University of West Bohemia
Faculty of Applied Sciences

Methods for Signal Classification and their Application to the Design of Brain-Computer Interfaces

Ing. Lukáš Vařeka

Doctoral Thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in specialization Computer Science and Engineering

Supervisor: Ing. Pavel Mautner, Ph.D.

Department of Computer Science and Engineering

Pilsen 2018

Západočeská univerzita v Plzni
Fakulta aplikovaných věd

Metody pro klasifikaci signálu a jejich využití na návrh rozhraní mozek-počítač

Ing. Lukáš Vařeka

Disertační práce k získání akademického titulu doktor v oboru Informatika a výpočetní technika

Školitel: Ing. Pavel Mautner, Ph.D.

Katedra informatiky a výpočetní techniky

Plzeň 2018

Prohlášení / Declaration

Předkládám tímto k posouzení a obhajobě disertační práci zpracovanou na závěr doktorského studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni. Prohlašuji tímto, že tuto práci jsem vypracoval samostatně, s použitím odborné literatury a dostupných pramenů uvedených v seznamu, jenž je součástí této práce.

/

I present a dissertation thesis to be reviewed and defended. The thesis was created at the end of the doctoral study at the Faculty of Applied Sciences, University of West Bohemia. I hereby declare that I have created this work alone if not noted otherwise and that all used references are properly cited in a list at the end of this thesis.

Plzeň, 20. 4. 2018 Ing. Lukáš Vařeka

Abstract

The aim of the thesis was to evaluate neural networks for classification of brain reactions to stimuli in P300-based brain-computer interfaces. Several neural network classification algorithms were proposed and compared with traditionally used classifiers in this field. The algorithms tested included supervised modifications of self-organizing maps (SOMs), Adaptive Resonance Theory (ART), and models from deep learning category (stacked autoencoders).

In the first part of the thesis, state-of-the-art signal processing and classification techniques for P300 brain-computer interfaces (BCIs) are introduced. P300 event-related potential BCIs are described in more detail. In the second part of the thesis, algorithms based on neural networks are proposed. Subsequently, the experiments designed to obtain the data used to evaluate proposed algorithms are described. State-of-the art Windowed means paradigm method was used for feature extraction. Different modifications of SOMs, ART networks and stacked autoencoders were used for classification. These models have so far never been explored in P300 BCIs and represent a promising alternative to traditional linear classifiers (Linear Discriminant Analysis, Support Vector Machines). Moreover, an on-line BCI was implemented to provide a real-time opportunity to test the proposed algorithms. Finally, achieved results, ongoing work and possibilities for future work are discussed. Stacked autoencoders were able to match or outperform state-of-the-art classification techniques.

Abstrakt

Cílem dizertační práce bylo ověřit vhodnost neuronových sítí pro klasifikaci reakcí mozku na stimuly v rozhraní mozek-počítač (BCI). Navrhl jsem několik algoritmů založených na neuronových sítích, které byly následně srovnány s nejlepšími současně využívanými algoritmy v této oblasti. Testovány byly modifikace Kohonenových map (SOM) a Adaptive Resonance Theory (ART) umožňující učení s učitelem a také modely z kategorie deep learning (zřetězené autoenkodéry).

V první části práce jsou popsány v současnosti používané metody na zpracování signálu a klasifikaci v BCI systémech. BCI založené na evokované komponentě P300 jsou probrány detailněji. Druhá část práce nejprve pojednává o algoritmech založených na neuronových sítích, které by bylo možné využít ke klasifikaci v oblasti BCI. Následně jsou získána data ověřující navržené postupy. Pro extrakci příznaků byla použita metoda Windowed means založená na rozdělení signálu do předem určených oken. Modifikace SOM, ART a deep learning byly použity pro klasifikaci komponenty P300. Všechny tyto modely nebyly dosud v oblasti BCI založených na komponentě P300 použity a nabízí alternativy ke klasickým algoritmům (Linear Discriminant Analysis, Support Vector Machines). Kromě toho je popsáno online BCI, které umožnilo otestovat navržené algoritmy v reálném čase. Práce je uzavřena výsledky, jejich diskuzí a stručným přehledem současných a plánovaných projektů. Autoenkodéry dosáhly stejných nebo lepších výsledků než ostatní algoritmy.

Acknowledgement

I would like to thank my Ph.D. supervisor Ing. Pavel Mautner, Ph.D. for introducing me to this interesting topic and for valuable advice and ideas that inspired my experimental work.

I am also very thankful to all other members of the neuroinformatics research group, especially Ing. Roman Mouček, Ph.D. for his continuous and friendly support, and for reading this Ph.D. thesis and providing helpful tips before submitting.

Knowledge exchange is vital to increase the impact of research. My gratitude goes to Doc. Ing. Josef Kohout, Ph.D. as the Head of the VP2 Division of the New Technologies for the Information Society for providing motivation and assisting with funding to participate in research abroad.

I especially thank my great parents, grandmother and my dearest wife Kateřina for standing by my side when I needed it, and for believing that I will finish my Ph.D. studies successfully after all those years.

Contents

List of Figures

List of Tables

1 Introduction

A growing interest has been devoted to understanding the human brain. Brain research investigates it from different perspectives. In medicine and in neurobiological research, many brain imaging and monitoring techniques provide us with knowledge that was previously unavailable, e.g. electroencephalography (EEG), magnetoencephalography (MEG), positron emission tomography (PET), functional magnetic resonance imaging (fMRI) and optical imaging. Although modern neuroimaging techniques have helped to discover new knowledge by measuring blood flow in the brain, traditional EEG still maintains its position because it is relatively cheap, non-invasive and has very high temporal resolution. Typically, EEG can measure changes in brain activity on a millisecond-level.

One of the most promising technical applications of EEG are brain-computer interfaces (BCIs). They allow a direct communication between the brain and the computer without traditional pathways using muscles. This is especially important for paralyzed people that do not have any other possibility to communicate with the outside world. However, brain-computer interface design is not straightforward since the intention of the user cannot be directly read from EEG. Instead, special training of the subject is often required. This thesis focuses on the BCIs that are based on specific brain reactions to stimuli, commonly referred to as P300 event-related potentials (ERPs). P300-based BCIs are very popular since they do not require special training of the subjects, instead, visual or auditory stimulation is required. Unfortunately, signal-to-noise ratio is typically low. Many approaches have been proposed to boost the reliability of P300 BCIs. Besides an efficient stimulation protocol and well chosen training data, correct feature extraction and classification techniques are necessary to achieve good results both in accuracy and speed. Recently, new algorithms for classification based on neural networks have emerged. In my opinion, they provide a promising alternative for signal classification in P300-based BCIs.

The main goal of the thesis is to verify if neural networks are suitable for ERP classification in P300 BCIs. Only multi-layer perceptron has been studied and extensively evaluated regarding this application. However, there is a variety of promising neural network models that are generally in the unsupervised learning category, e.g. Adaptive Resonance Theory

(ART) or self-organizing maps (SOMs). These models can be adjusted to perform supervised classification. The clustering phase these methods include may allow us to better analyze feature vectors. Furthermore, in recent years, deep learning has emerged as one of the most important fields of machine learning. For deep learning models, training methods that combine supervised and unsupervised learning were shown to boost classification accuracy. For evaluation purposes, a P300 experiment is designed and performed in the EEG laboratory of the Department of Computer Science and Engineering, University of West Bohemia. Based on the obtained datasets, state-of-the-art feature extraction and classification methods used for comparison are implemented. Then, an extensive evaluation of existing supervised extensions for SOM and ART is performed and where applicable, alternative solutions are proposed and implemented. In addition, some deep learning models (e.g. stacked autoencoders) are also compared and evaluated.

To verify proposed neural networks in an on-line BCI, an application for a simple BCI experiment 'Guess the number' has been developed. This experiment is primarily intended for educational purposes. Some of the algorithms mentioned above have also been tested on a group of approximately 200 school-age children datasets obtained from the 'Guess the number' experiment.

The first part of this thesis introduces ERPs, BCIs and the current state-of-the-art preprocessing, feature extraction and classification algorithms that have been studied for this application. Existing neural network classification algorithms that correspond to the main goal of this thesis are described in more detail. Section 2 introduces electroencephalographic signal and event-related potentials. Brain-computer interfaces and different approaches for their design are explained in Section 3. State-of-the-art techniques for preprocessing and feature extraction of the EEG/ERP data are introduced in Section 4. Related classification techniques are explored in Section 5. In Section 6, problems of the current P300 BCIs are discussed, and a novel approach for BCI design is proposed. Based on that discussion, the aims of the Ph.D. thesis are specified in Subsection 6.4.

Subsequently, the second applied part of the thesis follows. Its aim is to present experiments that have been conducted to verify if neural network algorithms can match the performance of state-of-the-art classifiers for the P300 component detection. This part of

the thesis is introduced in Section 7 that explains how the data necessary for subsequent experiments were obtained. Moreover, preprocessing and feature extraction are discussed and related data-driven methods proposed. Section 8 describes classification using neural networks. Subsequently, the results for different models are presented and discussed. Section 9 describes how an on-line BCI system for verification of the classifiers presented has been designed, implemented, and tested. Conclusion and future work are in Section 10.

2 Electroencephalography

2.1 Introduction

Electroencephalography (EEG) is a technique based on recording the brain electrical activity along the scalp. EEG measures voltage fluctuations which result from ionic current flows within the neurons of the brain [1]. The resulting EEG activity reflects the summation of the synchronous activity of many groups of neurons that have similar spatial orientation. The neurons of the cortex are thought to produce most of the EEG signal because they are well-aligned and fire together. In contrast, activity from deep sources in the brain is generally more difficult to detect. Unfortunately, the sources of the signal can be recovered from the EEG signal only approximately [2]. EEG uses electrodes for measuring EEG signals from multiple areas on the skull, the signal at each electrode is a time variation of the electrical potential difference between the electrode and the reference electrode. The recorded signal is stored as electroencephalogram for evaluation.

EEG is commonly used in clinical practice, neurological and psychological research. The main advantage of EEG is its good resolution in time domain and its non-invasiveness. However, this technique has also drawbacks. One of the biggest disadvantages is the fact that the EEG signal represents many sources of neural activity. Most of this activity is undesired so signal-to-noise ratio is typically very low. [1]

2.2 Recording of the EEG signal

The conventional electrode setting for both research and clinical purposes is called 10-20 system. It typically includes 21 electrodes (excluding the earlobe electrodes). The earlobe electrodes, connected to the left and right earlobes, are often used as the reference electrodes. It is also possible to attach the reference electrode at the root of the nose. The 10-20 system considers some constant distances by using specific anatomic landmarks from which the measurement would be made and then uses 10 or 20% of that specified distance as the electrode interval. The odd electrodes are on the left and the even ones on the right. The system is illustrated in Fig. 2.1. [3]

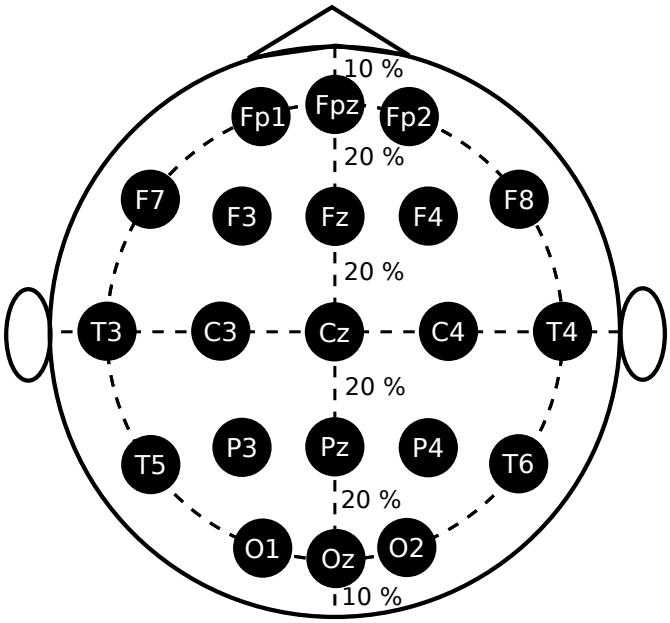


Figure 2.1: 10-20 system including 21 electrodes [3].

Since almost all EEG systems are computer-based, the conversion from analog to digital EEG is required. The conversion is performed by means of multichannel analog-to-digital converters. Fortunately, the effective bandwidth for EEG signals is limited to approximately 100 Hz. Therefore, to satisfy the Nyquist criterion, a minimum frequency of 200 Hz is often enough for sampling of the EEG signals. [3]

2.3 Normal EEG activity

Although EEG is stochastic, certain brain rhythms commonly manifest in the EEG signal. In healthy adults, the amplitudes and frequencies of such signals change from one state of consciousness to another, e.g. wakefulness or sleep. The characteristics of the waves may also change with age. There are five major brain waves distinguished by their different frequency ranges. These frequency bands from low to high frequencies are called delta, theta, alpha, beta (depicted in Fig. 2.2 [3]), and gamma:

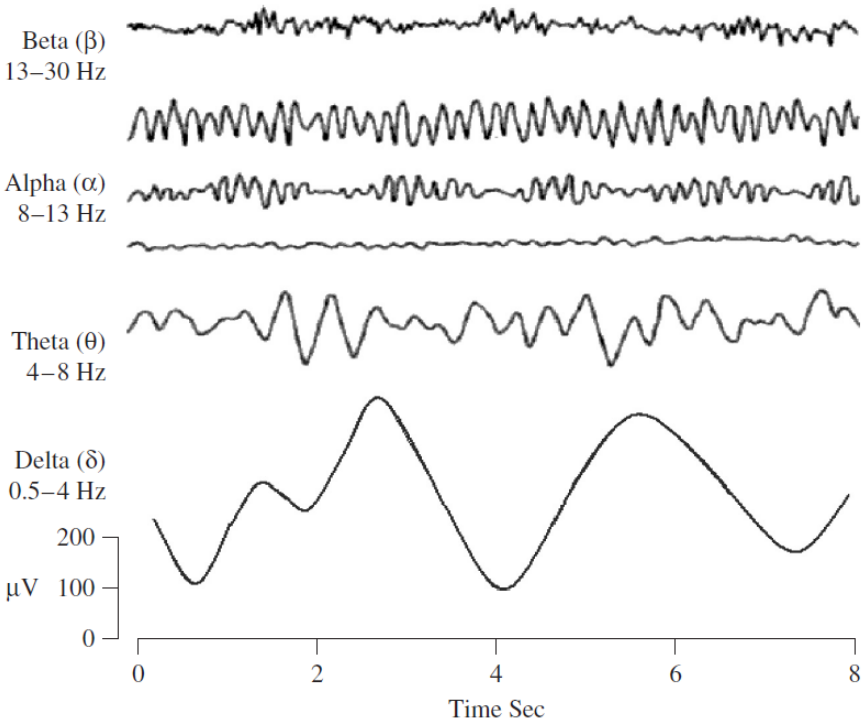


Figure 2.2: Brain rhythms in healthy adults [3].

Delta waves Delta waves lie within the range of 0.5 - 4 Hz. These waves are primarily associated with deep sleep and may also be present in the waking state.

Theta waves Theta waves lie within the range of 4 - 7.5 Hz. They appear as consciousness slips towards drowsiness. Theta waves have been associated with creative inspiration and deep meditation.

Alpha waves Alpha waves appear in the rear half of the head and are usually found over the occipital region of the brain. For alpha waves the frequency lies within the range of 8 - 13 Hz, and commonly appears as a round or sinusoidal shaped signal. Alpha waves have been thought to indicate a relaxed awareness without any attention or concentration. The alpha wave is the most prominent rhythm in brain activity. Most subjects produce alpha waves with their eyes closed. It is reduced or eliminated by opening the eyes, by hearing unfamiliar sounds, by anxiety, mental concentration or attention.

Beta waves A beta wave is the electrical activity of the brain varying within the range of 14 - 26 Hz. It is the usual waking rhythm of the brain associated with active thinking, active attention, focus on the outside world, or problem solving, and is found in normal adults.

Gamma waves The gamma range is associated with the frequencies above 30 Hz (mainly up to 45 Hz). Although the amplitudes of these rhythms are very low and their occurrence is rare, detection of these rhythms can be used for confirmation of certain brain diseases.

2.4 Event-related potentials

Event-related potentials (ERPs) are the changes of the EEG signal associated with something that occurs either in the external world or within the brain itself. An ERP may be triggered by an external stimulation, or emitted by the brain as it makes a decision or initiates a response. They are further classified as exogenous or endogenous. Exogenous ERPs are determined by the physical characteristics of the stimulus while endogenous ERPs are determined by their psychological effects. There are several ERPs differing by their latency, polarity and amplitude. The labeling reflects their polarity (P for the positive, N for the negative) and the latency (time after stimulus). For example, the N100 is a negative event-related potential occurring approximately 100 ms after the event in the signal. Since latency may vary among individuals and even among different recording situations within the same individual, the waveform is often identified by its typical latency. Therefore, the names of ERP components may also be based on sequential numbering of the peaks (e.g. P1, N1, P2, N2, P3). [4]

Some ERPs are associated with any type of visual or audible events (e.g. the N100

component), the others are triggered only by the events following some semantic pattern (e.g. the P300 or the N400 component). The ERP experiments usually aim at eliciting certain ERP components using regular stimulation.

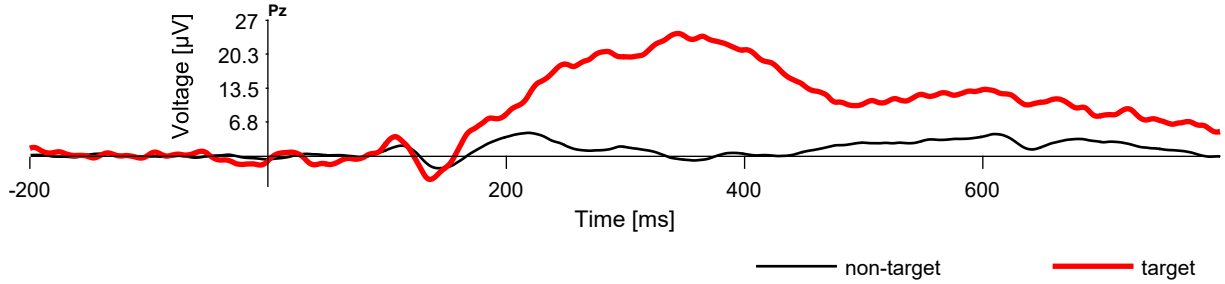


Figure 2.3: Comparison of averaged EEG responses to common (non-target) stimuli and rare (target) stimuli. There is a clear P300 component following the target stimuli.

2.4.1 P300

For example, oddball paradigm [5] is commonly used for the P300 elicitation. In this technique, low-probability target stimuli are mixed with high-probability non-target stimuli. Both stimuli trigger a reaction which can be measured and detected shortly after the event in the EEG signal and consists of multiple ERP components. However, the target stimuli tend to cause a different reaction, with the P300 waveform (sometimes referred to as the P3 component) being most significant. Fig. 2.3 shows an example of averaged event-related potentials for target and non-target stimuli. The P300 waveform is probably related to the process of decision making - it is elicited when the subject classifies the last stimulus as the target (for example by silent counting). The P300 is usually the strongest ERP component and it occurs 250 - 400 ms after the target stimulus as a positive peak. Its amplitude and latency may be influenced by different factors. For example, the P300 amplitude gets larger as target probability gets smaller. The amplitude is also larger when subjects devote more effort to a task. [5]

2.5 Artifacts

Unfortunately, event-related potentials or different useful information in the signal are usually hidden in noise. In EEG, disturbing signals are commonly referred to as artifacts. The main

artifacts can be divided into patient-related (physiological, biological) and system (technical) artifacts [3].

Physiological artifacts include any biological activity arising from other sources than the brain. There are several types of biological artifacts, e.g. blinks, eye movements, muscle activity, and skin potentials. These artifacts can be problematic in two ways. First, they are typically very large compared to the ERP signals and may greatly decrease signal-to-noise ratio of the averaged ERP waveform. Second, some types of artifacts may be systematic rather than random, occurring in some conditions more than others and being time-locked to the stimulus so that the averaging process does not eliminate them. For example, some stimuli may be more likely to elicit blinks than others, which could lead to differences in amplitude in the averaged ERP waveforms. [5]

Detection of eye-blinking artifacts is especially important since the artifacts may distort the data to an unacceptable extent [5]. Depending on the position of the reference electrode, a blink can be seen in the EEG signal as a positive or negative peak appearing at EOG electrode (if any) and a peak with opposite polarity appearing at the scalp electrodes. Deflection decreases with the increasing distance between the eyes and the electrode. A typical eye-blink response is represented by a peak with the amplitude of 50 - 100 μV with the duration of 200 - 400 ms [5]. Figure 2.4 shows an example of a blink in the signal.

The most important method to deal with physiological artifacts is to ask the subject to sit comfortably, not to move and to limit blinking. There is no substitute for clean data [5]. However, in many cases, it is also important to reject or correct the artifacts that necessarily more or less distort the data.

The system artifacts include 50 Hz power supply interference, impedance fluctuation, cable defects, electrical noise from the electronic components, and unbalanced impedances of the electrodes [3]. They are usually easier to eliminate than the biological artifacts, e.g. power supply interference can be eliminated using temporal filtering [5].

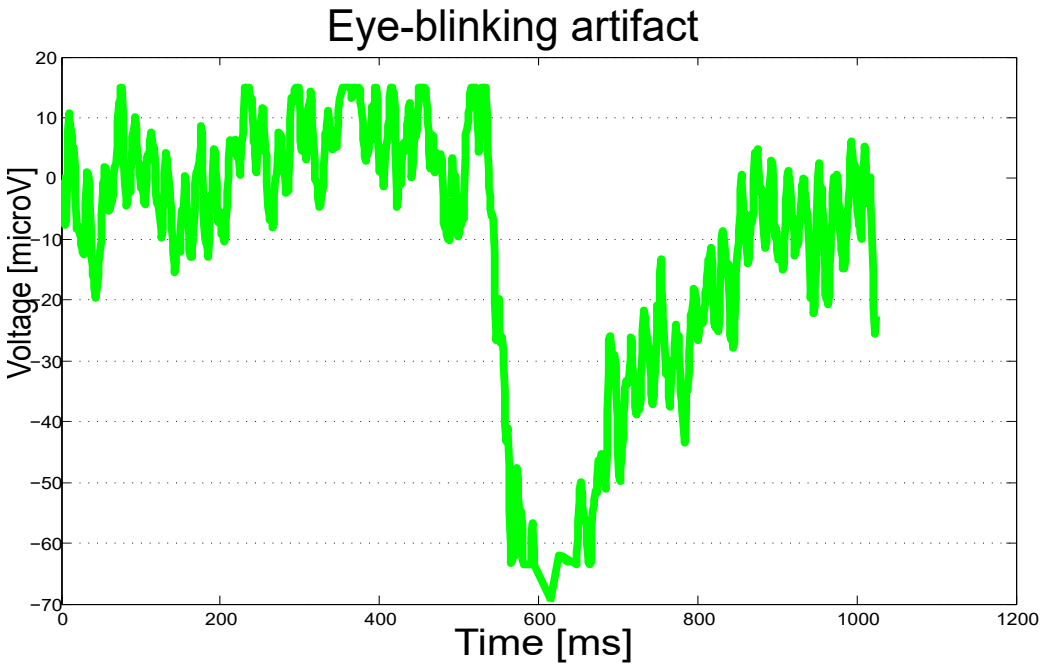


Figure 2.4: An eye-blink in EEG signal. The x-axis presents time and y-axis voltage in μV.

3 Brain-computer interfaces

Recent advances in cognitive neuroscience and brain imaging techniques have started to provide us with the ability to interface directly with the human brain. The scientific interest in brain-computer interfaces is primarily driven by the needs of people with neuromuscular diseases (e.g. amyotrophic lateral sclerosis, brainstem stroke, brain or spinal cord injury, cerebral palsy, muscular dystrophies, multiple sclerosis, and numerous other diseases) [6]. Typically, these diseases damage the neural pathways that control muscles. Nearly two million people are affected in the United States alone, and far more around the world [7]. Those most severely affected may lose all voluntary muscle control and may be completely locked in to their bodies, unable to communicate with the outside world [8].

In the first international meeting on BCI technology, which took place in 1999, at the Rensselarville Institute of Albany (New York), Jonathan R. Wolpaw formalized the definition of the BCI system [9]:

> A brain-computer interface (BCI) is a communication or control system in which the user's messages or commands do not depend on the brain's normal output channels. That is, the message is not carried by nerves and muscles, and, fur-

thermore, neuromuscular activity is not needed to produce the activity that does carry the message.

Instead of using brain's normal output pathways, users explicitly try to manipulate their brain activity to produce signals that can be used to control computers. The used recording techniques include, besides electroencephalography (EEG) and more invasive electrophysiological methods, magnetoencephalography (MEG), positron emission tomography (PET), functional magnetic resonance imaging (fMRI), and optical imaging. However, MEG, PET, fMRI, and optical imaging are still technically demanding and expensive. Furthermore, PET, fMRI, and optical imaging, which depend on blood flow, have long time constants and therefore, they are less appropriate for on-line communication. Currently, only EEG and related methods can function in most environments, and require relatively simple and inexpensive equipment. [8]

Any BCI has input (e.g. electrophysiological activity from the user), output (i.e. device commands), components that translate input into output, and a protocol that determines the operations. The EEG signal is acquired by electrodes on the scalp and processed to extract specific signal features (e.g. amplitudes of evoked potentials) that reflect the decision of the user. These features are translated into commands that operate a device (e.g. a simple word processing program). The user must develop and maintain good correlation between his or her intent and the signal features employed by the BCI and the BCI must select and extract features that the user can control and must translate those features into device commands correctly and efficiently. [8]

3.1 Different paradigms for BCIs

Present-day BCIs generally fall into 5 groups based on the electrophysiological signals they use: visual evoked potentials, slow cortical potentials, μ rhythms, the P300 ERPs, and steady-state visual evoked potentials [8]:

3.1.1 Visual evoked potentials (VEP)

Visual evoked potentials (VEP) are ERPs with short latency that represent the exogenous response of the brain to a rapid visual stimulus. They are characterized by a negative peak around 100 ms (N100) followed by a positive peak around 200 ms (P200) [10]. The N100 component is significantly modulated by attention [5]. These potentials were used by the system introduced by Vidal in the 1970s [11] that used the VEP recorded from the scalp over visual cortex to determine the direction of eye gaze. Therefore, the VEP-based communication systems depend on the individual ability to control gaze direction. They perform the same function as systems that determine gaze direction from the eyes themselves, and can be categorized as dependent BCI systems. [8]

3.1.2 Slow cortical potentials

Low-frequency voltage changes generated in cortex occur over 0.5 to 10.0 s and are called slow cortical potentials (SCPs). Negative SCPs are typically associated with movement and other functions involving cortical activation, while positive SCPs are usually associated with reduced cortical activation. People can learn to control SCPs and thus, they can control movement of an object on a computer screen. [8]

3.1.3 μ rhythms

These electrical activities are observable inside a frequency range from 8 Hz to 12 Hz (μ). These signals are associated with those cortical areas most directly connected to the motor output of the brain and can be willingly modulated with a movement, a preparation for movement or an imaginary mental movement. Movement is typically accompanied by a decrease in the μ activity. Its opposite, rhythm increase, occurs in the post-movement period and with relaxation. Since the changes are independent of activity in the normal output channels of peripheral nerves and muscles, the increases or decreases of this rhythm have been used several times as a support for a BCI. However, the amplitude of μ oscillation is usually much smaller than that of visual alpha activity. Since both signals share frequency characteristics, motor μ activity estimated at the sensor level is likely to be contaminated

by visual alpha activity due to volume conduction [12]. Therefore, methods for EEG source separation should be considered to recover uncontaminated μ activity. The BCIs based on event-related desynchronization (ERD) of the μ wave are sometimes referred to as ERD/ERS BCIs. [1, 10, 13]

3.1.4 P300 Event-related potentials

As previously mentioned, the P300 is an event-related potential elicited by oddball paradigm. Because of its amplitude and the fact that the P300 is a cognitive reaction to outside events, many brain-computer interfaces are based on the P300 detection [14]. However, the detection of the P300 is challenging because the P300 component is usually hidden in underlying EEG signal. [5]

This BCI paradigm was successfully used for attention-based typewriting introduced by [15]. The Matrix speller consists of a 6x6 symbol matrix. The symbols are arranged in rows and columns. Throughout the course of a trial, the rows and columns are flashed one after the other in a random sequence. Since the neural processing of a stimulus can be modulated by attention, the ERP elicited by target intensifications is different from the ERP elicited by non-target intensifications. Fig. 3.5 shows examples of the P300 spellers based on two different scenarios.

Recently, it has been shown that the P200 component can also contribute to improvement of the accuracy of P300 spellers. Therefore, it has been recently recommended to focus on the ERP signal as a whole rather than considering the P300 component only. [16]

3.1.5 Steady-State Visual Evoked Potentials (SSVEP)

These signals are natural responses to visual stimulation at specific frequencies. When the human eye is excited by a visual stimulus ranging from 3.5 Hz to 75 Hz, the brain generates an electrical activity at the same (or multiples of the) frequency of the visual stimulus. The SSVEP signals are strongly modulated by a selective spatial attention process: these signals are well defined within the extent, determined by the visual attention. Outside this area, flashing visual stimuli do not generate the same activity. They are used for understanding which stimulus the subject is looking at in case of stimuli with different flashing frequency. [10]

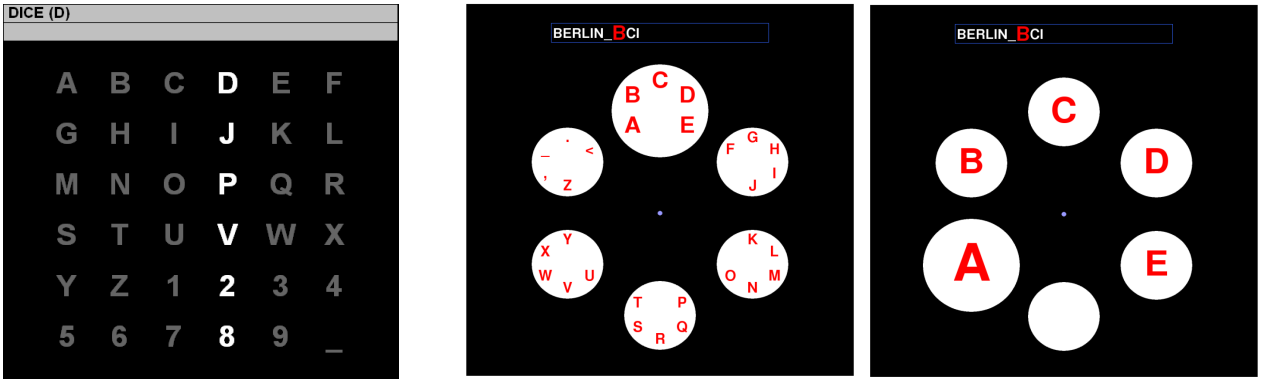


Figure 3.5: Comparison of two P300 speller experiments. The screenshot on the left shows the original P300 speller as it was suggested by [15]. The screenshots on the right show one of many improvements of the original scheme - Hex-o-Spell [16]. Symbols can be selected in the mental typewriter Hex-o-Spell in a two level procedure [16]: 1) At the first level, a disc containing a group of 5 symbols is selected. 2) For the second level, the symbols of the selected group are distributed to all discs (animated transition). The empty disc can be used to return to the group level without selection. Selection of the backspace symbol allows to erase the last written symbol.

3.2 Summary of existing BCI paradigms and new emerging approaches

All BCI systems currently face some challenges that prevent them from being accepted by the majority of the population. Typically, the most limiting factor is their classification accuracy and even more low transfer bit-rate, especially when compared with other means of communication that are available to healthy people. However, different BCI paradigms exhibit different bit-rates and different training times before any particular BCI system can be used successfully. The comparison of BCI paradigms is depicted in Figure 3.6. [17]

In recent years, numbers of BCI research projects and publications in this area have been increasing rapidly. A combination of different BCI types called a hybrid BCI is a new trend in BCI research. The main goal of hybrid BCIs is overcoming the limitations and disadvantages of the conventional BCI systems. When presenting two or more BCI type to the user, in the combination, the user has the chance to get more efficient response through utilizing the BCI type that is more appropriate for him or her. It also can decrease fatigue, as the user can shift to another BCI option. It has been demonstrated that accuracy is improved in the hybrid condition.

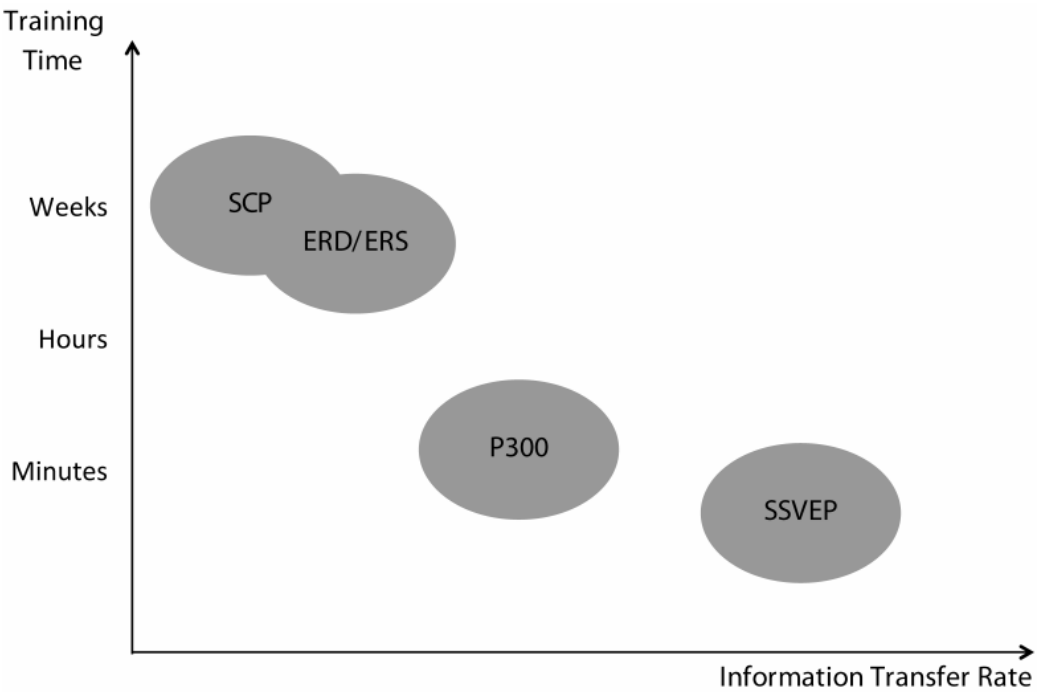


Figure 3.6: A general comparison of SCP, ERD/ERS (μ rhythms), P300, and SSVEP regarding their training time and information transfer rate. Generally, SSVEP and P300 BCIs seem to be the most promising in terms of transfer rate and training times. [17]

For example, P300 and SSVEP BCI were introduced as hybrid in an asynchronous BCI system in [18]. The advantage of this combination is that the stimuli for evoking both patterns can be shown on one screen simultaneously. The P300 paradigm considered in this study is a P300 speller. Only one frequency is allocated for SSVEP paradigm that was associated with the background color. Its flashing evokes the SSVEP response. During the classification, P300 and SSVEP signals are separated by a band pass filter. The SSVEP is utilized as a control state detection. When the user is gazing at the screen, the SSVEP is detected and it is assumed that the user intends to send a command. The system detects P300 target selection and related control state simultaneously. [17]

3.3 BCI illiteracy

Besides other existing challenges, the BCI systems unfortunately do not work for all users. A universal BCI that works for everyone has never been developed. Instead, about 20% of subjects have troubles using a typical BCI system. Some groups have called this phenomenon

"BCI illiteracy". Some possible solutions have been proposed, such as improved signal processing, training, and new tasks or instructions. However, these approaches have not resulted in a BCI that works for all users, probably because a small minority of users cannot produce detectable patterns of brain activity necessary to a particular BCI approach.

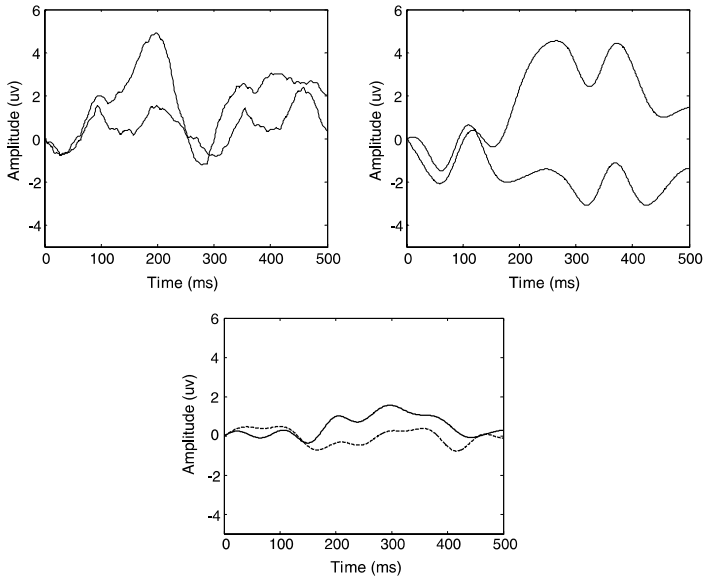


Figure 3.7: The P300 waveforms for different subjects. Only the subject whose response is in the top right figure has a strong P300. [6]

While all people have brains with the same cortical processing systems, in roughly the same locations, there are individual variations in the brain structure. In some users, neuronal systems needed for control might not produce electrical activity detectable on the scalp. This is not because of any problem with the user. Their necessary neural populations may be healthy and active, but the activity they produce is not detectable by EEG. The key group of neurons may be located too deep for EEG electrodes or too close to another, louder group of neurons. For example, about 10 % of subjects do not produce a robust P300. [6]

Consider the examples in Fig. 3.7 which depicts ERP activity from three users of a P300 BCI. Each figure represents an average of many trials. The top left panel shows a subject who did not have a strong P300. The solid and dashed lines look similar in the time window when the P300 is typically prominent, which is about 300 - 500 ms after the flash. However, these two lines differed during an earlier time window. The top right panel shows a subject who did have a strong P300. The bottom panel shows a subject whose ERPs look similar for

target and non-target flashes throughout the time window. This subject cannot use a P300 BCI.

Therefore, before using any particular BCI, the health state and individual needs of the subject should be evaluated to find the best way to restore communication. For example, in our ongoing Czech-Bavarian BASIL project, the procedure depicted in Figure 3.8 is used.

3.4 Design of BCI systems

In order to control a BCI, the user must produce various brain activity patterns that are identified by the system and translated into commands. Typically, this identification relies on a classification algorithm [8]. The reliability of the identification depends on preprocessing, feature extraction and classification of the brain signal.

The purpose of preprocessing is to improve signal-to-noise ratio (SNR) of the brain signal. Feature extraction selects the most relevant features for classifiers. For classification, different approaches are commonly used. Support vector machines (SVM), multi-layer perceptrons (MLP) and linear discriminant analysis (LDA) are among the most frequently used methods [19]. Recently, there has been growing tendency to treat feature extraction and classification as a complex algorithm rather than as separate operations [20].

Since this thesis focuses on P300 BCIs, Fig. 3.9 depicts the structures of P300 BCIs. From general BCIs, they only differ in one step: for further processing it is necessary to extract parts of the EEG signal that are time-locked to stimuli (commonly referred to as ERP trials or epochs).

3.4.1 Evaluation of BCI systems

Probably the most common measure of BCI performance is classification accuracy. It is simply calculated as the number of correct predictions relative to all predictions. However, in many cases, distribution of patterns is not equal when considering their class labels. In those case, accuracy alone is a poor measure of classification performance. For example, if we have 90 % of non-targets in the testing set, any classifier classifying each pattern as non-target would have 90 % accuracy. Therefore, to better evaluate results of classification,

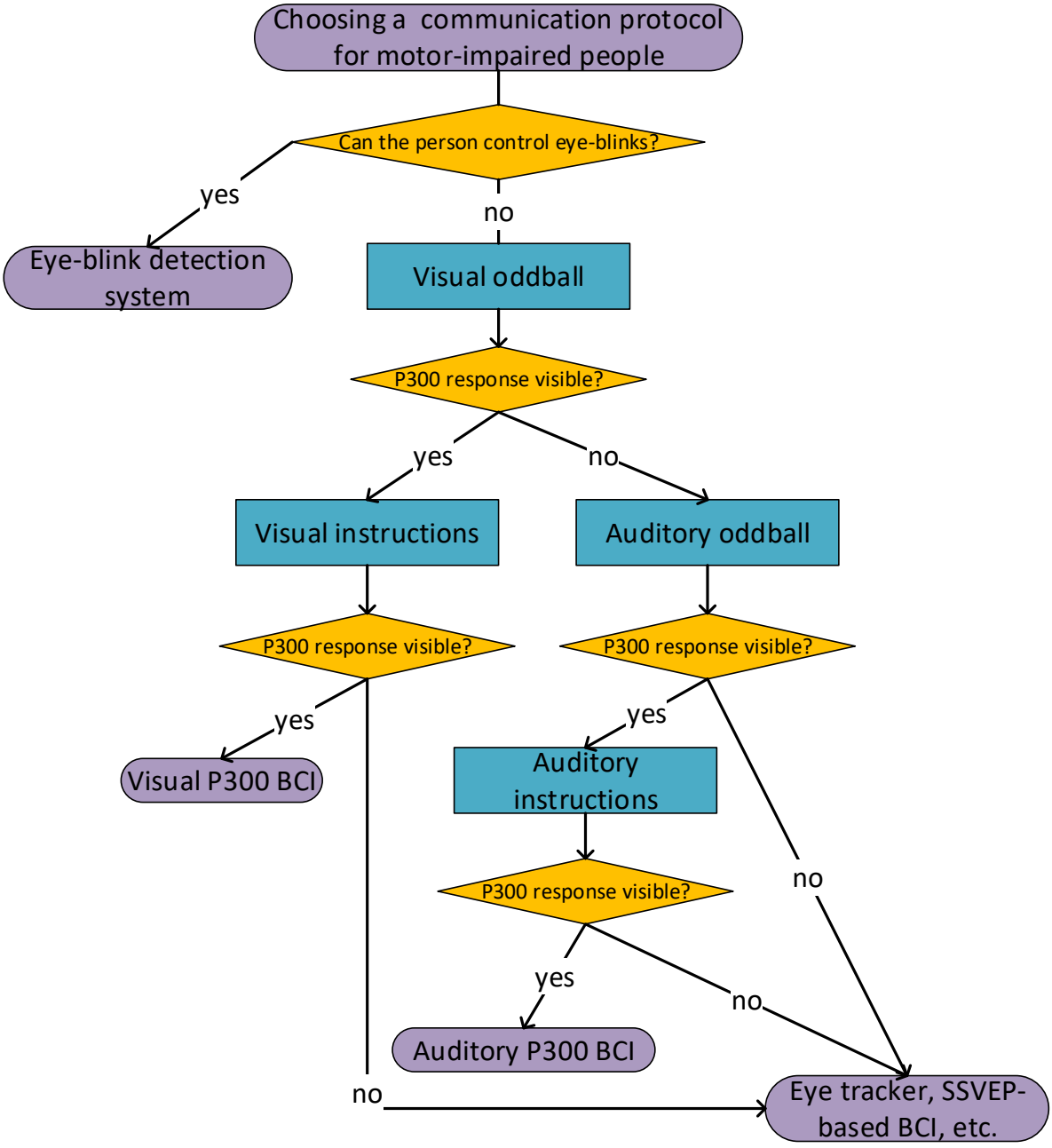


Figure 3.8: One of many possible procedures for finding a suitable communication pathway for people with significant motor impairments. Usually, if the subject can control some muscle movements, such as eye blinks for yes/no questions, this pathway is preferred to BCIs because the accuracy of understanding the subject intention is not limited by the accuracy of the classifier used.

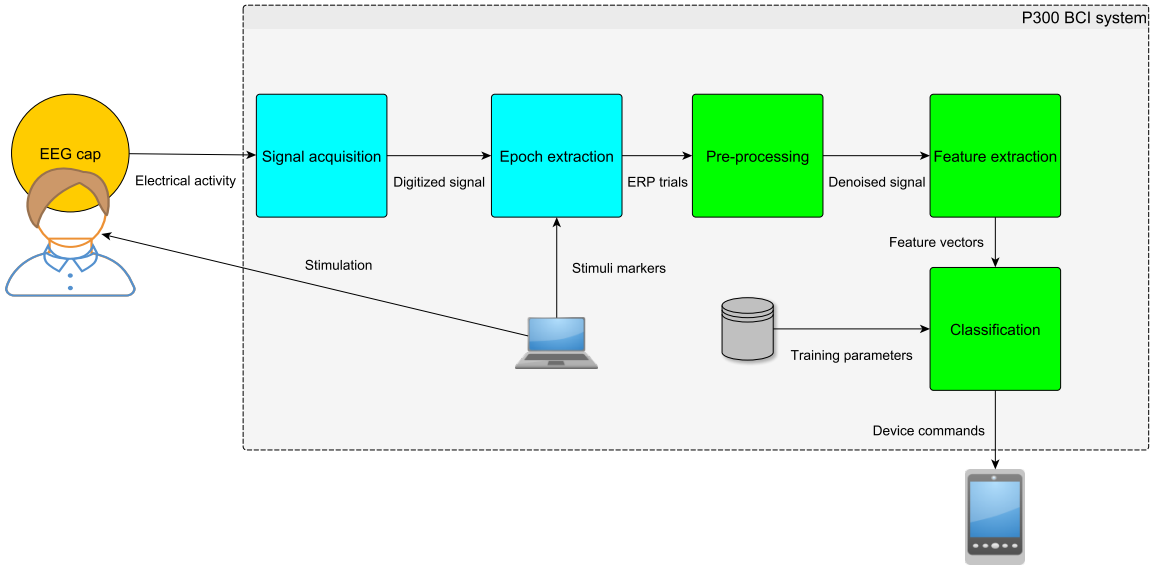


Figure 3.9: Diagram of the P300 BCI system. The EEG signal is captured, amplified and digitized using equidistant time intervals. Then, parts of the signal time-locked to stimuli (i.e. epochs or ERP trials) must be extracted. Preprocessing and feature extraction methods are applied to the resulting ERP trials in order to extract relevant features. Classification uses learned parameters (e.g. distribution of different classes in the training set) to translate the feature vectors into commands for different device types.

precision (positive predictive value) and recall (true positive rate or sensitivity) are also commonly reported. Suppose that we have t_p - number of true positive detections, t_n - number of true negative detections, f_p - number of false positive detections, and f_n - number of false negative detections. For the P300 component detection, t_p represents the number of targets that were correctly classified, t_n corresponds to the number of non-targets that were correctly classified. Similarly, f_p and f_n correspond to numbers of misclassified non-targets and targets, respectively. These measures can be calculated in the following way [21]:

$$ACCURACY = \frac{t_p + t_n}{t_p + t_n + f_p + f_n} \tag{3.1}$$

$$PRECISION = \frac{t_p}{t_p + f_p} \tag{3.2}$$

$$RECALL = \frac{t_p}{t_p + f_n} \tag{3.3}$$

4 Preprocessing and feature extraction techniques for P300 BCIs

4.1 Introduction

ERP components are characterized by their temporal evolution and the corresponding spatial potential distributions. Therefore, as raw material, we have the spatio-temporal matrix $\mathbf{X}^{(k)} \in \mathbb{R}^{\mathbb{M}.\mathbb{T}}$ for each trial k with M being the number of channels and T being the number of sampled time points. It can be helpful for classification to reduce the dimensionality of those features, e.g. by averaging across time in certain intervals, or by removing non-informative channels. The time intervals may result from sub-sampling, or they may be specifically chosen, preferably such that each interval contains one ERP component in which the spatial distribution is approximately constant [16]. The subset of channels can be chosen according to the used BCI paradigm, e.g. for the P300 speller it appears that 8-channel electrode set (Fz, Cz, P3, Pz, P4, PO7, PO8, Oz) is sufficient [20].

In case there is only one time interval, we call the features purely spatial. In this case, the dimensionality of the feature corresponds to the number of channels. Otherwise, when a single channel is selected, the feature is the time course of scalp potentials at a given channel, sampled at time intervals [16].

4.1.1 Feature vector properties

To design an EEG-based BCI system, the following properties must be considered [19]:

- *low SNR*: BCI features are noisy since the EEG signal generally has poor signal-to-noise ratio.
- *high dimensionality*: Usually, several features are generally extracted from several channels over several time segments before being concatenated into a single feature vector.
- *time information*: BCI features are usually based on time domain since brain activity patterns are related to specific time variations of EEG (e.g. the P300 component).

4.2 Temporal features

4.2.1 Introduction

Temporal preprocessing and feature extraction generally treat the EEG signal as an one-dimensional signal. The signal is typically sampled in equal time intervals, e.g. 1 ms. The purely temporal signal may be taken from the most informative channel regarding the P300 classification (e.g. the Pz channel [5]).

The most used methods to improve signal-to-noise ratio of the EEG/ERP signal include averaging, temporal filtering, discrete wavelet transform and matching pursuit. [22, 23]

4.2.2 Averaging

As mentioned above, one of the main challenges of classification in P300 BCIs is low signal-to-noise ratio (SNR). In single trials, the P300 response is hard to distinguish from the background noise. This is due to artifacts in the signal caused for example by blinking which disrupts the signal a lot, or due to the latency of ERP which may slightly change from trial to trial even for the same subject. The problem with SNR may be overcome by averaging together many subsequent trials associated with the same stimulus [23]. This strategy is common in P300 BCI systems. The averaging generally suppresses random noise and makes the ERP with a repeated pattern stand out. The problem with averaging is that more averaged trials mean slower data transfer. For example, the artifacts significantly increase the amount of trials needed to detect ERPs (usually the P300) with reasonable accuracy. Several solutions have been proposed to solve this problem, e.g. artifact removal or ANOVA averaging [24]. Fig. 4.10 shows how averaging gradually increases SNR.

4.2.3 Temporal filtering

Temporal filtering is absolutely necessary for EEG/ERP processing [5]. First, to fulfill the Nyquist Theorem, the rate of digitization must be at least twice as high as the highest frequency in the signal being digitized in order to prevent aliasing. Since the real filters do not have rectangular frequency response, the common practice is to set the digitization rate to be at least three times as high as the cut-off value of the filter [5].

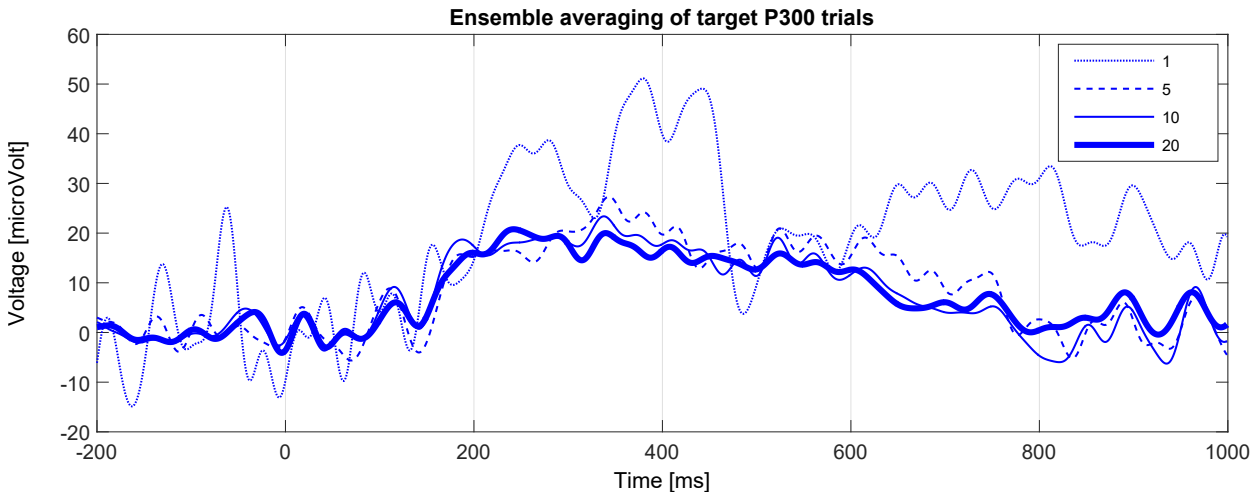


Figure 4.10: This figure shows the effect of averaging P300 target trials together. The dotted line (i.e. an unaveraged single trial) indicates that without averaging, background random oscillation is reflected in an ERP waveform. Random EEG activity is gradually suppressed when local group averages of 5, 10, or 20 trials are calculated as dashed and solid lines depict.

The second main goal of filtering is the reduction of noise, and this is considerably more complicated. The basic idea is that the EEG consists of a signal plus some noise, and some of the noise is sufficiently different in frequency distribution from the signal. That noise can be suppressed simply by eliminating certain frequencies. For example, most of the relevant portion of the ERP waveform consists of frequencies between 0.01 Hz and 30 Hz, and contraction of the muscles leads to an EMG artifact that primarily consists of frequencies above 100 Hz. Therefore, the EMG activity can be eliminated by suppressing frequencies above 100 Hz and this causes a very little change to the ERP waveform. However, as the frequency distribution of the signal and the noise become more similar, it becomes more difficult to suppress the noise without significantly distorting the signal. For example, alpha waves can provide a significant source of noise, but because they are around 10 Hz, it is difficult to filter them without significantly distorting the ERP waveform. [5]

High-pass frequency filters may be used to remove very slow voltage changes of non-neural origin during the data acquisition process. Specifically, factors such as skin potentials caused by sweating and drifts in electrode impedance can lead to slow changes in the baseline voltage of the EEG signal. It is usually a good idea to remove these slow voltage shifts by filtering frequencies lower than approximately 0.01 Hz. This is especially important when

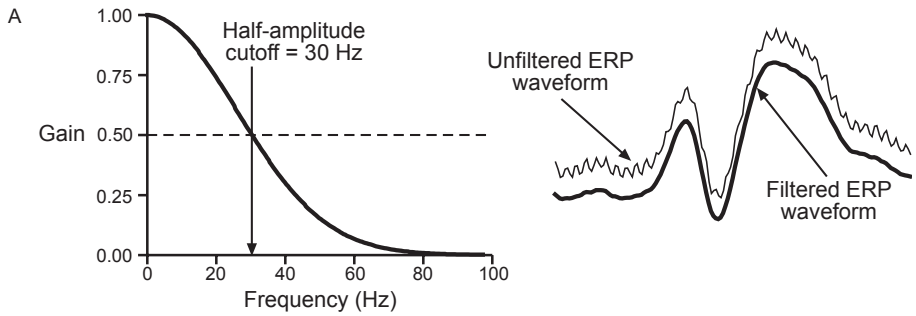


Figure 4.11: This figure illustrates the effects of low-pass filtering. The frequency response of the low-pass filter is on the left, the original P300 response and the low-pass-filtered response on the right. Although the filter deals with high-frequency distortions, it also decreases the amplitude of peaks and shifts their latencies. [5]

obtaining recordings from patients or from children, because head and body movements are one common cause of these shifts in voltage. [5]

Although filters may increase SNR in the temporal domain, they also more or less distort the ERPs. For example, as Fig. 4.11 shows, the low-pass filtering causes changes in the latency of the P300 component. The low pass filters also decrease the amplitude of peaks in the signal.

4.2.4 Discrete Wavelet Transform

Wavelets [25] were proposed by J. Morlet as a method for seismic data processing. The mathematical foundation was written by J. Morlet with A. Grossman. The theory of wavelets is based on signal decomposition using a set of functions that is generated by one or two base functions using dilatation or translation (modifying scale and position parameters).

The most commonly used is the Discrete Wavelet Transform (DWT) which has linear computational complexity. It is based on restricting position and scales. Typically, they are based on powers of 2. [25]

The DWT of a signal is calculated by applying several filters to the signal. Suppose $\boldsymbol{x}$ is an input signal, $\boldsymbol{g}$ is an impulse response of a low pass filter. Then a convolution is calculated:

$$\boldsymbol{y}[n] = (\boldsymbol{x} * \boldsymbol{g})[n] = \sum_{k=-\infty}^{\infty} \boldsymbol{x}[k]\boldsymbol{g}[n-k] \tag{4.4}$$

The input signal $\boldsymbol{x}$ is also passed through a high-pass filter $\boldsymbol{h}$. Both the output signals

of filtering are downsampled by factor of 2 to remove redundancy. The downsampled filter output of the low-pass filter $\boldsymbol{g}$ can be further processed by passing it again through a new low-pass filter $\boldsymbol{g}$ and a high-pass filter $\boldsymbol{h}$ with the half the cut-off frequency. The process may be repeated iteratively based on the desired level of discrete wavelet transform and the input signal dimension. The input signal dimension must be a multiple of 2^n where n is the number of decomposition levels. DWT is illustrated in Fig. 4.12.

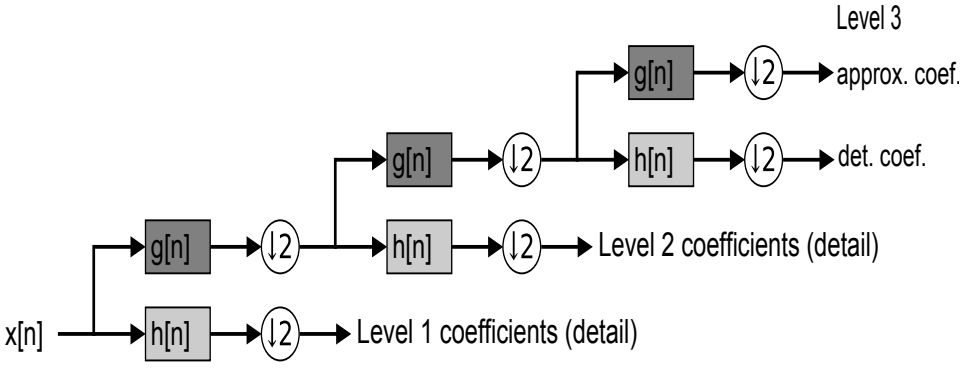


Figure 4.12: 3-Level Discrete Wavelet Transform. At each iteration, the input is passed through a high-pass filter $\boldsymbol{h}$ and a low-pass filter $\boldsymbol{g}$ and subsequently downsampled by factor of 2. The process can be repeated iteratively for downsampled output of low-pass filtering. The downsampled signal output of high-pass filtering is referred to as detail coefficients, the downsampled output of the low-pass filtering after the last iteration is referred to as approximation coefficients.

DWT for P300 BCIs Wavelet Transform is suitable for ERP analysis because of its good resolution in both time and frequency domain [26]. In [27], the activity of the average ERP was analyzed using 5-level DWT. The wavelet coefficients related to the ERPs were identified and the remaining ones were set to zero. The inverse transform was applied to obtain a de-noised average ERP. The algorithm could also be applied to single trials. The authors identified the approximation coefficients of level 5 to be most correlated with the P300 component. However, the application of that processing to the detection of the P300 component was not explored in [27].

Another solution is presented in [28]. It is based on blind source separation of 14 EEG channels using Independent Component Analysis. For feature extraction, 11-level DWT using Daubechies-4 wavelet was applied to the Independent Component 2 that was correlated with

the P300 component. The P300 detection accuracy was 60 % when false positive events were taken in account.

Our research group has also published a few papers regarding the benefits of DWT for the P300 detection (see [29] and [30]). For the detection of the P300 component, cross-correlation was calculated between a wavelet (scaled to correspond to the P300 component) and the ERP signal only in the corresponding part of the signal, where the P300 could be situated. If the maximum correlation coefficient exceeded threshold, the P300 was considered detected. The threshold was empirically set for each wavelet. Haar and Simmlet8 wavelets were tested. The best results were achieved for the Simmlet8 wavelet — the classification accuracy reached 70 % when applied to averages of ten ERP trials.

In one of my previous publications [Author1], multi-level DWT was applied to purely temporal single trials. 6-level or 7-level approximation coefficients were used as the feature vectors. Fig. 4.13 illustrates the pre-processing. High-frequency details almost completely disappeared from the approximation coefficients of level 6 when Daubechies7 wavelet was used for decomposition. This approach is consistent with [27] except for the level of decomposition (6 instead of 5). The level 6, however, still seems to preserve the shape of the P300 component and further halves the number of samples. The accuracy of approximately 75 % was achieved. Furthermore, it could be increased up to more than 90 % when averaging together each six neighboring trials [Author1]. Consequently, when using only approximation coefficients of multi-level discrete wavelet transform, the method seemed suitable for the P300 data processing and feature extraction by performing both de-noising driven by correctly chosen wavelet (suitable wavelets have shapes resembling the P300 component) and dimensionality reduction.

4.2.5 Matching Pursuit

Matching pursuit was introduced by Mallat and Zhang [31] and has many interesting applications including image processing and recognition, or seismic data processing. It is an iterative, greedy process that searches for the best match between a single dictionary element and the input signal. The search for the best match is repeated with the signal residue from the previous step until a stopping rule is satisfied [32]. It is general because it requires min-

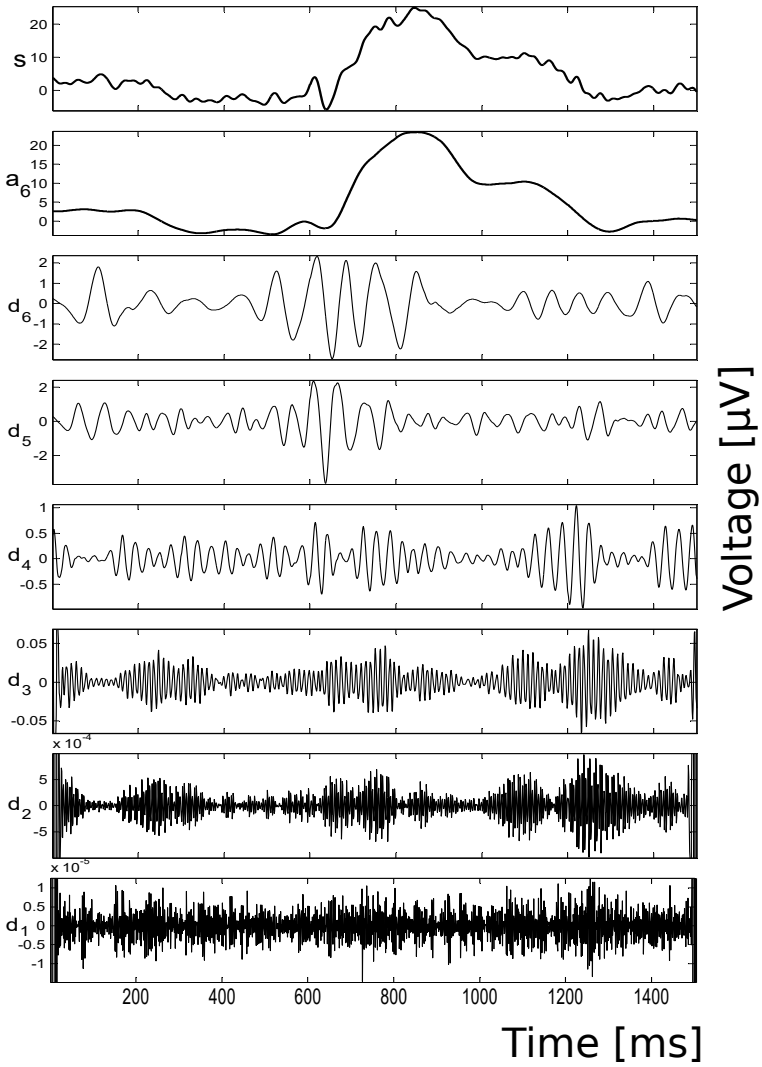


Figure 4.13: Discrete Wavelet Transform of an averaged target epoch (depicted as the first waveform in this figure) using Daubechies7 wavelet [Author1]. The signal reconstructed from the approximation coefficients of level 6 (below the original signal) contains the de-noised signal. The first 500 ms of the signal correspond to pre-stimulus interval, this part is not affected by event-related potentials and therefore, it oscillates near zero on average. The P300 component is located more than 300 ms after the stimulus onset. The reconstructed details denoted as d_1 to d_6 do not visually seem to be significant for the P300 detection.

imal assumptions on the dictionary vectors which must only belong to a Hilbert space [32]. However, the Gabor dictionary build from a Gaussian window is often used. Suppose we have a function g as follows:

$$g(t) = e^{-\pi t^2} \tag{4.5}$$

The Gabor atom has the following definition:

$$g_{\gamma f} = g_{s,u,v,w}(t) = g(\frac{t-u}{s})\cos(vt+w) \tag{4.6}$$

where s means scale, u latency, v frequency and w phase. These four parameters uniquely define each individual atom.

Let us have an input vector $\boldsymbol{x}$ and define $\boldsymbol{g_\gamma}$ as a vector containing function values of $g_{\gamma f}$ that is sampled at time points corresponding to the sampling of $\boldsymbol{x}$. Matching pursuit [31] decomposes any input vector $\boldsymbol{x}$ into a linear expansion of waveforms. The waveforms are selected from a redundant dictionary D of functions (typically consisting of Gabor atoms $\boldsymbol{g_\gamma}$). Let $\boldsymbol{\gamma} = (s,u,v,w)$. In matching pursuit, $\boldsymbol{g_{\gamma_0}}$ is chosen from D in order to best match significant structures of the signal by means of maximizing scalar product with the original signal:

$$|< \boldsymbol{x}, \boldsymbol{g_\gamma} >| \tag{4.7}$$

Then $\boldsymbol{x}$ may be written as:

$$\boldsymbol{x} =< \boldsymbol{x}, \boldsymbol{g_{\gamma_0}} > \boldsymbol{g_{\gamma_0}} + \boldsymbol{Rx} \tag{4.8}$$

and the process is repeated on the residual $\boldsymbol{Rx}$.
After M iterations, the signal is decomposed into a set of Gabor atoms [32]:

$$\boldsymbol{x} = \sum_{k=0}^{M-1} < \boldsymbol{R^k x}, \boldsymbol{g_{\gamma_k}} > \boldsymbol{g_{\gamma_k}} + \boldsymbol{R^M x} \tag{4.9}$$

$\boldsymbol{R^k x}$ is the residual after k iterations of matching pursuit. $\boldsymbol{R^M x}$ is the residual after M iterations. Typically, when $||\boldsymbol{R^M x}||$ is less than a prescribed threshold, matching pursuit stops [32].

Matching Pursuit for P300 BCIs Matching pursuit has not yet been extensively explored regarding the processing the EEG/ERP signal. However, it has been used for continuous EEG processing [33]. Since biological signal processing is one of the most important matching pursuit applications, it seems promising in the case of ERPs as well: the Gabor atoms are very flexible and can resemble any part of the signal including the localized ones

such as the P300 component.

Some of the atoms found may be associated with ERPs, the others may correspond to some artifacts or noise in the signal. The interpretation of the Gabor atom can be based on its parameters. The position is especially important since each ERP component has its typical delay from the stimulus onset. However, significant differences in parameters may occur in different subjects and even the same subject may show different parameters for the same ERP component.

Filtering of the signal using a reconstruction of a few matching pursuit atoms appears to be the most promising. This approach was proposed in [24], described also in [30] and successfully tested on supervised classifiers [22]. In [Author2], a multi-layer perceptron with eight input, five hidden and one output neuron was used to test the method on the data set of 752 epochs from five healthy participants. The accuracy of approximately 77 % for single trials could be increased up to over 90 % when 6 trials were selected for averaging. Recall was significantly higher than precision, so false positives were a bigger problem than error rate.

An example of matching pursuit decomposition is in Fig. 4.14.

4.3 Spatio-temporal features and filtering

4.3.1 Introduction

Although purely temporal features may be sufficient if the most informative channel has high signal-to-noise ratio, in many cases, it is beneficial to combine the results of more channels to improve SNR. This approach relies on assumption that measured signals are mutually independent [34]. While the model assuming independent sources in the brain is considered partially inaccurate, they are also not completely dependent on each other and in many cases, assuming the independence can still lead to impressive results ([16], [35]). The text in the following chapters explains how different brain and non-brain sources contribute to the EEG signal measured at various EEG channels. Moreover, related implications and reasons for selecting multiple EEG channels when extracting EEG features are discussed.

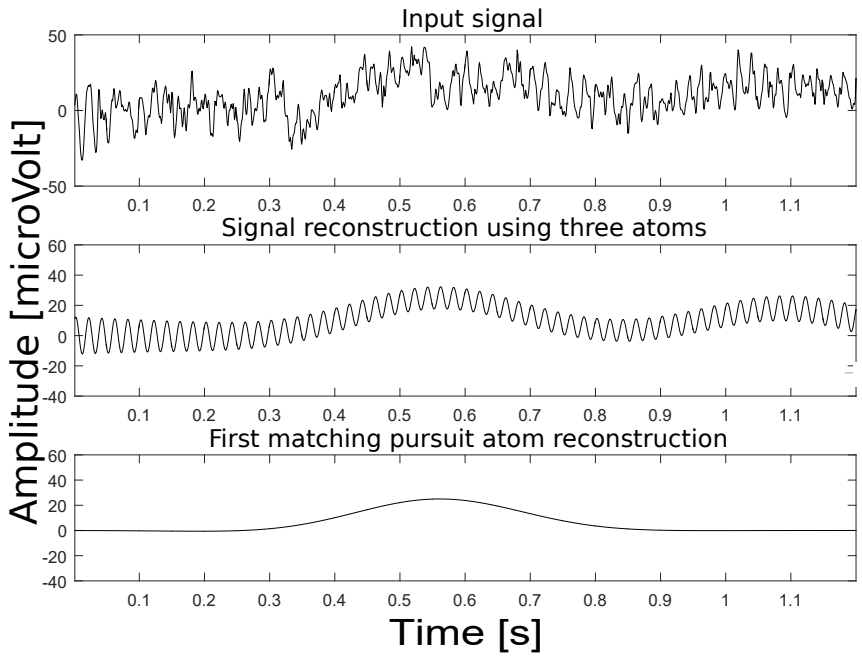


Figure 4.14: Matching pursuit decomposition of an ERP epoch into three Gabor atoms. The first plot depicts an original ERP. The second plot displays a reconstruction of an ERP epoch using three Gabor atoms. The last plot shows reconstruction of the Gabor atom selected in the first iteration of matching pursuit. Apparently, this Gabor atom corresponds to a low-frequency peak with the latency of approximately 550 ms. This atom could represent an ERP component such as the P300 and therefore, it could be e.g. used as an approximation of an ERP component. Another Gabor atom represents a high-frequency distortion (50 Hz) in the signal.

4.3.2 Blind source separation

The basic macroscopic model of EEG generation [36] assumes the tissue to be a resistive medium and only considers effects of volume conduction, while neglecting the marginal capacitive effects. Therefore, each single current source $s(t)$ contributes linearly to the scalp potential measured at one EEG channel $x(t)$, i.e.:

$$x(t) = as(t) \tag{4.10}$$

with a representing the individual propagation of the source $s(t)$ towards $x(t)$ obtained at one particular EEG channel [16]. Since there are multiple sources contributing ($\boldsymbol{s}(t) = (s_1(t), s_2(t), ...)^T$), there is a matrix of propagation vectors $\mathbf{A} = [\boldsymbol{a_1}, \boldsymbol{a_2}, ...]$ and the overall surface potential results in [16]:

$$\boldsymbol{x}(t) = \mathbf{A}\boldsymbol{s}(t) + \boldsymbol{n}(t) \tag{4.11}$$

In Equation 4.11, $\boldsymbol{n}(t)$ is noise (i.e. it will not be a subject of investigation). The propagation matrix $\mathbf{A}$ is often called the forward model, as it relates the source activities to the signal acquired at different sensors.

The reverse process of relating the sensor activities to originating sources is called backward modeling and aims at computing a linear estimate of the source activity from observed signals [16]:

$$\hat{\boldsymbol{s}}(t) = \mathbf{W}^T\boldsymbol{x}(t) \tag{4.12}$$

A single source $\hat{s(t)}$ is therefore obtained as a linear combination of the spatially distributed information from the multiple sensors. A solution can be obtained only approximately, e.g. by means of least mean squares estimator [16]. Therefore, the goal of backward modeling is to improve signal to noise ratio of signals of interest (e.g. ERP components). The rows $\boldsymbol{w}^T$ of the matrix $\mathbf{W}^T$ are commonly referred to as spatial filters [16].

For BCI systems, the exact recovery of the sources in the brain is not required. Instead, we use linear projection that combines information from multiple channels into the one-dimensional signal whose time course can be analyzed with conventional temporal methods. The vector $\boldsymbol{w}$ can be chosen using one of the blind source separation methods to amplify the desired ERP component, e.g. the P300, and thus increase signal-to-noise ratio [37]. One of the benefits of spatial filtering is that some channels may contribute to a reduction of noise in the informative channels. Fig. 4.15 shows that incorporating one noisy channel into feature vector may improve linear separability and thus classification accuracy. [16]

4.3.3 Independent Component Analysis

Independent component analysis (ICA) [34] is a concept that can be applied to any set of random variables to find a linear transform that maximizes the statistical independence of the output components. ICA is defined as an optimization problem to minimize the mutual information between the source components [34]. An efficient algorithm using higher order statistics was presented to measure the notion of non-Gaussianity that corresponds to

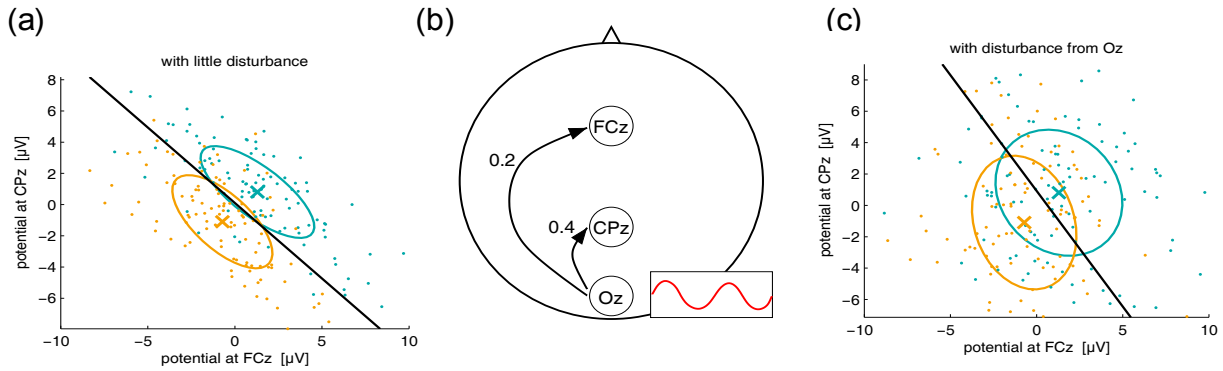


Figure 4.15: The benefits of spatial filtering. (a) Two dimensional Gaussian distributions were generated to simulate scalp potentials at two electrodes with relative high signal-to-noise ratio. (b) A disturbing signal is generated: simulated visual alpha at channel Oz, which is propagated to channels CPz and FCz. (c) This results in a substantially decreased separability in the original two dimensional space. However when classifying 3D data that includes data from sensor Oz, it becomes possible to subtract the noise from the informative channels CPz and FCz and classification becomes possible with the similar accuracy as for undisturbed data in (a). [16]

statistical independence.

To understand how non-Gaussianity relates to statistical independence, it is necessary to understand the central limit theory, which states that the sum of many independent processes tends towards a Gaussian distribution. Therefore, if $\boldsymbol{s}(t)$ is assumed to be a set of truly independent sources, the observed mixed signal $\boldsymbol{x}(t)$ will be more Gaussian by the central limit theory. A single estimated source $\hat{s}_i(t)$ is a linear mixture of $\boldsymbol{x}(t)$ given by the weights in the spatial filter $\boldsymbol{w}_i$. The $\boldsymbol{w}_i$ that maximizes the non-Gaussianity of $\hat{s}_i(t)$ is used to find the closest approximation to the true independent source $s_i(t)$. Therefore, the optimization criteria is to find the unmixing matrix that maximizes non-Gaussianity in all of the source components.

There are many different implementations of ICA that each use different metrics for statistical independence, e.g. FastICA which has good performance and uses kurtosis [23] as a measure of non-Gaussianity.

Furthermore, ICA can be implemented using neural networks, namely Infomax. [38]

ICA for P300 BCIs ICA has been proposed for ERP component analysis by [39]. This approach requires multiple EEG channels as inputs. This approach is explored and yields good results, however, it has high computational complexity. Therefore, it is inappropriate for on-line BCI systems. In [35], this problem has been addressed by applying ICA in training mode only. During the testing phase, a priori knowledge about spatial distribution (i.e. ICA demixing matrix) is used to decompose the signal efficiently. Although this technique has proven to boost classification accuracy of the P300 ERP component significantly (up to 100% classification accuracy with 5 - 8 averaged trials), it has also drawbacks. The ICA is trained on an individual subject and the information obtained cannot be easily applied to different subjects which may have different latencies and amplitudes of the ERP components.

4.3.4 Windowed means paradigm

To provide a simple method for feature extraction that would consider both time and spatial features, the Windowed means paradigm was proposed [16]. It is a simple method that is fast to compute and yields very good results.

The Windowed means paradigm (WMP) is a general method for capturing slow-changing cortical potentials, most importantly in reaction to events (i.e. ERPs). Signal processing usually includes band-pass filtering and occasionally spatial filtering that is simply performed by selecting subsets of EEG channels. Subsequently, windowed averages of pre-processed signal data, per channel, are computed and used as features for the subsequent machine learning stage. Obviously, the dimensionality of the feature vectors is (number of channels) x (number of time windows). The high dimensionality of features can easily lead to over-training. Therefore, either very robust classifiers need to be used, or the number of windows and EEG channels must be carefully optimized. [16]

The paradigm can be applied to event-related slow-changing brain dynamics, including, besides P300s, for example, the perception of self-induced errors, machine-induced errors, or prediction of movement intent. [16]

5 Classification methods for P300 BCIs

5.1 Introduction

The purpose of classification is to divide the feature space into regions that correspond to different classification classes (decisions). For example, if a feature vector $\boldsymbol{x}$, corresponding to an unknown pattern, falls in the region of "target" class, it is classified as "target", otherwise it is classified as "non-target". This does not necessarily mean that the decision is correct. If it is not correct, misclassification has occurred. The patterns (feature vectors) whose true class is known and which are used for the design of the classifier are known as training patterns (training feature vectors). [40]

While performing a pattern recognition task, classifiers may be facing several problems related to the features properties. In the field of BCI, there is a major problem with the curse-of-dimensionality.

The amount of data needed to properly describe the different classes increases exponentially with the dimensionality of the feature vectors. Actually, if the number of training data is small compared to the size of the feature vectors, the classifier will most probably give poor results. It is recommended to use, at least, five to ten times as many training samples per class as the dimensionality. Unfortunately, this requirement is problematic in BCI because typically, training sets are small and dimensionality is high. [19]

5.2 Linear classifiers

Linear classifiers use linear functions to separate classes. Let us focus on the two-class case and consider linear discriminant functions. Suppose we have l-dimensional feature space, a weight vector $\boldsymbol{\omega} = [\omega_1, \omega_2, ..., \omega_l]^T$ and a threshold θ. Then the corresponding decision hypersurface is a hyperplane [40]:

$$g(\boldsymbol{x}) = \boldsymbol{\omega}^T \boldsymbol{x} + \theta = 0 \tag{5.13}$$

For any $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ on the decision hyperplane, Equation 5.14 directly implies that the difference vector $\boldsymbol{x}_1 - \boldsymbol{x}_2$ (i.e. the decision hyperplane) is orthogonal to the vector $\boldsymbol{\omega}$ [40].

$$0 = \boldsymbol{\omega}^T \boldsymbol{x}_1 + \theta = \boldsymbol{\omega}^T \boldsymbol{x}_2 + \theta \Longrightarrow \boldsymbol{\omega}^T (\boldsymbol{x}_1 - \boldsymbol{x}_2) = 0 \tag{5.14}$$

The most popular linear classifiers for BCIs include Linear Discriminant Analysis and Support Vector Machines. [19]

5.2.1 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA, also known as Fisher's LDA) is one of the most popular linear classifiers. The separating hyperplane is obtained by seeking the projection that maximizes the distance between the two classes means and minimize the interclass variance [41]. To solve a N-class problem ($N > 2$), several hyperplanes are used. [19]

This technique has a very low computational complexity which makes it suitable for online BCI systems. Furthermore, this classifier is simple to use and generally provides good results. Consequently, LDA has been successfully used in a great number of BCI systems such as motor imagery based BCI, P300 speller, multiclass or asynchronous BCI. [19]

For known Gaussian distributions with the same covariance matrix for all classes, it can be shown that Linear Discriminant Analysis (LDA) is the optimal classifier in the sense that it minimizes the risk of misclassification for new samples drawn from the same distributions. LDA is equivalent to Least Squares Regression. [16]

LDA for P300 BCIs The optimality criterion generally relies on three assumptions. The following text discusses the criterion regarding the P300 BCIs [16]:

1. Features of each class are Gaussian distributed: According to experience presented in [16], features of ERP data satisfy this condition quite well and the method is quite robust to deviations from the normality assumption. For other type of features it is necessary to find preprocessing to approximately achieve Gaussian distributions.

2. Gaussian distributions of all classes have the same covariance matrix: This assumption implies the linear separability of the data. It is approximately satisfied for many ERP data sets as the ongoing activity is typically independent of the different conditions under investigation. But this is not necessarily so. When comparing ERPs related

to different visual stimuli, the visual alpha rhythm may be modulated by each type of stimuli in a different way. Fortunately, LDA is quite robust in cases of different covariance matrices. On the other hand, modeling a separate covariance matrix for each class leads to a nonlinear separation that is much more sensible to errors in the estimation of the covariance matrices and therefore often yields inferior results to LDA unless a large number of training samples is available. [16]

3. True class distributions are known: This assumption is obviously never fulfilled in any real application. Means and covariance matrices of the distributions have to be estimated from the data. Due to the inevitable errors in those estimates the optimality statement might not hold at all, even when the assumptions (1) and (2) are met quite well. This is typically the case, when the number of training samples is low compared to the dimensionality of the features. [16]

It has been shown that for high-dimensional data, the estimated covariance matrix may be imprecise. Therefore, LDA with shrinkage has been proposed to compensate for the problem. [16]

5.2.2 Support Vector Machines

A Support Vector Machine (SVM) [42] also uses a discriminant hyperplane to separate classes. Such a hyperplane is not unique. A classifier may converge to any of the possible solution. For SVM, the selected hyperplane is the one that maximizes the margins, i.e., the distance from the nearest training points. Maximizing the margins is known to increase the generalization capabilities [40]. In Figure 5.16, the margin for direction "1" is $2z_l$ and the margin for direction "2" is $2z_2$. Our goal is to search for the direction that gives the maximum possible margin. For any linear classifier defined by Equation 5.13, the distance between a point $\boldsymbol{x}$ and a hyperplane $g(\boldsymbol{x})$ can be calculated as [40]:

$$z = \frac{|g(\boldsymbol{x})|}{||\boldsymbol{\omega}||} \tag{5.15}$$

Let us now assume a two-class linearly separable classification task with two classes, c_1 and c_2. We can scale the weight vector $\boldsymbol{\omega}$, threshold θ so that the value of $g(\boldsymbol{x})$, at the nearest

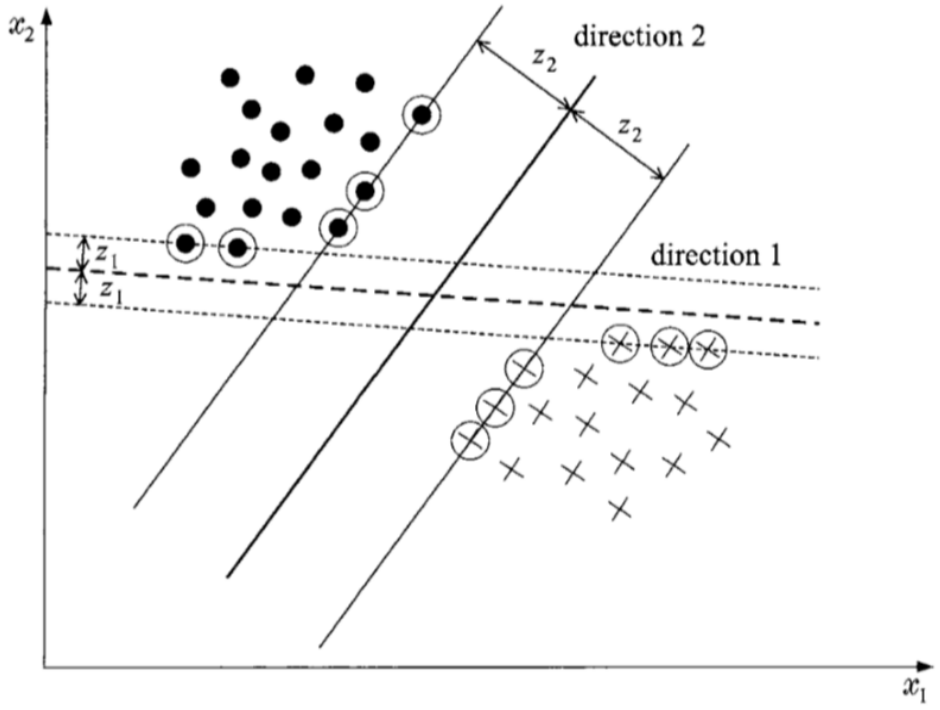


Figure 5.16: The figure depicts a linearly separable classification problem. However, there are multiple solutions for the decision hyperplane. The margin for direction 2 is larger than the margin for direction 1. Therefore, it is the preferable solution for the Support Vector Machine. [40]

points circled in Figure 5.16, is equal to 1 for the class c_1 and, thus, equal to -1 for the class c_2. Assuming these conditions, the following can be stated:

- The margin equals: $\frac{1}{||\boldsymbol{\omega}||} + \frac{1}{||\boldsymbol{\omega}||} = \frac{2}{||\boldsymbol{\omega}||}$
- We require:

$$\boldsymbol{\omega}^T \boldsymbol{x} + \theta >= 1, \forall \boldsymbol{x} \in c_1 \tag{5.16}$$

$$\boldsymbol{\omega}^T \boldsymbol{x} + \theta <= -1, \forall \boldsymbol{x} \in c_2 \tag{5.17}$$

For each input vector $\boldsymbol{x_i}$, we denote the corresponding class indicator by y_i (+1 for c_1, -1 for c_2). Our task can now be summarized as: compute the parameters $\boldsymbol{\omega}$, θ of the hyperplane so that to [40]:

- minimize the norm $J(\boldsymbol{\omega}) = \frac{1}{2}||\boldsymbol{\omega}||^2$
- subject to $y_i(\boldsymbol{\omega}^T \boldsymbol{x_i} + \theta) >= 1, i = 1, 2, ...$

Obviously, minimizing the norm makes the margin maximum. This is a quadratic optimization task subject to a set of linear inequality constraints. [40]

If the data is not linearly separable, the formulation can be modified to become a soft-margin classifier. Misclassifications are now allowed with a given penalty that is regulated by the penalty parameter that must be chosen in advance [23]. SVMs are discussed in more detail in [40].

SVMs for P300 BCIs In [43], the authors report the results of a comparison of different classification algorithms, which show that stepwise linear discriminant analysis (SWLDA) and support vector machines (SVMs) perform better compared to the other classifiers. This conclusion is supported by [19].

5.3 Non-linear classifiers

Non-linear classifiers have non-linear decision boundaries. They may be superior to linear classifiers if the features are not linearly separable.

5.3.1 Multi-layer perceptron

The perceptron [44] is the simplest artificial neural network - it is essentially an artificial neuron; it simulates the functioning of a single biological neuron. The artificial neuron has the following definition:

$$y = \sigma(\sum_{i=1}^{n} \omega_i x_i + \theta) \tag{5.18}$$

where y is the output of the neuron, ω_i are the weights of the neuron, x_i are the inputs of the neuron, θ is the threshold or the bias and σ is the neural activation function. The aim of the activation function is to map neural input to its output. One of the traditionally used neural activation functions is a sigmoid (depicted in Fig. 5.17). Although even one perceptron can be trained and used for classification, its real advantage is revealed when multiple perceptrons are connected into neural networks.

Multi-layer perceptron (MLP) is a widely used neural network. It follows a supervised

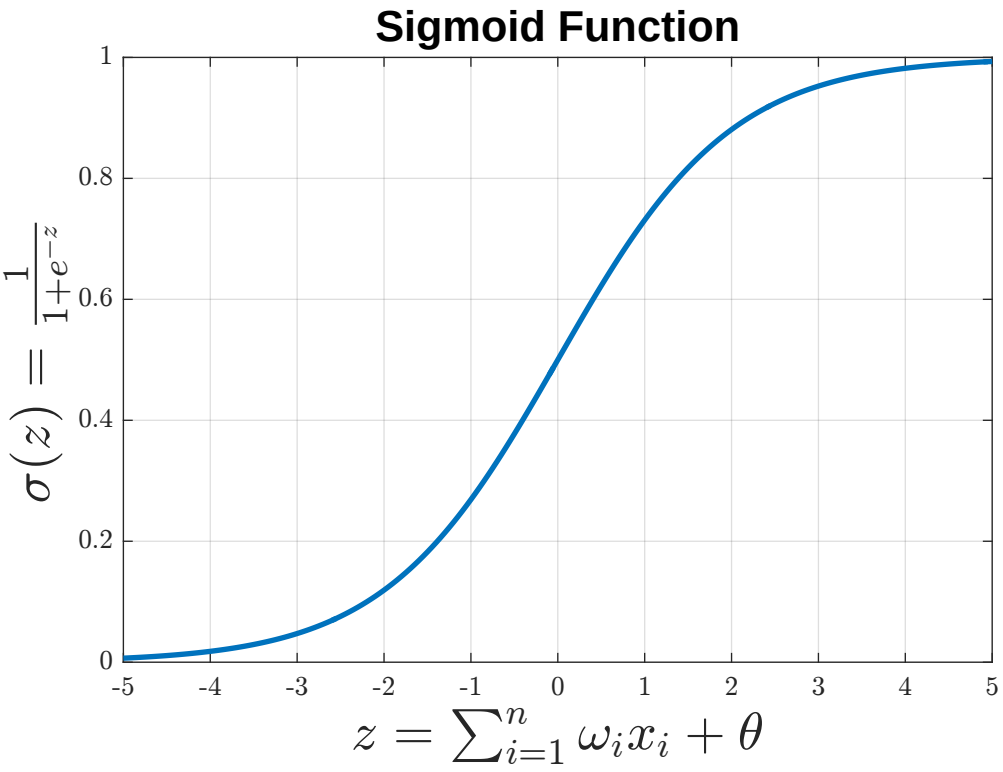


Figure 5.17: Sigmoid activation function defined as $\sigma(z) = \frac{1}{1+e^{-z}}$. It has some useful properties — its output is always positive, converges to zero for large negative inputs and it is differentiable.

learning model. From structural point of view, it is based on perceptrons connected in a form of more layers. Output of each neuron is connected to inputs of all neurons in the next layer. An example of classification using MLP is shown in Fig. 5.18. [44]

Since one perceptron can classify feature vectors using one decision hyperplane, two perceptrons in the same layer represent two hyperplanes. Adding an additional layer enables the neural network to separate a more complex shape. [44]

Softmax The softmax function [45] is often used as the last (output) layer of feed-forward neural networks. It transforms the outputs of each unit to be between 0 and 1, just like a sigmoid function. Furthermore, it divides each output so that the total sum of all outputs is equal to 1. Therefore, the outputs of the softmax function corresponds to a categorical probability distribution, i.e. can be interpreted as the probability that any of the output

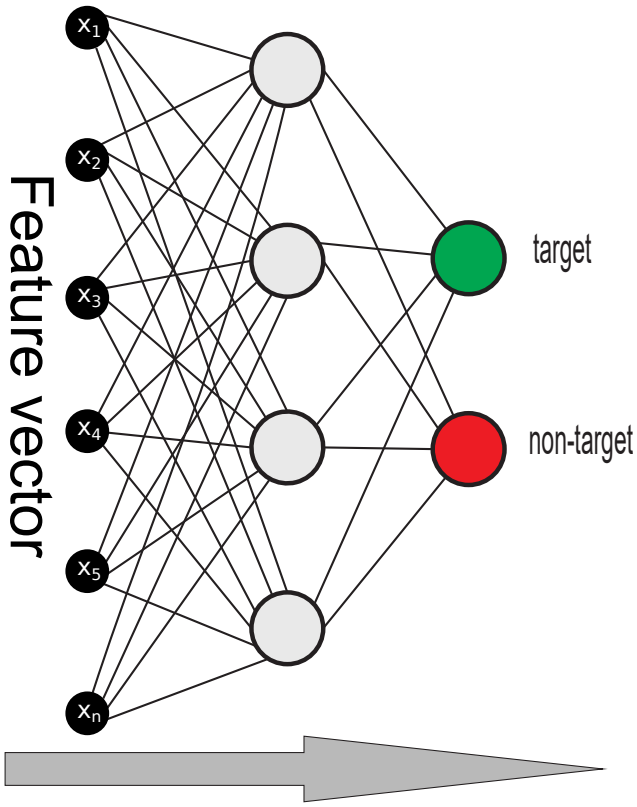


Figure 5.18: The figure depicts how the ERPs can be classified using multi-layer perceptron. Feature vectors $\boldsymbol{x} = (x_1, x_2, ..x_n)$ are accepted with the input layer and propagated throughout the network with one hidden layer depicted as gray circles. The decision about the class can be based on comparing the outputs of two output neurons, the higher output decides the class.

classes are true. Mathematically, when $\boldsymbol{z}$ is a vector of the inputs of the output layer and j indexes output units, $j = 1, 2, ..K$. Then softmax has the following definition:

$$softmax(\boldsymbol{z})_j = \frac{e^{\boldsymbol{z}_j}}{\sum_{k=1}^{K} e^{\boldsymbol{z}_k}} \tag{5.19}$$

Backpropagation In the 80s, the discovery of Backpropagation (BP) algorithm sparked a renewed interest in artificial neural networks. The algorithm is based on error minimization that leads to a gradual update of weights and thresholds. The parameters are updated starting from the last layer of MLP and finishing with the first layer. Since BP is based on derivative calculation of the error (loss) function, it requires the activation function to be differentiable. Besides BP, several improvements to the original algorithm were published. For example, Scaled conjugate gradient method [46] has gained popularity because it is fully-

automated, includes no critical user-dependent parameters and is considerably faster than BP. [44]

Error functions To allow BP or any related algorithm to minimize predictive error of multi-layer perceptron or other related neural networks, it is required that error (also referred to as loss, or cost) function is defined. Mean square error (MSE) and cross-entropy are commonly used error functions [47]. Suppose that n is the number of training inputs, $j = 1, 2, ...$ output neurons, $\boldsymbol{x}$ is the input vector, $\boldsymbol{a}$ is the output vector when $\boldsymbol{x}$ is the input, $\boldsymbol{y}(\boldsymbol{x})$ is the desired output when $\boldsymbol{x}$ is the input. Consequently, y_j is the desired output of the j-th output neuron when $\boldsymbol{x}$ is the input and a_j are the corresponding actual output values. Then, MSE and cross-entropy error functions are defined as follows [47]:

$$MSE = \frac{1}{2n} \sum_{\boldsymbol{x}} ||\boldsymbol{y}(\boldsymbol{x}) - \boldsymbol{a}||^2 \tag{5.20}$$

$$CROSS\_ENTROPY = \frac{1}{n} \sum_{\boldsymbol{x}} \sum_{j} [y_j \ln a_j + (1 - y_j) \ln(1 - a_j)] \tag{5.21}$$

Both MSE and cross-entropy are in line with expected error function behavior. They are non-negative, differentiable, and when actual outputs are close to desired outputs, both functions are close to zero. When used together with sigmoid activation functions σ, cross-entropy has an advantage of having its derivatives depend on $\sigma(z)$ and not $\sigma'(z)$ and thus prevents learning slowdown for large z as explained e.g. in [47].

MLP for P300 BCIs Multi-layer perceptrons can approximate any continuous function. Furthermore, they can also classify any number of classes. This makes MLP very flexible classifiers that can adapt to a great variety of problems. Therefore, MLP, which are among the most popular networks used in classification, have been applied to almost all BCI problems. However, the fact that MLP are universal classifiers makes them sensitive to overtraining, especially with such noisy and non-stationary data as EEG. Therefore, careful architecture selection and regularization is required. [19]

In [22], a voting classifier containing MLP was applied to the P300 classification problem.

When using matching pursuit or wavelet transform for feature extraction, accuracy of more than 70 % was achieved on single trials and over 90 % for averaging.

5.4 Deep learning

Deep learning models have emerged as a new area of machine learning since 2006 [48]. For complex and non-linear problems, deep learning models have proven to beat traditional classification approaches (e.g. SVM) that are affected by the curse of dimensionality [49]. These problems could not be efficiently solved by using multi-layer perceptron with many layers (commonly referred to as deep neural networks) trained using backpropagation. The more layers the neural network contains, the lesser the impact of the backpropagation on the first layers. The gradient descent then tends to get stuck in local minima or plateaus which is why no more than two layer were used in most practical applications. This problem is commonly referred to as vanishing gradient. Deep learning is based on simple and effective strategies that has emerged to deal with this problem. One of them is an unsupervised pre-training and another solution is based on Rectified Linear Unit (ReLu) activations. [48, 49]

An unsupervised pre-training has gained a lot of attention because many times, there is a large amount of unlabeled data and only its small subset is labeled to allow supervised learning. Training can usually be divided into two steps: unsupervised pre-training and supervised fine-tuning. Unsupervised pre-training is typically performed using greedy layer-wise algorithm and its goal is to find a better initial configuration of weights and thresholds. Supervised fine-tuning stage follows to further adjust weights using backpropagation. [49]

ReLu activation functions represent another strategy to solve the vanishing gradient problem. This strategy is based on observation that a sigmoid activation function also has its drawbacks, especially when used for neurons in hidden layers. As seen in Fig. 5.17, for large inputs z, outputs calculated as $\sigma(z) = \frac{1}{1+e^{-z}}$ converge to 1. Therefore, its derivative $\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$ is close to zero for large inputs. This leads to a slow training because gradient weight updates are small despite large inputs. The ReLu function computes the function $\sigma(z) = max(0, z)$. In other words, the neural activation is simply thresholded at zero. For positive activations, the derivative is 1 and the gradient no longer vanishes. [48]

Nonreccurent deep networks models, suitable for unsequenced data, generally fall into the

following categories [49]:

- Stacked autoencoders
- Deep belief networks (stacked restricted Boltzmann machine)
- Deep Kernel Machines
- Deep Convolutional Networks

In this thesis, only stacked autoencoders are described and subsequently evaluated. They provide an efficient method for layer-wise unsupervised pre-training and still do not require any other training algorithm than backpropagation. Other deep learning models were extensively described and discussed in the literature, for example in [48, 49]. Deep Kernel Machines and Deep Convolution Networks are not based on unsupervised pre-training and therefore, they do not fit with the aims of this thesis. However, Deep Convolutional Networks (CNN) have revolutionized image classification and already have been successfully applied to the P300 detection [50]. Thus they provide an interesting field for further exploration. On the other hand, they perform best for very large datasets (tens of thousands trials) using high-dimensional feature vectors. Obtaining that large homogeneous P300 datasets is very difficult. Deep belief networks can also be pre-trained in an unsupervised way. However, it is known in the literature [51] that both their performance and internal representations of the features encoded is comparable to stacked autoencoders.

Stacked autoencoders A single autoencoder (AA) is a two-layer neural network (see Fig. 5.19). The encoding layer encodes the inputs of the network and the decoding layer decodes (reconstructs) the inputs. Consequently, the number of neurons in the decoding layer is equal to the input dimensionality. The goal of an AA is to compute a code $\boldsymbol{h}$ of an input instance $\boldsymbol{x}$ from which $\boldsymbol{x}$ can be recovered with high accuracy. This models a two-stage approximation to the identity function [49]:

$$f_{dec}(f_{enc}(\boldsymbol{x})) = f_{dec}(\boldsymbol{h}) = \hat{\boldsymbol{x}} \approx \boldsymbol{x} \tag{5.22}$$

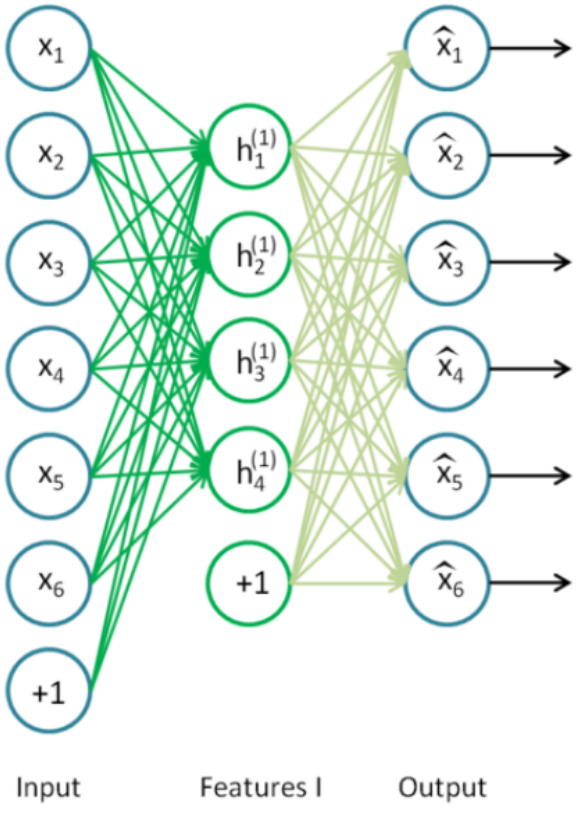


Figure 5.19: Autoencoder. The input layer (x_1, x_2, .. , x_6) has the same dimensionality as the output (decoding layer). The encoding layer ($h_1^{(1)}$, .., $h_4^{(1)}$) has a lower dimensionality and performs the encoding. [52]

with f_{enc} being the function computed by the encoding layer and f_{dec} being the function computed by the decoding layer. As depicted in Fig. 5.19, $h_i^{(1)} = \sigma(\sum_{n=1}^{6} \omega_{ni} x_n + \theta)$ where ω_{ni} is the weight assigned to the connection between x_n and h_i^1, θ is the bias, and σ is the transfer function. Similar equations are used for next autoencoder layers.

The number of neurons in the encoding layer is lower than the input dimensionality. Therefore, in this layer, the network is forced to remove redundancy from the input by reducing dimensionality. The single autoencoder (being a shallow neural network) can easily be trained using standard backpropagation algorithm with random weight initialization. [52]

Stacking of autoencoders in order to boost performance of deep networks was originally proposed by [53].

A key function of stacked autoencoders is unsupervised pre-training, layer by layer, as input is fed through. Once the first layer is pre-trained (neurons $h_1^{(1)}$, $h_2^{(1)}$, .., $h_4^{(1)}$ in Fig. 5.19),

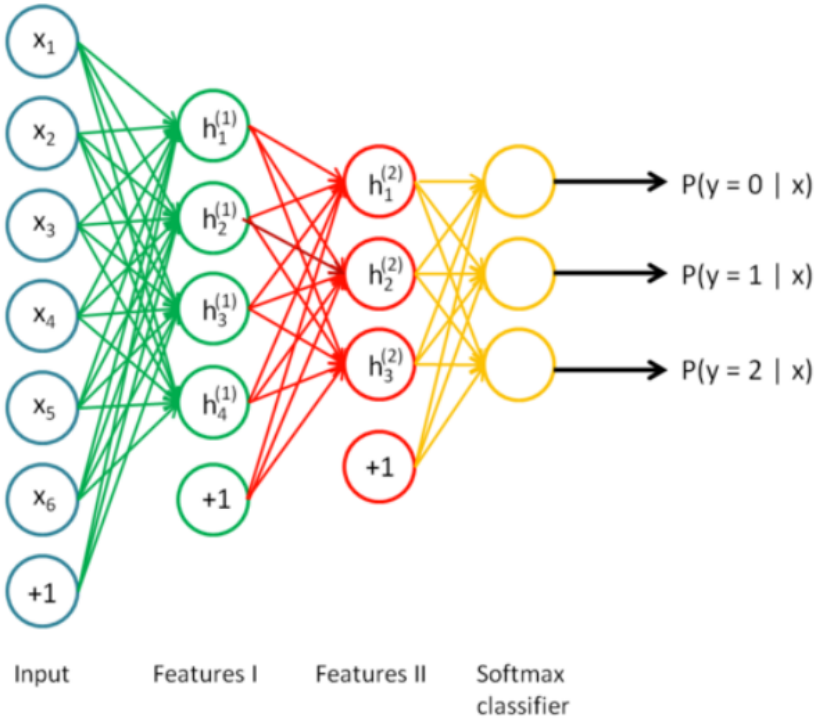


Figure 5.20: Stacked autoencoder [52]. The weights leading to the neurons in the first hidden layer $h_1^{(1)}$ to $h_4^{(1)}$ and from this layer towards the second hidden layer ($h_1^{(2)}$ to $h_3^{(2)}$) are pre-trained in an unsupervised way. The last layer represents a softmax classifier. The whole neural network can be finally fine-tuned using backpropagation.

it can be used as an input of the next autoencoder. The final layer can deal with traditional supervised classification and the pre-trained neural network can be fine-tuned using backpropagation. A stacked autoencoder is depicted in Fig. 5.20. [52]

In [54], a modification of stacked autoencoders was proposed. The authors defined Stacked denoising autoencoders (SdA). The approach is based on corrupting the input with random noise and training the autoencoder to recover the uncorrupted input. In other words, all steps of training the autoencoder are unchanged except the additional noise that is added to the input vectors. Therefore, the neural network does not simply reconstruct the original input but also performs denoising to obtain the original input. The tests revealed that SdA yield better classification results than standard stacked autoencoders.

Regularization and sparse autoencoders To prevent overtraining in stacked autoencoders, various regularization techniques were proposed. Sparse autoencoders represent one

example [52]. They are based on a sparsity constraint for the hidden units. The goal is to make hidden neurons inactive most of the time. With n being the number of training samples, $\boldsymbol{x}^{(i)}$ the i-th input, $h_j^{(k)}(\boldsymbol{x}^{(i)})$ the activation of the hidden unit j in the k-th layer when the network is given an input $\boldsymbol{x}^{(i)}$. Let $\hat{\rho_j}$ be the average activation of a hidden unit j:

$$\hat{\rho_j} = \frac{1}{n}\sum_{i=1}^{n}[h_j^{(k)}(\boldsymbol{x}^{(i)})] \tag{5.23}$$

We would like to enforce the constraint:

$$\hat{\rho_j} = \rho \tag{5.24}$$

ρ is a sparsity parameter, typically a small value close to zero, e.g. $\rho = 0.15$. In other words, we would like the activation of each hidden neuron j to be close to e.g. 0.15. To achieve this, an extra penalty term is added to the error function that penalizes $\hat{\rho_j}$ deviating significantly from ρ. Kullback-Leibler (KL) divergence [52] is probably the most commonly used method (s is the number of hidden neurons) [52]:

$$SPARREG = \sum_{j=1}^{s}\rho\log\frac{\rho}{\hat{\rho_j}} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho_j}} \tag{5.25}$$

Consequently, the error function is modified using the sparsity regularizer (β controls weights of the sparsity penalty term) [52]:

$$E_{sparse} = E + \beta(SPARREG) \tag{5.26}$$

When training a sparse autoencoder, it is possible to make the sparsity regularizer small by increasing the weights and decreasing the activations in hidden layers. Adding a regularization term on the weights to the error function prevents it from happening. This can be called L2 regularization term and is defined for all hidden layers k, training samples i and variables in the input data m as [55]:

$$L2REG = \frac{1}{2}\sum_{k}\sum_{i}\sum_{m}(\omega_{im}^{(k)})^2 \tag{5.27}$$

Stacked autoencoders for P300 BCIs As far as I know, stacked autoencoders have not been commonly applied to the P300 detection problem. However, in [56], the authors applied both deep belief networks and stacked autoencoders to classify the P300 component in the Guilty Knowledge Test. This test is used to determine if certain information is stored in the brain by detecting the P300 component. Mean accuracies for deep belief network, stacked autoencoders, and support-vector machines were 86.9 %, 86.01 % and 83.33 %, respectively. However, the authors optimized the parameters on the testing dataset instead of using a separate validation set for parameter optimization.

5.5 Clustering-based neural networks

Apart from traditional approaches based on supervised learning, clustering-based neural networks may also be used for BCI design. They generally perform cluster analysis based on finding similarities among feature vectors. In most cases, these classifiers do not require any supervision. However, to perform classification, information about class labels has to be taken into account. This can be achieved for example by interpreting the resulting clusters, or by using supervised modifications of originally unsupervised algorithms. Consequently, algorithms that can also be used for P300 BCI classification are discussed. So far, these methods have not been frequently used for P300 BCIs.

5.5.1 Self-organizing maps

Self-Organizing Maps (SOM) are neural networks in the unsupervised-learning category. In its original form the SOM was invented by the founder of the Neural Networks Research Centre, Professor Teuvo Kohonen in 1981-82, and numerous versions, generalizations, accelerated learning schemes, and applications of the SOM have been developed since then.

The SOM converts complex, nonlinear statistical relationship between high-dimensional data items into simple geometric relationship on a low-dimensional display [57]. To allow this, a topological structure among the cluster units is assumed. There are m cluster units (also commonly referred to as Kohonen units, or neurons), arranged in a one- or two-dimensional array and the input signals are n-tuples.

The weight vector for a cluster unit (sometimes referred to as a codebook vector) serves as a model input pattern associated with that cluster. During the self-organization process, the cluster unit whose weight vector matches the input pattern most closely (typically, by means of the square of the minimum Euclidean distance) is chosen as the winner. The winning unit and typically also its neighboring units (in terms of the topology of the cluster units) update their weights. The architecture and corresponding algorithm can be used to cluster a set of p continuous-valued vectors $\boldsymbol{x} = (x_1, x_2, .., x_n)$ into m clusters [44]. The architecture of the SOM network is shown in Fig. 5.21.

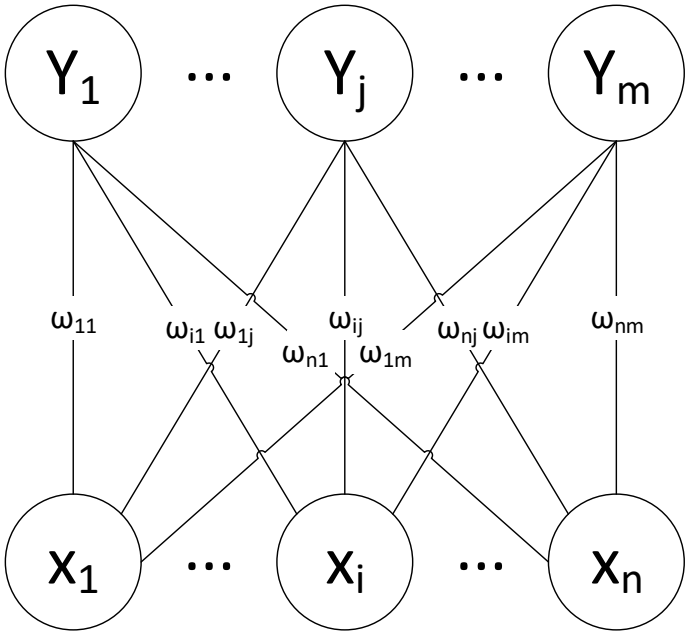


Figure 5.21: SOM network ($\boldsymbol{x}$ - input vector, $\boldsymbol{w}$ - weight vectors, $\boldsymbol{Y}$ - cluster units). [44]

SOMs for ERP decomposition SOMs have not been used for P300 BCIs. However, they have already been proposed for ERP decomposition and the results were promising [58]. In addition, SOMs were used for classification of features in continuous EEG [59].

5.5.2 LASSO model

The LASSO model (Learning ASsociations by Self-Organization) [60] was designed for supervised learning of associated patterns. The main innovation of this model is that output data

are presented to the map for its organization simultaneously with input data. Therefore, the LASSO model can be called input-supervised with self-organization control.

The principle of the LASSO model is illustrated in Fig. 5.22. It consists of three layers: an input layer (n_I dimensional), a Kohonen layer (represented by traditional SOM units), and an output layer (n_O dimensional). Each SOM cluster unit is connected to all the units of the two other layers. The weights of connections from input units to SOM units are called $\boldsymbol{\omega}^{IK}$. Unlike the traditional SOM learning model, connections from SOM units to output units are bidirectional: the weights from the SOM layer to output are called $\boldsymbol{\omega}^{KO}$ while $\boldsymbol{\omega}^{OK}$ is used for the opposite direction. [60]

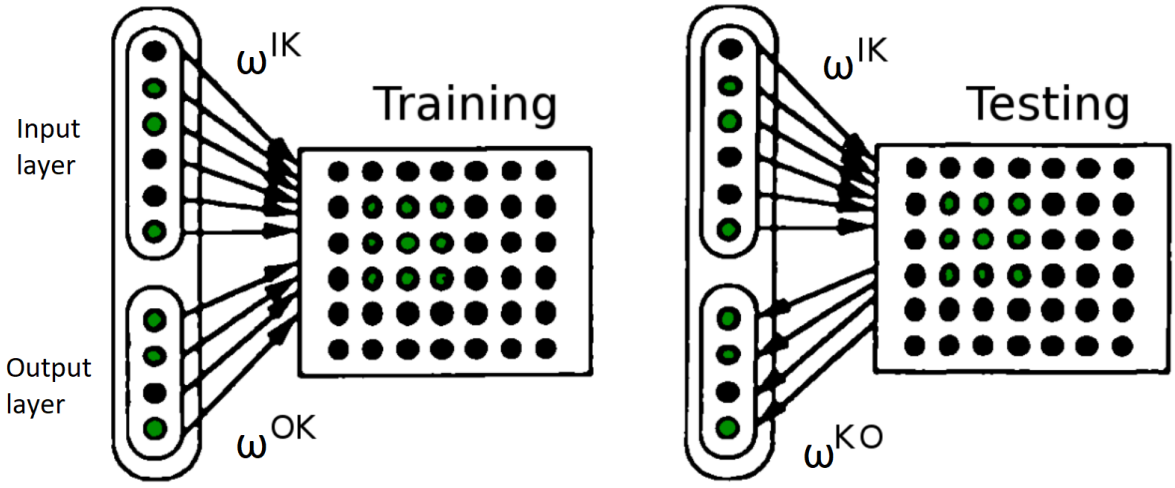


Figure 5.22: The principle of the LASSO model. On the left, the training phase is illustrated. Both original feature vector $\boldsymbol{\omega}^{IK}$ and known class label vector $\boldsymbol{\omega}^{OK}$ are connected into one big feature vector that is used for the SOM training. In the testing phase (on the right), only feature vector is used and the expected class label vector $\boldsymbol{\omega}^{KO}$ is filled from the winning neuron. [60]

In the learning phase, the weights $\boldsymbol{\omega}^{IK}$ and $\boldsymbol{\omega}^{OK}$ are learned simultaneously using self-organization. Weights are gradually adjusted using input and desired output presented together at each learning step. At the end of the learning phase, the weights $\boldsymbol{\omega}^{OK}$ are used for computing associated outputs: the weights $\boldsymbol{\omega}^{KO}$ are set to $\boldsymbol{\omega}^{OK}$. [60]

After the learning phase, input and output layer are distinguished. The SOM map is now used to associate missing data (output pattern) with partial information (input pattern).

An output is associated with the input pattern in the following way [60]:

1. An input pattern $\boldsymbol{x}$ is presented onto the input layer (n_I dimensional)

2. Minimizing the Euclidian distance in the n_I dimensional input space, the n_O other dimensions are ignored during the winning SOM cluster unit J determination. The achieved Euclidian distance in the n_I dimensional input space is called d_I.

3. Selection of the activation group referring to all SOM-units that contribute to setting up the system output. Unit j is selected if $d_D(j, J) < s_D$ and $\frac{d_I^2(\boldsymbol{\omega}_j^{IK}, \boldsymbol{x}) - d_I^2(\boldsymbol{\omega}_J^{IK}, \boldsymbol{x})}{d_I^2(\boldsymbol{\omega}_J^{IK}, \boldsymbol{x})} < p_I$ where d_D is Euclidian distance in the SOM unit space (typically 2-dimensional for the grid layout of SOM units), s_D is the threshold and p_I is the relative distance.

   Activations of non-selected units are set to 0 while activations of selected units are determined in proportion to their representativeness of the input pattern:

$$Y_j = \frac{d_I^2(\boldsymbol{\omega}_J^{IK}, \boldsymbol{x})}{d_I^2(\boldsymbol{\omega}_j^{IK}, \boldsymbol{x})} \tag{5.28}$$

4. The output pattern is calculated as a sum of output weight vectors in the activation group (AG), weighted by the activation level given to the SOM units:

$$\boldsymbol{y} = \frac{1}{\sum_{j \in AG} Y_j} \sum_{j \in AG} Y_j \boldsymbol{\omega}_j^{KO} \tag{5.29}$$

The LASSO for P300 BCIs To my best knowledge, the LASSO model has been used neither for the P300 detection, nor for any other EEG signal processing. However, its application to the P300 data can be quite straightforward. The input layer with any feature vector could be concatenated with the output layer containing the target or non-target class label (e.g. (1, 0) for target and (0, 1) for non-target). The SOM network is then trained in an unsupervised way using those concatenated feature vectors. In the testing phase, for each input pattern, an activation group is found and expected class labels (output patterns) are determined using Equation 5.29.

5.5.3 Adaptive Resonance Theory

The ART (Adaptive Resonance Theory) network developed by Carpenter and Grossberg [61] is also based on clustering. Its output is a direct information about an output class. There are several ARTs (ART 1, ART 2, ARTMAP) differing by architecture and input feature vector type (binary or real valued) they are able to process. The simplified architecture of this network is illustrated in Fig. 5.23.

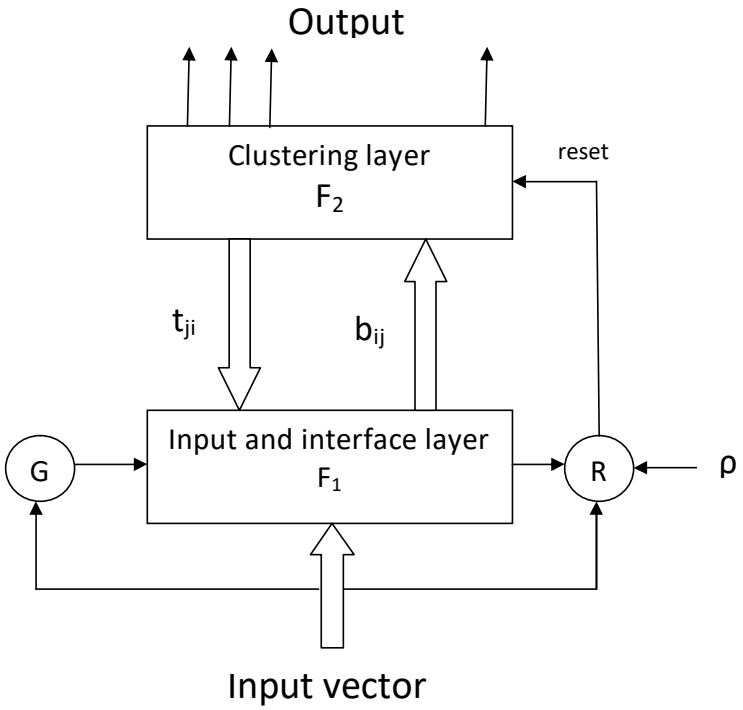


Figure 5.23: The simplified architecture of ART 2 network.

The network consists of two layers of elements labeled $\mathbf{F_1}$ (input units) and $\mathbf{F_2}$ (cluster units), each fully interconnected with the others, and unit **G** and **R** (called *gain control unit* and *reset unit*), which are used to control the processing of the input data vector and the creating of the clusters.

The input and interface layer $\mathbf{F_1}$ has the same number of processing units as it is the size of the feature vector. The clustering layer $\mathbf{F_2}$ consists of as many units as the maximum number of clusters. $\mathbf{F_1}$ and $\mathbf{F_2}$ layers are connected using the set of weight vectors. Weight vectors are modified according to the properties of the input feature vector. For detailed description of ART network and the training algorithm see [44].

ART for ERP decomposition In one of my previous research publications, the ART 2 network was used for clustering of features obtained by processing the ERP data. The features were matching pursuit atoms, i.e. any waveforms that corresponded to significant EEG signal changes (e.g. ERP components such as the P300, or artifacts). Clustering and evaluation of the results was performed to verify if the ART 2 network can separate ERP components from artifacts and other noise. The method was tested for the P300 component and the results were presented in [Author3]. It achieved relatively low accuracy for the P300 component detection (between 50 % and 60 %). In contrast, precision was around 70 %. However, the main application of the ART 2 network seemed to be signal filtering for averaging. For example, when averaging only reconstructed patterns that belonged to the P300 cluster, the resulting average was smoother and better depicted the latency of the P300 waveform when compared with the standard target group average.

Furthermore, Fuzzy ARTMAP (i.e. a modified ART network to allow supervised learning) has been proposed for BCI systems. [19]

5.5.4 Fuzzy ARTMAP

The Fuzzy ARTMAP model [62] is a model that was proposed by the authors of the ART model. It is a generalization of an ARTMAP system [63] that was proposed for supervised learning of binary feature vectors. In contrast, the Fuzzy ARTMAP system is more general as it learns to classify inputs by a fuzzy set of features, or a pattern of fuzzy memberships values between 0 and 1 indicating the extent to which each feature is present. Each fuzzy ARTMAP system includes a pair of fuzzy adaptive resonance theory modules (ART_a and ART_b) that create stable recognition categories in response to sequences of input patterns [62]. The first ART module receives input feature patterns $\boldsymbol{x}$ while the second ART module receives the associated class label codes $\boldsymbol{y}$. Both modules are linked by an associative learning network and an internal controller that ensures autonomous system operation in real time.

Fuzzy ARTMAP for P300 BCIs To my best knowledge, the Fuzzy ARTMAP (FA) model has not been used for the P300 classification. In [64], however, the authors proposed the FA model for mental task classification using power spectral density of the EEG signal. The

average classification rate was above 90 %. The authors report better results in comparison with other feature extraction methods but they did not perform any comparison with state-of-the-art classifiers.

5.5.5 Simplified Fuzzy ARTMAP

The SFAM model (Simplified Fuzzy ArtMap) [65] deals with a significant drawback of the original fuzzy ARTMAP model — its inventors had introduced complicated architectures for their networks instead of presenting them as simple algorithms. Moreover, the original model can generally associate any pairs of features which creates unnecessary complexity for classification tasks where the output patterns are simple class labels. As a solution, redundancies are removed and this model more resembles feed-forward neural networks. [65]

The main idea of the SFAM model is as follows:

1. Find the nearest subclass prototype J that resonates with the input pattern $\boldsymbol{x}$ (winner). To resonate, the following condition has to be fulfilled: $\frac{|\boldsymbol{x} \wedge \boldsymbol{\omega}_J|}{|\boldsymbol{x}|} \geqslant \rho$ (see Fig. 5.24 and its description for context).

2. If the labels of that subclass prototype and the input pattern match, move the prototype closer to the input pattern.

3. If not, reset the winner, temporarily increase the resonance threshold (ρ) and try the next winner.

4. If the winner is uncommitted (unlabeled and with its weights set to 1), assign $\boldsymbol{x}$ to be its prototype and label it using the known input class label.

The algorithm is illustrated in Fig. 5.24.

6 Summary of state-of-the-art methods and additional research possibilities

The previous text mentioned the most common, but by no means all approaches that have been applied to P300 BCIs. Literature reviews of the field [19, 23] did not show any single

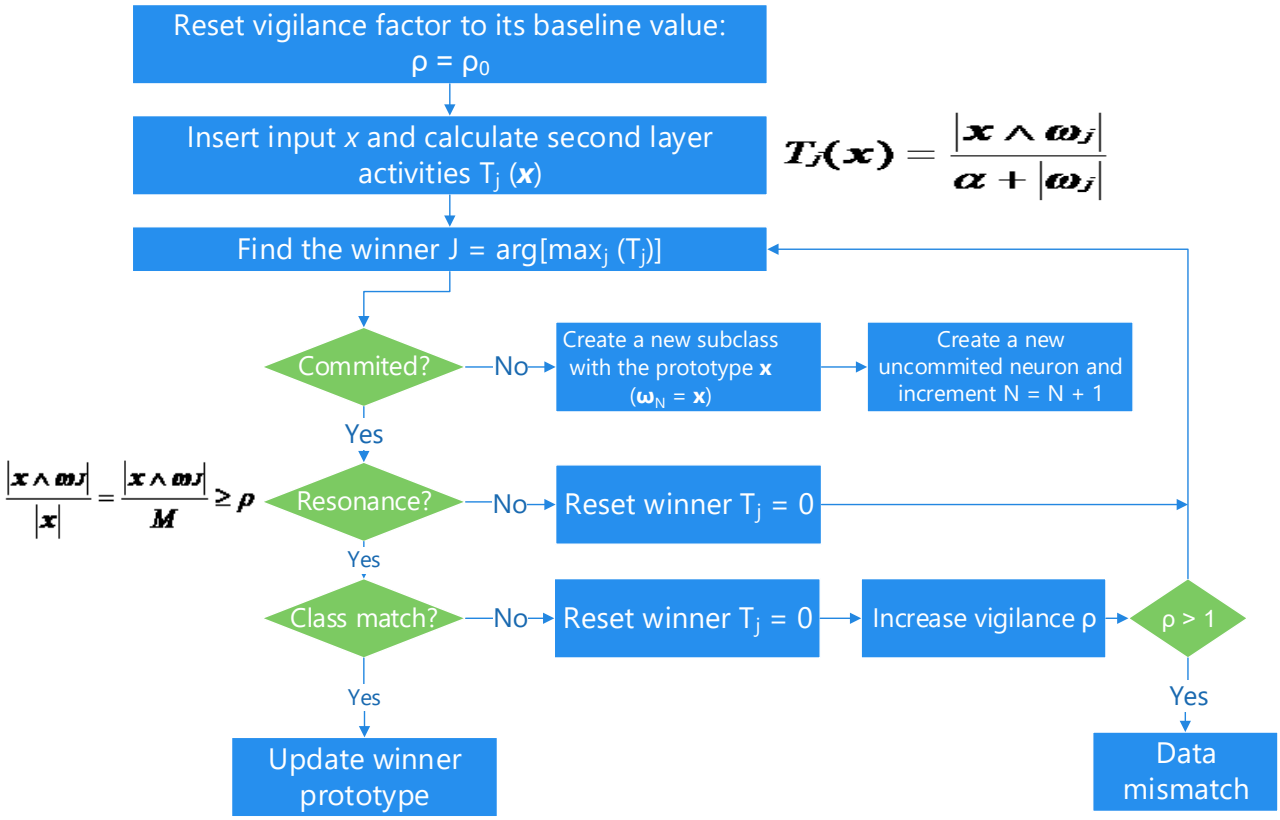


Figure 5.24: Simplified Fuzzy ARTMAP training algorithm for one feature pattern. T_j, calculated as $T_j(\boldsymbol{x}) = \frac{|\boldsymbol{x} \wedge \boldsymbol{\omega}_j|}{\alpha + |\boldsymbol{\omega}_j|}$ is activity of the j-th neuron in the second layer, $\boldsymbol{x}$ is the input vector, $\boldsymbol{\omega}_j$ is the weight vector of the j-th neuron in the second layer, $\wedge$ is the fuzzy AND operator, α represents a choice parameter, the norm $|.|$ is L1 norm [65]. In case of class match, the winner prototype is updated using the β parameter: $\boldsymbol{\omega}_J^{new} = \beta(\boldsymbol{x} \wedge \boldsymbol{\omega}_J^{old}) + (1 - \beta)\boldsymbol{\omega}_J^{old}$. Otherwise, to temporarily increase vigilance, the parameter epsilon is used: $\rho = \frac{|\boldsymbol{x} \wedge \boldsymbol{\omega}_J|}{|\boldsymbol{x}|} + \epsilon$.

P300 detection system to be the state-of-the-art. In [19], linear and non-linear classifiers were discussed regarding their benefits for EEG-based BCIs. Clearly, non-linear classifiers (e.g. multi-layer perceptrons) may be able to separate classes that are not linearly separable. However, linear classifiers may outperform non-linear classifiers if the features are very noisy since linear classifiers are simpler and thus less prone to overtrain. No clear conclusion can be made and there is ongoing discussion which approach is superior [16].

Many papers report different approaches for P300 BCIs, however, it is difficult to directly compare them because they use data recorded from various laboratories and subjects. However, there is a P300 benchmark dataset provided from the BCI Competition 2003 [66] and some papers report their results on this dataset. Several approaches were able to achieve

100% classification accuracy using only 4-8 averaged trials on the BCI Competition 2003 data.

Although the P300 speller has been studied extensively and is one of the well established BCI systems, a recent review of the field [20] concludes that more work still needs to be done to optimize the speed, accuracy, and consistency before the P300 speller is practical to use with disabled patients. This becomes even more relevant when considering that paralyzed patients can display widely varying ERP responses. A reliable BCI system must be able to adapt to the unique ERP responses of each subject and be robust enough to handle the variations between trials within a subject. It is a standard practice to train the BCI system for each new subject, allowing it to learn only the characteristics of that individual's ERP. Therefore, some approaches might have difficulty if they use a priori information to make assumptions about the temporal and spatial characteristics of the standard P300 response, especially when applied to abnormal ERPs from paralyzed patients. [23]

Therefore, a universal BCI system should not only rely on a priori information about expected event-related response, but should also be able to adapt and provide reasonable accuracy for different subjects. This problem is commonly addressed for supervised classifiers which require training for each individual subject. The problem with this approach is that it takes more time when new training data have to be collected with each new user. Furthermore, when traditional supervised methods are used, all attention is concentrated on separating the classes using class labels, and any other information is ignored by the classifier.

6.1 Unsupervised neural network models for P300-based BCIs

Despite the extensive research in this field, unsupervised neural networks (e.g. self-organizing maps or Adaptive Resonance Theory) have so far not been used for P300 BCIs. Instead of using class labels from a supervisor, unsupervised neural networks learn representation of different kinds of data types that occur in the datasets. Furthermore, since no assumptions of the class structure of the data are made, the networks may discover new clusters that have not been apparent before. Therefore, the method may also contribute to understanding of the related feature vectors. Self-organizing maps were successfully applied to the recognition of topographic patterns of EEG spectra in [59]. Six classes in total were used, for continuous

alpha activity, flat EEG, theta activity, eye movements, muscle activity and bad electrodes contact. The authors concluded that SOMs were able to recognize similar topographic patterns in different EEGs, also in EEGs not used for the training of the map. According to [19], Learning Vector Quantization is the closest approach that has been investigated regarding P300 BCIs. In [67], supervised LVQ1 has successfully been applied to P300 data. This further supports the hypothesis that similar models may be useful for P300 BCIs. Moreover, in [68], the authors proposed SOM-based clustering of P300 feature vectors obtained using wavelet transformation. However, the authors did neither implement nor test their proposed system. In [69], another SOM-based P300 detection system was proposed and evaluated. Its goal was to detect start and end times of the P300 component in children that were previously estimated by experts.

Unsupervised ANN, e.g. self-organizing maps can be trained on the data from a simple odd-ball experiment. At least two clusters for target and non-target responses, and possibly an additional "noise" cluster should appear after training. One cluster should be associated with target features, another one with non-target features and in addition, the rest will probably be undecidable. An expert can associate the clusters with classification classes, or features with target classes can be propagated through the network to create the associations. For each subject, these clusters are expected to be distributed differently. The percentage of training features associated with the undecidable cluster may indicate to which extent the subject is suitable for P300 BCIs. The trained neural network could be applied to a more complex BCI paradigm, e.g. the P300 speller. If the classification class of a single trial pattern cannot be decided, the trial can be averaged with the next corresponding trial to gradually increase SNR.

However, the original unsupervised models (e.g. ART and SOM) only perform blind clustering. After clustering, it is not directly possible to assign cluster to the classification class labels. Therefore, in this thesis, supervised modifications of these models are evaluated and some of them implemented and verified.

6.2 Proof of concept — clustering and classification of ERP data using the SOM network

To evaluate the usability of SOMs for P300 event-related data classification, a preliminary experiment was performed and a SOM-based method for classification designed. Details were described in [Author4]. The flowchart of processing steps used is shown in Fig. 6.25. A preselected subset of publicly available P300 datasets [Author5] was used. The selection of the datasets used was based on visual inspection of the quality of the P300 component.

The following procedure was applied for preprocessing and feature extraction:

1. **Epoch extraction**. First, EEG data were split into ERP epochs using stimuli markers (for both target and non-target stimuli). The intervals for ERP epochs extraction were 100 ms before the onset of stimulus and 1000 ms after its onset.

2. **Baseline correction**. From each epoch, the 100 ms pre-stimulus interval was used for baseline correction, i.e. subtracting average of that pre-stimulus interval from the whole epoch.

3. **Shuffling**. The extracted epochs were randomly shuffled.

4. **Channel selection**. Only the Cz channel was used for subsequent processing.

5. **Artifact removal**. The epochs possibly damaged by eye-blinking artifacts were automatically rejected using the amplitude threshold set to 45 μV.

6. **Band-pass filtering and downsampling**. Each epoch was band-pass-filtered with the cut-off frequencies of 0.2 and 10 Hz to improve signal-to-noise ratio, and subsequently downsampled by factor of 4.

7. **Vector normalization**. Finally, the scales of the pre-processed epochs were normalized by dividing each amplitude by the maximum amplitude in that epoch. An example of a preprocessed epoch after this step is depicted in Fig. 6.26 as a blue line.

8. **Feature extraction**. The feature extraction was based on matching pursuit of the preprocessed ERP epoch (from the Cz EEG channel). Five iterations were calculated.

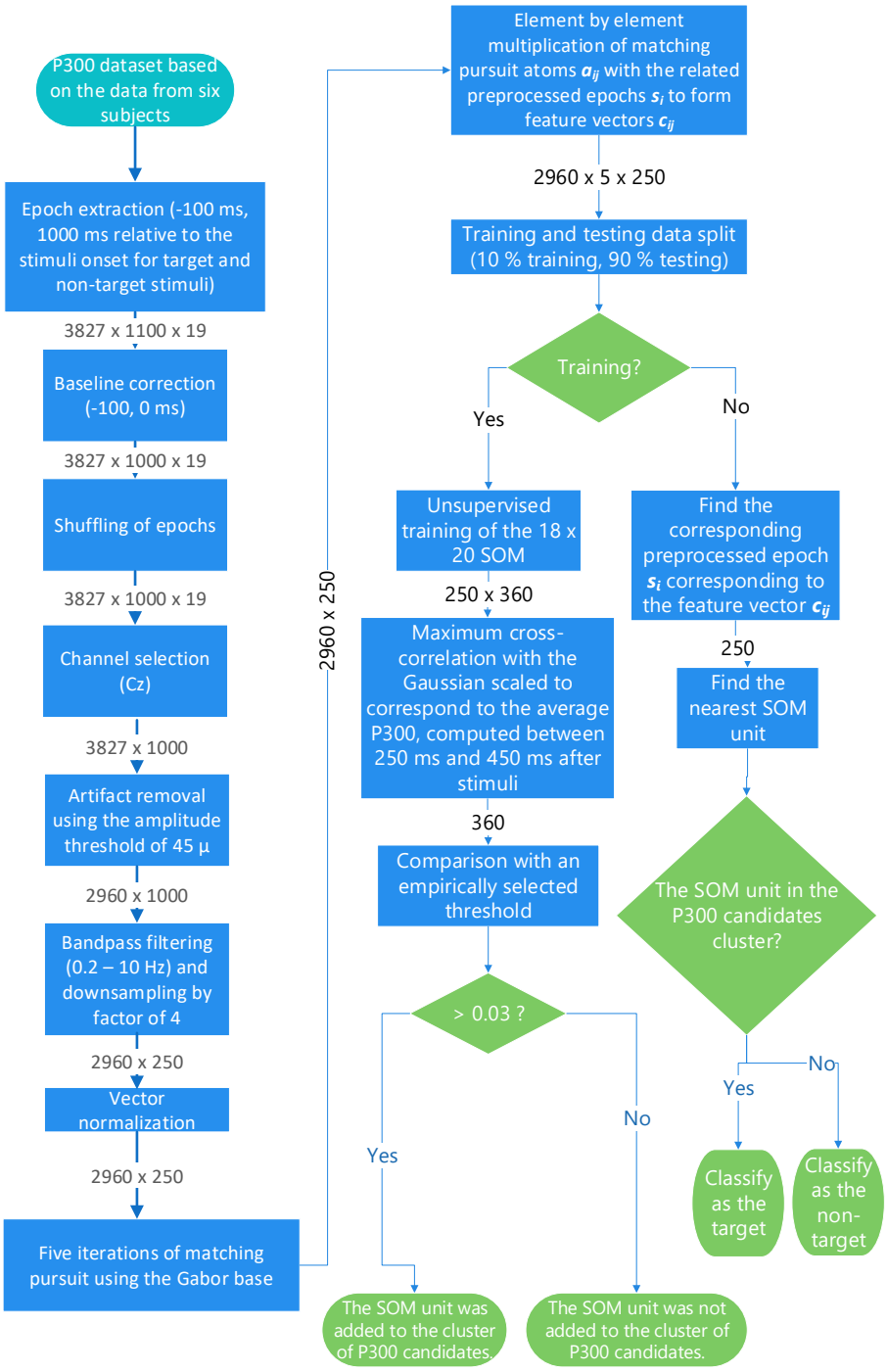


Figure 6.25: The flowchart depicting methods used to verify if the SOM network can be used for clustering and classification of the event-related potential data. The arrows connecting processes include information about the dimensionality of the data being exchanged between those processes.

Although matching pursuit is traditionally based on Gabor atoms, it is also possible to apply the same principle to any other normalized base. In the subsequent experiments,

the Gaussian base was used, i.e. the Gabor base with both v and w parameters set to 0. I decided to use this base because Gaussian functions seem to better resemble the shape of event-related potentials [5]. The Gaussian atoms $\boldsymbol{a}_{ij}$ obtained from matching pursuit were further transformed to represent feature vector that could be interpreted as ERP component candidates $\boldsymbol{c}_{ij}$ using element-by-element multiplication $\odot$ in Equation 6.30.

$$\boldsymbol{c}_{ij} = abs(\boldsymbol{a}_{ij}) \odot \boldsymbol{s}_i \tag{6.30}$$

In Equation 6.30, $\boldsymbol{a}_{ij}$ is the j-th atom decomposed by matching pursuit from the i-th epoch and $\boldsymbol{s}_i$ is the i-th preprocessed input ERP epoch. In the equation, $i = 1..n$ where n is the number of epochs and $j = 1..5$ corresponds to the iterations of matching pursuit. An example of one feature vector extraction using the first iteration of matching pursuit is shown in Fig. 6.26.

9. **Training and testing set split**. Subsequently, 10% of the target and non-target feature vectors were randomly chosen, shuffled and merged into a training dataset. The remaining feature vectors were used for testing. All feature vectors $\boldsymbol{c}_{ij}$ had associated class labels (target or non-target) related to the original ERP epoch labels.

The self-organizing map was trained with the training feature vectors $\boldsymbol{c}_{ij}$ obtained using the procedure described above. The size of the map was automatically adjusted to 18 x 20 using the SOM Toolbox heuristic based on the number of input vectors [70]. The network was trained using the batch algorithm. Both rough training phase and fine-tuning of the network were calculated to optimize clustering. Fig. 6.27 shows the codebook vectors $\boldsymbol{n}_k$ ($k = 1..360$) of the trained SOM network. Note that there appears to be a cluster of similar features correlated and possibly corresponding to the P300 component.

To separate the P300 component candidates from background noise-related feature vectors, cross-correlation between the codebook vectors of the SOM and the P300 component (approximated by manually selected and appropriately scaled Gaussian function) was calculated, but only in the corresponding time intervals where the P300 component is usually located (i.e. 250 - 450 ms after the stimulus [5]). The maximum correlation coefficient was

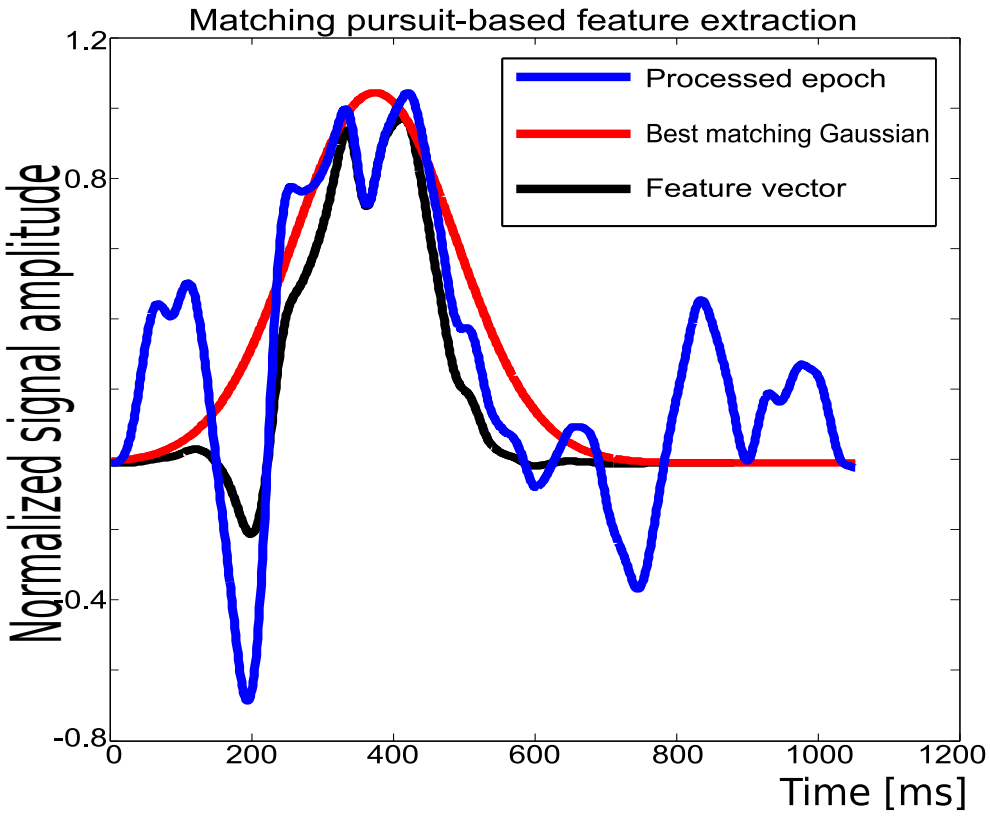


Figure 6.26: The figure depicts how each single feature vector is extracted from the preprocessed epoch (blue). Matching pursuit selects the best matching Gaussian atom. The atom that was selected in the first iteration is depicted in red. To get the feature vector (plotted in black), element-by-element multiplication of the epoch with the selected Gaussian atom was calculated.

compared with a threshold. The threshold represents the decision border between two classification classes (the P300 component detected/not detected). The threshold was empirically set to 0.03. Any SOM cluster unit $\boldsymbol{n}_k$ that was associated with a higher maximum correlation coefficient was tagged as the P300 candidate cluster unit. The histogram in Fig. 6.28 illustrates the threshold decision problem.

Evaluation of the off-line BCI system To evaluate the proposed algorithm in terms of classification accuracy, the following procedure was designed.

First, preprocessed epochs $\boldsymbol{s}_i$ from the testing dataset were extracted using the procedure described above. Subsequently, these vectors ($\boldsymbol{s}_i$) were directly applied to the trained SOM. For each $\boldsymbol{s}_i$, the nearest SOM cluster unit was found. Subsequently, it was verified, if the SOM correctly decided whether that vector belongs to the P300 cluster, or not. The decision

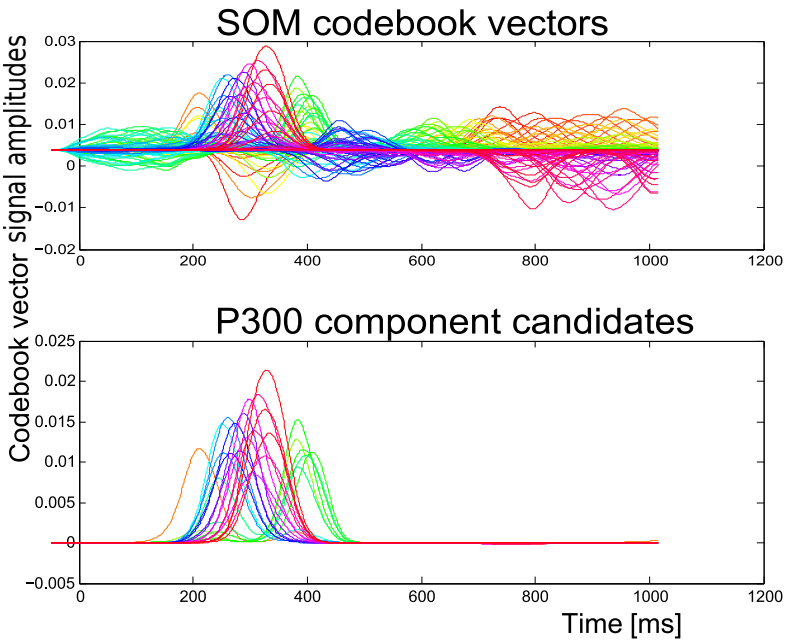
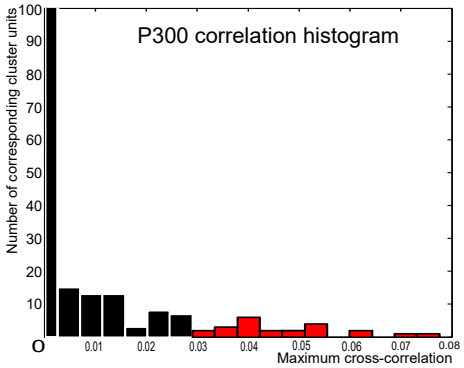


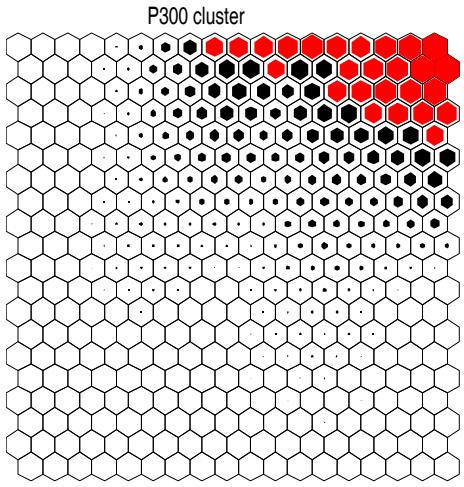
Figure 6.27: All weight vectors from SOM cluster units $\boldsymbol{n}_k$ ($k = 1..360$) are plotted in the upper plot. Note that many of them have latency and polarity that seem to correspond to the P300 component. Unlike the other codebook vectors, these vectors seem to correspond to non-random activity time-locked to stimuli. The bottom plot only depicts the codebook vectors that were tagged as the P300 component candidates.

Table 6.1: Accuracy of classification using the trained SOM network. The LED P300 datasets [Author5] were used. The selection of subjects was based on visual inspection of the P300 component.

Subject ID	Accuracy (%)
86	71.4
93	71.0
94	75.1
98	70.6
99	71.6
100	73.2



(a) Cross-correlation histogram



(b) The P300 cluster in the SOM

Figure 6.28: The histogram illustrates how the normalized Gaussian function scaled to resemble the P300 component correlates with the SOM codebook vectors. The threshold was empirically set to 0.03 based on visual assessment of the related waveforms. Any cluster units that were associated with higher correlations were tagged as the P300 component candidates (in red). The second picture shows the distribution of the P300 candidates on the map. The cluster units in red belong to the P300 component cluster. Sizes of black and red dots inside the SOM units are proportional to the achieved cross-correlation.

was based on comparing the tag of the preprocessed epoch with the tag of the winning cluster unit. Tab. 6.1 contains the results. For all measured subjects, the accuracy was over 70 % for single trials. Fig. 6.29 illustrates how the SOM responded to the testing data. Most epochs were associated with the correct cluster of neurons. Therefore, it appears that SOM can be used for the P300 classification problem. Furthermore, it works as a universal classifier for six subjects with no more training for each individual participant required. However, there are also limitations apparent from experimental design. First, only the datasets containing

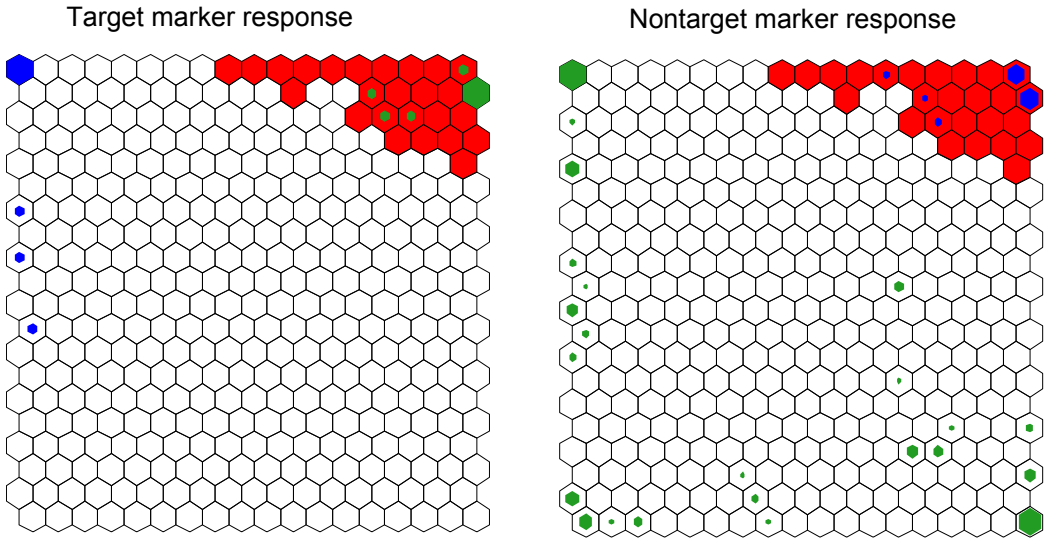


Figure 6.29: This figure illustrates how the trained SOM responded to the testing dataset. To see the difference, only target epochs (i.e. containing the P300), or only non-target epochs (i.e. not containing the P300 components) were applied to the SOM network. On the left, it is clearly observable, that more than half of the features were correctly detected as targets. However, some feature vectors were misclassified. In the non-target area, the features were more equally spread. In absolute values, slightly more feature vectors were classified as containing the P300 than in the target group. However, since there were much more non-targets than targets, in relative values, the percentage of misclassification for non-targets was lower. Note that the first SOM unit located in the top left corner responded to both target and non-target epochs, creating a lot of false negative detections. It can be speculated that this SOM unit responded to epochs damaged by artifacts, regardless of their class labels. If the output matched the expected output, it is depicted in green, otherwise it is depicted in blue. SOM units that were assigned to the cluster of the P300 candidates, are depicted in red. Sizes of blue and green dots are proportional to the number of preprocessed epochs $\boldsymbol{s}_i$ for which the SOM unit was the winner.

apparent P300 component were included for training and testing and therefore, the SOM network may perform worse for participants with low P300 amplitude or high P300 latency. Moreover, matching pursuit has high computational complexity limiting the usability of this algorithm for on-line BCI systems. To conclude, more robust experimental design and faster feature extraction algorithm must be designed and implemented to make SOM-based algorithm practically applicable for on-line BCIs.

6.3 Deep learning for P300 BCIs

Theoretical results [71] seem to indicate that to better classify complicated input data that can represent high-level abstractions (e.g., in vision, language, etc.), deep architectures (i.e. feedforward neural networks with many layers) may be necessary. Furthermore, there is growing evidence that in challenging related tasks (e.g. computer vision, natural language processing, or information retrieval), deep learning methods significantly outperform comparable but shallow neural networks, and often match or outperform the state-of-the-art classification results. This success has been explained e.g. by using unsupervised pre-training that was proven to improve the adjustments of neural network weights and to consistently boost classification accuracy. [72]

Since deep learning models utilizing unsupervised pre-training has worked well for many complex, not linearly separable feature vectors, the P300 classification problem could also be worth exploring. The P300 data are by nature very complex, defined both in time and space (i.e. time course spread throughout various EEG channels). With each layer, deep learning can gradually reduce the complexity of the classification problem to outperform other classifiers, especially neural networks trained using backpropagation with random weight initialization.

6.4 Aims of the Ph.D. Thesis

To summarize, the idea of using unsupervised neural networks for ERPs is described in the following steps:

1. Preprocess the signal and extract the features to maximize signal-to-noise ratio.
2. Select a suitable unsupervised neural network and train it on the extracted features.
3. Based on expert knowledge, or on an automatic procedure, identify target and non-target responses in the trained neural networks.
4. Verify the proposed approach by designing an on-line BCI system and testing the trained network on different subjects.

5. Compare the proposed classification approach with state-of-the-art classification techniques, e.g. Linear Discriminant Analysis, or Support Vector Machines.

6.5 Steps for neural network evaluation using P300 data

I have designed and tested off-line BCI systems to compare different classification models and evaluate their benefits. For this purpose, a whole process that also includes data acquisition, feature extraction and classification needs to described. Finally, an on-line BCI system has been implemented to verify the proposed models outside the laboratory in real time. The following chapters extend my publications, e.g. [Author6], [Author7], [Author5] and [Author8]. However, since the publications use various feature extraction and classification methods, in this thesis, I perform comparison of all related classification algorithms based on one feature extraction method. Both the data used for experiments and Matlab scripts for data loading, feature extraction, classification and plotting results are publicly available ([Author5] and [73]) so the reproducibility of the algorithms is ensured.

7 Experimental design, data acquisition and analysis

As mentioned in Section 3.4, it requires feature extraction and classification before any BCI system can be developed, or any study aiming to propose novel pattern recognition algorithms designed. However, as the very first step, the data for both training and testing phases must be obtained. This requires well prepared conditions for data recording since there is no substitution for clean data [5]. This thesis focuses on the P300-based BCIs. Consequently, an odd-ball paradigm was used to obtain the data for subsequent off-line analysis. The process of data acquisition and the datasets obtained were described in detail in [Author5]. The datasets further described in this section were used for the BCI experiments described in Section 8.

7.1 Laboratory

The EEG/ERP laboratory at the Department of Computer Science and Engineering of the University of West Bohemia was used to perform experiments subsequently described in this thesis. It is equipped with a soundproof cabin, an automobile simulator, a BrainVision amplifier, different kinds of stimulation devices, a computer with the BrainVision Recorder software tool for storing the data and standard EEG caps placed according to a 10-20 system with 19 electrodes. The infrastructure was described in detail in [Author9].

7.2 Recording Software

The BrainVision Recorder 1.2 was used for recording and storing the EEG/ERP data in the BrainVision format [74]. The Recorder was initialized using the following parameters:

- the sampling rate was set to 1 kHz,
- the number of channels was set to 19,
- the resolution was set to 0.1 μV,
- the recording low-pass filter was set with the cut-off frequency of 250 Hz.

The impedance threshold was set to 10 $k\Omega$, and the real impedances for each experiment are stored in text files.

7.3 Stimulation Device

For the data acquisition, the odd-ball stimulation device [75] was used. It was also designed at the Department of Computer Science and Engineering. The main part of the stimulation device is a box containing three high-power Light-Emitting Diodes (LEDs) differing in their color: red, green and yellow.

The core of the stimulator is an 8bit micro-controller that generates the required stimuli. The control panel consists of a LCD display and a set of push-buttons that are used to set the parameters of the stimulation protocol. The stimulator also generates additional synchronization signals for the EEG recorder.

The stimulator has typically been used for modified odd-ball paradigm experiments (three stimulus paradigm [75]). Apart from traditional target and non-target stimuli, the device can also randomly insert distractor stimuli. The distractor stimuli are usually used to elicit the subcomponent of the P300 waveform (called P3a) [76].

7.4 Stimulation protocol

The stimulator described above was used in the stimulation protocol. In our experiments, the stimulator settings were used as follows: each diode flashed once a second and each flash took 500 ms. The probabilities of the red, green and yellow diodes flashing were 83%, 13.5% and 3.5%, respectively. Between two occurrences of target stimulus (green diodes flashing), at least one non-target stimulus appeared. Otherwise, the order of stimuli was completely random.

The participants were sitting 1 m from the stimulator for 20 minutes. The experimental protocol was divided into three phases, each containing 30 target stimuli and each running for five minutes long. There was a short break between the phases. The participants were asked to sit comfortably, not move and to limit their eye blinking. They were instructed to pay attention to the stimulation.

7.5 Participants

A group of 25 healthy volunteers participated in the experiments. However, only the data from 19 subjects were used in subsequent experiments. Five subjects were rejected even before storing the data, so they are unavailable in our data publication [Author5]. Those subjects were blinking excessively, inattentive and in some cases, the experiment was ended earlier. High impedance was also one of the reason for rejection, because it was typically associated with a complete data loss on one or more electrode. One subject was rejected based on the lack of the P300 response. All subjects signed an informed consent.

7.6 Usage notes

The anonymized datasets supporting the results of the subsequent experiments are available in the EEG/ERP Portal under the following URL: `http://eegdatabase.kiv.zcu.cz/` [77]. The experimental IDs that have been used for subsequent experiments are 85, 86, 87, and 91-106.

Since the data were stored in the BrainVision (BV) format [74], related software tools have to be used to read and further process the data. In our experience, EEGLAB, an open-source Matlab toolbox for EEG signal processing [78], is one of the preferred options. It is necessary to download the BVA-io plugin (available at `http://sccn.ucsd.edu/wiki/EEGLAB`) in order to easily import the data stored in the BV format into EEGLAB. Another option is to use the EEGLoader library (available at `https://github.com/stebjan/eegloader`) that provides a simple interface for reading the BV format.

7.7 Training and testing data split

As a raw material, we have the data from 19 subjects. I decided to split the data into training and testing sets as an early step of processing. This allows me to analyze the data, find a feature extraction method with a favorable signal-to-noise ratio without making any assumption about the testing dataset. To select the dataset for training, reasonable quality and well pronounced and diverse P300 component is the aim. Based on such criteria, the training set was concatenated using the data from eight subjects (experimental IDs 95, 96, 99, 100, 102, 104, 105 and 106, none of them included in the testing dataset). The remaining datasets were used for testing. Although it is very common to train and test classifiers using the data from same subjects, this split can help to verify if it is possible to build a universal P300 classifier for the P300 detection, without having to retrain it with each new user.

7.8 Feature extraction discussion

Many papers have been published that describe different algorithms for feature extraction. However, articles comparing such algorithms are still lacking. Therefore, feature extraction should be chosen based on a specific experimental paradigm in order to maximize signal

to noise ratio. I have successfully used matching pursuit and discrete wavelet transform to extract features from the selected EEG channel (typically the Pz channel). However, there is growing evidence that including more channels into feature vectors can provide important information for classification, although it cannot always be explained which particular phenomenon contributed to classification accuracy [16]. For example, the P300 component can be confirmed by being located at central and parietal channels (e.g. the Pz channel) while simultaneously not being prominent at mostly independent and uncorrelated channels (such as frontal electrodes or occitipal electrodes). Moreover, it is possible that, for example, the tested participant blinks or performs small muscle movements following one stimuli more than after another stimuli, and those non-brain signals also show up in EEG and may contribute to increasing classification performance. Although Independent Component Analysis would be needed to approximately separate brain or non-brain components, it is very time-consuming and therefore, unsuitable for on-line BCIs. Fortunately, it is probably not necessary to perform it in BCI applications, since as explained in [16], when given enough data by selecting multiple channels, linear classifiers perform spatial filtering during training. Therefore, I decided to use the Windowed means paradigm method that is based on selecting certain time windows and EEG channels of interest. In summary, reasons for this choice are as follows:

- Its relative simplicity guarantees fast computational speed which is crucial for on-line BCIs.
- Moreover, when visualizing feature vectors, the experimenter can relate average feature vector amplitudes to specific time intervals and EEG channels.
- The method is flexible because any time interval and EEG channel of interests can be selected.
- Therefore, this method allows us to adjust feature vector dimensionality based on the amount of data we have available for training and based on the specific classification algorithm.

7.9 Data-driven parameter selection for the Windowed means paradigm

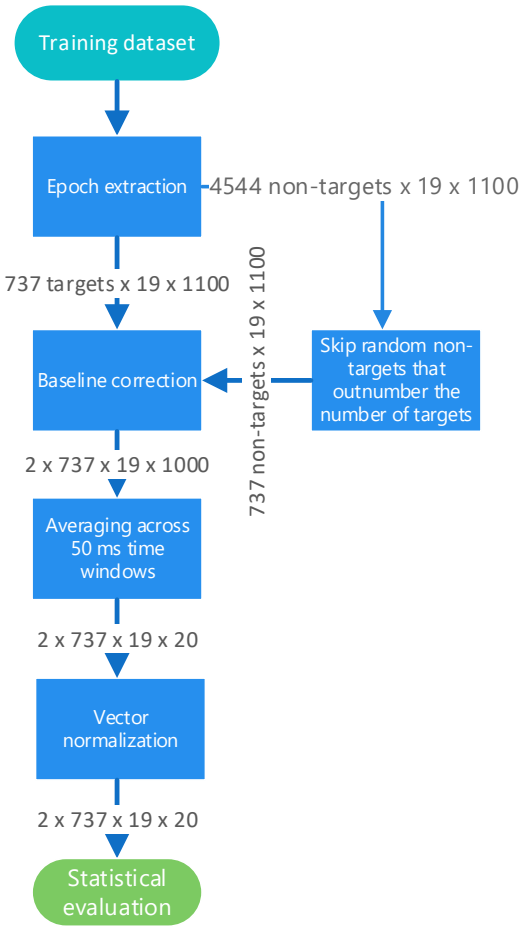


Figure 7.30: Flowchart depicting preprocessing steps used to extract both target and non-target features that can be statistically investigated. Those with significant differences between target and non-target can then be chosen for feature extraction. The arrows connecting processes include information about the dimensionality of the data.

The Windowed means paradigm is a general method for feature extraction based on averaging selected time windows in specific EEG channels. The question is how to select time intervals and channels for the P300 data described above. I decided to base that decision on statistical analysis of the training dataset. The workflow depicted in Fig. 7.30 was used to extract feature candidates. Epoch extraction with the intervals 100 ms before stimuli and 1000 ms after stimuli was performed for both target (the green diodes flashing (S 2)) and non-target (the red diodes flashing (S 4)) markers. However, since there are more non-targets than targets, non-target epochs were randomly skipped so that the numbers of target

and non-target epochs were equal. Baseline correction between -100 ms and 0 ms relative to stimuli onsets followed. Subsequently, the epoch was split into 20 time windows between 0 ms and 1000 ms. For 19 EEG channels, there were $19 * 20 = 380$ potential features. Two-sample t-test was used to investigate statistical significance of differences between averaged target and non-target features. The significance level was set low enough to satisfy Bonferroni correction for multiple comparisons, $\alpha = 0.05/380 \approx 0.0001316$.

Figure 7.31: This plot shows a comparison between target and non-target features for the Pz EEG channel. X-axis is scaled to correspond to time in ms. Statistically significant differences ($p < 0.0001316$) are highlighted using the pink area. Note that most differences occur around 400 ms where the visual P300 component is usually located.

For illustration, results for the Pz channel are depicted in Figure 7.31. Results for all EEG channels with highlighted significant differences are shown in Figure 7.32. Using the significance levels set to 0.0001316, the resulting feature vector dimensionality was 83.

			Time intervals relative to the stimuli onset [ms]																		
			0 - 50	50 - 100	100 - 150	150 - 200	200 - 250	250 - 300	300 - 350	350 - 400	400 - 450	450 - 500	500 - 550	550 - 600	600 - 650	650 - 700	700 - 750	750 - 800	800 - 850	850 - 900	900 - 950	950 - 1000
EEG channels	1	Fp1	0,4103	0,2740	0,7934	0,6950	0,2692	0,3470	0,0006	0,0538	0,3569	0,4356	0,7674	0,7512	0,9869	0,7923	0,8842	0,5920	0,9216	0,6353	0,9711	0,5027
	2	Fp2	0,3766	0,0899	0,3606	0,1349	0,0282	0,0003	0,0000	0,0000	0,0028	0,4710	0,2389	0,1225	0,2999	0,1789	0,8685	0,8307	0,9375	0,8608	0,7855	0,6652
	3	F3	0,7442	0,2178	0,0105	0,9335	0,6823	0,0000	0,0000	0,0000	0,0003	0,3751	0,5087	0,6737	0,2257	0,5184	0,0223	0,1001	0,0479	0,1849	0,1090	0,0590
	4	F4	0,7989	0,0706	0,0876	0,0697	0,0838	0,0000	0,0000	0,0000	0,0000	0,0825	0,0687	0,0606	0,4032	0,4362	0,4703	0,1646	0,2807	0,4343	0,6846	0,1987
	5	C3	0,2365	0,1566	0,0002	0,0094	0,9610	0,0000	0,0000	0,0000	0,0000	0,0001	0,0019	0,0384	0,8303	0,7624	0,1375	0,2487	0,1650	0,5316	0,1970	0,1894
	6	C4	0,6561	0,0667	0,0446	0,7666	0,1368	0,0000	0,0000	0,0000	0,0000	0,0001	0,0004	0,0010	0,0457	0,1531	0,7960	0,5862	0,9146	0,8027	0,7208	0,4234
	7	P3	0,2390	0,1191	0,0000	0,0000	0,1124	0,0000	0,0000	0,0000	0,0000	0,0000	0,0001	0,0124	0,9556	0,8992	0,8549	0,6224	0,7293	0,5063	0,8369	0,3925
	8	P4	0,4617	0,0324	0,6516	0,0801	0,3907	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0068	0,0987	0,1763	0,1741	0,1880	0,0659	0,2410	0,9794
	9	O1	0,4086	0,1563	0,1338	0,0000	0,0000	0,0029	0,0143	0,0001	0,0010	0,0057	0,0119	0,0355	0,7959	0,6745	0,9416	0,8225	0,6224	0,7667	0,4504	0,0911
	10	O2	0,7564	0,0568	0,1665	0,0014	0,0001	0,0083	0,0017	0,0000	0,0000	0,0000	0,0000	0,0001	0,0056	0,1175	0,1199	0,0827	0,1452	0,0575	0,2411	0,7913
	11	F7	0,4111	0,1294	0,0170	0,0125	0,0282	0,1574	0,0012	0,0010	0,3641	0,3792	0,2649	0,1383	0,0265	0,1038	0,0086	0,0678	0,0608	0,1068	0,0589	0,0243
	12	F8	0,2604	0,2174	0,3921	0,5845	0,1269	0,0024	0,0000	0,0000	0,0000	0,0000	0,0001	0,0003	0,0031	0,0023	0,0839	0,7624	0,5858	0,7508	0,3515	0,4807
	13	T3	0,0597	0,0404	0,0000	0,0000	0,0000	0,5734	0,0787	0,0000	0,0002	0,0085	0,2377	0,5304	0,6373	0,7991	0,1701	0,3095	0,5717	0,5653	0,3074	0,1898
	14	T4	0,0914	0,8788	0,5175	0,0728	0,7223	0,1407	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0003	0,0007	0,0123	0,2663	0,1449	0,2641	0,0794	0,7634
	15	T5	0,0869	0,0988	0,0000	0,0000	0,0000	0,0127	0,0444	0,0000	0,0000	0,0009	0,0306	0,1340	0,7489	0,8986	0,1813	0,5858	0,4534	0,6636	0,3578	0,2117
	16	T6	0,7191	0,2389	0,4679	0,0113	0,0049	0,0280	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0016	0,0080	0,0341	0,0149	0,0050	0,0076	0,1249
	17	Fz	0,6016	0,0668	0,0479	0,0307	0,2402	0,0000	0,0000	0,0000	0,0109	0,9547	0,6168	0,5520	0,5335	0,8229	0,0520	0,0917	0,0945	0,1884	0,1419	0,0553
	18	Cz	0,4000	0,1299	0,0100	0,0855	0,0316	0,0000	0,0000	0,0000	0,0000	0,0002	0,0011	0,0029	0,1436	0,1799	0,7169	0,9237	0,8423	0,9277	0,5735	0,2609
	19	Pz	0,2922	0,0774	0,0019	0,0561	0,6958	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0005	0,1553	0,3365	0,6079	0,3553	0,4709	0,2737	0,7282	0,5249

$p < 0.0001316$

Figure 7.32: The table shows p-values from two-sample t-test verifying whether there is no difference between target and non-target averaged time windows throughout all EEG channels. Significant differences for predefined significance level 0.0001316 (forming feature vectors of dimensionality 83) are highlighted.

8 Development and evaluation of neural networks for the P300 detection

This section describes P300 data classification based on neural networks. The outlined procedures were first proposed in two publications. In [Author6], ART and SOM-based modifications were described. Evaluation of stacked autoencoders was published in [Author7]. The aim of this section is to combine those approaches and present an evaluation of all related classification methods.

8.1 Procedure used to evaluate classification

For preprocessing and feature extraction, the procedure as described in Subsection 7.9 was used. Using the selection of averaged time windows and channel depicted in Fig. 7.32, the dimensionality of feature vector was 83.

From each subject, all target feature vectors were used for subsequent processing. The corresponding number of non-targets was randomly selected. Consequently, the training dataset contained 737 target and 737 non-target feature vectors. Only the training set was used for classifiers training. There were no further weight updates in the testing mode. Therefore, it could be observed if once trained classifiers can can also perform well for other

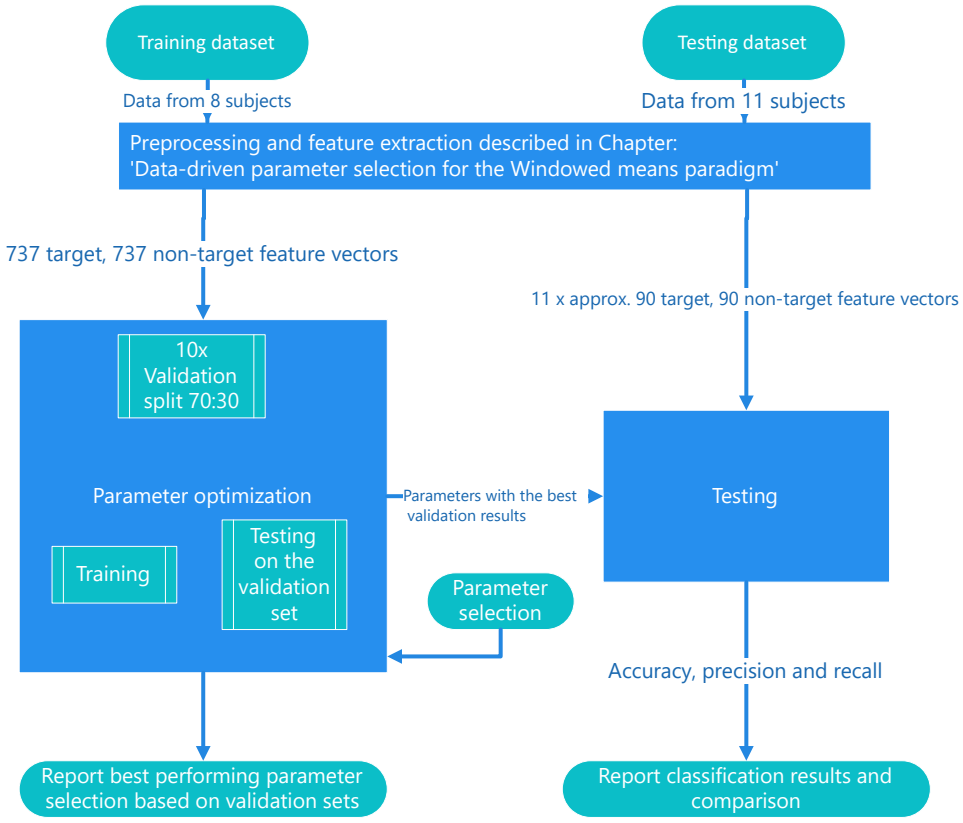


Figure 8.33: Flowchart depicting evaluation of classification models regarding the P300 detection. Preprocessing and feature extraction described above were used.

subjects.

To optimize parameters for various classification models, it is usually a good practice to create a validation set that is used to check classification results before parameter optimization is finalized and the classifier is applied to the testing set. 30 % randomly selected subset of the training dataset was used for validation. During validation, various manually inserted classification parameters, that are discussed later, were compared. To obtain smoother results, training and testing on the validation set was repeated ten times with ten random splits between training and validation sets. Validation classification results were averaged. Best performing parameters were chosen based on the validation set classification accuracy. After the parameters for each classification model were found, the models were tested on the testing dataset. During the testing phase, the data from each participant were evaluated separately. Similarly to the training dataset, all target trials were included but only the corresponding number of first non-target trials were used. The number of trials varied slightly for each

subject. However, for most subjects, approximately 90 target and 90 non-target trials were extracted. Finally, overall testing classification results were also calculated. The procedure used for training, validation and testing is depicted in Fig. 8.33.

8.2 Classification algorithms

Different classification models were proposed and subsequently compared with traditionally used classifiers.

8.2.1 Self-organizing maps

Self-organizing maps are typically used for unsupervised learning. An existing supervised modification LASSO [60] and two modifications of SOM I have designed (referred to as SOM1 and SOM2) are evaluated. The SOM toolbox [70] was used for the implementation. The following SOM-based methods were tested:

- The SOM1 method was a simple procedure I have designed to turn an unsupervised SOM into a supervised classifier. It was based on assigning the most likely class label for each SOM neuron (codebook vector). The motivation to test this method was an assumption that once the SOM training using self-organization is finished, each SOM unit represents a distinct ERP pattern that may correspond to both target and non-target labels. Such distinct patterns could maintain some stability and thus provide reasonable estimation for testing data as well.

  As the first step of the SOM1 method, the SOM network was trained using self-organization in the standard way. The second process was performed to label the SOM cluster units with the most likely outputs. A memory unit containing two variables (t for number of targets, n for number of non-targets) was associated with each SOM unit. One iteration of the labeling process was performed as follows ($\boldsymbol{x}$ was the input vector, ω_{ij} was the weight between i-th input and j-th neuron):

  1. In the same way like it would be performed for traditional SOM learning: for each

SOM unit j, compute distance $D(j)$ from the input vector $\boldsymbol{x}$:

$$D(j) = \sum_i (\omega_{ij} - x_i)^2 \tag{8.31}$$

2. Find a cluster unit j such that $D(j)$ is a minimum, and denote it J. (J is the closest cluster unit.)
3. If the expected class of feature vector $\boldsymbol{x}$ is target, set $J_T \leftarrow J_T + 1$. Otherwise, set $J_N \leftarrow J_N + 1$

Consequently, for each SOM unit, numbers of associated target and non-target features were obtained.

Finally, to compensate for possible differences in numbers of targets and non-targets, for each SOM unit J in the map, the following computation was performed: $J_T \leftarrow J_T / number\_of\_targets$ and $J_N \leftarrow J_N / number\_of\_nontargets$.

In the testing phase, the algorithm was as follows:

1. The closest cluster unit J was found for an input $\boldsymbol{x}$.
2. The class label was assigned to the input $\boldsymbol{x}$ as a result of comparison between the number of targets and non-targets previously associated with that cluster unit during the training process. If J_T was higher than J_N, the input $\boldsymbol{x}$ was classified as the target. Otherwise, it was classified as the non-target.

- The previous method was modified to the SOM2 method by adding k-means clustering. The SOM network was trained using self-organization in the same way as for the SOM1 method. Subsequently, k-means clustering was performed to create clusters c from all SOM unit vectors $\boldsymbol{\omega}_j$. Instead of assigning statistics to each SOM cluster unit, statistics were merged for each cluster. In other words, for each labeling iteration:

  1. The winner SOM unit J was found.
  2. The cluster C was assigned if $J \in C$.
  3. Based on the training feature vector label, C_T or C_N was incremented.

Finally, any possible differences between numbers of targets and non-targets were compensated as for the SOM1 method.

This method allowed each SOM cluster to be associated with one prevailing classification class label. In the testing phase, the cluster containing the winning neuron made the decision about the classification class of that input pattern.

- The LASSO model [60] described above was used by connecting input feature vectors with expected output: 0 for non-targets, and 1 for targets. In contrast with the SOM1 and SOM2 methods, labeling is not another step following self-organization. Instead, connected input and output patterns are simultaneously used for self-organization. This approach is an interesting alternative — we evaluate if input-supervised learning alone ensures that the LASSO model reliably classifies previously unknown testing data. Moreover, by setting different parameters, the effects of varying activation group size are observed.

8.2.2 Simplified Fuzzy ARTMAP

Similarly to Self-organizing maps, Adaptive Resonance Theory (ART) is also typically used for unsupervised clustering. In the theoretical part of this thesis, the Fuzzy ARTMAP model and Simplified Fuzzy ARTMAP (SFAM) were described. However, since SFAM aimed at being equivalent to the original Fuzzy ARTMAP map in terms of classification results, I evaluate only the SFAM model [65]. The motivation behind using this model is that the number of outputs classes is not specified and fixed before training. Instead, based on mutual similarities and differences among input feature vectors, new prototypes can be created during run-time and these prototypes are assigned with labels. This could be beneficial when compared to the fixed-sized SOM network. On the other hand, there are many more parameters to configure so the empirical tuning can be quite complicated. For subsequent experiments, Matlab implementation [79] was used.

8.2.3 Stacked autoencoders

Stacked autoencoders provide important improvements when compared to a traditional multi-layer perceptron algorithm. The improvements aim at increasing parameter updates for layers that are closer to the input layer. First, by using linear activation functions instead of the sigmoid combined with the mean square error loss function, vanishing gradient problem is limited. Second, using layer-wise unsupervised pre-training, improved initial settings of weights and biases is found before supervised fine-tuning is performed. These improvements could contribute to better P300 classification. Matlab Neural Network Toolbox was used for the implementation of stacked autoencoders [55].

8.2.4 Traditionally used classifiers for comparison

To evaluate possible benefits of using unsupervised neural networks, two traditionally used classifiers for the P300 detection were also used: Linear Discriminant Analysis (LDA) [16] and multi-layer perceptron. The BCILAB [80] implementation of LDA was used.

8.3 Classification parameter discussion and optimization

8.3.1 SOM1

The SOM1 method only depends on the architecture and parameters of SOM. For simplicity, I assumed a square 2D map with a hexagonal topology. The SOM was initialized using linear initialization and the Gaussian neighborhood function was used to control changes around the winner unit. The SOM was trained in two phases: first rough training and then fine-tuning. These parameters were described in [70] and recommended as default values. The most important remaining parameter of the SOM is then its size. The number of SOM units in the network determines the number of patterns that the SOM is able to learn. Given the fact that the classification result is based on statistics attached to each SOM unit in the training phase, it is likely that for a fixed training set, there is an optimal number of SOM units. A too small SOM size could mean that not enough patterns can be recognized by the network. On the other hand, too many SOM units would not have enough training data to be assigned to most SOM units as winners. I repeated the process of validation for

sizes between 2 and 11. Accuracy, precision and recall achieved using the validation set are shown in Fig. 8.34. The maximum accuracy was achieved for the size of 8 (corresponding to 8 x 8 = 64 SOM units) and this parameter was selected for subsequent experiments. Good performance (especially high recall) has also been achieved for the 2 x 2 network size. However, the accuracy was slightly lower. Moreover, four SOM units would not be enough to justify k-means clustering in the SOM2 method.

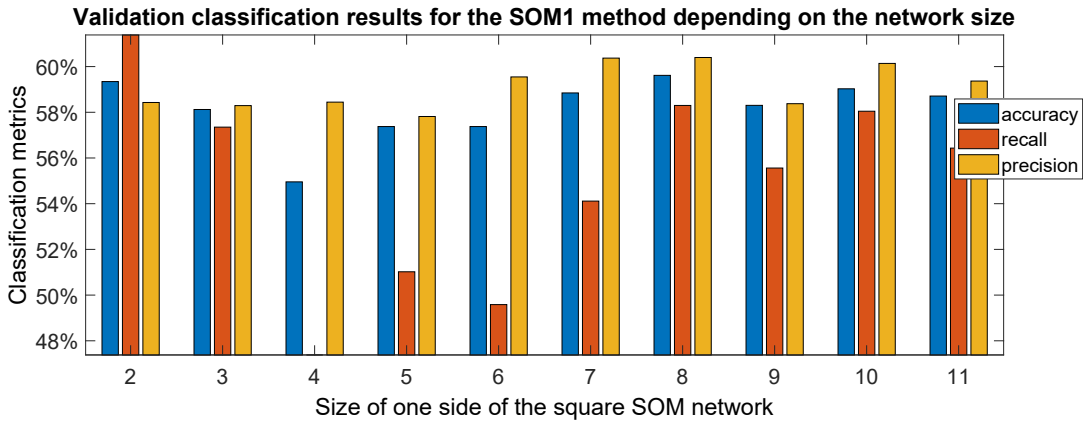


Figure 8.34: Validation results for the SOM1 method depending on its size.

8.3.2 SOM2

The SOM2 method was configured in the same way as the SOM1 method. The size of the network was set to 8 x 8 as well. The remaining parameter potentially affecting performance is the number of clusters k. From classification perspective, the clusters that are used to group SOM cluster units are effectively simplifying classification. For example, with two clusters allowed, the 8 x 8 network is learned to distinguish 64 patterns but subsequently, the SOM units are only assigned into two clusters and only the cluster of the winner SOM unit determines the classification output. Validation results depending on the number of clusters are shown in Fig. 8.35. The maximum accuracy with balanced precision and recall was achieved for five clusters. Interesting results in terms of recall were achieved for three clusters, too. However, the accuracy was slightly lower and tests revealed that for each of those three clusters, the numbers of target and non-target hits were only slightly different in the training session, possibly limiting classification abilities. Consequently, the SOM2 method with five clusters was used in the testing phase.

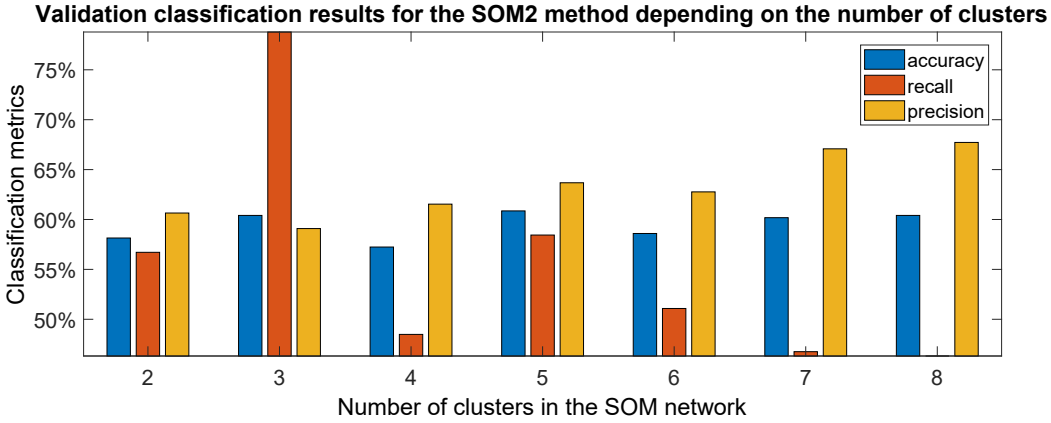


Figure 8.35: Validation results for the SOM2 method depending on the number of clusters.

Table 8.2: Validation classification results for the LASSO classifier

Validation acc./prec./rec. (%)		Grid distance s_D				
		1	2	3	4	5
Rel. distance p_I	0.05	56.5 / 41.6 / 62.7	57.5 / 47.2 / 62.3	55.9 / 49.4 / 59.4	55.7 / 51.2 / 58.7	55.4 / 51.1 / 58.4
	0.175	58.4 / 49.4 / 63	55.7 / 51.5 / 58.6	57.2 / 53.7 / 60.2	57.2 / 58 / 59.3	57.2 / 60.2 / 58.9
	0.3	55.2 / 45 / 59.4	57 / 46.7 / 61. 2	58.6 / 55 / 61.7	58.6 / 58.9 / 60.7	61.8 / 64.5 / 63.1
	0.425	56.6 / 46.3 / 61.1	58.8 / 49.8 / 62.8	58.4 / 50.6 / 62.6	58.1 / 55.4 / 61	62 / 64.1 / 63.5
	0.55	57 / 48.5 / 61.2	58.4 / 49.8 / 62.8	58.6 / 51.5 / 62.6	59.7 / 55 / 63.2	62.2 / 63.6 / 63.9

8.3.3 LASSO

As described in Chapter 5.5.2, the LASSO algorithm can be configured using two parameters. s_D is the threshold for the SOM unit space distance. For typical 2D situation, it can be simply interpreted as the distance in the grid to the winner SOM unit. In contrast, p_I is the relative distance in the input feature vector space. Both parameters can be used to restrict the activation group. For lower thresholds, less SOM units would belong to the activation group and consequently, less SOM units would affect the output of classification. In the extreme case, for $s_D = 1$, only the winner SOM unit makes the decision about the classification output. I tried to perform validation for different values of p_I and s_D to see how they affect the results of classification. The underlying SOM was configured in the same way as for both SOM1 and SOM2 methods. As Table 8.2 shows, there is a clear pattern of improving classification results with increasing LASSO parameters. Therefore, for the validation set, large activation groups seem beneficial for accuracy. It would be possible to try even higher parameters, however, this seems inconsistent with the aim of the LASSO in which only SOM unit close to the winner make significant contributions to the classification output. For testing, p_I was set to 0.55 and s_D was set to 5.

Table 8.3: Effect of vigilance ρ of the SFAM model on the number of created prototypes and validation classification results.

ρ	Number of prototypes	Accuracy (%)	Precision (%)	Recall (%)
0.02 - 0.76	26	59.3	62.1	56.7
0.8	30	59.7	61.5	61.5
0.84	58	56.3	57.8	61
0.88	117	53.4	56.1	49.8
0.92	241	52	53.7	59.7

8.3.4 Simplified Fuzzy ARTMAP (SFAM)

There are a lot of parameters that influence the performance of the Fuzzy ARTMAP (both original and simplified variants). To simplify the process of empirical optimization, I decided to set up most parameters based on recommendations in the literature. Consequently, as proposed in [65], alpha was set to 0.001 and epsilon to 0.001 to limit the number of prototypes and to prevent a false data mismatch alarm. Beta was set to 1 to allow fast learning. The number of training epochs was set to 50 because based on my experiments, the network remained stable after 50 epochs. Vigilance (also referred to as threshold of recognition) is a remaining important parameter because it affects creation of new prototypes (neurons in the second layer). Lower vigilance means more likely resonance while higher vigilance means less likely resonance and more likely prototype creation. In other words, settings of vigilance can lead to either more general memories (low vigilance, fewer, more general categories), or to detailed memories (high vigilance, many categories) [65]. I tried to set up vigilance in range between 0.04 and 0.92 with step 0.04 and observed both numbers of prototypes created and validation classification results. The results are in Table 8.3. Apparently, the number of prototypes was unchanged for vigilance between 0.02 and 0.76 but then started increasing rapidly. In contrast, classification results appear to decrease with possible but small exception for $\rho = 0.8$. The higher number of prototypes obviously does not seem to lead to better classification results. Finally, in line with these results, ρ was set to 0.8.

8.3.5 Stacked autoencoder (SAE)

There are many parameters that affect training and performance of stacked autoencoders. Consequently, a combination of recommendations in the literature and empirical testing was

used to achieve optimal performance based on the validation set.

Global training parameters were set to comply with deep learning best practices. The linear activation function and mean square error (MSE) loss function were used for all layers. During training, this combination should retain high enough derivatives for higher errors and thus limit the vanishing gradient problem.

Other parameters of the stacked autoencoder (number of layers, number of neurons in each layer, and number of iterations for the hidden layers) were empirically optimized using the results on the validation set. Since the training was indeterministic and the results often varied for similar settings of the network, the process was based on manually designing different neural network architectures (with different number of layers and number of neurons in each layer) and comparing the accuracy achieved on the validation set.

Finally, the following procedure was used to train the feed-forward neural network. The maximum number of training epochs for each autoencoder was limited to 200. Based on experimental results, this number was high enough because for each autoencoder, training stopped before reaching this threshold when gradient fell below 10^{-6}. The following architecture with empirically optimized parameters was used:

1. The first autoencoder with 70 hidden neurons was trained.

2. The second autoencoder with 55 hidden neurons was connected with the first autoencoder to form a 83-70-55-83 neural network, and trained.

3. The third autoencoder with 40 hidden neurons was connected with the second autoencoder to form a 83-70-55-40-83 neural network, and trained.

After the training of each autoencoder, the input feature vectors were encoded using that autoencoder to form input vectors of the next autoencoder. Using the output of the last autoencoder, the softmax supervised layer was trained with 200 training iterations and cross-entropy loss function. Finally, the whole pre-trained 83-70-55-40-2 network was fine-tuned using scaled conjugate gradient method (SCG). The structure of the stacked autoencoder is depicted in Fig. 8.36.

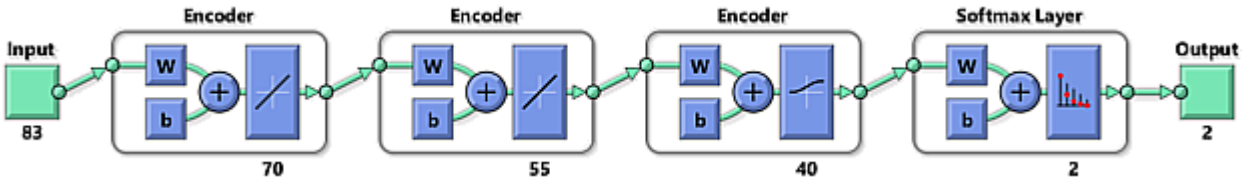


Figure 8.36: The structure of the SAE neural network.

Furthermore, regularization parameters were needed for the network globally to reduce overtraining and adjust the weight update. We have defined sparsity regularization in Equation 5.25 and L2 regularization in Equation 5.27. Then the MSE error function was modified to penalize poorly regularized network configurations:

$$MSE_{REG} = MSE + \lambda * (L2REG) + \beta * (SPARREG(\rho)) \tag{8.32}$$

Based on recommendations [55] and after empirical tuning, ρ was set to 0.2. Subsequently, based on the validation set, I verified how modifying λ and β parameters affects classification performance. I decided to keep β to λ ratio to 1000:1. Consequently, effects of increasing regularization controlled by the β parameter were observed based on the validation set (see Figure 8.37). The maximum validation accuracy in combination with high and balanced precision and recall were achieved for $\beta = 15$ and $\lambda = 0.015$. These values were set for subsequent testing. When regularization increased more, classification performance started to decrease rapidly.

8.3.6 Multi-layer perceptron (MLP)

The same numbers of neurons for each layer with the same parameters as for the SAE were used for MLP. However, the phase of unsupervised pre-training was not included. Instead, the randomly initialized network was trained using SCG. Consequently, benefits of unsupervised pre-training and SAE regularization could be evaluated.

8.3.7 Linear Discriminant Analysis (LDA)

For the training of LDA, the standard estimator for a covariance matrix is the empirical covariance. This estimator can fail for high-dimensional data with limited number of train-

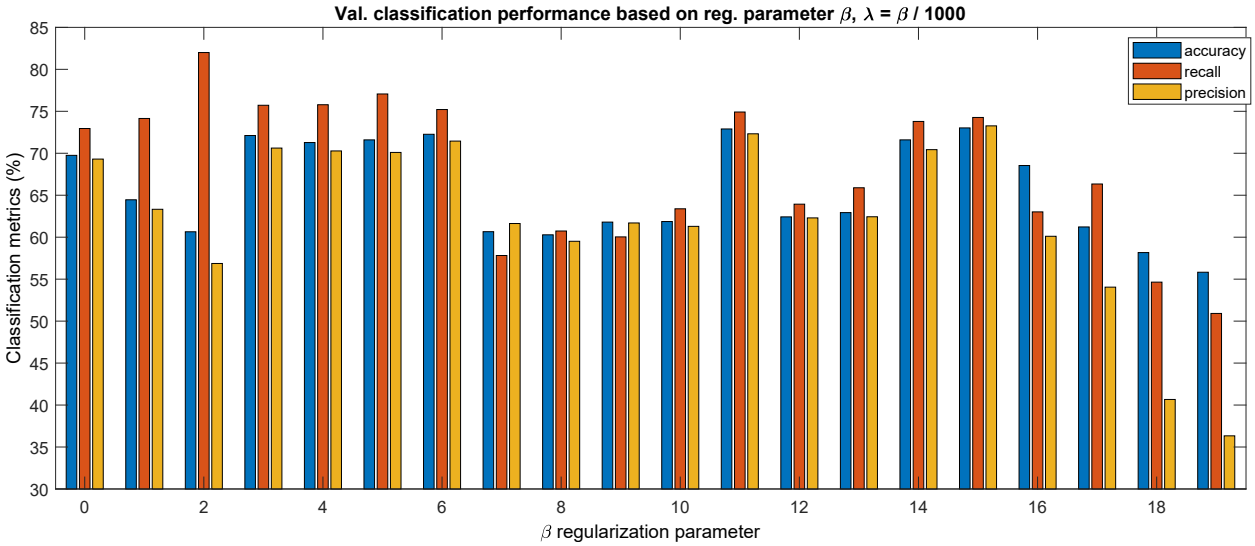


Figure 8.37: Validation results for the SAE network depending on regularization parameters β and λ. $\lambda = \frac{\beta}{1000}$

ing samples because the number of parameters to be estimated is quadratic in the number of dimensions. Shrinkage is a common solution for high-dimensional feature vectors [16]. Consequently, shrinkage regularization was used for testing.

8.4 Results

In the testing phase of the experiment, the data from each experiment were evaluated separately. The parameters were selected based either on the literature, or on empirical tuning using the validation set as described above. In summary, for any parameter to be set empirically, maximum validation accuracy in line with reasonable precision and recall was requested.

Since neural networks have indeterministic training, both training and subsequent testing phases were repeated 20 times for each classification algorithm. In each run, 70 % of the training data were randomly chosen (the remaining 30 % for validation were ignored since the parameter optimization was already performed). Consequently, training data were slightly different with each repetition. Testing results were evaluated by means of accuracy, precision and recall. First, the results were evaluated for each testing dataset and each classification model. Fig. 8.38, 8.39 and Fig. 8.40 depict average accuracies, precisions, and recalls,

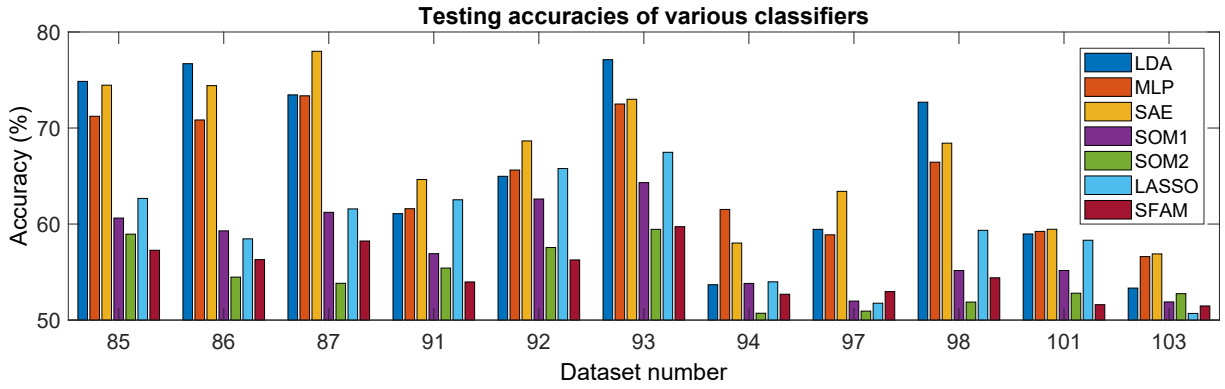


Figure 8.38: Accuracy of tested classifiers for individual testing datasets.

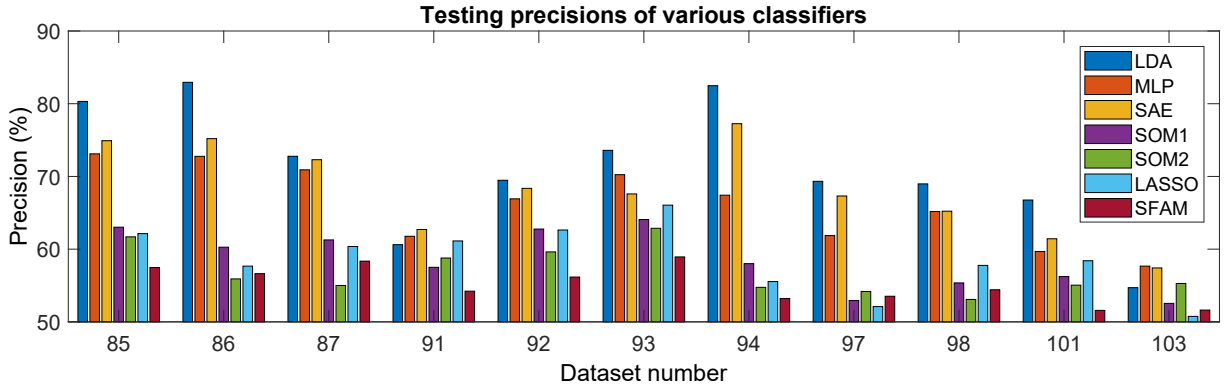


Figure 8.39: Precision of tested classifiers for individual testing datasets.

respectively. Table 8.4 depicts classification performances for each training model across all measured subjects. Both averages and corresponding standard deviations are depicted. Moreover, it was evaluated how each classifier performs when classifying local group averages of two to six neighboring epochs instead of single trials (Fig. 8.41).

Furthermore, for each training model, training and testing times were evaluated (Intel Core i7, 64 GB RAM, SSD hard drive). The results are shown in Table 8.5.

8.5 Discussion

As depicted in Table 8.4, average classification accuracies remained between 65 % and 70 % even for better performing classifiers. These accuracies may appear fairly low but it should be taken into account that the P300 component was detected in single trials while typically, BCI systems average many subsequent trials to limit the background noise. As Fig. 8.41 shows, classification accuracy increased considerably (up to 78 %) when classifying local

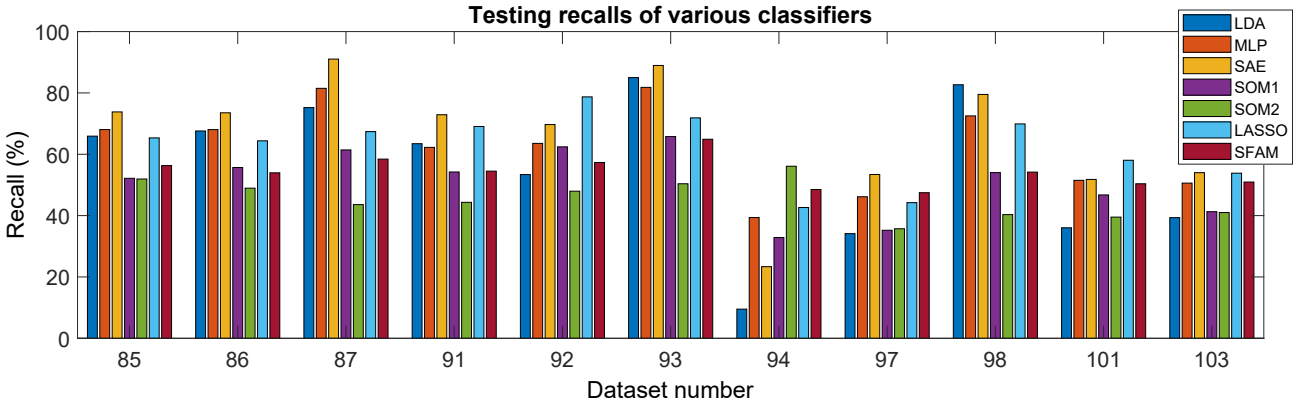


Figure 8.40: Recall of tested classifiers for individual testing datasets.

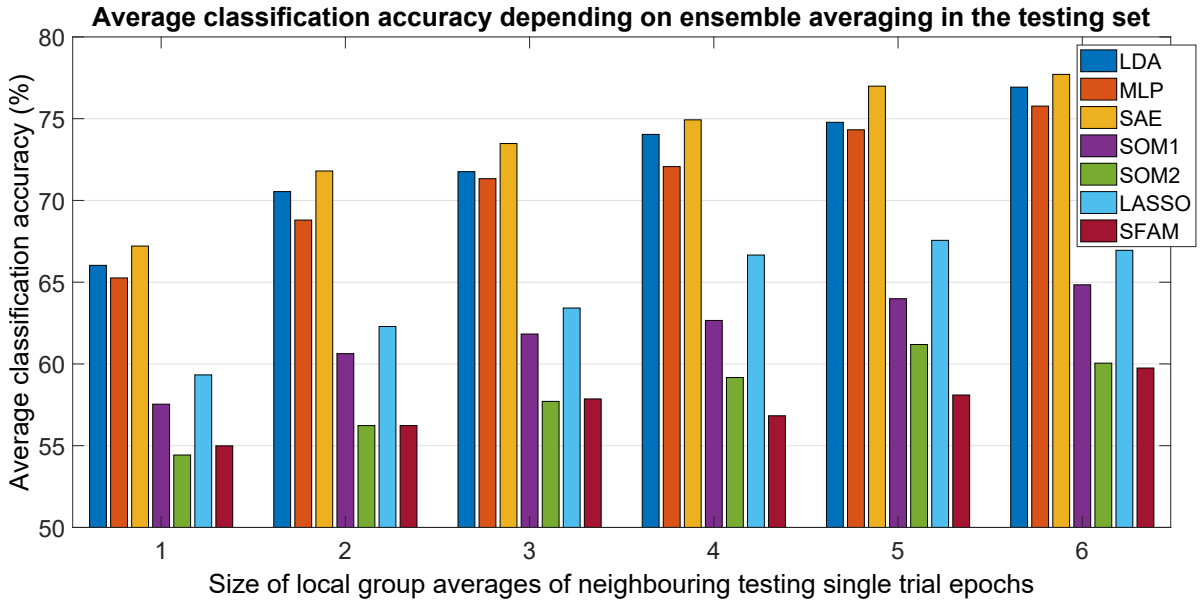


Figure 8.41: Average accuracies across all measured subjects when classifying local group averages of neighboring epochs. As random noise gets suppressed, there is an apparent increasing trend in accuracy for all classifiers, especially for LDA, MLP and SAE.

Table 8.4: Average classification performance for different classifiers across all subjects in the testing set (20 runs averaged, standard deviations in brackets).

Classifier	Average accuracy (SD)	Average precision (SD)	Average recall (SD)
LDA	66.03 (0.85)	71.09 (1.06)	55.66 (1.56)
MLP	65.26 (1.4)	66.14 (2.06)	62.3 (2.76)
SAE	67.21 (1.04)	68.16 (1.83)	66.54 (3.22)
SOM1	57.54 (1.4)	58.55 (1.88)	51.07 (3.06)
SOM2	54.43 (0.97)	56.93 (5.15)	45.44 (13.65)
LASSO	59.33 (1.84)	58.6 (1.64)	62.31 (4.02)
SFAM	54.99 (1.32)	55.1 (1.6)	54.26 (3.39)

Table 8.5: Average calculation speed for different classifiers (Intel Core i7 4930K, 64 GB RAM, AMD Radeon R7 200 Series was used, training was performed using CPU). To obtain the training calculation time, the time needed for the whole training was calculated. In the testing mode, the time needed to evaluate all feature vectors from one subject was counted.

Classifier	Training calculation time (ms)	Testing calculation time (ms)
LDA	49	1
MLP	1882	14
SAE	8176	11
SOM1	62	1
SOM2	369	1
LASSO	54	215
SFAM	4162	16

group averages instead of single trials. Furthermore, instead of personalized training data used in most BCI systems, these P300 classification models were trained to be universal, i.e. trained and tested on different subjects. When examining standard deviations, LDA maintained slightly more stability among different runs, since its training is deterministic and only the training data were partially modified.

The results in both Figure 8.38 and Table 8.4 indicate that stacked autoencoders (SAE) were able to slightly outperform other classifiers including LDA and MLP. The improvement is more pronounced in recall than in precision. As Figure 8.38 illustrates, SAEs yielded higher accuracy than all other classifiers in 6 out of total 11 subjects. Consequently, it appears that stacked autoencoders were able to match or outperform current state-of-the-art classifiers for the P300 detection in accuracy and in recall. This small improvement can probably be explained by improved training in SAE that also includes unsupervised pre-training and improved regularization. Moreover, in my related publication [Author7], I compared SAE with LDA and MLP using predefined time windows and all EEG channels. The feature vector dimensionality was 209. With this configuration, using the same testing data, SAE average accuracy was 69.2 % which was significantly higher than MLP accuracy 64.9 % and LDA accuracy 65.9 % (McNemar statistical tests; $p < 0.01$). Consequently, it appears that SAE is resilient to the curse of dimensionality. In fact, dimensionality reduction is a part of the SAE training algorithm [81]. The results achieved are consistent with promising results reported by [82]. For deep belief networks, the authors reported 60 % to 90 % for other methods compared with 69 % and 97 % for deep learning. The achieved results encourage using deep

learning models for the P300 component detection with applications to P300-based BCIs.

In contrast with SAE, neither SOM nor ART-based models matched state-of-the-art classifiers. The LASSO model yielded the best accuracy (approximately 59.33 % on average). In my opinion, these generally poor results are caused by too high feature vector dimensionality which prevent SOM and ART-based models to create sufficient number of distinct patterns that are later associated with class labels. This hypothesis is supported by the results that I achieved in [Author6]. In that paper, using predefined time windows and Fz, Cz and Pz channels, the feature vector dimensionality was only 27. With this settings and the same testing datasets, average accuracy was 64.09 % for the LASSO model and 62.1 % for the SOM1 method, while SOM2 and SFAM still yielded poor results (58.58 % and 53.88 %, respectively). Apparently, k-means clustering in the SOM2 method did not lead to classification improvement. The SFAM method yielded much smaller accuracy for the testing data than for the validation data, suggesting that this classifier was prone to overtraining. In contrast, the LASSO method can be especially vulnerable to the curse of dimensionality, since it is based on merging a feature vector with its corresponding class label. A higher feature vector dimensionality leads to relatively smaller impact of one-dimensional class labels on self-organization.

Although classification accuracy is very important for the reliability of P300 BCI systems, only BCIs with reasonably fast bit-rates are comfortable to be used for disabled users. Therefore, all used methods were compared based on the average time needed to evaluate all feature vectors from one subject in the testing phase (average time was measured from the acceptance of all feature vectors from one subject until the decision about their classification class). Since real-world BCI systems should be able to evaluate ERP trials on-line, computational time for the processing and classification of feature vectors should not be higher than inter-stimulus intervals. According to our experience, to be comfortable to use, inter-stimulus intervals should be at least 200 ms. In the literature, only slightly lower inter-stimulus intervals are used for the P300 speller (for example, 175 ms in [83]). Consequently, any system that evaluates a single ERP trial faster may be used. Fortunately, once the BCI system is trained, classifying a single feature vector is usually not very time consuming. As Table 8.5 clearly shows, all algorithms can be used in an on-line BCI system. When implementing any

universal BCI system, training times are usually less important for their usability. For those systems, training has to be performed only once. However, if a more personalized system is requested, the training process would have to be repeated for each new user. Table 8.5 depicts that feed-forward neural networks (MLP and SAE) and the SFAM model were the only classification models that required more than one second to be trained. Nevertheless, training times are still quite low for all models including SAE which can be explained by a relatively small training set.

For future work, more issues remain to be addressed. Deep learning is apparently relatively resilient to the curse of dimensionality and is known to outperform other classifiers when large collections of diverse data are used for training. This is especially relevant for the P300 data, considering large variability in the P300 amplitudes and latencies among different subjects and different sessions. Therefore, collecting much larger P300 datasets has the potential to make the improvements in classification accuracy even more pronounced. However, achieving this goal is still a challenge since many EEG laboratories are unwilling to share their data. Even the data that are publicly available have been recorded using different electrode placements and different stimulation paradigms, limiting their potential for merging into bigger training sets. For possibly improved performance of stacked autoencoders, a large collection of unlabeled ERP data could help too — a lot of unlabeled data could be used for unsupervised pre-training, while a smaller labeled dataset would be used for supervised fine-tuning.

In the current experiment, even though SAE was better than LDA and MLP, still, only four participants reached an accuracy above 70 % which is often seen as a minimum to use a P300-based BCI [84]. It can therefore be evaluated how an individualized BCI system (i.e. the system trained on the data from its particular user) would perform and if better performance of SAEs outweighs their increased training times. In [82], pre-training possibilities for deep belief networks are discussed. The authors proposed that the weights of a new neural network could be initialized using the results of pre-training based on another subject. The same principle could be applied to stacked autoencoders. This could lead to possibly increased classification performance. Another possible strategy for increasing bit-rate would be to shorten the inter-stimulus interval. Although shorter intervals could lead to lower P300

amplitudes, SAE can classify high-dimensional feature vectors and could detect only slight differences in the feature vectors. Furthermore, it could also be interesting to explore stacked denoising autoencoders, convolution neural networks or other deep learning training models.

9 Implementation and testing of an example BCI application

The goal of the experiments described above was to verify different methods based on neural networks for P300 BCIs and to compare them with traditional classification approaches. This section presents an implementation of an on-line and off-line BCI and its evaluation in a real-world environment. The example BCI is based on the 'Guess the number' experiment.

9.1 Guess the number experiment

The 'Guess the number' experiment is based on visual stimulation, and was originally developed to demonstrate the benefits of using BCI to public. The participant in the experiment is asked to choose a number between 1 and 9 and concentrate on it (i.e. this number is the target stimulus). Then, the subject is exposed to visual stimuli that consist of nine numbers randomly appearing on the monitor. During the experiment, both EEG signal and stimuli markers are recorded. Concurrently, experimenters observe average event-related potential (ERP) waveforms for each number, search for the P300 component, and try to guess the number thought. Their guess is finally verified when the participant is asked to reveal the thought number. The hardware used and the course of the experiment are illustrated in Fig. 9.42. Although originally, only human experts tried to guess the number thought, it is useful to have a simple BCI application that can make the decision automatically. Such an application would not be useful in clinical practice, e.g. for disabled patients. However, it is based on the same principle as any other P300 BCI systems, and the numbers in the scenario can easily be replaced e.g. by commands in the hospital settings (switch on the TV, ask for nurse help, etc.)

Figure 9.42: **a)** Medium 10/20 EEG cap. **b)** The BrainVision V-Amp amplifier. **c)** Course of the 'Guess the number' experiment. Researchers observe event-related potentials while they are averaged in the BrainVision Recorder. The notebook on the right was used to control the stimulation. The subject sitting in the chair is exposed to visual stimuli. The subject's face has been blurred to protect his/her privacy.

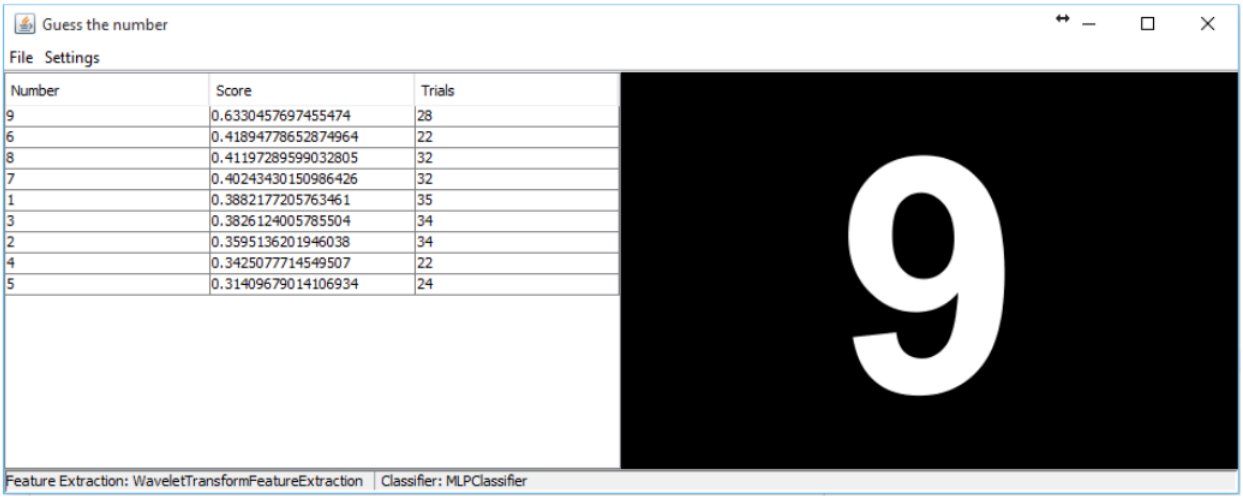


Figure 9.43: Front-end of the Guess the number application. On the right, the number chosen by the classifier is depicted. On the left, all numbers are listed ordered by classification score (neural network output) in descending order. The higher the average score for the number is, the more likely was the number chosen by the measured subject.

9.2 Guess the number - application for on-line and off-line BCI

I participated in the development of an application for experiments described above. It is a desktop application written in Java language using Swing for its graphical user interface (a screenshot is shown in Fig. 9.43). The purpose of this application is to enable off-line (experimental data are collected and analyzed after the experiment is performed) and on-line (data are streamed into the application during an experiment) classification. Both on-line and off-line data are processed in the same way. Therefore, off-line classification allows users to test preprocessing, feature extraction, and classification algorithms. Subsequently, the most suitable combination of algorithms can be selected for on-line classification. The application is still under development and available on GitHub [85]. The current aim is to extend the application to be able to control different BCIs based on various experimental scenarios.

9.3 On-line and off-line experiments and evaluation

For the purpose of tuning this BCI application, the system was trained using the data obtained with the Guess the number experiments. Subsequently, the system was tested on other datasets following the same protocol. The system was evaluated mainly in off-line

mode using the previously recorded data. However, sporadically, the system was also run on on-line data during the experiments to verify that on-line and off-line classification provide the same results.

9.4 Measurements performed to obtain the data

The measurements were conducted in elementary and secondary schools mainly in the Pilsen region, the Czech Republic between autumn 2014 and spring 2015. They were taken at the time of regular school hours, typically in the morning. Each experiment was performed in a classroom that was dedicated for health entertaining and educating program, also including Neurosky brain games, ECG monitoring, modeling of body muscles, etc. Unfortunately, the environment was usually quite noisy since many children were in the room at the same time. From each experiment, BrainVision files were stored. Furthermore, important metadata about each participant were also stored. Most importantly, the number thought by the participant was stored to allow researchers to verify the classification accuracy later. The overall dataset consisted of 239 measurements from different subjects with the average age of 13.2. Details about the experiment were published in [Author10].

9.5 Comparison with the human expert

Typically, the reported accuracies of various BCI classification systems are mutually compared. However, the unique design of the Guess the number experiment allows us to compare classification results with the human experts classification as well. During the experiments, the data were evaluated by a human expert with neuroinformatics background — in 67.6% experiments, the expert was able to predict the correct number at the first attempt. The expert observed epochs as they were gradually averaged for each stimulus (guessed number) in the BrainVision Recorder software [74] and searched for the most likely target. Each experiment ended either when the expert was convinced about the number guessed (either correctly or incorrectly) or he was unable to make the decision.

9.6 Off-line classification of the Guess the number data using stacked autoencoders

The following classification experiment is based on two publications ([Author11] and [Author12]) which also contain its detailed description.

The preprocessed data were split into training, validation and testing datasets. The training and validation dataset contained data from 13 subjects. These subjects were selected manually based on their P300 response to target stimuli. From each subject, all target trials were extracted. Additionally, the corresponding number of non-targets was randomly selected. The number of target stimuli varied for each subject. However, in total, 262 target and 262 non-target randomly shuffled trials were used for training and validation. Moreover, 20 % of the set (i.e. the validation set) were used for classification parameter optimization and were not used for training. In contrast, the testing set contained 206 datasets corresponding to 206 measured subjects. The rest of the collected data were manually excluded because the related signal was highly damaged. Since only three EEG channels (Fz, Cz, and Pz) were available in the data, the exact same configuration of feature extraction and classification from Section 8 could no longer be used. Instead, the workflow depicted in Figure 9.44 was used for signal processing and classification.

After epoch extraction and baseline correction were performed, dimensionality was reduced by using only 512 samples between 175 ms and 687 ms following the stimuli. This reduction also ensured that the dimensionality of preprocessed epoch was a power of 2. Subsequently, for each EEG channel separately, 5-level DWT using the Daubechies-8 wavelet followed. Approximation coefficients of level 5 from each EEG channel were merged to form feature vectors of dimensionality 48. Finally, the scales of the features were normalized using vector normalization. Subsequently, the stacked autoencoders were designed to be compared with other classifiers, including LDA, MLP, and SVM.

Stacked autoencoders were implemented using the Deeplearning4j library [86]. Their parameters were tuned empirically based on the validation set. Finally, the network was configured in the following way:

- both layer-wise unsupervised pre-training and supervised fine-tuning was performed,

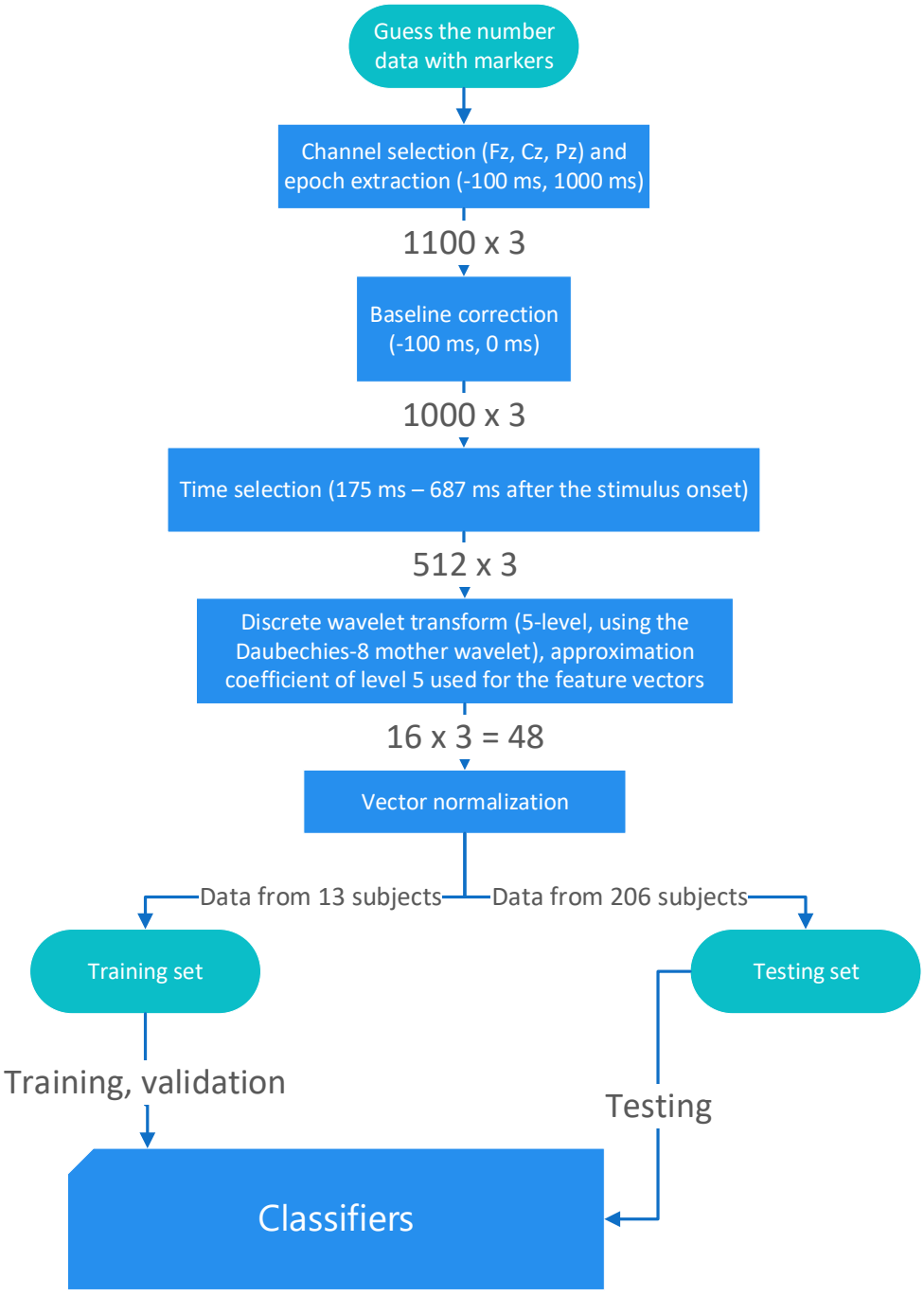


Figure 9.44: Flow chart of feature extraction and classification methods used for 'Guess the number' experiments. The arrows connecting procedures are denoted by epoch or feature vector dimensionality.

- for both, the number of iterations was set to 1500,
- the optimization algorithm was set to Stochastic Gradient Descent,
- the learning rate was set to 0.005,
- the Nestorovs updater with momentum 0.9 was used.

The network contained three layers and its architecture was 48 - 24 - 12 - 2. The weights of all layers were initialized using the Xavier method. The first two layers used the Root-mean-square error cross entropy loss function and relu activation. The last layer used the Negative likelihood loss function and softmax activation. Thus, vanishing gradient problem was minimized and the resulting neural network did not suffer from overtraining.

In the same way as for Section 8, configuration of LDA and SVM was based both on the related literature and experiments on the validation set. The regularization of LDA was set to shrinkage and for SVM, C-SVC type of SVM with linear kernel function, cost 425, weight 1 and seed 1 was used. MLP was configured in the same way as SAE, except unsupervised pre-training phase was not included.

For each classifier and each dataset, one step towards the decision about the most likely number thought was made as follows:

1. The classifier receives a feature vector $\boldsymbol{x}$ and a stimulus code corresponding to the related displayed number c.
2. The classifier returns a score $s = classify(\boldsymbol{x})$, typically in range between 0 and 1. The higher the score is, more likely the epoch belongs to the target class.
3. Score s is added onto the c position of the array containing scores: $scores[c] \leftarrow scores[c]+s$. Moreover, counter of scores is incremented $cnt[c] \leftarrow cnt[c]+1$.

Finally, for each number c in range 1-9, all scores are averaged $scores[c]/cnt[c]$. The number c with the highest averaged score is the winner. The decision of the classifier is then compared with the known number thought stored in the metadata.

The results of classification available in Table 9.6 are sorted by classification success rate. Stacked autoencoders outperformed other classifiers. Comparatively good performance was

Table 9.6: Classification performance using the testing set. Each success rate was computed as a proportion of correctly guessed thought numbers to the number of all testing datasets (i.e. 206).

Method	Classification success rate
SAE	79.4%
MLP	76.7%
LDA	75.6%
SVM	73.7%
Human Expert [Author11]	64.4%

achieved despite lower feature vector dimensionality. Because of this decrease in feature vector dimensionality, to avoid overtraining, the architecture of the SAE had to be simplified when compared to the SAE as described in Subsection 8.2. However, the algorithm still maintained both acceptable generalization and capability to distinguish complex target to non-target differences.

10 Conclusion

The main aim of this thesis was to evaluate benefits of using neural networks for classification in P300-based brain-computer interfaces (BCIs). The first part of the thesis introduced the theoretical background of BCIs and related fields, such as electroencephalography, event-related potentials, and algorithms for signal feature extraction. Subsequently, an overview of classification methods was presented. Traditionally, linear classifiers, such as Linear Discriminant Analysis or Support Vector Machines are used in BCIs because they are easy to implement, fast and yield good results, especially when trained on a limited amount of low-dimensional feature vectors. However, they commonly suffer from the curse of dimensionality that limits their usability for high-dimensional and complex feature vectors. Moreover, they can perform poorly when applied to linearly inseparable feature vectors. Neither unsupervised learning models, such as SOM, or ART, nor deep learning algorithms were extensively used for the P300 classification. SOM and ART models are not suitable in their original unsupervised form, since they only perform clustering with no information from the supervisor. However, I performed an evaluation of existing supervised modifications of those models, designed my own modifications where applicable, and empirically found parameters that

yielded the best results.

To integrate and compare results that I published in [Author6] and in [Author7], I designed an adaptive and data-driven feature extraction method based on statistical comparison between target and non-target time windows across EEG channels. Based on a pre-defined significance level, 83-dimensional feature vectors were created that were subsequently used for training, validation and testing. As an additional benefit, I designed my classification methods to work universally for any testing subject, without need for individual training.

Most classification models proposed for P300 BCIs were able to classify the P300 data successfully. However, using the proposed feature extraction, neither SOM-based, nor ART-based methods reached the performance achieved in [Author6]. The most likely explanation is that these models are not resilient to the curse of dimensionality. The LASSO model yielded the highest accuracy from SOM-based methods (59.33 % on average). Based on the achieved results, neither ART-based models, nor SOM-based can be recommended for P300 BCIs at the present state.

Apart from ART and SOM neural network models, I also explored deep learning models. Deep neural networks have emerged in recent years and for many applications, they are known to outperform not only standard neural network training models, but also other state-of-the-art classifiers. I decided to implement stacked autoencoders because of their useful properties: they use real-valued inputs and combine unsupervised and supervised learning to maximize classification accuracy. Indeed, the results achieved showed that with the increasing dimensionality of feature vectors, the relative benefits of stacked autoencoders over both LDA and MLP increased. In my opinion, their advantage over LDA is caused by their lack of the curse of dimensionality. The advantage over MLP is based on unsupervised pre-training that improves neural network weight updates, especially for first layers. The main disadvantage of these neural networks is relatively time-consuming training, but for BCIs with reasonable response times in on-line mode, training times do not matter much. Therefore, consistently with the experience of authors in other machine learning fields, I would recommend using stacked autoencoders for P300-based BCIs. However, despite the improvement, average single trial classification accuracy stayed slightly below 70 % improved up to 78 % using epoch averaging. Both experimental results and existing deep learning

literature highlight the need for larger amount of training data, especially when developing a universal P300 classifier that can successfully perform for unknown subjects without further training. However, such amount of data is currently not publicly available, suggesting a challenging task for the future.

Moreover, this thesis presents a Java-based BCI project that is currently under development. The application contains implemented methods for deep learning, as well as traditional linear classifiers. In its present state, it can be used to classify the data from the 'Guess the number' experiment. However, its modification will be able to operate under various BCI paradigms. Based on both on-line and off-line data from this experiment, I managed to confirm the improvement of SAE in comparison with other classification algorithms.

10.1 Summary of achievements and evaluation of thesis goals

The main goals of thesis are specified in Section 6.4. The following points summarize my achievements regarding those goals (the most important achievements are highlighted in bold):

- After an extensive evaluating of the literature and related experiments, the feature extraction based on the Windowed means paradigm was proposed and applied in subsequent data processing. **To find suitable time windows and related EEG channels, I proposed a statistics-based data-driven method.**

- Neural networks based on self-organizing maps (SOM), Adaptive Resonance Theory (ART) and stacked autoencoders were used for the classification of event-related potential data. Since both SOM and ART models in their original form only perform unsupervised learning, **I proposed two supervised modifications of the SOM algorithm (SOM1 and SOM2)**. Other already existing but rarely used supervised modifications LASSO and Simplified fuzzy ARTMAP were also used. For comparison, the data were split into training, validation and testing sets. Parameter settings was discussed. Neither SOM, nor ART-based algorithms outperformed state-of-the-art. In contrast, **stacked autoencoders were able to outperform both multi-layer perceptron and LDA**. They do not suffer from the curse of dimensionality and un-

supervised pre-training and improved regularization contributed to better classification performance. These results are consistent with the experience of other researcher in deep learning applications. **Both the data used [Author5] and related Matlab algorithms [73] are publicly available to ensure the reproducibility of the algorithms described.**

- The algorithms were implemented in Matlab and Java. **Furthermore, the Java-based project Guess the number allows to run the algorithms both in an on-line BCI and as an off-line classification test. In the off-line test based on over 200 datasets, it was confirmed that stacked autoencoders outperform both traditional classifiers and the human expert that evaluated data by observing the waveforms.** The project is still under development. Finally, stacked autoencoders were also successfully used for the P300 data validation [Author13].

10.2 Ongoing projects and future work

10.2.1 The BASIL Czech-Bavarian project

The models proposed (especially stacked autoencoders) are used for on-line BCIs applied to both healthy people and patients in hospitals that may suffer from various neuromuscular disorders. The BASIL (Brainwave driven assistance system for motor-impaired people) Czech-Bavarian project running between 2016 and 2019 is one of the best opportunities to apply the advanced machine learning techniques with a possibly promising impact in the region. The Bavarian Sensorik company is also involved in the project and helps with the development of hardware. More specifically, a BCI device is planned that can also be used in home settings and allows the patients to control their environment (with actions including, but not limited to, opening the window, turning on the TV, calling for medical help, etc.). One of the experimental protocols tested is depicted in Fig. 10.45. The device consists of a monitor that exposes the user to visual stimuli (e.g. P300-based or SSVEP-based), optional headphones for those who could have problems with observing visual stimuli, a wireless headset for measuring brain signals, a portable computer (such as Raspberry PI) for mixing stimuli and raw EEG data, pre-processing, feature extraction and classification. Moreover,

Figure 10.45: One of the tested visual P300-based stimulation protocols for the Czech-Bavarian project. The subject chooses one of the actions (to open the door, to eat, to ask for help, to switch the light on, to make a phone call, to turn on the radio, to use the toilet, to turn on the TV, to open the window). Subsequently, rows and columns randomly flash. The goal of the classification algorithm is to find the row and the column with the P300 component and consequently, to find the requested action.

the computer sends feature vectors and other relevant data and metadata to the cloud. In the cloud, classification algorithms are stored and trained — both universal and personalized for different users. The classifier may then be requested from the computer. Subsequently, its parameters (such as deep neural network weights) can be received by the cloud computer. The simplified architecture of the system in progress is depicted in Fig. 10.46.

10.2.2 Predicting navigational decisions through P300 event-related potentials using BCI

This study was designed during my research stay (April to May 2017) at the Division of Psychology, University of Stirling in Scotland. Besides me, a Ph.D. student Simon Ladouce also participates. A visual P300-based brain-computer interface is tested to predict decisions during free navigation in the real-world. Eight participants were equipped with a mobile

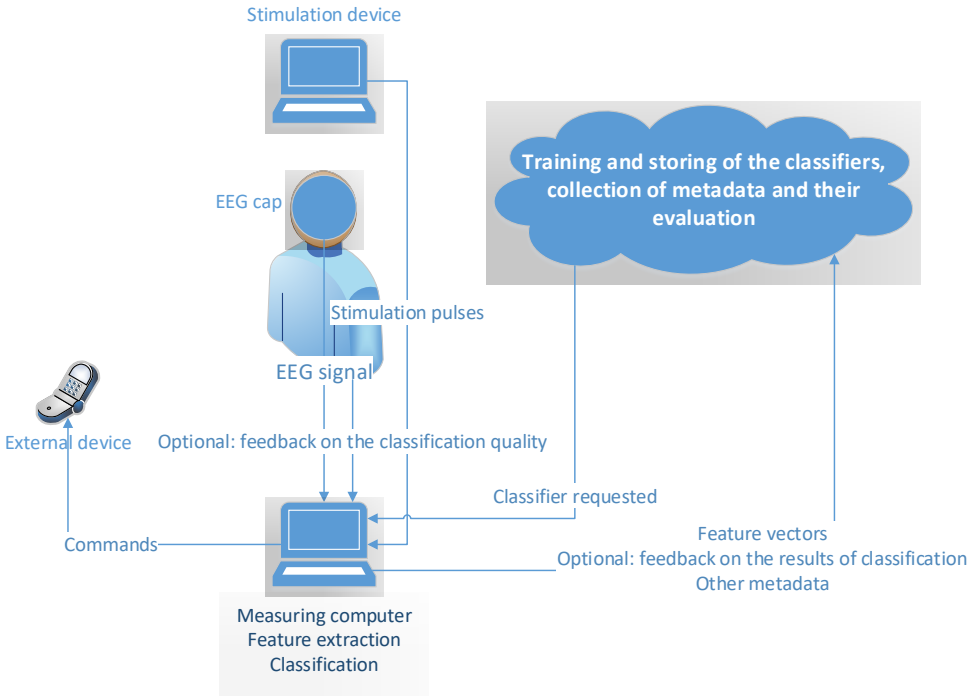


Figure 10.46: The simplified architecture of the planned BCI system developed under the BASIL project. The subject has an intention of an action to be performed by the external device. The monitor presents visual action-related stimuli to the subject. In the measuring computer, the EEG signal captured using a wireless EEG cap is mixed with the related stimuli markers and the resulting split EEG trials are further processed and transformed into feature vectors. The action requested by the user is detected by a deep learning-based classifier and is executed by the external device. Methods for collecting feedback from the user about the correctness of the action performed are currently under discussion. Furthermore, beyond a traditional BCI architecture, the feature vectors and the related feedback and other metadata are sent to the cloud computer. The cloud computer stores the data, trains classifiers, stores different classifiers for different subjects, analyzes both data and metadata, and sends classifier parameters to the local device on demand.

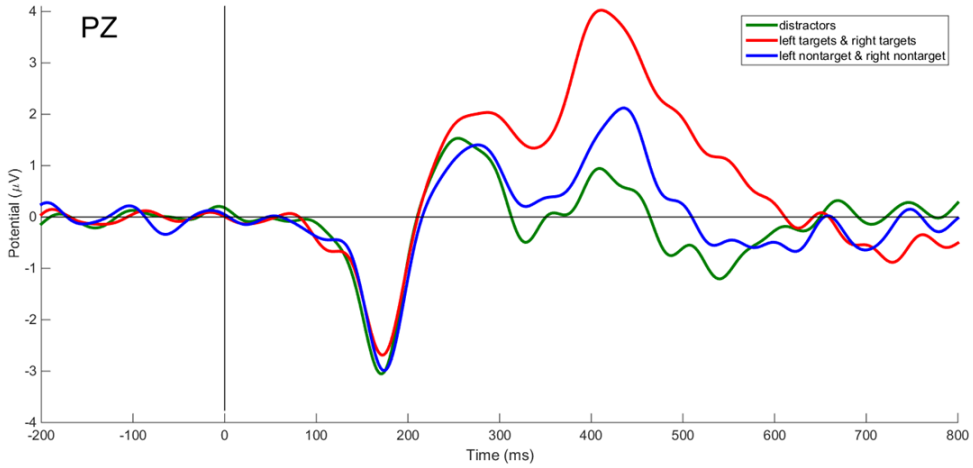


Figure 10.47: Grand averages for left arrows, right arrows, and distractor stimuli are shown. To produce the figure, the data from the training session were collected from all participants, cleaned using band-pass filtering and Independent Component Analysis, and subsequently averaged. For each participant, 150 left arrows, 150 right arrows, and 300 distractors were used. Although non-target arrow stimuli also produce an ERP response, it is apparently weaker than the reaction to target stimuli. Therefore, classifiers should be able to discriminate between them. Apparently, building P300-based BCIs during free navigation should be possible.

EEG system and presented with visual stimuli (left and right arrows) on a tablet as they were navigating through the corridors of a university campus. Event-related potentials features were extracted off-line from a first training session to train classifiers to discriminate target from non-target feature vectors during a follow-up testing session. Variations of the visual evoked ERP paradigm used (with/out distractors) are to be compared in terms of performance (accuracy, precision and recall). Methodological considerations and future directions for the application of BCI in the real-world will be discussed. In Figure 10.47, grand averages for left arrows, right arrows, and distractor stimuli are depicted.

10.2.3 Ideas for future research

Since stacked autoencoders performed well in P300 classification tasks, other models from deep learning category seem worth evaluating too. For example, stacked denoising autoencoders could be able to better handle noise in the EEG data. Convolution neural networks could be useful as well, although in my experience, they require more training data than we

can currently obtain.

Although classification algorithms are important for the performance of P300-based BCIs, there are also other ways to improve these systems. For example, in [87], mismatch negativity paradigm was used instead of classic oddball paradigm. The authors claim that this modification increases amplitudes of the ERP components (such as P300 and N200) and boosts the accuracy and bit-rate of P300 BCIs. Furthermore, in [88] and in [89], face flashing paradigm based on flashing characters with superimposed pictures of well-known faces significantly outperformed the commonly used character flashing paradigm. Alternatively, in [90], multimodal BCIs were proposed that significantly outperform classic P300 BCIs. Moreover, as reported in [91], the reliability of P300-based BCIs depends on environment during the experiment: noisy everyday environment slightly lowers the accuracy of P300 BCIs. The health state of the participant affected the accuracy even more. Participants with neuromuscular disorder who need BCI the most, achieved the lowest accuracy. This problem, however, can be partially solved by tuning the BCI parameters, such as using longer inter-stimulus intervals [91]. In future work, the state-of-the-art approaches for BCIs may be integrated to further improve performance, especially when applied to paralyzed subjects.

References

[1] T. F, "Electroencephalography: Basic principles, clinical applications and related fields," *Archives of Neurology*, vol. 40, no. 3, 1983. [Online]. Available: http://dx.doi.org/10.1001/archneur.1983.04050030085025

[2] S. Klein and B. M. Thorne, *Biological psychology.* New York, USA: Worth., 2006.

[3] S. Sanei and J. A. Chambers, *EEG Signal Processing.* Wiley-Interscience, Sep 2007. [Online]. Available: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\%5C\&path=ASIN/0470025816

[4] T. W. Picton, O. G. Lins, and M. Scherg, "The recording and analysis of event-related potentials." in *Handbook of Neuropsychology*, F. Boller and J. Grafman, Eds. Amsterdam: Elsevier, 1995, vol. 10, pp. 3–73.

[5] S. Luck, *An introduction to the event-related potential technique*, ser. Cognitive neuroscience. MIT Press, 2005. [Online]. Available: http://books.google.cz/books?id=J_QgAQAAIAAJ

[6] D. S. Tan and A. Nijholt, Eds., *Brain-Computer Interfaces - Applying our Minds to Human-Computer Interaction*, ser. Human-Computer Interaction Series. Springer, 2010.

[7] R. T. Abresch, J. J. Han, and G. T. Carter, "Rehabilitation management of neuromuscular disease: the role of exercise training." *J Clin Neuromuscul Dis*, vol. 11, no. 1, pp. 7–21, 2009. [Online]. Available: http://www.biomedsearch.com/nih/Rehabilitation-management-neuromuscular-disease-role/19730017.html

[8] D. J. McFarland and J. R. Wolpaw, "Brain-computer interfaces for communication and control," *Commun. ACM*, vol. 54, no. 5, pp. 60–66, May 2011. [Online]. Available: http://doi.acm.org/10.1145/1941487.1941506

[9] J. Wolpaw, N. Birbaumer, W. Heetderks, D. McFarland, P. Peckham, G. Schalk, E. Donchin, L. Quatrano, C. Robinson, and T. Vaughan, "Brain-computer interface tech-

nology: a review of the first international meeting," *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 2, pp. 164–173, 2000.

[10] F. Beverina, G. Palmas, S. Silvoni, F. Piccione, and S. Giove, "User adaptive BCIs: SSVEP and P300 based interfaces." *PsychNology Journal*, vol. 1, no. 4, pp. 331–354, 2003. [Online]. Available: http://dblp.uni-trier.de/db/journals/psychnology/psychnology1.html

[11] J. J. Vidal, "Real-time detection of brain events in EEG," *Proceedings of the IEEE*, vol. 65, no. 5, pp. 633–641, 1977. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1454811

[12] S. Yin, Y. Liu, and M. Ding, "Amplitude of Sensorimotor Mu Rhythm Is Correlated with BOLD from Multiple Brain Regions: A Simultaneous EEG-fMRI Study," *Frontiers in Human Neuroscience*, vol. 10, p. 364, 2016.

[13] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw, "BCI2000: a general-purpose brain-computer interface (BCI) system." *IEEE transactions on bio-medical engineering*, vol. 51, no. 6, pp. 1034–1043, Jun. 2004. [Online]. Available: http://dx.doi.org/10.1109/tbme.2004.827072

[14] U. Hoffmann, J.-M. Vesin, T. Ebrahimi, and K. Diserens, "An Efficient P300-based Brain-Computer Interface for Disabled Subjects," *Journal of neuroscience methods*, vol. 167, no. 1, pp. 115–125, Jan. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.jneumeth.2007.03.005

[15] L. A. Farwell and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalography and Clinical Neurophysiology*, vol. 70, no. 6, pp. 510–523, Dec. 1988. [Online]. Available: http://dx.doi.org/10.1016/0013-4694(88)90149-6

[16] B. Blankertz, S. Lemm, M. S. Treder, S. Haufe, and K.-R. Müller, "Single-trial analysis and classification of ERP components - a tutorial," *NeuroImage*, vol. 56, no. 2, pp. 814–825, 2011.

[17] S. Amiri, A. Rabbi, L. Azinfar, and R. Fazel-Rezai, "A review of P300, SSVEP, and hybrid P300/SSVEP brain- computer interface systems," in *Brain-Computer Interface Systems - Recent Progress and Future Prospects*, R. Fazel-Rezai, Ed. Rijeka: InTech, 2013, ch. 10. [Online]. Available: http://dx.doi.org/10.5772/56135

[18] R. C. Panicker, S. Puthusserypady, and Y. Sun, "An asynchronous P300 BCI with SSVEP-based control state detection," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 6, pp. 1781–1788, June 2011.

[19] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for EEG-based brain-computer interfaces," *Journal of neural engineering*, vol. 4, no. 2, Jun. 2007. [Online]. Available: http://dx.doi.org/10.1088/1741-2560/4/2/R01

[20] J. N. Mak, Y. Arbel, J. W. Minett, L. M. McCane, B. Yuksel, D. Ryan, D. Thompson, L. Bianchi, and D. Erdogmus, "Optimizing the P300-based brain-computer interface: current status, limitations and future directions," *Journal of Neural Engineering*, vol. 8, no. 2, pp. 025 003+, Apr. 2011. [Online]. Available: http://dx.doi.org/10.1088/1741-2560/8/2/025003

[21] D. L. Olson and D. Delen, *Advanced Data Mining Techniques*, 1st ed. Springer Publishing Company, Incorporated, 2008.

[22] L. Vareka, "Neural networks in ERP signal processing (in Czech)," Diploma Thesis, University of West Bohemia, 2011.

[23] Z. Cashero, *Comparison of EEG Preprocessing Methods to Improve the Performance of the P300 Speller*. Proquest, Umi Dissertation Publishing, 2012.

[24] T. Rondik, "Methods of erp signals processing (in czech)," Diploma Thesis, University of West Bohemia, 2010.

[25] G. Kaiser, *A friendly guide to wavelets*. Cambridge, MA, USA: Birkhauser Boston Inc., 1994.

[26] E. Bartnik, K. Blinowska, and P. Durka, "Single evoked potential reconstruction by means of wavelet transform," *Biological Cybernetics*, vol. 67, pp. 175–181, 1992, 10.1007/BF00201024. [Online]. Available: http://dx.doi.org/10.1007/BF00201024

[27] R. Quiroga and H. Garcia, "Single-trial event-related potentials with wavelet denoising," *Clinical Neurophysiology*, vol. 114, no. 2, pp. 376–390, Feb. 2003. [Online]. Available: http://dx.doi.org/10.1016/S1388-2457(02)00365-6

[28] J. Ramirez-Cortes, V. Alarcon-Aquino, G. Rosas-Cholula, P. Gomez-Gil, and J. Escamilla-Ambrosio, "Anfis-based P300 rhythm detection using wavelet feature extraction on blind source separated EEG signals," in *Intelligent Automation and Systems Engineering, LNEE, Vol. 103*. Springer New York, 2011.

[29] T. Rondik and J. Ciniburk, "Comparison of various approaches for P3 component detection using basic methods for signal processing," in *BMEI*, 2011, pp. 698–702.

[30] R. M. Tomas Rondik, Jindrich Ciniburk and P. Mautner, "ERP components detection using wavelet transform and matching pursuit algorithm," in *Applied Electronics 2011*, 2011, pp. 1–4.

[31] S. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, pp. 3397–3415, 1993.

[32] S. E. Ferrando, L. A. Kolasa, and N. Kovačević, "Algorithm 820: A flexible implementation of matching pursuit for gabor functions on the interval," *ACM Trans. Math. Softw.*, vol. 28, no. 3, pp. 337–353, Sep. 2002. [Online]. Available: http://doi.acm.org/10.1145/569147.569151

[33] P. Durka and K. Blinowska, "Analysis of EEG transients by means of matching pursuit," *Annals of Biomedical Engineering*, vol. 23, no. 5, pp. 608–611, Sep. 1995. [Online]. Available: http://dx.doi.org/10.1007/bf02584459

[34] P. Comon, "Independent component analysis, a new concept?" *Signal Process.*, vol. 36, no. 3, pp. 287–314, Apr. 1994. [Online]. Available: http://dx.doi.org/10.1016/0165-1684(94)90029-9

[35] N. Xu, X. Gao, B. Hong, X. Miao, S. Gao, and F. Yang, "BCI Competition 2003–Data set IIb: enhancing P300 wave detection using ICA-based subspace projections for BCI applications." *IEEE transactions on bio-medical engineering*, vol. 51, no. 6, pp. 1067–1072, Jun. 2004. [Online]. Available: http://dx.doi.org/10.1109/tbme.2004.826699

[36] P. L. Nunez and R. Srinivasan, *Electric Fields of the Brain: The Neurophysics of EEG, 2nd Edition*, 2nd ed. Oxford University Press, USA, Dec. 2005. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/019505038X

[37] L. P. Clay, L. C. Parra, C. D. Spence, Adam, and C. Paul Sajda, "Recipes for the linear analysis of EEG," *NeuroImage*, vol. 28, pp. 326–341, 2005. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.62.919

[38] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Comput.*, vol. 7, no. 6, pp. 1129–1159, Nov. 1995. [Online]. Available: http://dx.doi.org/10.1162/neco.1995.7.6.1129

[39] S. Makeig, T.-P. Jung, A. J. Bell, D. Ghahremani, and T. J. Sejnowski, "Blind separation of auditory event-related brain responses into independent components," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 94, no. 20, pp. 10 979–10 984, 1997.

[40] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Third Edition*, 3rd ed. Academic Press, Mar. 2006. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0123695317

[41] K. Fukunaga, *Introduction to Statistical Pattern Recognition, Second Edition (Computer Science & Scientific Computing)*, 2nd ed. Academic Press, Oct. 1990. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0122698517

[42] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995. [Online]. Available: http://dx.doi.org/10.1007/BF00994018

[43] D. J. Krusienski, E. W. Sellers, F. Cabestaing, S. Bayoudh, D. J. McFarland, T. M. Vaughan, and J. R. Wolpaw, "A comparison of classification techniques for the P300 Speller," *Journal of Neural Engineering*, vol. 3, no. 4, pp. 299–305, Dec. 2006. [Online]. Available: http://dx.doi.org/10.1088/1741-2560/3/4/007

[44] L. Fausett, Ed., *Fundamentals of neural networks: architectures, algorithms, and applications*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1994.

[45] J. Yang. (2017, 2) Relu and softmax activation functions. Accessed: 2018-02-11. [Online]. Available: https://github.com/Kulbear/deep-learning-nano-foundation/wiki/ReLU-and-Softmax-Activation-Functions

[46] M. F. Moeller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, no. 4, pp. 525 – 533, 1993. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608005800565

[47] M. A. Nielsen, "Neural networks and deep learning," 2018. [Online]. Available: http://neuralnetworksanddeeplearning.com/

[48] L. Deng and D. Yu, "Deep learning: Methods and applications," *Foundations and Trends in Signal Processing*, vol. 7, pp. 197–387, 2013. [Online]. Available: http://dx.doi.org/10.1561/2000000039

[49] L. Arnold, S. Rebecchi, S. Chevallier, and H. Paugam-Moisy, "An introduction to deep-learning," in *Advances in Computational Intelligence and Machine Learning, ESANN'2011*, Apr. 2011, pp. 477–488. [Online]. Available: http://liris.cnrs.fr/publis/?id=5173

[50] H. Cecotti and A. Graser, "Convolutional neural networks for P300 detection with application to brain-computer interfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 433–445, March 2011.

[51] A. de Giorgio, "A study on the similarities of deep belief networks and stacked autoencoders," Master's thesis, KTH, School of Computer Science and Communication (CSC), 2015.

[52] A. Ng, J. Ngiam, C. Y. Foo, Y. Mai, and C. Suen, "UFLDL Tutorial," *http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial*, 2010.

[53] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. MIT Press, 2007, pp. 153–160. [Online]. Available: http://papers.nips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.pdf

[54] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1953039

[55] MATLAB, *MATLAB version 8.6.0 (R2015b) - Neural Network Toolbox*. Natick, Massachusetts: The MathWorks Inc., 2015.

[56] J. P. Kulasingham, V. Vibujithan, and A. C. D. Silva, "Deep belief networks and stacked autoencoders for the p300 guilty knowledge test," in *2016 IEEE EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, Dec 2016, pp. 127–132.

[57] T. Kohonen, *Self-organization and associative memory: 3rd edition*. New York, NY, USA: Springer-Verlag New York, Inc., 1989.

[58] T. Rondik and P. Mautner, "Clustering of Gabor atoms describing event-related potentials," in *HEALTHINF 2013 International Conference on Health Informatics*, 2013, pp. 309–314.

[59] S. L. Joutsiniemi, S. Kaski, and A. T. Larsen, "Self-organizing map in recognition of topographic patterns of EEG spectra," *IEEE Transactions on Biomedical Engineering*, vol. 42, pp. 1062–1068, 1995.

[60] S. Midenet and A. Grumbach, "Learning ASsociations by self-organization: The LASSO model," *Neurocomputing*, vol. 6, no. 3, pp. 343 – 361, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0925231294900698

[61] G. A. Carpenter and S. Grossberg, "The handbook of brain theory and neural networks," M. A. Arbib, Ed. Cambridge, MA, USA: MIT Press, 1998, ch. Adaptive resonance theory (ART), pp. 79–82. [Online]. Available: http://dl.acm.org/citation.cfm?id=303568.303586

[62] G. Carpenter, S. Grossberg, N. Markuzon, J. Reynolds, and D. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *Neural Networks, IEEE Transactions on*, vol. 3, no. 5, pp. 698–713, Sep 1992.

[63] G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network," *Neural Networks*, vol. 4, pp. 565–588, 1991.

[64] R. Palaniappan, R. Paramesran, S. Nishida, and N. Saiwaki, "A new brain-computer interface design using fuzzy ARTMAP," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 10, no. 3, pp. 140–148, Sept 2002.

[65] M.-T. Vakil-Baghmisheh and N. Pavešić, "A fast simplified Fuzzy ARTMAP network," *Neural Process. Lett.*, vol. 17, no. 3, pp. 273–316, Jun. 2003. [Online]. Available: http://dx.doi.org/10.1023/A:1026004816362

[66] B. Blankertz, K. Muller, G. Curio, T. Vaughan, G. Schalk, J. Wolpaw, A. Schlogl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schroder, and N. Birbaumer, "The BCI competition 2003: progress and perspectives in detection and discrimination of EEG single trials," *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 1044–1051, 2004.

[67] N. Liang and L. Bougrain, "Non-identity Learning Vector Quantization applied to evoked potential detection," in *Deuxième conférence française de Neurosciences Computationnelles, "Neurocomp08"*, Marseille, France, Oct. 2008, iSBN : 978-2-9532965-0-1. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00331590

[68] M. Kaur, P. Ahmed, A. Soni, and M. Q. Rafiq, "Wavelet transform use for P300 signal clustering by self-organizing map," *International Journal of Scientific and*

Engineering Research, vol. 4, no. 11, pp. 1631–1638, 11 2013. [Online]. Available: https://pdfs.semanticscholar.org/48f7/ce67250f50639b329f55ef7c9f79adef8f25.pdf

[69] C. Morales, C. M. Held, P. A. Estevez, C. A. Perez, S. Reyes, P. Peirano, and C. Algarin, "Single trial P300 detection in children using expert knowledge and SOM," *Conf Proc IEEE Eng Med Biol Soc*, vol. 2014, pp. 3801–3804, 2014.

[70] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas, "Self-organizing map in Matlab: the SOM toolbox," in *In Proceedings of the Matlab DSP Conference*, 1999, pp. 35–40.

[71] Y. Bengio and Y. LeCun, "Scaling learning algorithms towards AI," in *Large Scale Kernel Machines*, L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, Eds. MIT Press, 2007. [Online]. Available: http://www.iro.umontreal.ca/~lisa/pointeurs/bengio+lecun_chapter2007.pdf

[72] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *J. Mach. Learn. Res.*, vol. 11, pp. 625–660, Mar. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1756025

[73] L. Vareka, "The p300 classifier comparator," 2018. [Online]. Available: https://github.com/NEUROINFORMATICS-GROUP-FAV-KIV-ZCU/P300ClassifierComparator

[74] Brain Products GmbH, "BrainVision Recorder 1.2," Jul. 2012. [Online]. Available: http://www.brainproducts.com/productdetails.php?id=21

[75] K. Dudacek, P. Mautner, R. Moucek, and J. Novotny, "Odd-ball protocol stimulator for neuroinformatics research," in *Applied Electronics (AE), 2011 International Conference on*, Sept., pp. 1–4.

[76] J. Polich, "Updating P300: an integrative theory of P3a and P3b." *Clinical neurophysiology*, vol. 118, no. 10, pp. 2128–2148, Oct. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.clinph.2007.04.019

[77] P. Jezek and R. Moucek, "System for EEG/ERP data and metadata storage and management," *Neural Network World*, vol. 22, no. 3, pp. 277–290, 2012. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84864866156&partnerID=40&md5=358d6c8f8b31b89255ee48c2273a85f5

[78] A. Delorme and S. Makeig, "EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics including Independent Component Analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[79] E. Akbas, "Simplified Fuzzy ARTMAP neural network," Jul. 2006. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/11721-simplified-fuzzy-artmap-neural-network

[80] A. Delorme, T. Mullen, C. Kothe, Z. A. Acar, N. Bigdely-Shamlo, A. Vankov, and S. Makeig, "EEGLAB, SIFT, NFT, BCILAB, and ERICA: New tools for advanced EEG processing," *Intell. Neuroscience*, vol. 2011, pp. 10:10–10:10, Jan. 2011. [Online]. Available: http://dx.doi.org/10.1155/2011/130714

[81] L. Zamparo and Z. Zhang, "Deep autoencoders for dimensionality reduction of high-content screening data." *CoRR*, vol. abs/1501.01348, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1501.html#ZamparoZ15

[82] A. Sobhani, "P300 classification using deep belief nets," Master's thesis, Colorado State University, 2014.

[83] E. W. Sellers, D. J. Krusienski, D. J. McFarland, T. M. Vaughan, and J. R. Wolpaw, "A P300 event-related potential brain-computer interface (BCI): the effects of matrix size and inter stimulus interval on performance," *Biol Psychol*, vol. 73, no. 3, pp. 242–252, Oct 2006.

[84] C. E. Lakey, D. R. Berry, and E. W. Sellers, "Manipulating attention via mindfulness induction improves P300-based brain-computer interface performance," *J Neural Eng*, vol. 8, no. 2, p. 025019, Apr 2011.

[85] L. Vareka, T. Prokop, J. Stebetak, and R. Moucek, "Guess the number (GitHub repository)," 2015. [Online]. Available: https://github.com/NEUROINFORMATICS-GROUP-FAV-KIV-ZCU/guess_the_number

[86] A. Gibson, C. Nicholson, J. Patterson, M. Warrick, A. D. Black, V. Kokorin, S. Audet, and S. Eraly, "Deeplearning4j: Distributed, open-source deep learning for Java and Scala on Hadoop and Spark," 5 2016. [Online]. Available: https://figshare.com/articles/deeplearning4j-deeplearning4j-parent-0_4-rc3_8_zip/3362644

[87] J. Jin, E. W. Sellers, S. Zhou, Y. Zhang, X. Wang, and A. Cichocki, "A P300 brain-computer interface based on a modification of the mismatch negativity paradigm." *Int. J. Neural Syst.*, vol. 25, no. 3, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/ijns/ijns25.html#JinSZZWC15

[88] T. Kaufmann, S. M. Schulz, A. Koblitz, G. Renner, C. Wessig, and A. Kubler, "Face stimuli effectively prevent brain-computer interface inefficiency in patients with neurodegenerative disease," *Clin Neurophysiol*, vol. 124, no. 5, pp. 893–900, May 2013.

[89] J. Jin, B. Z. Allison, Y. Zhang, X. Wang, and A. Cichocki, "An ERP-based BCI using an oddball paradigm with different faces and reduced errors in critical functions," *Int J Neural Syst*, vol. 24, no. 8, p. 1450027, Dec 2014.

[90] Y. Li, J. Pan, J. Long, T. Yu, F. Wang, Z. Yu, and W. Wu, "Multimodal BCIs: Target detection, multidimensional control, and awareness evaluation in patients with disorder of consciousness," *Proceedings of the IEEE*, vol. 104, no. 2, pp. 332–352, Feb 2016.

[91] P. Ortner, S. Putz, S. Bruckner, and Guger, "Accuracy of a P300 speller for different conditions: A comparison," in *Proceedings of the 5th International Brain-Computer Interface Conference, Graz, Austria*, 9 2011, Aufsatz / Paper in Tagungsband (referiert).

Author's publications

[Author1] L. Vareka and P. Mautner, "Off-line analysis of the P300 event-related potential using discrete wavelet transform," in *36th International Conference on*

Telecommunications and Signal Processing, TSP 2013, Rome, Italy, 2-4 July, 2013, 2013, pp. 569–572. [Online]. Available: http://dx.doi.org/10.1109/TSP.2013.6613998

[Author2] L. Vareka, "Matching pursuit for P300-based brain-computer interfaces," in *35th International Conference on Telecommunications and Signal Processing, TSP 2012, Prague, Czech Republic, July 3-4, 2012*, 2012, pp. 513–516. [Online]. Available: http://dx.doi.org/10.1109/TSP.2012.6256347

[Author3] L. Vareka and P. Mautner, "The event-related potential data processing using ART 2 network," in *The 5th International Conference on BioMedical Engineering and Informatics (BMEI 2012)*, 2012, pp. 467–471.

[Author4] ——, "Self-organizing maps for event-related potential data analysis," in *HEALTHINF 2014 - Proceedings of the International Conference on Health Informatics, ESEO, Angers, Loire Valley, France, 3-6 March, 2014*, 2014, pp. 387–392. [Online]. Available: http://dx.doi.org/10.5220/0004885103870392

[Author5] L. Vareka, P. Bruha, and R. Moucek, "Event-related potential datasets based on a three-stimulus paradigm," *GigaScience*, vol. 3, no. 1, p. 35, 2014. [Online]. Available: http://www.gigasciencejournal.com/content/3/1/35

[Author6] L. Vareka and P. Mautner, "Modications of unsupervised neural networks for single trial p300 detection," *Neural Network World*, vol. 28, pp. 1–16, 2018.

[Author7] ——, "Stacked autoencoders for the P300 component detection," *Frontiers in Neuroscience*, vol. 11, p. 302, 2017. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fnins.2017.00302

[Author8] ——, "Using the Windowed means paradigm for single trial P300 detection," in *2015 38th International Conference on Telecommunications and Signal Processing (TSP)*, July 2015, pp. 1–4.

[Author9] R. Moucek, P. Bruha, P. Jezek, P. Mautner, J. Novotny, V. Papez, T. Prokop, T. Rondik, J. Stebetak, and L. Vareka, "Software and hardware infrastructure

for research in electrophysiology," *Frontiers in Neuroinformatics*, vol. 8, p. 20, 2014. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fninf.2014.00020

[Author10] R. Moucek, L. Vareka, T. Prokop, J. Stebetak, and P. Bruha, "Event-related potential data from a guess the number brain-computer interface experiment on school children," *Scientific Data*, vol. 4, Mar 2017.

[Author11] L. Vareka, T. Prokop, J. Stebetak, and R. Moucek, "Guess the number - applying a simple brain-computer interface to school-age children," in *Proceedings of the 9th International Joint Conference on Biomedical Engineering Systems and Technologies - Volume 4: BIOSIGNALS*, 2016, pp. 263–270.

[Author12] L. Vareka, T. Prokop, P. Mautner, R. Moucek, and J. Stebetak, "Application of stacked autoencoders to P300 experimental data," in *Proceedings of the 16th International Conference on Artificial Intelligence and Soft Computing, ICAISC 2017*, 2017.

[Author13] R. Moucek, L. Hnojsky, L. Vareka, T. Prokop, and P. Bruha, "Experimental design and collection of brain and respiratory data for detection of driver's attention," in *Proceedings of the 10th International Joint Conference on Biomedical Engineering Systems and Technologies - HEALTHINF*, 2017.

[Author14] L. Vareka, P. Bruha, R. Moucek, P. Mautner, L. Cepicka, and I. Holeckova, "Developmental coordination disorder in children - experimental work and data annotation," *GigaScience*, vol. 2, no. 1, 2017. [Online]. Available: https://doi.org/10.1093/gigascience/gix002

[Author15] L. Vareka, P. Bruha, and R. Moucek, "Single channel eye-blinking artifacts detection," in *2013 International Conference on Applied Electronics*, Sept 2013, pp. 1–4.