evaluations in Section 5.3 are designed to get insight into the influence of unfreezing GPT2's decoder units on the learning process of the model. Since overfitting is found to be an issue in most of the experimental evaluations, much thought is put into how to reduce this as well as into how this is influenced by the language-pretraining. A straightforward solution to the overfitting issue is to keep all of GPT2's decoder units frozen. The more of them that are frozen, the less details the model can learn and thus, the more overfitting should be reduced. Of course, the bias-variance trade-off is a known phenomenon and it shows that it is not necessarily so that overfitting should always be nullified (Kohavi & Wolpert, 1996). In the context of this thesis, this translates to the fact that freezing all of GPT2's decoder units is not necessarily optimal. There can rather exist a configuration of frozen and unfrozen layers that results in a better trade-off between bias and variance. Two hypotheses are established. Overfitting in deep learning models is known to happen in deeper layers. In those layers, models analyse the most detailed parts of their input. From such a point of view, a first hypothesis states that unfreezing GPT2's more shallow decoder units such that these can learn to capture general patterns from the EEG data, while freezing the deeper decoder units such that these can not get fixated on details, is likely to result in better average test scores when compared to a model that

is completely frozen. Compared to a completely frozen model, such a configuration would allow for more adaptation in the more shallow decoder units, countered by a reduction in overfitting caused by freezing the deeper decoder units. On the other hand, a second hypothesis states that when unfreezing the other way around, through the same reasoning, average test scores are likely to be lower than those of a completely frozen model. Compared to a completely frozen model, such a configuration would mostly increase overfitting. It is important to note that these two hypotheses also capture a factor of transfer learning. In the case of this thesis, freezing and unfreezing decoder units directly relates to the degree, and more specifically the depth, with which GPT2's language-pretraining can transfer to EEG classification. Regrettably, while the results of the experimental evaluations do not falsify these two hypotheses, there is no statistical basis for claiming that they are true. Traditional one-way ANOVA tests find that there are no statistically significant differences between the average test scores found by a variety of configurations created by unfreezing GPT2's decoder units from shallow to deep and the other way around. Nonetheless, the power of a one-way ANOVA test in the case of a 9-participant dataset is questionable. In addition, there is an inarguable disproportion between the complexity of the model and the size of the dataset. It can be concluded that both of these phenomena might be clouding the effect of unfreezing the model's layers. Drawing further conclusions regarding the two hypotheses requires additional experimental evaluations.

## 6.3 Future work

Before taking the work that is presented in this thesis further, it would be interesting to confirm its findings on larger EEG motor imagery datasets. In doing so, the balance between the model's complexity and the dataset would be better, allowing for a clearer analysis of the established hypotheses. Examples of datasets that could be interesting to analyse are the dataset of Kaya et al. (2018), which consists of substantially more and longer recording sessions for 13 participants, and that of Schalk et al. (2004), which consists of recording sessions for 109 participants. It is important to note that, due to the size of GPT2, model training times were already substantial in the case of the Graz dataset A. Larger datasets are guaranteed to require even more time. While the source code that was developed for this thesis is already optimised in a variety of ways, extensive parallelisation of the training and evaluation process might be required for an analysis of larger datasets to become viable as a master's thesis. In any case, once the findings are verified and more insight into the influence of the language-pretraining has been gained, a variety of interesting research directions open up.

First and foremost, the study could be extended to other EEG decoding tasks. There is no guarantee that the findings that are presented here are valid beyond motor imagery. While there is no reason to believe that they are not, it should be interesting to confirm. The influence of an EEG decoding task's properties on the model's performance can provide valuable insight into its abilities. For example, an EEG decoding task for which it is rational to create longer event windows, such as in sleep staging (L.-X. Feng et al., 2021). In the context of this thesis, longer event windows translate to larger contexts for GPT2. An interesting aspect to investigate consists of analysing whether it is more beneficial to scale the model up, allowing for larger contexts, or to downsample the event windows, allowing smaller models to handle them. In the context of designing mobile, online BCIs, such an analysis could provide insight into whether or not a high sampling rate is required. Another example are EEG decoding tasks that benefit from higher spatial resolution, such as epilepsy detection (Srinivasan et al., 2007). In the context of this thesis, a higher spatial resolution translates to longer channel tokens. Since the smallest word embeddings that GPT2 has been trained with are of size 768, it can be interesting to

analyse GPT2's ability to extract spatial knowledge from channel tokens that represent a larger number of electrodes. A mobile, online BCI would never use a large number of electrodes but rather stick with, for example, the 22 electrodes of the Graz dataset A. Nonetheless, it should be an interesting research question.

Another direction that can be taken with this work is an optimisation study of the model. For an actual BCI to work, classification accuracy needs to be much higher than that which is achieved in this thesis. In some way, training the model with a larger dataset is already in line with such an objective. It could also be interesting to investigate how much data is needed to train a language-pretrained transformer to state-of-the-art performance. Other ideas could be tried as well. For example, different preprocessing methods could be applied to the EEG time-series data. Compared to the simple standardisation method that is used in this thesis, Barachant et al.'s (2012) Riemannian geometry and He and Wu's (2020) Euclidian space alignment could prove to be interesting. Additionally, since the size of the event windows translates directly to the size of GPT2's context window, it could be interesting to vary this. The winner of the fourth BCI competition implements event windows half the size of those that are used here. Different feature extraction techniques could be tested as well. A reasonable approach for such research would consist of starting with simple features, such as the PSD features that are used in this thesis, and gradually making them more complex, using the knowledge that was gained from previous iterations. It can be interesting to immediately investigate the natural language model's performance on complex representations such as BENDR and CSP. However, since classifying EEG using NLP models has never been done before, starting simple should be fruitful. A study that is structured in such a manner might also result in insight regarding transferable EEG characteristics.

Finally, in the spirit of the work of K. Lu et al. (2021), it should be interesting to formalise the language-pretraining for BCIs paradigm. Through a study of a variety of language-pretrained models applied to a variety of EEG decoding tasks, it might be possible to find much more concrete relations between these two domains. In addition, such a study should provide insight into whether or not language-pretrained NLP models, and the techniques that they consist of, can be seen as suitable methods for EEG decoding. In the context of the work of Kostas et al. (2021), it might even be interesting to pretrain a transformer using both language and EEG. A comparison of both types of pretraining for the same model should prove to be interesting as well.

# Bibliography


Abiri, R., Borhani, S., Sellers, E. W., Jiang, Y., & Zhao, X. (2019, January). A comprehensive review of EEG-based brain–computer interface paradigms. *Journal of Neural Engineering*, *16*(1), 011001. doi: 10/ggnvvh

Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A Learning Algorithm for Boltzmann Machines*. *Cognitive Science*, *9*(1), 147–169. doi: 10/bnff58

Aghaei, S. (2012, January). Evolution of the World Wide Web : From Web 1.0 to Web 4.0. *International journal of Web & Semantic Technology*, *3*(1), 1–10. doi: 10.5121/ijwest.2012.3101

Ahn, M., & Jun, S. C. (2015, March). Performance variation in motor imagery brain-computer interface: A brief review. *Journal of Neuroscience Methods*, *243*, 103–110. doi: 10.1016/j.jneumeth.2015.01.033

Al-Fahoum, A., & Al-Fraihat, A. (2014). Methods of EEG Signal Features Extraction Using Linear Analysis in Frequency and Time-Frequency Domains. *ISRN neuroscience*. doi: 10.1155/2014/730218

AL-Quraishi, M. S., Elamvazuthi, I., Daud, S. A., Parasuraman, S., & Borboni, A. (2018, October). EEG-Based Control for Upper and Lower Limb Exoskeletons and Prostheses: A Systematic Review. *Sensors (Basel, Switzerland)*, *18*(10), 3342. doi: 10.3390/s18103342

Alammar, J. (2018). *The Illustrated GPT-2 (Visualizing Transformer Language Models)* [Blog Post]. https://jalammar.github.io/illustrated-gpt2/#part-2-illustrated-self-attention.

Amin, S. U., Alsulaiman, M., Muhammad, G., Bencherif, M. A., & Hossain, M. S. (2019). Multilevel Weighted Feature Fusion Using Convolutional Neural Networks for EEG Motor Imagery Classification. *IEEE Access*, *7*, 18940–18950. doi: 10/ggkz2h

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

Ang, K. K., Chin, Z. Y., Zhang, H., & Guan, C. (2008, June). Filter Bank Common Spatial Pattern (FBCSP) in Brain-Computer Interface. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (pp. 2390–2397). doi: 10/bmcxv9

Arnold, A., Nallapati, R., & Cohen, W. W. (2007, October). A Comparative Study of Methods for Transductive Transfer Learning. In *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)* (pp. 77–82). doi: 10.1109/ICDMW.2007.109

Aznan, N. F. N., Connolly, J. D., Moubayed, N. A., & Breckon, T. (2019). Using Variable Natural Environment Brain-Computer Interface Stimuli for Real-time Humanoid Robot Navigation. *2019 International Conference on Robotics and Automation (ICRA)*. doi: 10.1109/ICRA.2019.8794060

Ba, J., Kiros, J., & Hinton, G. E. (2016). Layer Normalization. *arXiv preprint*.

Bablani, A., Edla, D. R., & Dodia, S. (2018, January). Classification of EEG Data using k-Nearest Neighbor approach for Concealed Information Test. *Procedia Computer Science*, *143*, 242–249. doi: 10/gncp7h

Baevski, A., Zhou, Y., Mohamed, A., & Auli, M. (2020). Wav2vec 2.0: A framework for self-supervised learning of speech representations. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 12449–12460). Curran Associates, Inc.

Bailey, D. L., Townsend, D. W., Valk, P. E., & Maisey, M. N. (2005). *Positron Emission Tomography: Basic Sciences*. Springer Science & Business Media.

Banville, H., Albuquerque, I., Hyvärinen, A., Moffat, G., Engemann, D.-A., & Gramfort, A. (2019, October). Self-Supervised Representation Learning from Electroencephalography Signals. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)* (pp. 1–6). doi: 10.1109/MLSP.2019.8918693

Banville, H., Chehab, O., Hyvärinen, A., Engemann, D.-A., & Gramfort, A. (2021, March). Uncovering the structure of clinical EEG signals with self-supervised learning. *Journal of Neural Engineering*, *18*(4), 046020. doi: 10.1088/1741-2552/abca18

Barachant, A., Bonnet, S., Congedo, M., & Jutten, C. (2012, April). Multiclass Brain–Computer Interface Classification by Riemannian Geometry. *IEEE Transactions on Biomedical Engineering*, *59*(4), 920–928. doi: 10.1109/TBME.2011.2172210

Bascil, M. S., Tesneli, A. Y., & Temurtas, F. (2016, September). Spectral feature extraction of EEG signals and pattern recognition during mental tasks of 2-D cursor movements for BCI using SVM and ANN. *Australasian Physical & Engineering Sciences in Medicine*, *39*(3), 665–676. doi: 10/gnbc96

Bashashati, A., Fatourechi, M., Ward, R., & Birch, G. (2007). A survey of signal processing algorithms in brain-computer interfaces based on electrical brain signals. *Journal of neural engineering*. doi: 10.1088/1741-2560/4/2/R03

Belkacem, A. N., Jamil, N., Palmer, J. A., Ouhbi, S., & Chen, C. (2020). Brain Computer Interfaces for Improving the Quality of Life of Older Adults and Elderly Patients. *Frontiers in Neuroscience*, *14*. doi: 10.3389/fnins.2020.00692

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2006, December). Greedy layer-wise training of deep networks. In *Proceedings of the 19th International Conference on Neural Information Processing Systems* (pp. 153–160). Cambridge, MA, USA: MIT Press.

Berger, H. (1929, December). Über das Elektrenkephalogramm des Menschen. *Archiv für Psychiatrie und Nervenkrankheiten*, *87*(1), 527–570. doi: 10.1007/BF01797193

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 24). Curran Associates, Inc.

Bergstra, J., Yamins, D., & Cox, D. (2013, February). Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. In *Proceedings of the 30th International Conference on Machine Learning* (pp. 115–123). PMLR.

Blitzer, J., Dredze, M., & Pereira, F. (2007, June). Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* (pp. 440–447). Prague, Czech Republic: Association for Computational Linguistics.

Blitzer, J., McDonald, R., & Pereira, F. (2006, July). Domain Adaptation with Structural

Correspondence Learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing* (pp. 120–128). Sydney, Australia: Association for Computational Linguistics.

Boto, E., Holmes, N., Leggett, J., Roberts, G., Shah, V., Meyer, S., ... Brookes, M. (2018). Moving magnetoencephalography towards real-world applications with a wearable system. *Nature*. doi: 10.1038/nature26147

Bowman, S. R., Pavlick, E., Grave, E., Durme, B. V., Wang, A., Hula, J., ... Chen, B. (2018). Looking for ELMo's friends: Sentence-Level Pretraining Beyond Language Modeling. *ArXiv*.

Bozinovski, S. (2020, September). Reminder of the First Paper on Transfer Learning in Neural Networks, 1976. *Informatica*, *44*(3). doi: 10.31449/inf.v44i3.2828

Bracewell, R. N. (1986). *The Fourier transform and its applications* (Vol. 31999). McGraw-Hill New York.

Bright, D., Nair, A., Salvekar, D., & Bhisikar, S. (2016, June). EEG-based brain controlled prosthetic arm. In *2016 Conference on Advances in Signal Processing (CASP)* (pp. 479–483). doi: 10.1109/CASP.2016.7746219

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... Amodei, D. (2020). Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 1877–1901). Curran Associates, Inc.

Brunner, C., Leeb, R., Muller-Putz, G. R., & Schlogl, A. (2008). BCI Competition 2008 – Graz data set A. *Dataset*.

Brunner, C., Naeem, M., Leeb, R., Graimann, B., & Pfurtscheller, G. (2007, June). Spatial filtering and selection of optimized components in four class motor imagery EEG data using independent components analysis. *Pattern Recognition Letters*, *28*(8), 957–964. doi: 10.1016/j.patrec.2007.01.002

Buch, E. R., Weber, C., Cohen, L., Braun, C., Dimyan, M., Ard, T., ... Birbaumer, N. (2008). Think to Move: A Neuromagnetic Brain-Computer Interface (BCI) System for Chronic Stroke. *Stroke*. doi: 10.1161/STROKEAHA.107.505313

Bundy, A., & Wallen, L. (1984). Augmented Transition Network. In A. Bundy & L. Wallen (Eds.), *Catalogue of Artificial Intelligence Tools* (pp. 7–7). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-642-96868-6_14

Burgess, R. C. (2003, January). Filters, Analog/Digital. In M. J. Aminoff & R. B. Daroff (Eds.), *Encyclopedia of the Neurological Sciences* (pp. 369–374). New York: Academic Press. doi: 10.1016/B0-12-226870-9/00302-6

Caruana, R. (1997, July). Multitask Learning. *Machine Learning*, *28*(1), 41–75. doi: 10.1023/A:1007379606734

Chai, R., Naik, G. R., Nguyen, T. N., Ling, S. H., Tran, Y., Craig, A., & Nguyen, H. T. (2017, May). Driver Fatigue Classification With Independent Component by Entropy Rate Bound Minimization Analysis in an EEG-Based System. *IEEE Journal of Biomedical and Health Informatics*, *21*(3), 715–724. doi: 10/ggzfcc

Chakladar, D. D., & Chakraborty, S. (2018, August). Multi-target way of cursor movement

in brain computer interface using unsupervised learning. *Biologically Inspired Cognitive Architectures*, *25*, 88–100. doi: 10/gfhd7f

Chang, C.-Y., Hsu, S.-H., Pion-Tonachini, L., & Jung, T.-P. (2018, July). Evaluation of Artifact Subspace Reconstruction for Automatic EEG Artifact Removal. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual International Conference*, *2018*, 1242–1245. doi: 10.1109/EMBC.2018.8512547

Chapelle, O., Schölkopf, B., Zien, A., & Bach, F. (Eds.). (2006). *Semi-Supervised Learning*. Cambridge, MA, USA: MIT Press.

Chen, K., Wang, J., Chen, L.-C., Gao, H., Xu, W., & Nevatia, R. (2015). ABC-CNN: An attention based convolutional neural network for visual question answering. *CoRR*, *abs/1511.05960*.

Chomsky, N. (1957). *Syntactic structures*. The Hague: Mouton.

Chung, J. E., Joo, H. R., Fan, J. L., Liu, D. F., Barnett, A. H., Chen, S., ... Frank, L. M. (2019, January). High-Density, Long-Lasting, and Multi-region Electrophysiological Recordings Using Polymer Electrode Arrays. *Neuron*, *101*(1), 21-31.e5. doi: 10/gfkhwv

Coenen, A., & Zayachkivska, O. (2013). Adolf Beck: A pioneer in electroencephalography in between Richard Caton and Hans Berger. *Advances in Cognitive Psychology*, *9*(4), 216–221. doi: 10.2478/v10053-008-0148-3

Cohen, T. S., Geiger, M., Köhler, J., & Welling, M. (2018, January). Spherical CNNs. In *ICLR*.

Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, *19*(90), 297–301. doi: 10.1090/S0025-5718-1965-0178586-1

Craik, A., He, Y., & Contreras-Vidal, J. (2019). Deep learning for electroencephalogram (EEG) classification tasks: A review. *Journal of neural engineering*. doi: 10.1088/1741-2552/ab0ab5

Cuffin, B. (1998, September). EEG dipole source localization. *IEEE Engineering in Medicine and Biology Magazine*, *17*(5), 118–122. doi: 10/b6mvrn

Dai, W., Yang, Q., Xue, G.-R., & Yu, Y. (2008, July). Self-taught clustering. In *Proceedings of the 25th international conference on Machine learning* (pp. 200–207). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/1390156.1390182

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019, June). Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *arXiv:1901.02860 [cs, stat]*.

D'Amour, A., Heller, K. A., Moldovan, D., Adlam, B., Alipanahi, B., Beutel, A., ... Sculley, D. (2020). Underspecification presents challenges for credibility in modern machine learning. *CoRR*, *abs/2011.03395*.

Daumas, M. (1965). Les machines à traduire de Georges Artsrouni. *Revue d'histoire des sciences*, *18*(3), 283–302. doi: 10.3406/rhs.1965.2427

Daumé, H., & Marcu, D. (2006, May). Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, *26*(1), 101–126.

Dayan, P., Hinton, G. E., Neal, R. M., & Zemel, R. S. (1995, September). The Helmholtz machine. *Neural Computation*, *7*(5), 889–904. doi: 10.1162/neco.1995.7.5.889

Delorme, A., & Makeig, S. (2004, March). EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. *Journal of Neuroscience Methods*, *134*(1), 9–21. doi: 10.1016/j.jneumeth.2003.10.009

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009, June). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255). doi: 10.1109/CVPR.2009.5206848

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, *29*(6), 141–142.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019, May). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*.

DiCarlo, J., Zoccolan, D., & Rust, N. (2012). How Does the Brain Solve Visual Object Recognition? *Neuron*. doi: 10.1016/j.neuron.2012.01.010

Di Flumeri, G., Aricò, P., Borghini, G., Sciaraffa, N., Di Florio, A., & Babiloni, F. (2019, March). The Dry Revolution: Evaluation of Three Different EEG Dry Electrode Types in Terms of Signal Spectral Features, Mental States Classification and Usability. *Sensors (Basel, Switzerland)*, *19*(6), 1365. doi: 10/gmq2ps

Dillen, A., Steckelmacher, D., Efthymiadis, K., Langlois, K., Beir, A. D., Marusic, U., ... Pauw, K. D. (2022, February). Deep learning for biosignal control: Insights from basic to real-time methods with recommendations. *Journal of Neural Engineering*, *19*(1), 011003. doi: 10.1088/1741-2552/ac4f9a

Djamal, E. C., & Lodaya, P. (2017, September). EEG based emotion monitoring using wavelet and learning vector quantization. In *2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)* (pp. 1–6). doi: 10/gm3tpg

Dornhege, G., del R. Millán, J., Hinterberger, T., McFarland, D. J., & Müller, K.-R. (2007). Evaluation Criteria for BCI Research. In *Toward Brain-Computer Interfacing* (pp. 327–342). MIT Press.

Dose, H., Møller, J. S., Iversen, H. K., & Puthusserypady, S. (2018, December). An end-to-end deep learning approach to MI-EEG signal classification for BCIs. *Expert Systems with Applications*, *114*, 532–542. doi: 10.1016/j.eswa.2018.08.031

Elbaz, G. (2012). *Common Crawl.*

Elsayed, G., Shankar, S., Cheung, B., Papernot, N., Kurakin, A., Goodfellow, I., & Sohl-Dickstein, J. (2018). Adversarial examples that fool both computer vision and time-limited humans. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 31). Curran Associates, Inc.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., & Bengio, S. (2010). Why Does Unsupervised Pre-training Help Deep Learning? *Journal of Machine Learning Research*, *11*(19), 625–660.

Ethier, C., Oby, E. R., Bauman, M. J., & Miller, L. E. (2012, May). Restoration of grasp following paralysis through brain-controlled stimulation of muscles. *Nature*, *485*(7398),

368–371. doi: 10/f3zbb4

Fadzal, C., Mansor, W., & Khuan, L. Y. (2011, June). Review of brain computer interface application in diagnosing dyslexia. In *2011 IEEE Control and System Graduate Research Colloquium* (pp. 124–128). doi: 10.1109/ICSGRC.2011.5991843

Fahimi, F., Zhang, Z., Goh, W. B., Lee, T.-S., Ang, K. K., & Guan, C. (2019, January). Inter-subject transfer learning with an end-to-end deep convolutional neural network for EEG-based BCI. *Journal of Neural Engineering*, *16*(2), 026007. doi: 10.1088/1741-2552/aaf3f6

Fazli, S., Popescu, F., Danóczy, M., Blankertz, B., Müller, K.-R., & Grozea, C. (2009, November). Subject-independent mental state classification in single trials. *Neural Networks: The Official Journal of the International Neural Network Society*, *22*(9), 1305–1312. doi: 10.1016/j.neunet.2009.06.003

Feng, L.-X., Li, X., Wang, H.-Y., Zheng, W.-Y., Zhang, Y.-Q., Gao, D.-R., & Wang, M.-Q. (2021). Automatic Sleep Staging Algorithm Based on Time Attention Mechanism. *Frontiers in Human Neuroscience*, *15*.

Feng, S. Y., Gangal, V., Wei, J., Chandar, S., Vosoughi, S., Mitamura, T., & Hovy, E. (2021). A Survey of Data Augmentation Approaches for NLP. *FINDINGS*. doi: 10.18653/v1/2021.findings-acl.84

Fisch, B. J. (2003, May). Current Practice of Clinical Electroencephalography, Third Edition. *Journal of Clinical Neurophysiology*, *20*(3), 225. doi: 10/ddt7q6

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, *7*(2), 179–188. doi: 10.1111/j.1469-1809.1936.tb02137.x

Fitzgibbon, S. P., Lewis, T. W., Powers, D. M. W., Whitham, E. W., Willoughby, J. O., & Pope, K. J. (2013, January). Surface Laplacian of Central Scalp Electrical Signals is Insensitive to Muscle Contamination. *IEEE Transactions on Biomedical Engineering*, *60*(1), 4–9. doi: 10/f4jh4n

Gao, Q., Dou, L., Belkacem, A. N., & Chen, C. (2017). Noninvasive Electroencephalogram Based Control of a Robotic Arm for Writing Task Using Hybrid BCI System. *BioMed research international*. doi: 10.1155/2017/8316485

Geethanjali, P., Mohan, Y., & Sen, J. (2012). Time domain Feature extraction and classification of EEG data for Brain Computer Interface. *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*. doi: 10.1109/FSKD.2012.6234336

Gehring, J., Auli, M., Grangier, D., Yarats, D., & Dauphin, Y. N. (2017, July). Convolutional Sequence to Sequence Learning. In *Proceedings of the 34th International Conference on Machine Learning* (pp. 1243–1252). PMLR.

Ghojogh, B., & Crowley, M. (2019). The Theory Behind Overfitting, Cross Validation, Regularization, Bagging, and Boosting: Tutorial. *ArXiv*.

Gibbs, F., Davis, H., & Lennox, W. G. (1935, December). The electro-encephalogram in epilepsy and in conditions of impaired consciousness. *Archives of Neurology & Psychiatry*, *34*(6), 1133–1148. doi: 10.1001/archneurpsyc.1935.02250240002001

Gibbs, F., Lennox, W. G., & Gibbs, E. L. (1936, December). The electro-encephalogram in diagnosis and in localization of epileptic seizures. *Archives of Neurology & Psychiatry*, *36*(6), 1225–1235. doi: 10.1001/archneurpsyc.1936.02260120072005

Girden, E. R. (1992). *ANOVA: Repeated measures* (No. 84). Sage.

Girshick, R. (2015, December). Fast R-CNN. In *2015 IEEE International Conference on Computer Vision (ICCV)* (pp. 1440–1448). doi: 10.1109/ICCV.2015.169

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014, June). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 580–587). IEEE Computer Society. doi: 10.1109/CVPR.2014.81

Glover, G. H. (2011, April). Overview of Functional Magnetic Resonance Imaging. *Neurosurgery clinics of North America*, *22*(2), 133–139. doi: 10/fbs94b

Goodfellow, I. J., Bengio, Y., & Courville, A. C. (2015). Deep Learning. *Nature*. doi: 10.1038/nature14539

Graimann, B., Allison, B., & Pfurtscheller, G. (Eds.). (2010). *Brain-Computer Interfaces: Revolutionizing Human-Computer Interaction*. Berlin Heidelberg: Springer-Verlag. doi: 10.1007/978-3-642-02091-9

Gramfort, A., Luessi, M., Larson, E., Engemann, D. A., Strohmeier, D., Brodbeck, C., ... Hämäläinen, M. S. (2013). MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience*, *7*(267), 1–13. doi: 10.3389/fnins.2013.00267

Grigoriev, A., & Grigorian, N. (2007, February). I.M. Sechenov: The Patriarch of Russian Physiology. *Journal of the History of the Neurosciences*, *16*(1-2), 19–29. doi: 10/dnctzt

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., ... Valko, M. (2020). Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 21271–21284). Curran Associates, Inc.

Guan, S., Zhao, K., & Yang, S. (2019, January). Motor Imagery EEG Classification Based on Decision Tree Framework and Riemannian Geometry. *Computational Intelligence and Neuroscience*, *2019*, e5627156. doi: 10.1155/2019/5627156

Guo, S., Lin, S., & Huang, Z. (2015, October). Feature extraction of P300s in EEG signal with discrete wavelet transform and fisher criterion. In *2015 8th International Conference on Biomedical Engineering and Informatics (BMEI)* (pp. 200–204). doi: 10/gnbc94

Haas, L. F. (2003, January). Hans Berger (1873–1941), Richard Caton (1842–1926), and electroencephalography. *Journal of Neurology, Neurosurgery & Psychiatry*, *74*(1), 9–9. doi: 10/dp9vn6

Haselsteiner, E., & Pfurtscheller, G. (2000, December). Using time-dependent neural networks for EEG classification. *IEEE Transactions on Rehabilitation Engineering*, *8*(4), 457–463. doi: 10/bz836v

He, H., & Wu, D. (2020, February). Transfer Learning for Brain-Computer Interfaces: A Euclidean Space Data Alignment Approach. *IEEE transactions on bio-medical engineering*, *67*(2), 399–410. doi: 10.1109/TBME.2019.2913914

Henaff, O. (2020, July). Data-efficient image recognition with contrastive predictive coding. In H. D. III & A. Singh (Eds.), *Proceedings of the 37th international conference on machine learning* (Vol. 119, pp. 4182–4192). PMLR.

Hinrichs, H., Scholz, M., Baum, A. K., Kam, J. W. Y., Knight, R. T., & Heinze, H.-J. (2020,

March). Comparison between a wireless dry electrode EEG system with a conventional wired wet electrode EEG system for clinical applications. *Scientific Reports*, *10*(1), 5218. doi: 10/gm3k2p

Hinton, G. E. (2002, August). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*(8), 1771–1800. doi: 10/ct8dwx

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006, July). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554. doi: 10/cjnhxz

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012, July). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580 [cs]*.

Hochreiter, S. (1998, April). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *6*(2), 107–116. doi: 10/cvjhdb

Hochreiter, S., & Schmidhuber, J. (1997, November). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. doi: 10/bxd65w

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., Laroussilhe, Q. D., Gesmundo, A., ... Gelly, S. (2019, May). Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning* (pp. 2790–2799). PMLR.

Hutchins, W. J. (2004). The Georgetown-IBM Experiment Demonstrated in January 1954. In R. E. Frederking & K. B. Taylor (Eds.), *Machine Translation: From Real Users to Research* (pp. 102–114). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-540-30194-3_12

Iivanainen, J., Stenroos, M., & Parkkonen, L. (2017). Measuring MEG closer to the brain: Performance of on-scalp sensor arrays. *NeuroImage*. doi: 10.1016/j.neuroimage.2016.12.048

Iivanainen, J., Zetter, R., Grön, M., Hakkarainen, K., & Parkkonen, L. (2019). On-scalp MEG system utilizing an actively shielded array of optically-pumped magnetometers. *NeuroImage*. doi: 10.1016/j.neuroimage.2019.03.022

İnce, R., Adanır, S. S., & Sevmez, F. (2021, September). The inventor of electroencephalography (EEG): Hans Berger (1873–1941). *Child's Nervous System*, *37*(9), 2723–2724. doi: 10/ggwv6m

Jasper, H. (1958). The ten-twenty electrode system of the international federation. *Electroencephalogr. Clin. Neurophysiol.*, *10*, 370–375.

Jiang, D., Li, W., Zhang, R., Cao, M., Luo, N., Han, Y., ... Li, X. (2021, June). A Further Study of Unsupervised Pretraining for Transformer Based Speech Recognition. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6538–6542). doi: 10.1109/ICASSP39728.2021.9414539

Jing, L., & Tian, Y. (2021). Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. doi: 10.1109/tpami.2020.2992393

Jones, K. S. (1994). Natural Language Processing: A Historical Review. In A. Zampolli, N. Calzolari, & M. Palmer (Eds.), *Current Issues in Computational Linguistics: In Honour of Don Walker* (pp. 3–16). Dordrecht: Springer Netherlands. doi: 10.1007/978-0-585-35958

-8_1

Jun, J. J., Steinmetz, N. A., Siegle, J. H., Denman, D. J., Bauza, M., Barbarits, B., ... Harris, T. D. (2017, November). Fully integrated silicon probes for high-density recording of neural activity. *Nature*, *551*(7679), 232–236. doi: 10.1038/nature24636

Jurcak, V., Tsuzuki, D., & Dan, I. (2007, February). 10/20, 10/10, and 10/5 systems revisited: Their validity as relative head-surface-based positioning systems. *NeuroImage*, *34*(4), 1600–1611. doi: 10.1016/j.neuroimage.2006.09.024

Kawala-Janik, A. (2013). *Efficiency evaluation of external environments control using bio-signals* (Doctoral dissertation, University of Greenwich). doi: 10/1/Aleksandra_Kawala-Janik_2013.pdf

Kawala-Sterniuk, A., Browarska, N., Al-Bakri, A., Pelc, M., Zygarlicki, J., Sidikova, M., ... Gorzelanczyk, E. J. (2021, January). Summary of over Fifty Years with Brain-Computer Interfaces—A Review. *Brain Sciences*, *11*(1), 43. doi: 10/gjjzqr

Kaya, M., Binli, M. K., Ozbay, E., Yanar, H., & Mishchenko, Y. (2018, December). A large electroencephalographic motor imagery dataset for electroencephalographic brain computer interfaces. *Scientific Data*, *5*(1), 180211. doi: 10/gff7bc

Khan, M. J., & Hong, K. (2017). Hybrid EEG–fNIRS-Based Eight-Command Decoding for BCI: Application to Quadcopter Control. *Front. Neurorobot.*. doi: 10/f9rctr

Kingma, D. P., & Ba, J. (2017, January). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*.

Kingwell, K. (2012, July). Neurally controlled robotic arm enables tetraplegic patient to drink coffee of her own volition. *Nature Reviews Neurology*, *8*(7), 353–353. doi: 10/gm8f7x

Kleber, B., & Birbaumer, N. (2005, March). Direct brain communication: Neuroelectric and metabolic approaches at Tübingen. *Cognitive Processing*, *6*(1), 65–74. doi: 10/fh4xbz

Klonowski, W. (2009, May). Everything you wanted to ask about EEG but were afraid to get the right answer. *Nonlinear Biomedical Physics*, *3*, 2. doi: 10.1186/1753-4631-3-2

Kohavi, R., & Wolpert, D. H. (1996). Bias plus variance decomposition for zero-one loss functions. In L. Saitta (Ed.), *Machine learning, proceedings of the thirteenth international conference (ICML '96), bari, italy, july 3-6, 1996* (pp. 275–283). Morgan Kaufmann.

Koles, Z. J., Lazar, M. S., & Zhou, S. Z. (1990, June). Spatial patterns underlying population differences in the background EEG. *Brain Topography*, *2*(4), 275–284. doi: 10.1007/BF01129656

Kostas, D., Aroca-Ouellette, S., & Rudzicz, F. (2021). BENDR: Using Transformers and a Contrastive Self-Supervised Learning Task to Learn From Massive Amounts of EEG Data. *Frontiers in Human Neuroscience*, *15*. doi: 10.3389/fnhum.2021.653659

Kostas, D., & Rudzicz, F. (2020, October). Thinker invariance: Enabling deep neural networks for BCI across more people. *Journal of Neural Engineering*, *17*(5), 056008. doi: 10.1088/1741-2552/abb7a7

Krauledat, M., Tangermann, M., Blankertz, B., & Müller, K.-R. (2008, August). Towards Zero Training for Brain-Computer Interfacing. *PLOS ONE*, *3*(8), e2967. doi: 10.1371/journal.pone.0002967

Kreilinger, A., Hiebel, H., & Müller-Putz, G. R. (2016, March). Single Versus Multiple Events Error Potential Detection in a BCI-Controlled Car Game With Continuous and Discrete Feedback. *IEEE Transactions on Biomedical Engineering*, *63*(3), 519–529. doi: 10/f8dx3j

Kriegeskorte, N. (2015). Deep Neural Networks: A New Framework for Modeling Biological Vision and Brain Information Processing. *Annual review of vision science*. doi: 10.1146/annurev-vision-082114-035447

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems* (Vol. 25). Curran Associates, Inc.

Krogh, A., & Hertz, J. (1991). A simple weight decay can improve generalization. In J. Moody, S. Hanson, & R. Lippmann (Eds.), *Advances in neural information processing systems* (Vol. 4). Morgan-Kaufmann.

Kübler, A. (2020, July). The history of BCI: From a vision for the future to real support for personhood in people with locked-in syndrome. *Neuroethics*, *13*(2), 163–180. doi: 10/gm28n3

Kukacka, J., Golkov, V., & Cremers, D. (2017). Regularization for deep learning: A taxonomy. *CoRR*, *abs/1710.10686*.

Lalor, E. C., Kelly, S. P., Finucane, C., Burke, R., Smith, R., Reilly, R. B., & McDarby, G. (2005, December). Steady-State VEP-Based Brain-Computer Interface Control in an Immersive 3D Gaming Environment. *EURASIP Journal on Advances in Signal Processing*, *2005*(19), 1–9. doi: 10.1155/ASP.2005.3156

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2020). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *ICLR*.

Lawhern, V., Hairston, W. D., McDowell, K., Westerfield, M., & Robbins, K. (2012, July). Detection and classification of subject-generated artifacts in EEG signals using autoregressive models. *Journal of Neuroscience Methods*, *208*(2), 181–189. doi: 10/f337zz

LeCun, Y., Bengio, Y., & Hinton, G. (2015, May). Deep learning. *Nature*, *521*(7553), 436–444. doi: 10.1038/nature14539

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989, December). Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, *1*(4), 541–551. doi: 10.1162/neco.1989.1.4.541

Lee, H. K., & Choi, Y.-S. (2019, December). Application of Continuous Wavelet Transform and Convolutional Neural Network in Decoding Motor Imagery Brain-Computer Interface. *Entropy*, *21*(12), 1199. doi: 10/gnbc9z

Lee, T.-J., & Sim, K.-B. (2015). Vowel Classification of Imagined Speech in an Electroencephalogram using the Deep Belief Network. *Journal of Institute of Control, Robotics and Systems*, *21*(1), 59–64. doi: 10/gmddwc

Leuthardt, E. C., Miller, K. J., Schalk, G., Rao, R. P. N., & Ojemann, J. G. (2006, June). Electrocorticography-based brain computer interface–the Seattle experience. *IEEE transactions on neural systems and rehabilitation engineering: a publication of the IEEE Engineering in Medicine and Biology Society*, *14*(2), 194–198. doi: 10/c8fcd3

Levene, H. (1961). Robust tests for equality of variances. *undefined*.

Li, M., Zhang, M., Luo, X., & Yang, J. (2016, August). Combined long short-term memory based network employing wavelet coefficients for MI-EEG recognition. In *2016 IEEE International Conference on Mechatronics and Automation* (pp. 1971–1976). doi: 10/gncp7g

Liang, S.-F., Shaw, F., Young, C., Chang, D., & Liao, Y.-C. (2010). A closed-loop brain computer interface for real-time seizure detection and control. *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. doi: 10.1109/IEMBS.2010.5627243

Lilliefors, H. W. (1967). On the Kolmogorov-Smirnov Test for Normality with Mean and Variance Unknown. *Journal of the American Statistical Association*, *62*(318), 399–402. doi: 10.2307/2283970

Lin, S.-K., Istiqomah, Wang, L.-C., Lin, C.-Y., & Chiueh, H. (2018). An Ultra-Low Power Smart Headband for Real-Time Epileptic Seizure Detection. *IEEE Journal of Translational Engineering in Health and Medicine*. doi: 10.1109/JTEHM.2018.2861882

Lin, Y.-P., & Jung, T.-P. (2017, June). Improving EEG-Based Emotion Classification Using Conditional Transfer Learning. *Frontiers in Human Neuroscience*, *11*, 334. doi: 10/ggbrpn

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*.

Long, J., Li, Y., Wang, H., Yu, T., Pan, J., & Li, F. (2012, September). A Hybrid Brain Computer Interface to Control the Direction and Speed of a Simulated or Real Wheelchair. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *20*(5), 720–729. doi: 10/f4b9t8

Lotte, F. (2012, November). A new feature and associated optimal spatial filter for EEG signal classification: Waveform Length. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)* (pp. 1302–1305).

Lotte, F., Bougrain, L., Cichocki, A., Clerc, M., Congedo, M., Rakotomamonjy, A., & Yger, F. (2018, June). A review of classification algorithms for EEG-based brain-computer interfaces: A 10 year update. *Journal of Neural Engineering*, *15*(3), 031005. doi: 10.1088/1741-2552/aab2f2

Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F., & Arnaldi, B. (2007). A review of classification algorithms for EEG-based brain-computer interfaces. *Journal of neural engineering*. doi: 10.1088/1741-2560/4/2/R01

Lu, K., Grover, A., Abbeel, P., & Mordatch, I. (2021, June). Pretrained Transformers as Universal Computation Engines. *arXiv:2103.05247 [cs]*.

Lu, W., Wei, Y., Yuan, J., Deng, Y., & Song, A. (2020). Tractor Assistant Driving Control Method Based on EEG Combined With RNN-TL Deep Learning Algorithm. *undefined*.

Lu, Z., Gao, N., Liu, Y., & Li, Q. (2018, October). The Detection of P300 Potential Based on Deep Belief Network. In *2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)* (pp. 1–5). doi: 10/gncp7j

Luck, S. J. (2014). *An Introduction to the Event-Related Potential Technique* (Second ed.). Cambridge, MA, USA: A Bradford Book.

Macé, E., Montaldo, G., Cohen, I., Baulac, M., Fink, M., & Tanter, M. (2011, August). Functional ultrasound imaging of the brain. *Nature Methods*, *8*(8), 662–664. doi:

10.1038/nmeth.1641

Marshall, D., Coyle, D., Wilson, S., & Callaghan, M. (2013). Games, Gameplay, and BCI: The State of the Art. *IEEE Transactions on Computational Intelligence and AI in Games*. doi: 10.1109/TCIAIG.2013.2263555

McCane, L. M., Sellers, E. W., McFarland, D. J., Mak, J. N., Carmack, C. S., Zeitlin, D., ... Vaughan, T. M. (2014, June). Brain-computer interface (BCI) evaluation in people with amyotrophic lateral sclerosis. *Amyotrophic Lateral Sclerosis & Frontotemporal Degeneration*, *15*(3-4), 207–215. doi: 10.3109/21678421.2013.865750

McCann, B., Keskar, N. S., Xiong, C., & Socher, R. (2018, June). *The Natural Language Decathlon: Multitask Learning as Question Answering* (No. arXiv:1806.08730). arXiv. doi: 10.48550/arXiv.1806.08730

Meng, J., Zhang, S., Bekyo, A., Olsoe, J., Baxter, B., & He, B. (2016). Noninvasive Electroencephalogram Based Control of a Robotic Arm for Reach and Grasp Tasks. *Scientific reports*. doi: 10.1038/srep38565

Menon, V., & Crottaz-Herbette, S. (2005). Combined EEG and fMRI studies of human brain function. *International Review of Neurobiology*, *66*, 291–321. doi: 10/ctzzhf

Michielli, N., Acharya, U. R., & Molinari, F. (2019, March). Cascaded LSTM recurrent neural network for automated sleep stage classification using single-channel EEG signals. *Computers in Biology and Medicine*, *106*, 71–81. doi: 10/gjzb9h

Mikołajczyk, A., & Grochowski, M. (2018). Data augmentation for improving deep learning in image classification problem. *2018 International Interdisciplinary PhD Workshop (IIPhDW)*. doi: 10.1109/IIPHDW.2018.8388338

Mikołajewska, E., & Mikołajewski, D. (2014, July). Non-invasive EEG-based brain-computer interfaces in patients with disorders of consciousness. *Military Medical Research*, *1*(1), 14. doi: 10.1186/2054-9369-1-14

Miller, K. J., Hermes, D., & Staff, N. P. (2020, July). The current state of electrocorticography-based brain–computer interfaces. *Neurosurgical Focus*, *49*(1), E2. doi: 10/gmkcw8

Moore, G. E. (2006, September). Cramming more components onto integrated circuits, Reprinted from Electronics, volume 38, number 8, April 19, 1965, pp.114 ff. *IEEE Solid-State Circuits Society Newsletter*, *11*(3), 33–35. doi: 10.1109/N-SSC.2006.4785860

Mrachacz-Kersting, N., Jiang, N., Stevenson, A., Niazi, I., Kostic, V., Pavlovic, A., ... Farina, D. (2016). Efficient neuroplasticity induction in chronic stroke patients by an associative brain-computer interface. *Journal of neurophysiology*. doi: 10.1152/jn.00918.2015

Müller-Putz, G., & Pfurtscheller, G. (2008). Control of an Electrical Prosthesis With an SSVEP-Based BCI. *IEEE Transactions on Biomedical Engineering*. doi: 10.1109/TBME.2007.897815

Murugappan, M., Murugappan, S., Balaganapathy, & Gerard, C. (2014, March). Wireless EEG signals based Neuromarketing system using Fast Fourier Transform (FFT). In *2014 IEEE 10th International Colloquium on Signal Processing and its Applications* (pp. 25–30). doi: 10/gnbc93

Musk, E., & Neuralink. (2019, October). An Integrated Brain-Machine Interface Platform With Thousands of Channels. *Journal of Medical Internet Research*, *21*(10), e16194. doi:

10/ggcntw

Naeem, M., Brunner, C., Leeb, R., Graimann, B., & Pfurtscheller, G. (2006, September). Seperability of four-class motor imagery data using independent components analysis. *Journal of Neural Engineering*, *3*(3), 208–216. doi: 10.1088/1741-2560/3/3/003

Nijboer, F., Sellers, E., Mellinger, J., Jordan, M., Matuz, T., Furdea, A., ... Kübler, A. (2008, August). A P300-based brain-computer interface for people with amyotrophic lateral sclerosis. *Clinical neurophysiology : official journal of the International Federation of Clinical Neurophysiology*, *119*(8), 1909–1916. doi: 10.1016/j.clinph.2008.03.034

Olivas-Padilla, B. E., & Chacon-Murguia, M. I. (2019, February). Classification of multiple motor imagery using deep convolutional neural networks and spatial filters. *Applied Soft Computing*, *75*, 461–472. doi: 10/gncp7f

Oostenveld, R., & Praamstra, P. (2001, April). The five percent electrode system for high-resolution EEG and ERP measurements. *Clinical Neurophysiology*, *112*(4), 713–719. doi: 10.1016/S1388-2457(00)00527-7

Oppenheim, A. V., Buck, J. R., & Schafer, R. W. (2001). *Discrete-time Signal Processing. Vol.2.* Upper Saddle River, N.J. : Prentice Hall.

Ortiz Suárez, P. J., Sagot, B., & Romary, L. (2019, July). Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019* (pp. 9–16). Leibniz-Institut für Deutsche Sprache. doi: 10.14618/ids-pub-9021

Oussous, A., Benjelloun, F.-Z., Lahcen, A. A., & Belfkih, S. (2018). Big Data technologies: A survey. *J. King Saud Univ. Comput. Inf. Sci.*. doi: 10.1016/J.JKSUCI.2017.06.001

Özdenizci, O., Wang, Y., Koike-Akino, T., & Erdoğmuş, D. (2019). Transfer Learning in Brain-Computer Interfaces with Adversarial Variational Autoencoders. *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*. doi: 10.1109/NER.2019.8716897

Pan, S. J., & Yang, Q. (2010, October). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359. doi: 10/bc4vws

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc.

Perez, L., & Wang, J. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. *ArXiv*.

Perrin, F., Pernier, J., Bertrand, O., & Echallier, J. F. (1989, February). Spherical splines for scalp potential and current density mapping. *Electroencephalography and Clinical Neurophysiology*, *72*(2), 184–187. doi: 10/frdn6x

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018, June). Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (pp. 2227–2237). New Orleans, Louisiana: Association for Computational Linguistics. doi: 10.18653/v1/N18-1202

Pfurtscheller, G., & da Silva, F. L. (2017). *EEG Event-Related Desynchronization and Event-*

*Related Synchronization*. Oxford University Press.

Pfurtscheller, G., & Lopes da Silva, F. H. (1999, November). Event-related EEG/MEG synchronization and desynchronization: Basic principles. *Clinical Neurophysiology: Official Journal of the International Federation of Clinical Neurophysiology*, *110*(11), 1842–1857. doi: 10/ffwmrh

Pfurtscheller, G., Neuper, C., Muller, G., Obermaier, B., Krausz, G., Schlögl, A., ... Schrank, C. (2003, July). Graz-BCI: State of the art and clinical applications. *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society*, *11*, 177–80. doi: 10/dpn8fh

Pratt, L. Y. (1992). Discriminability-Based Transfer between Neural Networks. In *Advances in Neural Information Processing Systems* (Vol. 5). Morgan-Kaufmann.

Pratt, L. Y., & Jennings, B. (1996). A survey of transfer between connectionist networks. *Connection Science*, *8*(2), 163–184. doi: 10.1080/095400996116866

Pratt, L. Y., Mostow, J., & Kamm, C. A. (1991, July). Direct transfer of learned information among neural networks. In *Proceedings of the ninth National conference on Artificial intelligence - Volume 2* (pp. 584–589). Anaheim, California: AAAI Press.

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., & Huang, X. (2020). Pre-trained Models for Natural Language Processing: A Survey. *ArXiv*. doi: 10.1007/s11431-020-1647-3

Radford, A., & Narasimhan, K. (2018). Improving Language Understanding by Generative Pre-Training. *OpenAI*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. *OpenAI*.

Raducanu, B. C., Yazicioglu, R. F., Lopez, C. M., Ballini, M., Putzeys, J., Wang, S., ... Mitra, S. (2017, October). Time Multiplexed Active Neural Probe with 1356 Parallel Recording Sites. *Sensors (Basel, Switzerland)*, *17*(10), E2388. doi: 10.3390/s17102388

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, *21*(140), 1–67.

Raghu, M., Zhang, C., Kleinberg, J., & Bengio, S. (2019). Transfusion: Understanding transfer learning for medical imaging. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc.

Raghu, S., Sriraam, N., Temel, Y., Rao, S. V., & Kubben, P. L. (2020). EEG based multi-class seizure type classification using convolutional neural network and transfer learning. *Neural Networks*, *124*, 202–212. doi: 10.1016/j.neunet.2020.01.017

Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007, June). Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning* (pp. 759–766). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/1273496.1273592

Rakotomamonjy, A., Guigue, V., Mallet, G., & Alvarado, V. (2005). Ensemble of SVMs for Improving Brain Computer Interface P300 Speller Performances. In W. Duch, J. Kacprzyk, E. Oja, & S. Zadrożny (Eds.), *Artificial Neural Networks: Biological Inspirations – ICANN*

*2005* (pp. 45–50). Berlin, Heidelberg: Springer. doi: 10/bx78jr

Ramadan, R., & Vasilakos, A. (2017). Brain computer interface: Control signals review. *Neurocomputing*. doi: 10.1016/j.neucom.2016.10.024

Ramos-Murguialday, A., Broetz, D., Rea, M., Läer, L., Yilmaz, Ö., Brasil, F., ... Birbaumer, N. (2013). Brain–machine interface in chronic stroke rehabilitation: A controlled study. *Annals of neurology*. doi: 10.1002/ana.23879

Ranzato, M., Poultney, C., Chopra, S., & Cun, Y. (2006). Efficient learning of sparse representations with an energy-based model. In B. Schölkopf, J. Platt, & T. Hoffman (Eds.), *Advances in neural information processing systems* (Vol. 19). MIT Press.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., ... Song, Y. S. (2019, December). Evaluating Protein Transfer Learning with TAPE. *Advances in neural information processing systems*, *32*, 9689–9701.

Rashid, M., Sulaiman, N., P. P. Abdul Majeed, A., Musa, R. M., Ab. Nasir, A. F., Bari, B. S., & Khatun, S. (2020). Current Status, Challenges, and Possible Solutions of EEG-Based Brain-Computer Interface: A Comprehensive Review. *Frontiers in Neurorobotics*, *14*, 25. doi: 10/gj326h

Rebuffi, S.-A., Bilen, H., & Vedaldi, A. (2017). Learning multiple visual domains with residual adapters. In I. Guyon et al. (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc.

Redrovan, D. V., & Kim, D. (2018, March). Hand gestures recognition using machine learning for control of multiple quadrotors. In *2018 IEEE Sensors Applications Symposium (SAS)* (pp. 1–6). doi: 10.1109/SAS.2018.8336782

Rivest, F., & Kohar, R. (2020, January). A New Timing Error Cost Function for Binary Time Series Prediction. *IEEE transactions on neural networks and learning systems*, *31*(1), 174–185. doi: 10.1109/TNNLS.2019.2900046

Robbins, H., & Monro, S. (1951). A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, *22*(3), 400–407.

Roy, Y., Banville, H., Albuquerque, I., Gramfort, A., Falk, T. H., & Faubert, J. (2019, August). Deep learning-based electroencephalography analysis: A systematic review. *Journal of Neural Engineering*, *16*(5), 051001. doi: 10/ggc76t

Sameer, M., Gupta, A., Chakraborty, C., & Gupta, B. (2020). ROC Analysis for detection of Epileptical Seizures using Haralick features of Gamma band. *2020 National Conference on Communications (NCC)*. doi: 10/gnhcv2

Sankaranarayanan, S., Jain, A., Chellappa, R., & Lim, S. N. (2018, April). Regularizing Deep Networks Using Efficient Layerwise Adversarial Training. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1).

Sannelli, C., Vidaurre, C., Müller, K.-R., & Blankertz, B. (2019, January). A large scale screening study with a SMR-based BCI: Categorization of BCI users and differences in their SMR activity. *PLOS ONE*, *14*(1), e0207351. doi: 10.1371/journal.pone.0207351

Saxe, A. M., McClelland, J. L., & Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In Y. Bengio & Y. LeCun (Eds.), *2nd international conference on learning representations, ICLR 2014, banff, AB, canada, april 14-16, 2014,*

*conference track proceedings.*

Saxena, S., & Cunningham, J. P. (2019, April). Towards the neural population doctrine. *Current Opinion in Neurobiology*, *55*, 103–111. doi: 10.1016/j.conb.2019.02.002

Schalk, G., McFarland, D. J., Hinterberger, T., Birbaumer, N., & Wolpaw, J. R. (2004, June). BCI2000: A general-purpose brain-computer interface (BCI) system. *IEEE transactions on bio-medical engineering*, *51*(6), 1034–1043. doi: 10.1109/TBME.2004.827072

Schlögl, A., Lee, F., Bischof, H., & Pfurtscheller, G. (2005, August). Characterization of four-class motor imagery EEG data for the BCI-competition 2005. *Journal of Neural Engineering*, *2*(4), L14–L22. doi: 10/dgb66b

Schwemmer, M. A., Skomrock, N. D., Sederberg, P. B., Ting, J. E., Sharma, G., Bockbrader, M. A., & Friedenberg, D. A. (2018, November). Meeting brain–computer interface user performance expectations using a deep neural network decoding framework. *Nature Medicine*, *24*(11), 1669–1676. doi: 10.1038/s41591-018-0171-y

Sellers, E., & Donchin, E. (2006). A P300-based brain–computer interface: Initial tests by ALS patients. *Clinical Neurophysiology*. doi: 10.1016/j.clinph.2005.06.027

Sennrich, R., Haddow, B., & Birch, A. (2016, August). Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1715–1725). Berlin, Germany: Association for Computational Linguistics. doi: 10.18653/v1/P16-1162

Serdar Bascil, M., Tesneli, A. Y., & Temurtas, F. (2015, June). Multi-channel EEG signal feature extraction and pattern recognition on horizontal mental imagination task of 1-D cursor movement for brain computer interface. *Australasian Physical & Engineering Sciences in Medicine*, *38*(2), 229–239. doi: 10/gnbc95

Shakeel, A., Navid, M. S., Anwar, M. N., Mazhar, S., Jochumsen, M., & Niazi, I. K. (2015, December). A Review of Techniques for Detection of Movement Intention Using Movement-Related Cortical Potentials. *Computational and Mathematical Methods in Medicine*, *2015*, e346217. doi: 10.1155/2015/346217

Shannon, C. (1949, January). Communication in the Presence of Noise. *Proceedings of the IRE*, *37*(1), 10–21. doi: 10/ftzz7r

Sharmila, A., & Geethanjali, P. (2020). DWT Based Time Domain Features on Detection of Epilepsy Seizures from EEG Signal. In G. Naik (Ed.), *Biomedical Signal Processing: Advances in Theory, Algorithms and Applications* (pp. 181–200). Singapore: Springer. doi: 10.1007/978-981-13-9097-5_9

Shelhamer, E., Long, J., & Darrell, T. (2017). Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(4), 640–651. doi: 10.1109/TPAMI.2016.2572683

Shimodaira, H. (2000, October). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, *90*(2), 227–244. doi: 10.1016/S0378-3758(00)00115-4

Shin, J., Kwon, J., & Im, C.-H. (2018). A Ternary Hybrid EEG-NIRS Brain-Computer Interface for the Classification of Brain Activation Patterns during Mental Arithmetic, Motor Imagery, and Idle State. *Frontiers in Neuroinformatics*, *12*, 5. doi: 10/ggrkzk

Shorten, C., & Khoshgoftaar, T. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*. doi: 10.1186/s40537-019-0197-0

Shortliffe, E., & Barnett, G. (2006, January). Biomedical Data: Their Acquisition, Storage, and Use. In *Biomedical informatics* (pp. 46–79). doi: 10.1007/0-387-36278-9_2

Simonyan, K., & Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27). Curran Associates, Inc.

Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ICLR*.

Singh, S. P. (2014, March). Magnetoencephalography: Basic principles. *Annals of Indian Academy of Neurology*, *17*(Suppl 1), S107-S112. doi: 10/gcb3cc

Song, Z., Fang, T., Ma, J., Zhang, Y., Le, S., Zhan, G., ... Kang, X. (2021, February). Evaluation and Diagnosis of Brain Diseases based on Non-invasive BCI. In *2021 9th International Winter Conference on Brain-Computer Interface (BCI)* (pp. 1–6). doi: 10.1109/BCI51272.2021.9385291

Srinivasan, V., Eswaran, C., & Sriraam, N. (2007, May). Approximate Entropy-Based Epileptic EEG Detection Using Artificial Neural Networks. *IEEE Transactions on Information Technology in Biomedicine*, *11*(3), 288–295. doi: 10.1109/TITB.2006.884369

Stevenson, I. H., & Kording, K. P. (2011, February). How advances in neural recording affect data analysis. *Nature Neuroscience*, *14*(2), 139–142. doi: 10.1038/nn.2731

Steyrl, D., Kobler, R. J., & Müller-Putz, G. R. (2016). On Similarities and Differences of Invasive and Non-Invasive Electrical Brain Signals in Brain-Computer Interfacing. *Journal of Biomedical Science and Engineering*, *09*(08), 393. doi: 10.4236/jbise.2016.98034

Suarjaya, I. M. A. D. (2012, March). A New Algorithm for Data Compression Optimization. *International Journal of Advanced Computer Science and Applications (IJACSA)*, *3*(8). doi: 10.14569/IJACSA.2012.030803

Sun, L., Liu, Y., & Beadle, P. (2005, May). Independent component analysis of EEG signals. In *Proceedings of 2005 IEEE International Workshop on VLSI Design and Video Technology, 2005.* (pp. 219–222). doi: 10/c8q6qt

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015, June). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9). doi: 10.1109/CVPR.2015.7298594

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A Survey on Deep Transfer Learning. In V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, & I. Maglogiannis (Eds.), *Artificial Neural Networks and Machine Learning – ICANN 2018* (pp. 270–279). Cham: Springer International Publishing. doi: 10.1007/978-3-030-01424-7_27

Tangermann, M., Müller, K.-R., Aertsen, A., Birbaumer, N., Braun, C., Brunner, C., ... Blankertz, B. (2012). Review of the BCI Competition IV. *Frontiers in Neuroscience*, *6*, 55. doi: 10/ghhvkk

Tayeb, Z., Fedjaev, J., Ghaboosi, N., Richter, C., Everding, L., Qu, X., ... Conradt, J. (2019, January). Validating Deep Neural Networks for Online Decoding of Motor Imagery Movements from EEG Signals. *Sensors*, *19*(1), 210. doi: 10/ggkz2j

Thomson, D. (1982, September). Spectrum estimation and harmonic analysis. *Proceedings of the IEEE*, *70*(9), 1055–1096. doi: 10.1109/PROC.1982.12433

Townsend, G., LaPallo, B. K., Boulay, C., Krusienski, D., Frye, G. E., Hauser, C. K., ... Sellers, E. (2010). A novel P300-based brain–computer interface stimulus presentation paradigm: Moving beyond rows and columns. *Clinical Neurophysiology*. doi: 10.1016/j.clinph.2010.01.030

Turnip, A., & Junaidi, E. (2014, August). Removal artifacts from EEG signal using independent component analysis and principal component analysis. In *2014 2nd International Conference on Technology, Informatics, Management, Engineering Environment* (pp. 296–302). doi: 10/gnbc98

Uusitalo, M. A., & Ilmoniemi, R. J. (1997, March). Signal-space projection method for separating MEG or EEG into components. *Medical and Biological Engineering and Computing*, *35*(2), 135–140. doi: 10/dd6tm7

van de Laar, B., Gürkök, H., Plass-Oude Bos, D., Poel, M., & Nijholt, A. (2013, June). Experiencing BCI Control in a Popular Computer Game. *IEEE Transactions on Computational Intelligence and AI in Games*, *5*(2), 176–184. doi: 10.1109/TCIAIG.2013.2253778

van Dokkum, L. E. H., Ward, T., & Laffont, I. (2015, February). Brain computer interfaces for neurorehabilitation – its current status as a rehabilitation strategy post-stroke. *Annals of Physical and Rehabilitation Medicine*, *58*(1), 3–8. doi: 10/gjgt4f

van den Oord, A., Li, Y., & Vinyals, O. (2019, January). Representation Learning with Contrastive Predictive Coding. *arXiv:1807.03748 [cs, stat]*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is All you Need. In *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc.

Vidal, J. J. (1973). Toward Direct Brain-Computer Communication. *Annual Review of Biophysics and Bioengineering*, *2*(1), 157–180. doi: 10/ffs9t3

Vilela, M., & Hochberg, L. R. (2020). Applications of brain-computer interfaces to the control of robotic and prosthetic arms. *Handbook of Clinical Neurology*, *168*, 87–99. doi: 10.1016/B978-0-444-63934-9.00008-1

Waldert, S. (2016). Invasive vs. Non-Invasive Neuronal Signals for Brain-Machine Interfaces: Will One Prevail? *Frontiers in Neuroscience*, *10*.

Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., ... Tang, X. (2017, July). Residual Attention Network for Image Classification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 6450–6458). doi: 10.1109/CVPR.2017.683

Wang, H., Li, Y., Long, J., Yu, T., & Gu, Z. (2014, October). An asynchronous wheelchair control by hybrid EEG–EOG brain–computer interface. *Cognitive Neurodynamics*, *8*(5), 399–409. doi: 10/gc6vxq

Wang, Z., Song, Y., & Zhang, C. (2008). Transferred Dimensionality Reduction. In W. Daelemans, B. Goethals, & K. Morik (Eds.), *Machine Learning and Knowledge Discovery in Databases* (pp. 550–565). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-540-87481-2_36

Wei, J., & Zou, K. (2019). EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. *EMNLP*. doi: 10.18653/v1/D19-1670

Weisberg, S. (1980). Maximum Likelihood Estimation. In *Applied Linear Regression* (pp. 309–312). New York, NY: Wiley.

Welch, P. (1967, June). The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, *15*(2), 70–73. doi: 10/fjndmb

Wermter, S., Riloff, E., & Scheler, G. (Eds.). (1996). *Connectionist, statistical, and symbolic approaches to learning for natural language processing* (Vol. 1040). Springer. doi: 10.1007/3-540-60925-3

Winograd, T. (1971). *Procedures as a representation for data in a computer program for understanding natural language*. M.I.T. Project MAC.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Rush, A. M. (2020, October). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations* (pp. 38–45). Online: Association for Computational Linguistics.

Wolpaw, J., Birbaumer, N., Heetderks, W., McFarland, D., Peckham, P., Schalk, G., ... Vaughan, T. (2000, June). Brain-computer interface technology: A review of the first international meeting. *IEEE Transactions on Rehabilitation Engineering*, *8*(2), 164–173. doi: 10/dz4238

Wolpaw, J. R., McFarland, D. J., Neat, G. W., & Forneris, C. A. (1991, March). An EEG-based brain-computer interface for cursor control. *Electroencephalography and Clinical Neurophysiology*, *78*(3), 252–259. doi: 10/bj36jg

Wu, P., & Dietterich, T. G. (2004, July). Improving SVM accuracy by training on auxiliary data sources. In *Proceedings of the twenty-first international conference on Machine learning* (p. 110). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/1015330.1015436

Wyler, A. R. (1987). Electrocorticography. In H. G. Wieser & C. E. Elger (Eds.), *Presurgical Evaluation of Epileptics* (pp. 183–191). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-642-71103-9_31

Xu, G., Shen, X., Chen, S., Zong, Y., Zhang, C., Yue, H., ... Che, W. (2019). A Deep Transfer Convolutional Neural Network Framework for EEG Signal Classification. *IEEE Access*, *7*, 112767–112776. doi: 10.1109/ACCESS.2019.2930958

Xu, J., Mitra, S., Van Hoof, C., Yazicioglu, R. F., & Makinwa, K. A. A. (2017). Active Electrodes for Wearable EEG Acquisition: Review and Electronics Design Methodology. *IEEE Reviews in Biomedical Engineering*, *10*, 187–198. doi: 10.1109/RBME.2017.2656388

Yamins, D., Hong, H., Cadieu, C., Solomon, E., Seibert, D., & DiCarlo, J. (2014). Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*. doi: 10.1073/pnas.1403112111

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NeurIPS*.

Yger, F., Berar, M., & Lotte, F. (2017, October). Riemannian Approaches in Brain-Computer Interfaces: A Review. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *25*(10), 1753–1762. doi: 10.1109/TNSRE.2016.2627016

Yildiz, K. A., Shin, A. Y., & Kaufman, K. R. (2020, March). Interfaces with the peripheral nervous system for the control of a neuroprosthetic limb: A review. *Journal of NeuroEngineering and Rehabilitation*, *17*(1), 43. doi: 10/gh6458

Ying, X. (2019). An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*. doi: 10.1088/1742-6596/1168/2/022022

Yogatama, D., d'Autume, C. d. M., Connor, J. T., Kociský, T., Chrzanowski, M., Kong, L., ... Blunsom, P. (2019). Learning and Evaluating General Linguistic Intelligence. *ArXiv*.

Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015). Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*.

Young, C., Liang, S.-F., Chang, D., Liao, Y.-C., Shaw, F., & Hsieh, C.-H. (2011). A Portable Wireless Online Closed-Loop Seizure Controller in Freely Moving Rats. *IEEE Transactions on Instrumentation and Measurement*. doi: 10.1109/TIM.2010.2050358

Zabidi, A., Mansor, W., Lee, Y. K., & Che Wan Fadzal, C. W. N. F. (2012, September). Short-time Fourier Transform analysis of EEG signal generated during imagined writing. In *2012 International Conference on System Engineering and Technology (ICSET)* (pp. 1–4). doi: 10/gnbc92

Zadrozny, B. (2004, July). Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning* (p. 114). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/1015330.1015425

Zamir, A. R., Sax, A., Shen, W., Guibas, L. J., Malik, J., & Savarese, S. (2018). Taskonomy: Disentangling Task Transfer Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3712–3722).

Zanini, P., Congedo, M., Jutten, C., Said, S., & Berthoumieu, Y. (2018, May). Transfer Learning: A Riemannian Geometry Framework With Applications to Brain–Computer Interfaces. *IEEE Transactions on Biomedical Engineering*, *65*(5), 1107–1116. doi: 10.1109/TBME.2017.2742541

Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision – ECCV 2014* (pp. 818–833). Cham: Springer International Publishing. doi: 10.1007/978-3-319-10590-1_53

Zhang, H., Goodfellow, I., Metaxas, D., & Odena, A. (2019, May). Self-Attention Generative Adversarial Networks. In *Proceedings of the 36th International Conference on Machine Learning* (pp. 7354–7363). PMLR.

Zhang, R. (2019). Making Convolutional Networks Shift-Invariant Again. *ICML*.

Zhang, X., Yao, L., Huang, C., Sheng, Q. Z., & Wang, X. (2017). Intent Recognition in Smart Living Through Deep Recurrent Neural Networks. In *ICONIP*. doi: 10.1007/978-3-319-70096-0_76

Zhang, X., Yao, L., Sheng, Q. Z., Kanhere, S. S., Gu, T., & Zhang, D. (2018, March). Converting Your Thoughts to Texts: Enabling Brain Typing via Deep Feature Learning of EEG Signals. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)* (pp. 1–10). doi: 10.1109/PERCOM.2018.8444575

Zhang, X., Yao, L., Zhang, S., Kanhere, S., Sheng, M., & Liu, Y. (2019). Internet of Things Meets Brain–Computer Interface: A Unified Deep Learning Framework for Enabling Human-

Thing Cognitive Interactivity. *IEEE Internet of Things Journal*. doi: 10.1109/JIOT.2018.2877786

Zhang, Y., Nam, C. S., Zhou, G., Jin, J., Wang, X., & Cichocki, A. (2019, September). Temporally Constrained Sparse Group Spatial Patterns for Motor Imagery BCI. *IEEE Transactions on Cybernetics*, *49*(9), 3322–3332. doi: 10.1109/TCYB.2018.2841847

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., ... He, Q. (2021, January). A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, *109*(1), 43–76. doi: 10.1109/JPROC.2020.3004555

Zoph, B., Ghiasi, G., Lin, T.-Y., Cui, Y., Liu, H., Cubuk, E. D., & Le, Q. (2020). Rethinking pre-training and self-training. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (Vol. 33, pp. 3833–3845). Curran Associates, Inc.

# Appendices

# Appendix A

# Result Summary

Table A.1: Summary of all results achieved in this thesis.

| ***Optimisation runs*** | | | | | |
|---|---|---|---|---|---|
| **Data** | **Pretraining** | **Frozen** | **Dropout** | **Decay** | **Test Score $\mu \pm \sigma$** |
| time-series | yes | all | 0.1 | 1 | $41.6\,\% \pm 6.5\,\%$ |
| features | yes | all | 0.1 | 1 | $30.9\,\% \pm 5.7\,\%$ |

| ***Addressing overfitting*** | | | | | |
|---|---|---|---|---|---|
| **Data** | **Pretraining** | **Frozen** | **Dropout** | **Decay** | **Test Score $\mu \pm \sigma$** |
| time-series | yes | all | 0.2 | 0.9 | $35.9\,\% \pm 9.8\,\%$ |

| ***Randomly initialised*** | | | | | |
|---|---|---|---|---|---|
| **Data** | **Pretraining** | **Frozen** | **Dropout** | **Decay** | **Test Score $\mu \pm \sigma$** |
| time-series | no | none | 0.1 | 1 | $26.1\,\% \pm 1.5\,\%$ |
| features | no | none | 0.1 | 1 | $31.3\,\% \pm 5.2\,\%$ |

| ***Completely unfrozen*** | | | | | |
|---|---|---|---|---|---|
| **Data** | **Pretraining** | **Frozen** | **Dropout** | **Decay** | **Test Score $\mu \pm \sigma$** |
| time-series | yes | none | 0.1 | 1 | $38.4\,\% \pm 7.1\,\%$ |
| features | yes | none | 0.1 | 1 | $30.5\,\% \pm 5.3\,\%$ |

Table A.1: Summary of all results achieved in this thesis.

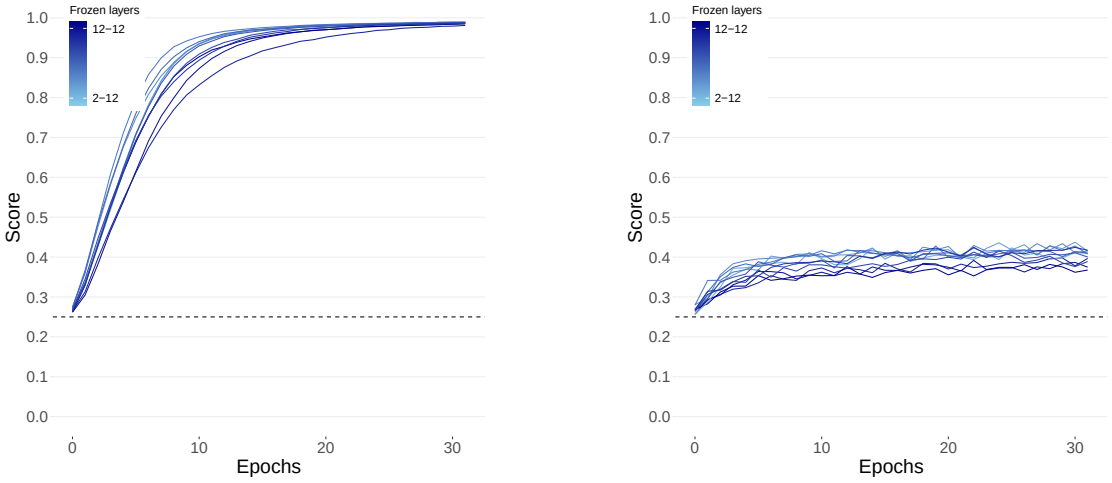
| ***Shallow-to-deep unfrozen*** | | | | | |
|---|---|---|---|---|---|
| **Data** | **Pretraining** | **Frozen** | **Dropout** | **Decay** | **Test Score $\mu \pm \sigma$** |
| time-series | yes | $2-12$ | 0.1 | 1 | $40.0\,\% \pm 8.6\,\%$ |
| time-series | yes | $3-12$ | 0.1 | 1 | $41.7\,\% \pm 6.0\,\%$ |
| time-series | yes | $4-12$ | 0.1 | 1 | $39.8\,\% \pm 6.1\,\%$ |
| time-series | yes | $5-12$ | 0.1 | 1 | $40.3\,\% \pm 7.8\,\%$ |
| time-series | yes | $6-12$ | 0.1 | 1 | $40.2\,\% \pm 6.6\,\%$ |
| time-series | yes | $7-12$ | 0.1 | 1 | $38.4\,\% \pm 5.9\,\%$ |
| time-series | yes | $8-12$ | 0.1 | 1 | $38.3\,\% \pm 7.1\,\%$ |
| time-series | yes | $9-12$ | 0.1 | 1 | $43.1\,\% \pm 9.6\,\%$ |
| time-series | yes | $10-12$ | 0.1 | 1 | $36.7\,\% \pm 5.0\,\%$ |
| time-series | yes | $11-12$ | 0.1 | 1 | $37.8\,\% \pm 1.0\,\%$ |
| time-series | yes | $12-12$ | 0.1 | 1 | $35.8\,\% \pm 6.7\,\%$ |
| features | yes | $2-12$ | 0.1 | 1 | $30.8\,\% \pm 6.4\,\%$ |
| features | yes | $3-12$ | 0.1 | 1 | $29.7\,\% \pm 6.4\,\%$ |
| features | yes | $4-12$ | 0.1 | 1 | $29.7\,\% \pm 6.8\,\%$ |
| features | yes | $5-12$ | 0.1 | 1 | $30.4\,\% \pm 6.4\,\%$ |
| features | yes | $6-12$ | 0.1 | 1 | $29.5\,\% \pm 4.9\,\%$ |
| features | yes | $7-12$ | 0.1 | 1 | $32.0\,\% \pm 4.4\,\%$ |
| features | yes | $8-12$ | 0.1 | 1 | $31.3\,\% \pm 6.6\,\%$ |
| features | yes | $9-12$ | 0.1 | 1 | $29.7\,\% \pm 4.9\,\%$ |
| features | yes | $10-12$ | 0.1 | 1 | $29.3\,\% \pm 6.2\,\%$ |
| features | yes | $11-12$ | 0.1 | 1 | $29.1\,\% \pm 4.0\,\%$ |
| features | yes | $12-12$ | 0.1 | 1 | $30.0\,\% \pm 6.2\,\%$ |

Table A.1: Summary of all results achieved in this thesis.

| ***Deep-to-shallow unfrozen*** | | | | | |
|---|---|---|---|---|---|
| **Data** | **Pretraining** | **Frozen** | **Dropout** | **Decay** | **Test Score $\mu \pm \sigma$** |
| time-series | yes | $1-11$ | 0.1 | 1 | $42.0\,\% \pm 5.7\,\%$ |
| time-series | yes | $1-10$ | 0.1 | 1 | $40.4\,\% \pm 6.2\,\%$ |
| time-series | yes | $1-9$ | 0.1 | 1 | $38.4\,\% \pm 7.1\,\%$ |
| time-series | yes | $1-8$ | 0.1 | 1 | $40.2\,\% \pm 7.0\,\%$ |
| time-series | yes | $1-7$ | 0.1 | 1 | $36.6\,\% \pm 8.2\,\%$ |
| time-series | yes | $1-6$ | 0.1 | 1 | $36.0\,\% \pm 9.1\,\%$ |
| time-series | yes | $1-5$ | 0.1 | 1 | $34.3\,\% \pm 6.1\,\%$ |
| time-series | yes | $1-4$ | 0.1 | 1 | $35.0\,\% \pm 6.4\,\%$ |
| time-series | yes | $1-3$ | 0.1 | 1 | $35.0\,\% \pm 6.5\,\%$ |
| time-series | yes | $1-2$ | 0.1 | 1 | $38.5\,\% \pm 8.6\,\%$ |
| time-series | yes | $1-1$ | 0.1 | 1 | $38.4\,\% \pm 6.8\,\%$ |
| features | yes | $1-11$ | 0.1 | 1 | $30.9\,\% \pm 5.5\,\%$ |
| features | yes | $1-10$ | 0.1 | 1 | $30.2\,\% \pm 5.2\,\%$ |
| features | yes | $1-9$ | 0.1 | 1 | $31.4\,\% \pm 6.5\,\%$ |
| features | yes | $1-8$ | 0.1 | 1 | $28.5\,\% \pm 4.5\,\%$ |
| features | yes | $1-7$ | 0.1 | 1 | $29.4\,\% \pm 4.7\,\%$ |
| features | yes | $1-6$ | 0.1 | 1 | $29.7\,\% \pm 4.5\,\%$ |
| features | yes | $1-5$ | 0.1 | 1 | $31.3\,\% \pm 3.6\,\%$ |
| features | yes | $1-4$ | 0.1 | 1 | $30.0\,\% \pm 4.7\,\%$ |
| features | yes | $1-3$ | 0.1 | 1 | $32.2\,\% \pm 6.2\,\%$ |
| features | yes | $1-2$ | 0.1 | 1 | $31.1\,\% \pm 5.3\,\%$ |
| features | yes | $1-1$ | 0.1 | 1 | $30.8\,\% \pm 4.4\,\%$ |

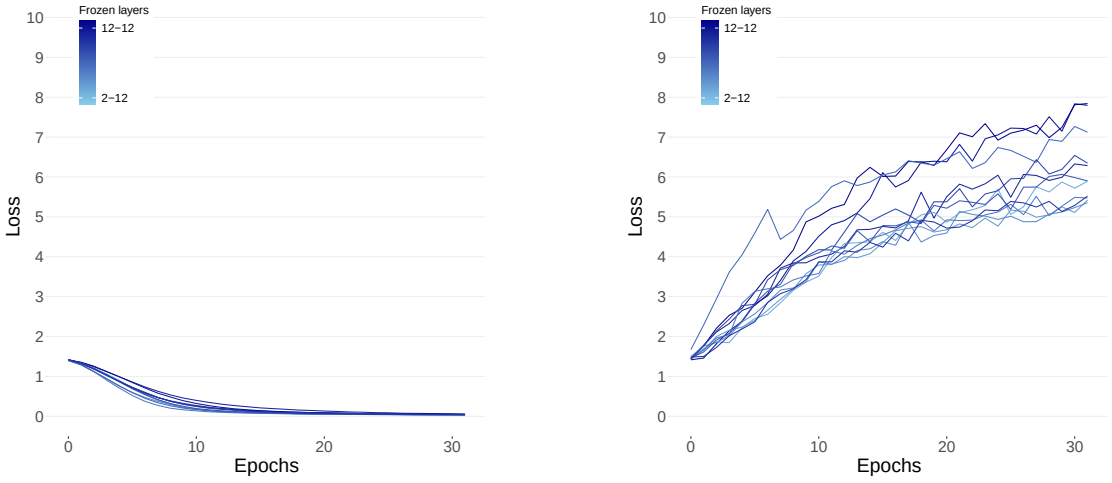
# Appendix B

# Training and validation scores and loss during unfreezing



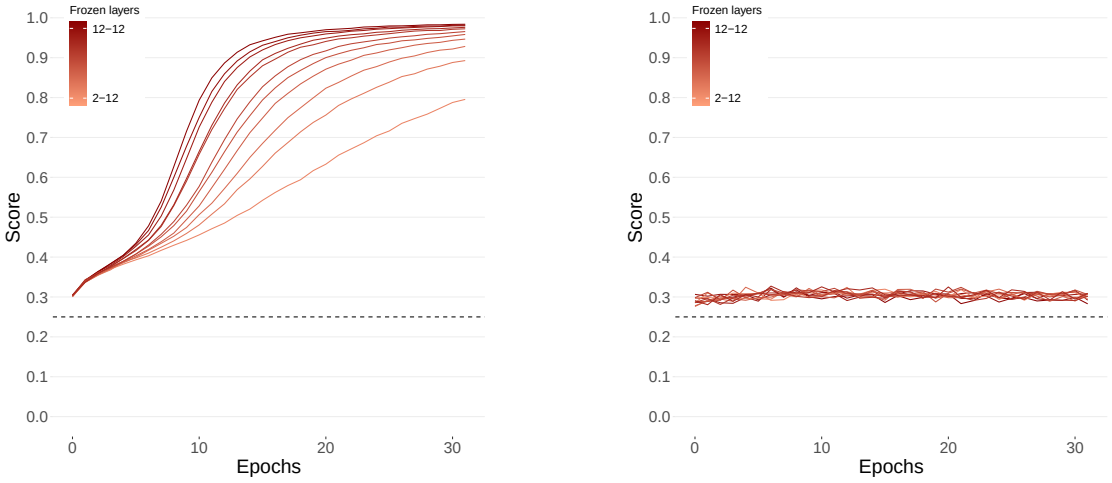

(a) Training (b) Validation

Figure B.1: Participant-wise average training and validation classification accuracy of the model during training on the time-series data. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained its decoder units are unfrozen from shallow to deep. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.



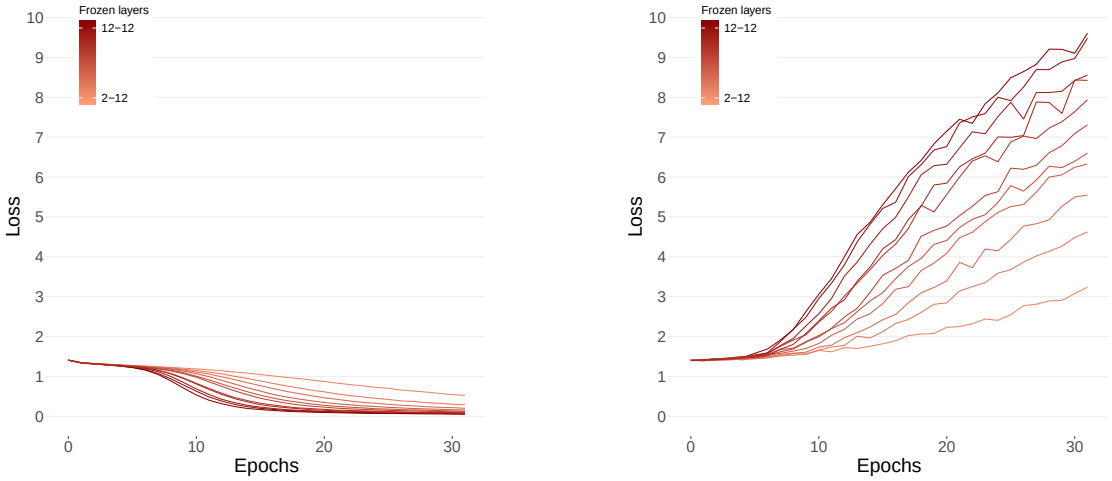

(a) Training (b) Validation

Figure B.2: Participant-wise average training and validation cross-entropy loss of the model during training on the time-series data. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from shallow to deep. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.



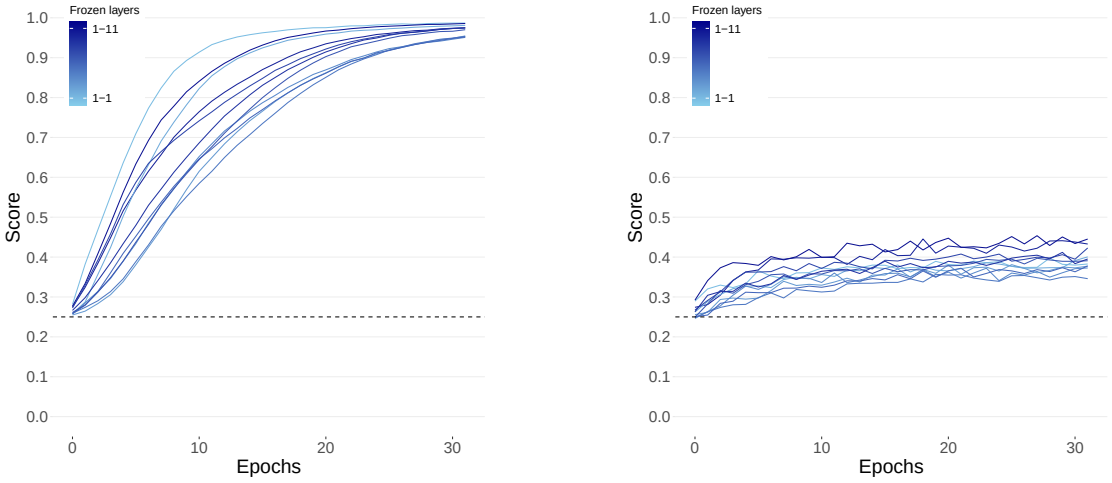

(a) Training (b) Validation

Figure B.3: Participant-wise average training and validation classification accuracy of the model during training on the PSD features. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from shallow to deep. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.



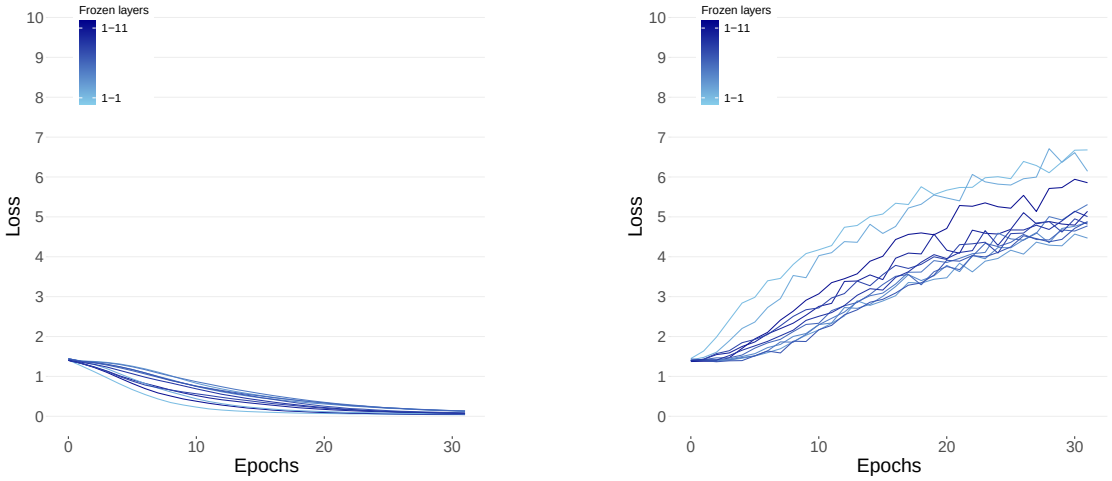

(a) Training (b) Validation

Figure B.4: Participant-wise average training and validation cross-entropy loss of the model during training on the PSD features. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from shallow to deep. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.



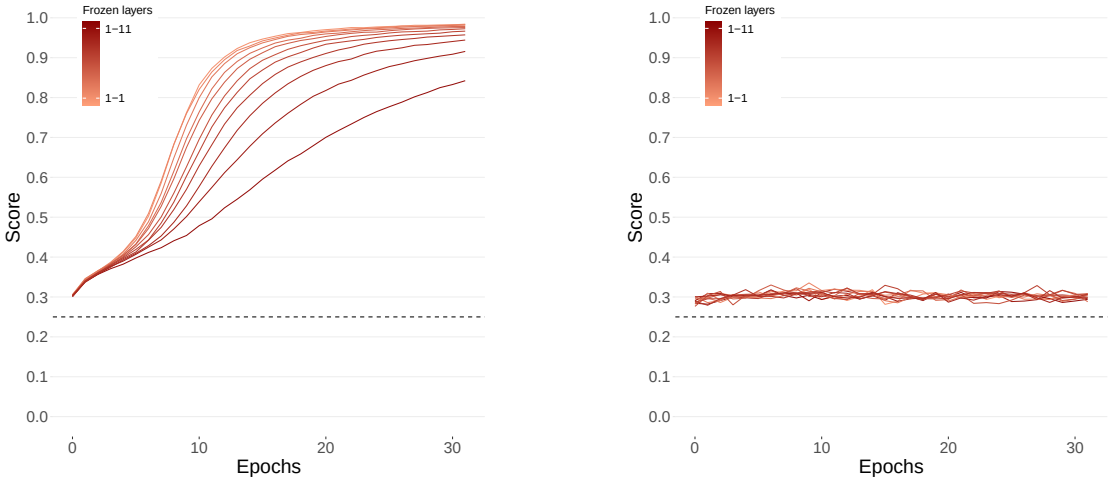

(a) Training (b) Validation

Figure B.5: Participant-wise average training and validation classification accuracy of the model during training on the time-series data. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from deep to shallow. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.



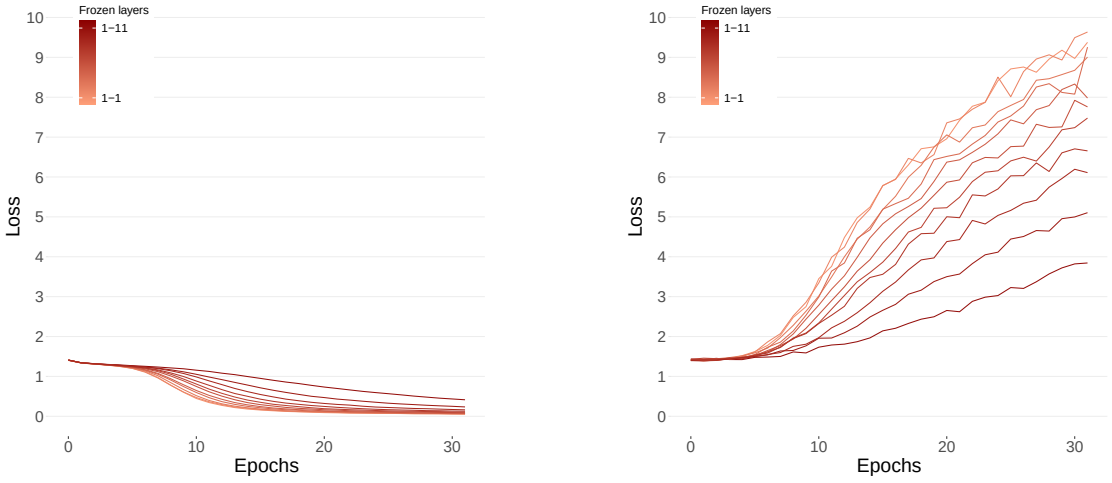

(a) Training (b) Validation

Figure B.6: Participant-wise average training and validation cross-entropy loss of the model during training on the time-series data. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from deep to shallow. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.




(a) Training (b) Validation

Figure B.7: Participant-wise average training and validation classification accuracy of the model during training on the PSD features. The dashed black line denotes random assignment at 25 %. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from deep to shallow. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.




(a) Training (b) Validation

Figure B.8: Participant-wise average training and validation cross-entropy loss of the model during training on the PSD features. The instance of GPT2 is language-pretrained and its decoder units are unfrozen from deep to shallow. The values of the hyperparameters of these runs correspond to the best found hyperparameter values given by Table 5.1.