Master's Thesis:

Designing a Brain Computer Interface Using an Affordable EEG Headset.

Author:
Patrick Teich

Patrick.Teich@fu-berlin.de

Supervisor: Prof. Dr. Raúl Rojas
Second examiner: Dr. Daniel Göhring

Berlin, January 3, 2015

Abstract

Brain computer interfaces (BCIs) are still not wide-spread. Compared to other means of control, they currently do not provide much usability. However, for highly motor-disabled people they are the only means of communication to the outside world. Using them to provide such people a way to command some devices, these devices could expand the abilities of a person. Thus, giving highly disabled people back some of their autonomy.

Considering the usually high price of recording technology necessary to build BCIs, this work examines the question, whether it is possible to also employ a consumer-priced EEG headset. Towards this goal an Emotiv EPOC EEG headset is used to design and build a proof-of-concept system that realises a BCI for arbitrary control over some device.

As a result, a prototype has been implemented that comprises analysis, transformation, and classification through a linear discriminant analysis of the recorded signals and that allows device control through the means of brain signals.

In addition to the evaluation whether the concept is working, a number of possible further improvements and considerations are given in order to achieve a more mature state.

Declaration of Academic Integrity

I hereby confirm that the present thesis

Designing a Brain Computer Interface Using an Affordable EEG Headset

is solely my own work and that if any text passages or diagrams from books, papers, the Web or other sources have been copied or in any other way used, all references – including those found in electronic media – have been acknowledged and fully cited.

Berlin, January 3, 2015

_____

Patrick Teich

Acknowledgements

At first I would like to thank Prof. Dr. Raúl Rojas and the Department of Mathematics and Computer Science of the Free University Berlin to have made this work possible as conclusion of my study. Further, I want to thank Dr. Daniel Göhring for readily examining it.
Great thanks go to Omar Mendoza Montoya and Dr. Adalberto Llarena for giving me hints and very helpful advice and feedback throughout the course of this work. Special thanks go to all the subjects, who provided me with valuable data and helpful hints. The last great thanks are for my friend and greatest reviewer, who also initially brought me into this project and therefore made this work possible.

Contents

1 Introduction

Brain computer interfaces (BCIs) are an interesting way to control a device. After several decades of research and many advances particularly made in recent years, we are still far from having BCI-applications used for daily tasks. Mainly they are applied for medical purposes and in research. One reason for this are their still extremely limited capabilities, far lower than those of common human device interfaces like button-based, touch-based, voice-recognition-based, or gesture recognition-based interfaces. However, opposed to these, BCIs have one really attractive property: They are the only known human device interfaces that do not require muscle movements. Therefore, BCIs are mostly beneficial to people, who lost the control over all muscles, that otherwise could have been used as means of communication. However, provided a person is able to employ at least one muscle to communicate, at our current state of technology, muscle-based interfaces are preferable, due to the lower information transfer rate of BCIs [1] Nonetheless, there are people like those affected by total locked-in syndrome, for whom BCIs offer the possibility to communicate with their environment and especially with assisting devices, that would increase a person's independence.

For everyone else, who would like to use BCIs, there, fortunately, exist already some consumer-price (affordable) devices, which would allow to build a BCI upon. These are so-called EEG-headsets, which can be used for brain signal recording by doing Electroencephalography (EEG), so a computer can receive data from a user's brain. Unfortunately, there are no serious applications due to the poor quality of signals those devices provide. They are usually used for simple games. The other common usage for them is research. The interesting challenge is to create reliable interfaces based on such recording devices, so more people can benefit from them.

However, let us first concentrate on people gravely in need for a BCI-based system. Which possibilities could a BCI provide? Let us assume we have an autonomous wheelchair, which already has the ability to navigate on itself and find its path to a given goal. In order to command it to a certain goal, different means of input have been added. As there are voice, the use of a game controller, or a control stick. For highly disabled people, however, these interfaces are of no use. They would need something they could use without muscle movements — they would need a BCI [1].
With that, such people would regain a part of their independence in mobility through the wheelchair. Through the BCI they would gain the ability to command. And that would be, of course, not limited to that wheelchair. It could easily be extended to an arbitrary computerised device. That would be a huge benefit.

Further, in this section the work's background is presented, as well as the problem to solve, related work, its goal and requirements to fulfil, and lastly, a short overview over the structure of BCIs is given.

1.1 Background

The present work was created within the "Intelligent Systems and Robotics" work group. It stems from the idea to provide people with locked-in syndrome or similar diseases with new independent

mobility. Also, it expands a small showcase "brain control" for the group's autonomous car and also the wheelchair [2]. It employs the Emotiv EPOC EEG headset[1] as a basis to design a brain computer interface, that is one of the few highly affordable EEG headsets available. Since development of a BCI is a demanding task, and without much prior knowledge of that field, my research concentrated at first on creating a simple proof-of-concept system.

Building intelligent systems is a task, where machines shall be provided with human-like abilities. One human-like ability is to recognise reoccurring patterns within a larger context of information. Machine learning and especially its sub-branch pattern recognition are the disciplines within which is dealt providing machines that ability. There, systems have been created that are able to "learn" patterns given to them in then recognise learned patterns in unknown data. Others discover patterns by grouping (or clustering) data. The latter are using so-called "Unsupervised learning" methods, where they do not receive any additional hints about existing patterns in the data. The former are using "Supervised learning" methods for which the data contains additional labels that mark the patterns to learn. They train how to recognise a certain pattern from one data point. After training, if provided with an unknown data point, they are able to recognise into which pattern the data points fits best [19]. I employ a supervised learning method during my work in order to recognise patterns in brain signals, such that that may be used to control a device.

Within these methods of supervised learning there are two kinds. Regression, where a function is estimated from the training data and then used for prediction of a continuous value dependent on an unknown data point. And classification, where a so-called classifier is trained to be able to assign an unknown data point to a pattern class (a discrete value) [19].
I decided to use classification, which means given a certain amount of EEG data from a user, a classifier decides which type of intention the user seemed to have.

The EPOC EEG headset already comes with a certain classifier that I first tried to use. In section 3.1 I subsume my efforts and reasons for building an own classifier as part of the BCI-design.

Within BCI research, systems are built upon classification or regression [1]. While the fundamental structure is always about the same, it differs e.g. in used recording technology, signal of interest, or signal translation method. (See section 1.5 for more details.)

I am using EEG-signals recorded using the above EEG headset and from these I do not constrain myself much to particular EEG signals, rather than using and analysing a whole spectrum of the provided data. Also, I limited myself to evaluating a binary classifier as basis for the final BCI. The classifier is also applied to the BCI competition data IV[2] in order to see whether my methods work on data that has somewhat proven quality.

Further, as applications of BCI's in general Nicolas et. al. [1] name word processors, brain control of wheelchairs or neuroprostheses, and games as examples.

[1] *http://emotiv.com/*

[2] *http://www.bbci.de/competition/iv/*

1.2 Problem

The problem of this work can be stated in short as, given the affordable Emotiv EPOC EEG headset, is it possible to create a brain computer interface upon that reliable enough to enable a user to command an arbitrary computerised device?

What BCIs do is to distinguish between different brain activity patterns and translating them into certain commands or actions. Depending on the quality of the employed recording device, the retrieved data represents such brain activity related signals with more or less noise. Such signals are in general quite noisy due to the number of signal sources (brain cells) that are located within the region one single sensor measures from. Also, interference between neighboured regions is high.

Also, I examined whether the use of the Emotiv EPOC software Suite as basis, which encapsulates a classifier for upto four different actions,suffices. That means, it is built to be able to distinguish four different mental states of a subject between one another and a neutral state. Then, it outputs to the classified mental state its assigned action.

1.3 Related Work

Reviews of BCI systems in general have been given in [1] and [3] among others. Waldert et. al. concentrated on extracting directional information only [4]. McFarland et. al. provide an overview over the feature extraction and translation methods for classification based systems [5]. Teplan concentrates on measurement in EEG-based systems [6] and Lotte et. al. on classification [7].

After [1] kinds of BCI applications range over word processors, adapted web browsers, brain control of a wheelchair or neuroprotheses, games, and more. Further, [8] identified and classified the following 4 main challenges of BCI systems in real world use:

(1) The information transfer rate of such systems is far too low.

(2) The error rate during input is very high due to highly variable brain signals.

(3) Autonomy is not really given, since sensors usually need to be applied by someone else.

(4) Cognitive load, which is that under distractions the brain generates different, more noisy signals. BCI systems need to work even under such a distracting environment.

Ferrez et. al. are using error-related potentials within the EEG recording shortly after an action to then apply reinforcement learning. That is an attempt towards a more adaptive BCI [9]. Vidaurre et. al. also strife to solve that so-called “BCI Illiteracy” problem, that there is a significant portion of people (about 15–20%), who are unable to control a BCI. Therefore, they also try to make BCIs more adaptive such that they do not solely rely on off-line calibration, when using supervised learning [10].

Regarding filter methods, Blankertz et. al. rely on Common Spatial Patterns (CSP) as proposed by Ramoser et. al. [11]. They extend the CSP approach to consider typical variations of a subject’s EEG signals between trials [12].

Carlson and Millán also used a BCI to control a wheelchair with “left” and “right” commands and additionally equipped the wheelchair with obstacle avoidance to ensure safe use [13]. They employ motor imagery as input employing Laplacian Filtering and power spectral density. Also, there is a thesis from Rebsamen [14], that covers a wheelchair control through BCI using P300-based destination selection. Mandel et. al. created a wheelchair control based on EEG and Steady-State Visual Evoked Potentials (SSVEPs). They stress that due to slight inaccuracy of the control method it is important to have the wheelchair equipped with a safety mechanism, that avoids collisions, as their wheelchair. Also they mention they had one subject that was not able to use their control, which falls under the so-called BCI illiteracy [15].

1.4 Goal and Requirements

As stated in the Problem section the goal is to answer, whether it is possible to create a brain computer interface for some device using the Emotiv EEG headset. Reaching that goal entails a state-of-the-art research, a BCI-concept, an implementation of that concept and an evaluation, whether and at best how far it is working. The only main requirement given is the use of that special EEG headset. As a side requirement — a desirable property — the resulting implementations should be constrained to non-commercial use due to certain licenses. However, that is less important at the current stage.

Starting point The more detailed preconditions were at first

- a 14-channel EEG headset
- the corresponding classification suite
- the corresponding software development kit for C
- exactly one person, who is able to really use the potential of the classification suite and
- the control of our wheelchair in mind.

1.5 BCI Structure

After [7] a Brain-Computer Interface (BCI) is a means of communication that enables a subject to send commands to some device only by the means of brain activity. And, therefore, they state that it is considered the only way of communication for people who are affected by some specific motor disabilities.

The general aim of a BCI is to "read" the user's intent, which is generally done using classifiers that take some representation of readings of brain signals and translate them into a class from a set of states or intentions.

It is especially necessary for people affected by ALS, brainstem stroke, brain or spinal cord injury, cerebral palsy, muscular dystrophies, multiple sclerosis. However, considering EEG-based BCI (as I do in this work), if a person is able to use even any muscle to communicate, that would allow faster communication [3]. Consequently, the only people with real benefits from such research are those suffering from these or similar diseases.

BCIs are differentiated through their signal acquisition technique, which may be invasive or non-invasive or their signal evocation method, which can be exogenous or endogenous (the way a subject is stimulated to create the desired signals) [1]. Such evoked potentials are fluctuations in the voltage measurements that occur either spontaneously (endogenous) from within the subject or after a certain event (exogenous) like a visual stimuli (event-related potential — ERP).

The general structure of a BCI system can be seen in figure 1, taken from [3]. It may be divided into three parts:

(1) the signal acquisition, which records signals from a subject and outputs their digital representation.

(2) the signal processing, which transforms the data such that features may be extracted. Then translates them into some representation suitable to control or command a device. E.g. by using a classifier, that takes the features as input and outputs some value hinting on a class, which may be interpreted as command or further processed.

(3) the algorithm that finally outputs commands.

As a small example let us take the study of classifiers for P300-based BCI spellers by Krusienski et. al. [16]. They acquired their data through a 64 electrode EEG system and bandpass-filtered the incoming signals from 0.1 to 60Hz. Through an offline evaluation step the best channel subset and preprocessing step was determined. Please refer to the actual paper for more information due to its specific nature. With the selected channels and preprocessing the feature vector had a length of 128. During experiment, subjects sitting upright viewed a matrix display as in figure 2. They were a given a word or a series of characters that should be spelled. Beforehand, data was collected to optimise feature weights to enable the system to provide precise feedback. Lastly, the signal processing contained one of 4 types of classifiers: Pearson's Correlation Model, Fisher's Linear Discriminant, Stepwise Linear Discriminant Analysis (SWLD), or Support Vector Machines (SVM). Since the classifiers only detect whether there is a P300 or not, a score is computed for each row and

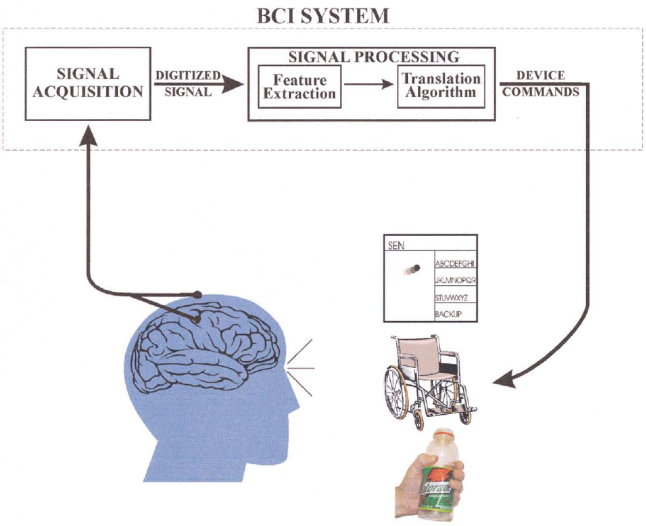


Figure 1: Overview over a BCI system in general from [3]

each column and the row and column with highest score is taken. That is the algorithm that outputs the command — a character — in this example.

As recording device there exist a number of possibilities with different properties. As there are MEG, EEG, fMRI, FNIRS. For its portability EEG, which is used in this work, is the most widely used. For more detailed information about the other types, see [1].

Further, after having decided how to record the brain signals, there are several signals that the brain can produce that may be used to interpret a subject's intent as in [1]. The types are:

(1) VEP (visual evoked potentials)

(2) SCP (slow cortical potentials)

(3) P300 evoked potentials

(4) Sensorimotor Rhythms

In the next step, when there is data from the desired signals, the feature extraction takes place. In [5] one can find a whole taxonomy of methods. There are a lot of different approaches like taking the features from the time domain using for instance Fourier transform or wavelets, or from the space domain trough Laplacian filter or Principal Component Analysis or alike, only to mention some. After that, the extracted features need to be translated to some reasonable output (mostly a class). There, also, is a large number of methods to choose from. Whether it is Linear Discriminant Analysis, Support Vector Machines, k-nearest-neighbors, or Hidden Markov Models, researchers tried out to apply many different existing classification models.

In a last step the output from the translation algorithm is used to control some device and give feedback to the subject.

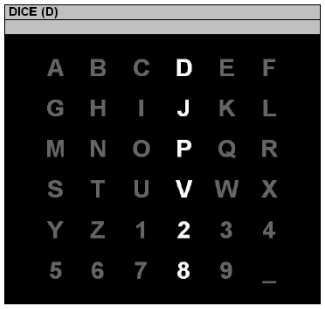


Figure 2: The letter matrix display from the P300-based BCI speller after [16]: A row or column intensifies for 100 ms every 175 ms. The letter in parentheses at the top of the window is the current target letter "D." A P300 should be elicited when the fourth column or first row is intensified. After the intensification sequence for a character epoch, the result is classified and online feedback is provided directly below the character to be copied.

From the technical point of view, the BCI is completed with these parts. For the subject who wants to use it, there is more to do:

(1) Mostly she needs to train the system, so it can adapt to her.

(2) She needs to learn how to produce input such that the system can detect her intent. It is possible to use specific mental tasks to help with that.

(3) That leads to usability problems: How to train the system most effectively and how to assist the subject effectively? (Feedback, Instructions)

The learning process is also referred to as BCI usage as a skill, which means that also for BCIs a user needs to master a certain skill level to use it. That especially means that BCIs are no means of reading one's mind rather they are tools to master [3].

After explaining the general parts of a BCI system, now I will go further into the details of each part. As already stated a BCI system in general consists of signal acquisition, signal processing, which, in turn, consists of feature extraction and a translation algorithm, and then the control of a device.

Signal Acquisition The signals, which mainly may be used for a BCI system come from the brain. However, there is a distinction in how those signals are acquired. There are different kinds of neuroimaging methods used, invasive or non-invasive, portable or non-portable methods. These are Electroencephalography (EEG), Magnetoencephalography (MEG), Electrocorticography (ECoG), Intracortical Neuron Recording, Functional Magnetic Resonance Imaging (fMRI), and Near Infrared Spectroscopy (NIRS). For a detailed description of each method's properties and usages, please refer to [1]. In the scope of this work, only EEG is considered.

Feature Extraction and Feature translation McFarland et. al. [5] already gave a taxonomy as provided in the Appendix. Please refer to their paper for references to each method.

Additionally, after Nicolas et. al. there are different frequency bands from which one can extract the features depending on the task. Each of these bands contain certain classes of signals. The bands are called delta (< 4Hz), theta (4–7 Hz), alpha (8–12 Hz), beta (12–30 Hz), and gamma (30–100 Hz). Also, in the alpha band there are the so-called mu rhythms that are strongly connected to motor activities [1].

Device control The possibilities how a control program may work are quite wide. Applications differ depending on the application and how which potentials are used. For instance, for a wheelchair there could be a direct control that lets the user steer the wheelchair on the feature translations output. So, if the user generated brain patterns are strongly classified as "left" the wheelchair may turn left, even with different strength. Or, it could turn stronger, if "left" is hold for a longer time. As another example, in a BCI speller (see [16] for an example), when there are usually only two classes, the user may select a character by first selecting a row and then a column, while the control program is highlighting first rows one after another and then the characters in that row after a selection took place.

2 Preliminaries

In this section the concepts behind the employed methods are shortly explained.

2.1 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a method to find a linear combination of features, such that two (or more classes) are ideally separated in terms of that features. More visually, one can imagine to draw a line onto which the elements of the classes are projected. LDA finds the line that achieves best possible separation between the classes on that line. See figure 3. The goal is to project two distributions into one dimension to make them easily distinguishable. Since, there is an infinite number of possible projections, one wants to find a projection that is optimal for distinguishing both. That means, one wants to achieve that the projected means are as far away as possible while the variances in both classes are as low as possible. Therefore, Fisher [17] formulated a criterion where he set the so-called inter-class variance to the sum of all classes' variances which projection should be minimised in respect to the projected means.

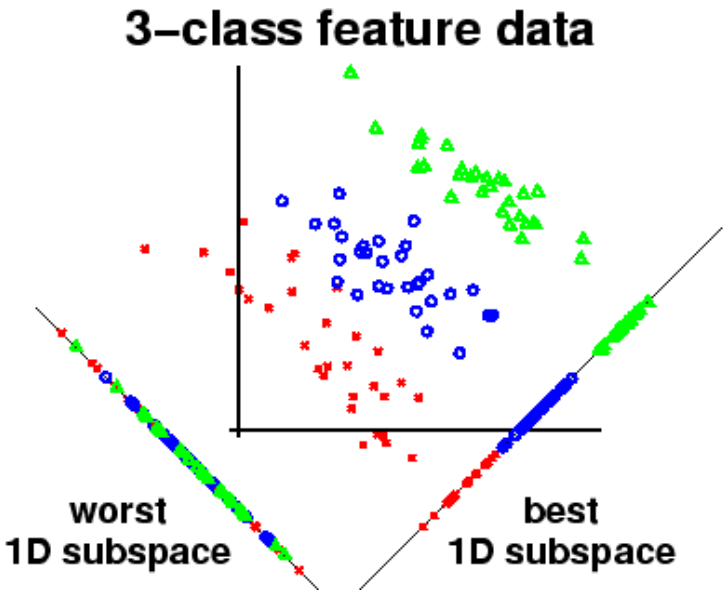


Figure 3: Different lines yield different projections. Taken from [18].

To have a simple classification mechanism the projection is done into the one dimensional space denoted by y onto a vector u. The projection is defined as $y(x) := u^T x$ [19].

So, as in [20] let for two classes μ_1, μ_2 be their respective mean, and Σ_1, Σ_2 their covariance matrices. Then, the Fisher criterion that fulfils the above goals can be formulated for an unknown projection vector u as:

$$S(u) := \frac{|u^T\mu_1 - u^T\mu_2|^2}{u^T\Sigma_1 u + u^T\Sigma_2 u}$$

This criterion should be maximised, so we differentiate in order to find the maximum location u_0. To make that part easier, let us first multiply by the denominator:

$$(u^T\Sigma_1 u + u^T\Sigma_2 u)S(u) = |u^T(\mu_1 - \mu_2)|^2$$

Also, since u and $\mu_1 - \mu_2$ are vectors, we can rewrite the right-hand side to only use multiplication:

$$(u^T\Sigma_1 u + u^T\Sigma_2 u)S(u) = u^T(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T u$$

Now, differentiation (with product rule on the left) and the fact, that covariance matrices are symmetric, yields:

$$(2(\Sigma_1 + \Sigma_2)u)S(u) + (u^T\Sigma_1 u + u^T\Sigma_2 u)\frac{dS}{du} = 2(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T u$$

Since, at a local maximum location $\frac{dS}{du} = 0$, we have:

$$(\Sigma_1 + \Sigma_2)uS(u_0) = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T u$$

Note, that for a given u, $(\mu_1 - \mu_2)u = \gamma$, where γ is a scalar, so:

$$(\Sigma_1 + \Sigma_2)uS(u_0) = \gamma(\mu_1 - \mu_2)$$

Now, we can derive u_0, as:

$$u_0 = \frac{\gamma}{S(u_0)}(\Sigma_1 + \Sigma_2)^{-1}(\mu_1 - \mu_2)$$

For the final discriminant we are not interested in the length of u_0 but only its direction, so we strip the (unknown) scaling factors and receive:

$$u := (\Sigma_1 + \Sigma_2)^{-1}(\mu_1 - \mu_2)$$

With that at hand, we can now define our linear classifier — the LDA-classifier. First, we choose a threshold y_0. Then, given a sample x from the feature space and classes C_0, C_1, check, whether $y(x) = u^T x \geq y_0$. If yes, classify x as belonging to class C_2 and to C_1 otherwise.

As a last point, Bishop [19] also refers to a formulation of the Fisher criterion for multiple classes as one possible way to support multiple classes with this classifier.

2.2 Mann-Whitney-Wilcoxon Test

The Mann-Whitney-Wilcoxon test (or Wilcoxon Rank-Sum test, or Mann-Whitney U test) is one non-parametric statistical test to determine, whether two independent random samples are from the same distribution. So, given values of two random variables, it determines whether both random variables follow the same underlying distribution. The test's null-hypothesis is that two random variables are from the same distribution. As every statistical test, this one also computes a p-value given empirical data about the values of both random variables. The computation is based on a rank for each value. The p-value can be interpreted using the following question:

If the groups are sampled from populations with identical distributions, what is the chance that random sampling would result in the mean ranks being as far apart (or more so) as observed in this experiment?

In other words, if we assume the same distribution for both random variables, how likely is it, that our observation (or a more extreme) would be a result from random sampling, or better, how likely is our observation (or a more extreme) to occur.

From that probability, one can conclude that, if it is extremely low, then the null-hypothesis is highly likely false.

The test assumes

(1) that the two samples are random and independent of each other, as well as the observations within each sample are independent from each other and

(2) that the observations are numeric or ordinal.

Also, it is important that both samples variances are close together.

The p-value is calculated as follows:

Algorithm 1 Mann-Whitney-Wilcoxon Test

input: two sample sets S_1 and S_2
output: p-value referring to the null-hypothesis
sorted $\leftarrow sort(S_1$ appended to $S_2)$
for each observation o in sorted **do**:
    occurrences_indices $\leftarrow$ [collect index of o' = o in sorted]
    $rank(o) \leftarrow mean$(occurrences_indices) ; assign the mean index in the sorted list as rank
let $(n_1, n_2) := (len(S_1), len(S_2))$, $R_1 := sum(\{rank(o) \mid o \in S_1\})$
$U_1 \leftarrow n_1 n_2 + \frac{(n_1(n_1+1))}{2} - R_1$
$U_2 \leftarrow n_1 n_2 - R_2$
$U := min(U_1, U_2)$
calculate the p-value from U and return it.

In order to calculate the p-value from U, one can use the normal distribution $\mathcal{N}(0, 1)$ which is an approximation of the standardised U statistic, if there are sufficiently many (more than 20) samples. Cheung and Klotz also give an algorithm to compute the exact distribution of the U statistic [21].

See [22], [23].

With such a test, I aim to remove variables from the problem that seem to be similar in both classes I want to distinguish. That should make the splitting easier and therefore yield better performance.

2.3 Power spectral density

To understand Power spectral density, we need some background of the type of signals we are dealing with. EEG signals are so-called continuous-time signals, that may be represented by voltage over time (the waveform) and also by spectra, energy, or power. Energy is as in physics the ability to do work, power is then the energy given over a certain amount of time. We measure power usually in watts. For a continuous-time complex-value signal $x(t)$ the power $P_x(t)$ at some instant time point t is calculated by:

$$P_x(t) = x(t)\bar{x}(t) = |x(t)|^2$$

where $\bar{x}(t)$ is the conjugate of $x(t)$. See [24], [25].

Having the power of a signal, its power spectrum is then the power expressed in terms of its Fourier transform. The power spectral density gives us then the distribution of the average (the expected) power over the frequencies of the signal [26].

For an exact mathematical definition, I just copied from Wikipedia[3]:
The power spectral density $S_{xx}(\omega)$ of a time-series signal $x(t)$ in a period with length T can be defined in terms of the truncated Fourier transform $\hat{x}_T(\omega) = \frac{1}{\sqrt{T}} \int_0^T x(t)e^{-x\omega t}dt$:

$$S_{xx}(\omega) = \lim_{T\to\infty} E\left[|\hat{x}_T(\omega)|^2\right]$$

where $\omega = 2\pi f$ is the angular frequency.

The employed algorithms to determine the power spectral density are a more involved and exceed the scope of the present work. I used an implementation for R from [27], where the underlying algorithm is cited.

2.4 Terminology

data refers to a sequence of vectors of numbers. The numbers are measurements obtained through some device or a label describing the whole vector. The label is optional since data of unknown kind exists.

data point refers to a single vector from the data.

channel refers to an input source for the BCI. Through such a channel data from a certain sensor placed on the scalp is received.

action refers to an intended classifier output that should represent an intent. Throughout the work there are mostly "neutral" and "non-neutral" actions with "neutral" as doing nothing or negative class and "non-neutral" as doing something or positive class.

[3] *http://en.wikipedia.org/w/index.php?title=Spectral_density&oldid=633803486*

signal is represented through data of a certain channel over time. It can also be only a projection into a subspace of the data over time, e.g. it can be only within a certain band of a certain period in the time-series of data points.

mental state refers to the state of a human being's brain in terms of patterns in the signals received through all channels. A certain mental state should be linked to an action that, in turn, is linked to a classifier output.

EEG headset refers to recording device consisting of a number of sensors, amplifiers, and possibly filters that is applied to one's scalp to retrieve data based on electrical signals that stem from one's brain and the environment.

significance is a statistical term. So the statistical significance refers to the following probability:

$$\Pr\left[\text{null hypothesis is true} \middle| \text{results that are at least as extreme as in a made test are obtained}\right]$$

where null hypothesis test-dependent and a made test is a certain calculation based on collected sample data. Significance is then present, if the given probability is sufficiently low. A significant result means, that it is highly likely that for the tested data the null hypothesis is false. Put simply within the context of this work, it refers to variables that are important in order to distinguish classes from the data.

condition is a statistical term. For some statistical tests the collected sample data needs to be clustered by conditions into groups that allow to make a distinction between two samples. With that the influence of a certain condition on data can be determined. The concept is similar to labels for data in the context of supervised machine learning.

motor imagery refers to a form of mental process that aims to manipulate special signals being recorded from the brain through an EEG headset. During that process, one tries to imagine muscle movement without actual movement.

system refers to the collection of modules that are composed to a BCI and that are actually doing work.

trial refers to a period of time during which data is collected assuming a certain action. As a result all data of one trial has the same label and the same condition. It may be used for training or testing purposes.

run refers to a number of consecutive trials collected with some pauses in between the trials with the goal of training of a classifier in mind.

session refers to a number of consecutive runs as setup for the system.

feature refers to a random variable that is extracted from the data in some form.

subject refers to a human being who is asked to produce sample data for the classifier as training or testing data during experiments.

user refers to a human being similar to a subject, however with emphasis on actually using the system for own benefits.

ground is a term from electrical engineering. It refers to a reference sensor used to determine measurements for other sensors.

3 Prior considerations

3.1 Emotiv Suite Tests

Besides an SDK[4], Emotiv provides a classification suite along with its headset. This suite contains three kinds of classifiers of which one is a general-purpose classifier. That one is called Cognitive Suite. The other special suites are able to detect some facial movements and blinking, as well as, the emotional state. At least, they are supposed to. From my informal testing during several weeks I noticed, that blinking detection works quite well, whereas the rest seems mostly random or only works for some persons well. Additionally, these specialised suites are not trainable. Therefore, I mostly stuck to the Cognitive Suite classifier. This one has also been used as input for a BCI to control our autonomous car [2]. Since I had one person at hand, who is actually able to make the Cognitive Suite classifier react at will, I tried out different ways to bring it to do the same for an arbitrary person.

I tried out different methods like motor imagery, slight muscle movements, muscle contraction, closing the lid (which works, but can be considered as cheating), or concentrating on some point. From my informal tests, I noticed the following:

- Imagining of movements, e.g. moving an object within the head, are working methods for some people.
- For different people, different methods need to be used (which fits other research [28])
- Concentrating on the feedback during training (like a moving orange cube) makes that feedback necessary to work with at least this classifier
- It is really painful to improve the classifier's performance through training, when the method is inconsistent. Also, on corrections the data is lost.
- For nearly every person it is extremely hard to get control over more than 2 actions.
- Less data seems to allow better control.

Aside from that I connected the classifier's output to the wheelchair's control and run it in 3 different control modes:

- Pushing step-wise, i.e. a trigger of "push" increments the wheelchair's impulse to move forward a certain distance.
- Continuous directions, that is, when triggering a "left" or "right" the wheelchair will steer as long as the action is hold.
- Step-wise directions, which will steer the wheelchair by a fixed angle on triggering a direction.

[4]Software development kit

All that did not lead to really satisfying result in the control over the wheelchair. The training of that classifier is made additionally hard considering the aspect, that the only correction one can make to its training is to clear all training data for an action. Consequently, it is necessary to follow a strict training during which the brain signals must be produced consistently with existing data.

So, I decided to design an own classifier, to, hopefully, overcome these limitations, i.e.:

(1) To be able to change preprocessing steps or feature extraction to those that work better with a different subject, and

(2) to be able to manipulate which training data (which trials) are used to train the classifier, so I may improve training success.

3.2 Method Choice

From the EEG data received through the headset, I first needed to decide, which features to use. As the data is given as time-series using Fourier transform was the easiest option for me. However, as the data was highly noisy, I switched to the power spectral density, which provided more stable transformed data, although a different kind of features. Other researchers have used the power spectral density mainly to decide whether a patient is in sleep or awake state as I read in [29]. Therefore, this decision might reveal itself as impractical at some point. Nonetheless, for the 0–1–classifier, I stuck to it.

To further reduce the dimensionality in feature space, I employed the Mann-Whitney-Wilcoxon test in order to determine which dimensions seem to be best to distinguish the two classes in the data. The use of a non-parametric statistical test in feature extraction is a quite low-level technique and also not mentioned in the discussed taxonomy. Nonetheless, as long as the data fits a statistical test's underlying assumptions, it is reasonable to use that statistical test. However, as a side note, during data analysis, I found that often the data violated the Mann-Whitney-Wilcoxon test's variance assumption. Therefore, further research is necessary here.

Having a feature vector ready to use, I employ linear discriminant analysis to achieve 0–1–classification. Regarding all modules developed during my research so far, use of LDA is not a requirement, as long as the classifier of choice outputs some kind of certainty value along with the assumed class value. Any classification method with that property should be easily applicable with my prototype system. Additionally, the LDA-classifier was also an option to use for reduction of dimensionality. However, in a usual case the number of dimensions to deal with even exceeds the number of samples that can be used to train a classifier, which is necessary to find the best dimensions in distinguishing both classes. Even, if I could restrict the dimensions to a small subspace of the original feature space it still remains hard to retrieve a lot more samples than dimensions. Therefore, a statistical test was the far better option. Although, it does not take into account correlations between dimensions. These can play an important role as stated in section 1.5.

4 Design and Implementation

This section describes the design process.

As the Emotiv suite did not provide me sufficient capabilities to adjust a system built upon it, an own classifier became necessary. Even restricting myself to only one action (i.e. “push”) did not enable me to find a method to reliably learn how to train the classifier, so that it responds as expected. As reference I had one person in our research group, who was, in fact, able to reliably make the classifier respond to up to 4 action (left, right, push, pull). It seems that he had the “natural talent” necessary for that kind of classifier. Or better, that he was really well suited for that kind of classifier.

Also, the kind of training makes a difference. After Dobrea and Dobrea the performance of the system differs significantly from subject to subject depending on the task chosen to train the classifier [28]. They conclude that besides subject-specific electrode positions and feature extraction, also, the BCI application itself (training method, feedback method) should be subject-specific. That is, my system needs to be highly configurable to fit the characteristics of a subject.

In course of the design 5 stages need to be considered data collection, data analysis, data classification, training method, and device control. Each of these stages will be described in the following.

Data collection

This comprises the methods to obtain data in a systematic manner, such it will be useful for the later classification task. The collection itself does not seem a difficult task, still some considerations have to be made. In order to apply supervised learning, a label needs to be assigned to data representing its supposed class. The data itself is sequence of vectors of which each vector contains a voltage value for each sensor attached. It looks as in table 1. All values except for the ones under label are Voltage.

AF3	F7	F3	FC5	T7	P7	O1	O2	P8	T8	FC6	F4	F8	AF4	label
4612	4287	4339	4624	4533	4489	4270	3809	4443	4514	4323	4438	4730	3883	0
⋮	⋮													
4650	4231	4292	4541	4533	4471	4262	3925	4392	4517	4349	4136	4682	3847	1
⋮	⋮													

Table 1: The collected data. To understand the channel names in the header, please look at figure 8. Values are rounded to integers in order to fit the page’s width. The original values had two number right of the point.

The classifier should be able to distinguish the mental state of a subject. That mental state is assumed to be represented by patterns in a subject's brain signals. So, for the supervised learning it is necessary to have a portion of data that should be interpreted as a certain action, in a sense that features taken from that portion of data will be considered as that certain action.

The data collection procedure follows a simple setup: First, the subject to collect the data from is wearing the EEG headset. Over a wireless connection a laptop can receive EEG data. One data collection session is divided up into a number of trials, where each trial contains data for one of two specific actions. Through a data collection application running on the same laptop the subject is instructed via an on-screen message which action to perform. For instance, she could be instructed to do a "push"-action and therefore concentrates on imagining herself moving forward with her full body. One second after giving the command the application starts to record and save the raw data from the headset. The recording period lasts for 5 seconds within which the subject should sustain her mental state. After that, there is a pause of 5 seconds and a new trial begins. For each trial the action to instruct is chosen at random.

Considerations A first consideration regarding the instructions is, that since a subject's brain will be affected by every perception she has, the instruction will certainly change the brain signals' patterns. For the later training such a change is considered noise. Therefore, it should be avoided in the data.

In my case, the instructions are given through a on-screen text. Regarding the discussed perception noise the actual method should not matter as they all affect the brain.
Also, since nobody knows where the desired patterns begin in the time sequence of the data, it is hard to tell, when the recording should start, after an instruction is given. I set that period to 1 second, so a subject can perceive, read, and process the instruction and then execute it. After that the trial goes on a fixed time. Since, it is hard to hold such a concentrated state for long, I set the trial length to 5 seconds, which seem to be a feasible period from my own tests and some experience of others in the work group.
Still, all the data in one trial is not necessarily of the desired action. However, the chosen trial length and throwing away apparent noise should already avoid a lot of "bad data".

Also, consider, that subjects should not be able to foresee an instruction to be able to adapt to the situation. At the time of usage, they always will have a certain intent that arises spontaneously and from that create the patterns. The collected (training) data should reflect that fact.

Lastly, since during data collection the subject needs to be highly concentrated, between two trials there is a pause. I fixed it to be 5 seconds long. Maybe, a random-length-pause would also have a positive effect by making the next trial even less adaptable. At least, it would require a subject to be concentrated even longer. I cannot tell which was a better alternative.

Anyway, the labels are then assigned to each single vector instead of to the whole trial. That makes it easier to treat the data as continuous and to extract data from random places.

Data analysis

This stage aims to filter out "bad" data to obtain more clean and less noisy data, which includes data transformation and finding well-suited features for the classification task.

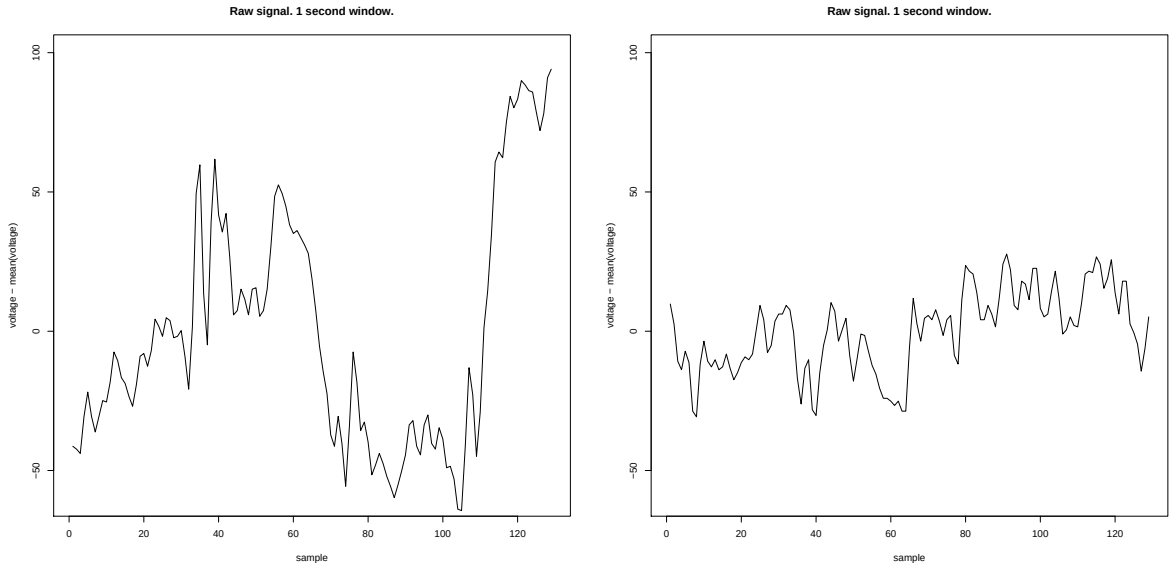
The headset outputs raw data in form of Voltage in μV. Its values range from about -4000 to 3000 depending on the channel and subject after subtracting the its mean for a whole session. In some cases there are extremely high peaks, which seem to relate to power shortages at the sensors. Such peaks usually happen, when a sensor is moved while a subject wears the headset. Usually, the signal is between -100 and 100. For the analysis and, in turn, for the classification later, I discard a window containing such "sensor artefacts".

The raw data then looks as in figure 4. Looking at many portions of it one may suppose there is some difference between signals of different actions. Even if that was the case, the difference would be hard to find due to the non-stationarity of EEG signals. That means, in contrast to stationary signals like sine waves EEG signals differ in their spectrum over time, so it is hard to impossible to find a start of a period, as there is no real period. For classification this means, that if I cannot find comparable periods, I cannot reliably classify, as there is nothing to compare a new period to.

Also, I need to remove the artefacts, as portions containing them are useless, since the do not reflect a subject's intent. However, from a continuous time signal it is very hard to remove some portions in between. As classification of such signals is already very hard, I decided to search for features in the frequency space created through Fourier transform. To retrieve that it is necessary to fix a period during which the signal is viewed as stationary. Such a period is called window.

Now, to come back to the removal of sensor artefacts, for each channel it is checked whether one window contains a huge peak, i.e. whether there is a slope steeper than a fixed threshold. Such a peak can be seen in figure 5.

Also, since I only check window-wisely, if the signal temporarily shifts its mean (for an unknown reason) windows that fit between two shifts are not detected as useless. That would require an analysis at the level of the whole data, or at least a detection of such a temporary shift. One may possibly observe the average mean of the signal and compare that to a window's mean for detection.



(a) For the “neutral” action. (b) For the “non-neutral” action.

Figure 4: 1 second raw signal in one channel without any transformation.

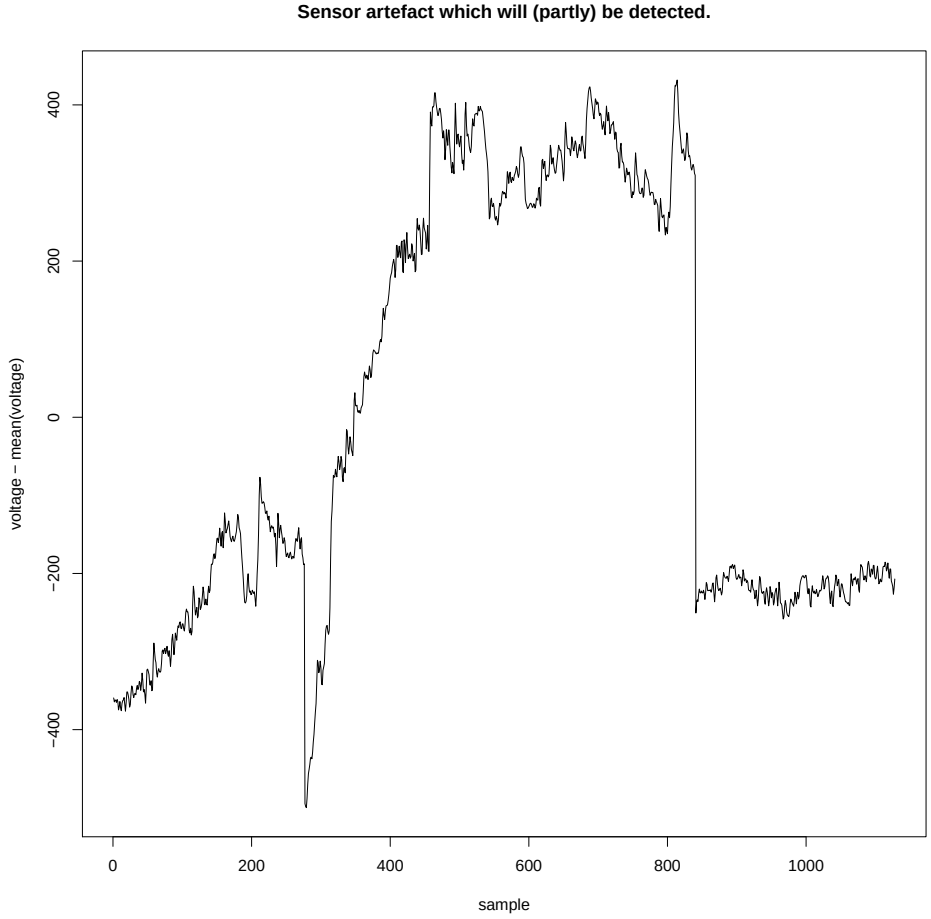


Figure 5: A sensor artefact with a temporary shift of the signal's mean.

With sensor artefacts removed, it already possible to extract features. The first and most simple approach is to use fast Fourier transform to obtain frequency amplitudes of a window. I was pointed by Omar [30], that the resulting data looked too noisy to be useful, when we compared several windows of two actions. Then he pointed out to use power spectral density, which should yield more stable results — as it did. The transformation can be seen in figure 7 for the same windows as in figure 4. In practice, this transformation is often used to determine whether a subject is in sleeping/ fatigue state or not as I read in [29] Furthermore, it is easy to replace the transformation and adapt the system to that. At least for the two-class case that should suffice.

As length of one window I decided to use one second, which correspond to 128 samples. That length affects the latency of the classifier on the one hand side, and its performance on the other. Also, that window length results in 64 frequencies from the power spectral density per channel. That makes 14×64 features per sample, which is a lot. If I find a subspace in which the classes are in each dimension easily distinguishable, overall noise when projected into one dimension is reduced and so the classes are easier to distinguish.

To reduce the number of dimensions, Omar [30] proposed use a non-parametric statistical test to find out which features are actually distinguishing both actions from the data. Also, to reduce variability not single windows are used, but an average of about 4 windows per trial. The averaging stabilises a feature's value for an action, so results from the test should be more reliable by averaging out noise effects.

This reduction of dimensionality process is explained in more detail below. In short, it uses a statistical test at each feature to determine features which are most likely from different distributions, and therefore, are well-distinguishable.

My prototype uses the Mann-Whitney-Wilcoxon test. For now, the employed test yields somewhat useful results. The test itself is applied to a sample statistic consisting of the averaged windows for each trial and action. For each pair (channel, frequency) it is tested, whether in that one value both populations are the same. That yields a p-value related to the null hypothesis of the test. All p-values are collected. Then, the 10 pairs (channel, frequency) with the lowest p-value (most likely rejecting the null hypothesis as in 2.2) are taken as part of the final feature vector. Also, in order to avoid that the test yields by chance that a certain value is significant, although it is not, I repeat the test several times with different windows and take the values most often considered significant. How that looks like at each step as visualised in figure 6. The process as schema is visualised in figure 18.

One problem here is, that due to unmet assumptions the test may fail to provide a reliable result. In my case that is that the employed statistical test assumes both conditions to have similar variances. However, especially in the cases I trained, that assumption is mostly not correct as one can see a bit later. A better alternative seems to be a hybrid test that also works, when variances differ, as proposed by Kasuya in [31], but I did not test that, yet.

Ten features remain to be chosen from all features after reducing the dimensions. From my data there were mostly four to six features that really could be considered useful. So, I decided for 10 as an arbitrary upper bound to reduce the risk of losing useful features when the data is far different. 10 dimensions is already quite large, if one considers that the collection of one sample of data takes at

least 1 second and it requires about 50 to 100 samples to achieve reliable abstraction with the training data [32]. Still, about 100 seconds seem feasible for one training run. In my training method a subject should usually at least perform 3 training and testing runs, so it adds up to 600 seconds or 10 minutes excluding some breaks in between.

The whole process is quite simple, as I do not use any calculated features, but only those the data provides through a single transformation. Here, I have been suggested to also try combined features so the classifier can also take into account dependencies between features. Some first trials with multiplying the 2 best and best and third-best found features did not yield a better result, however, the trials were extremely short. During evaluation process I noticed that the classifier has indeed problems distinguishing both actions with only these features, so more sophisticated features are necessary. Other work groups use additional filters or transformations to achieve that. As there is common spatial patterns [4], for example.

As a last point I found out during analysis, that during training a subject may not be able to recreate a brain signal pattern for the full course of a trial. In such cases, there are windows with lower quality data that may be chosen. It depends on the subject's performance how grave that problem can become. However, during development of the present prototype I could not find and implement a solution for that. I think about finding and removing outliers from the data using, for instance, the Mahalanobis distance. I also tried a statistical approach to remove outliers, where I employ the total power of a window. That is, summing up all values of a window's power spectral density. Over the total power I estimated the density function of the easy-to-create "neutral" action windows — as I was always working on "neutral" against "non-neutral" state classification. With that density estimation I determined how likely it was for a "non-neutral"-labelled window to be "neutral" by computing its total power and apply the estimated density function on it. If this probability exceeded a certain threshold it was considered "neutral" and filtered out. However, that process removed mostly about every single "non-neutral" window, due to the nature of the total power distributions in my data. See figure 11 for a visualisation of their nature.

Aside from the above offline analysis steps, during online usage there is also done parallel validation during testing phase. That is, multiple classifiers are trained using windows from different trials, so the contribution to the overall performance from one trial can be measured. With that I can determine some hint on the quality of a trial after testing phase.

To understand the channel names, have a look at figure 8 where all employed channels are displayed with their corresponding positions on the scalp.

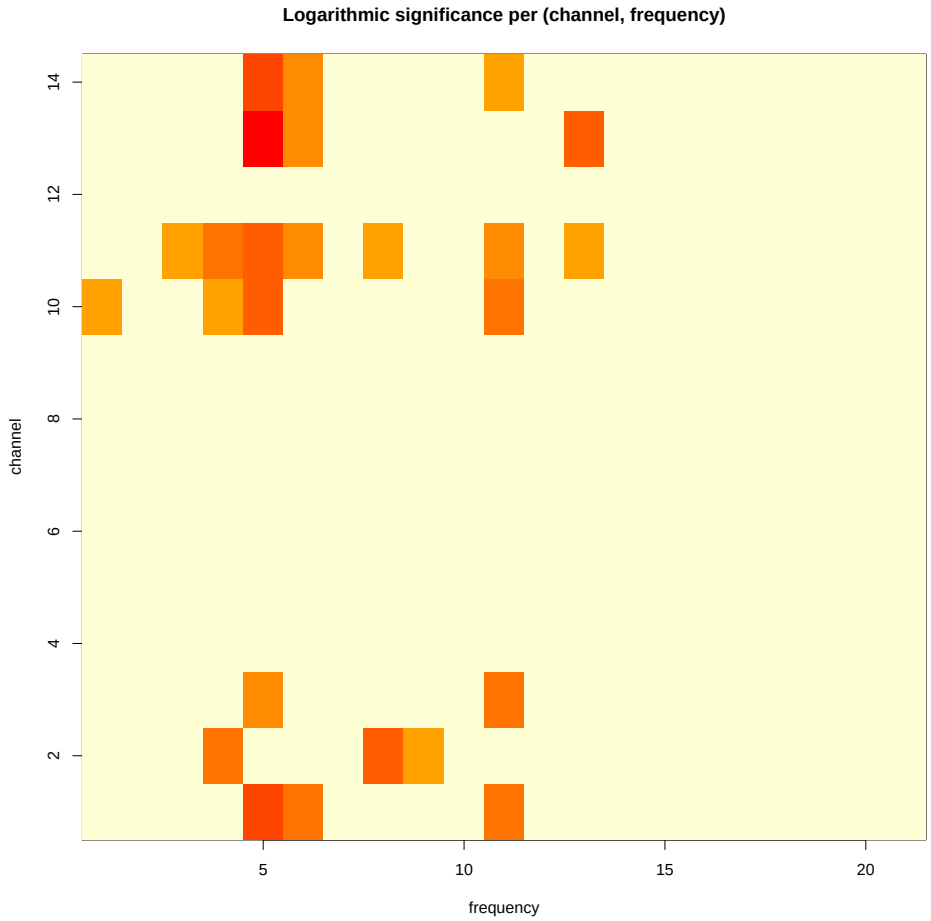
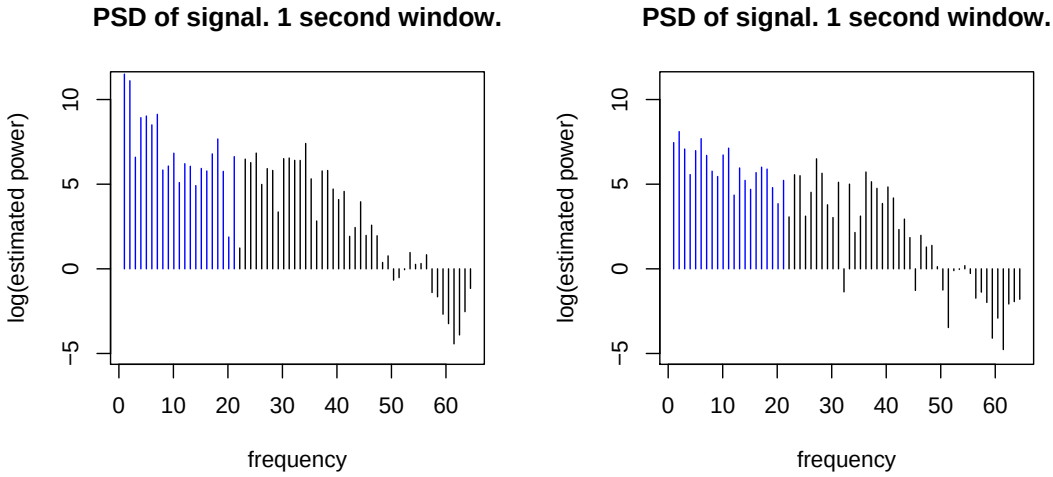


Figure 6: A colour map of the significance values during analysis. Each coloured rectangle represents a significant value (smaller than 0.05). It represents the significance considering only the frequency and channel at that position, checking whether these values from both classes are from the same distributions.



(a) For the “neutral” action.

(b) For the “non-neutral” action.

Figure 7: One second psd of the raw signal in one channel. The log of the power is displayed to better show relations between different frequencies. Also, the highlighted bars are the power values my prototype takes into account.

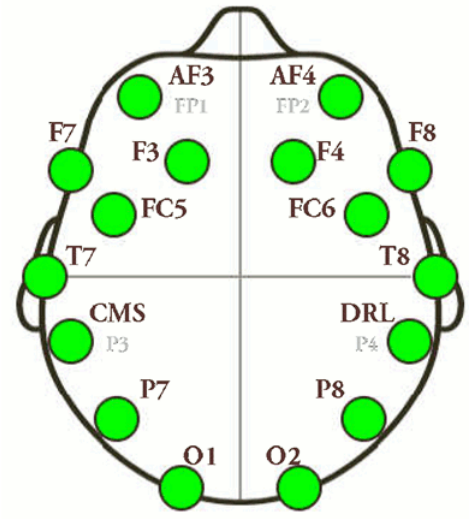


Figure 8: The EPOC EEG headset channels with their positions from [33]. Note that CMS and DRL are ground.

Data classification

In this stage the preprocessed data is used to create model that is able to determine a label for an unknown sample. The model is in general one of widely used classification models that are enumerated in section 8.3.
As classification algorithm I decided for linear discriminant analysis (LDA), since it is easy, stable, and can be trained really fast. Additionally, after [7] it also seems a good choice for EEG data, because of its high bias, which means, that the classifier does not respond strongly to changes in the training data. Since, that enables me to understandably improve classification performance (accuracy in reflecting the subject's intent) by improving training data quality. These properties make sense in classification of EEG data, since the signals are slightly different among trials and especially sessions, thus making the responses more stable.

So, by choosing training data correctly, I can easily improve the classifier's performance in reflecting a subject's intent. And linear discriminant analysis allows me to find out which data is best to choose for training, since I can train and use many classifier instances in parallel. This decision made it crucial for me to choose only the "good" training data from the trials. Since I am choosing windows from trials, I need some mechanism to choose only "good" ones.

For whole trials, I found a solution discussed under Training method below. Within trials I still select windows randomly. A possible solution has been mentioned above. The classification task itself is usually easy provided "good" data.

Beside the above, additional reasons for choosing LDA are that it

- has been employed successfully in different research works as in [12].
- can be trained and applied really fast.

As input for the classifier I use a 10-dimensional feature vector composed of certain values from the power spectral density (PSD) of all channels and from frequencies of 2Hz to 22Hz taken from a 1 second window. This special band has to do with the fact that mu (8–12Hz) and beta (18–25 Hz) rhythms can be controllable (see [34]) and that the sensorimotor rhythm is based within that band (after Omar [30]). The upper bound of the beta rhythm band is given quite differently in the literature, and I concentrated on capturing the sensorimotor rhythm at first, which is why my upper bound is lower than 25Hz. Still, that is merely a parameter. In more detail, the procedure is as follows

(1) Given a window with the length 1 second worth of EEG data of 14 channels, the PSD is computed for each channel.

(2) select only 2–22Hz of the resulting transformation.

(3) From a small set of averaged windows, determine which 10 values are best to distinguish both labels. Values are coordinates (channel, frequency).

(4) Then, for the feature extraction, use these 10 values, i.e. the power at the given frequency of the given channel, and stack them into a feature vector.

As a result, from each 1-second-window I extract a 10-dimensional feature vector, that is used as input for the LDA.

Further, it is important, how I determine values to choose. First, the reason to do so is, that with not much data it is highly desirable to have as low dimensionality as possible. In that way, I may provide enough data for training, such it may be considered sufficient. [7] cite a recommendation for five to ten times as many training samples per class as I have dimensionality. This makes at least 50 to 100 samples per class in my approach, to have a reference for training.

My procedure is structured as follows:

(1) First, collect some trials for each condition.

(2) From each trial choose a number of windows and create one average window of these.

(3) This results in one averaged window for each trial.

(4) Now, transform it into the PSD image.

(5) And then for each value (channel, frequency) determine the significance of the value in distinguishing between both conditions. Therefore, use a non-parametric statistical significance test. In my case, that is the wilcoxon test.

(6) This results in a significance value for each value. From these take the 10 best. Note, that 10 has been chosen, since there were rarely more than four or five features that could be considered significant from my experimental data.

(7) Each feature vector x is standardised by subtracting the vector of means $\hat{\mu}$ for the features and dividing by the vector of standard deviations $\hat{\sigma}$ for each feature, that was calculated from the training data used for the classifier. I.e. as input to the classifier give $x' := \frac{x - \hat{\mu}}{\hat{\sigma}}$. This is already done by the library.

As additional step, it would be desirable to reduce outliers from the training data before feeding it into the classifier. To do so, I would need to compute all possible feature vectors of one trial and then compute all Mahalanobis distances from their centre. Unfortunately, that computation is extremely costly and would make an online session a couple of hours long, so I skipped that. Although, it might be beneficial to do that analysis over time asynchronously to improve the selection of "good" training data. However, further research is necessary there. Or, different methods from [35] could be applicable.

Training method

This stage consists of preparing the classifier for the specialised for a certain subject. It also comprises methods of finding a suitable subset of the prepared data to use for classifier training.

For the filtering of the trials I implemented a special ranking into the training program. It works as follows:
During training, I train for N trials $N+1$ classifiers. That is, one classifier C_0 using windows from all trials and for each trial i a classifier C_i leaving out trial i.

All classifiers are trained using the exactly same windows, so I ensure they use the same data, but with different trials (so with a different amount of windows). During testing, all $N+1$ classifiers are then applied to the exact same data. With that all classifiers are trained and tested under the exact same conditions, only differing in the trials being used for training.

With that a quality measure is the defined as follows:
Let $p : C_{2,2} \mapsto \mathbb{R}$ be the performance measure that takes a classifier's confusion matrix after training and outputs the performance. That performance is usually from 0 to 1 using accuracy or precision and from -1 to 1 using Matthews Correlation Coefficient (MCC) as introduced by Matthews [36].

One subject suggested to use precision and recall as feedback during training, since it is easier to interpret. The main reason for the use of MCC was to have a comprised one-value measure, that reflects the performance in both classes. While precision and recall express a measure in terms of the positive class ignoring correct classifications of the negative class. For the subject the MCC is, indeed, not well-suited as performance measure, however, to for the trials quality hints it serves well as it allows me to clearly optimise in the direction of perfect classification. Using accuracy, precision, recall, or similar measures would overweight the classification performance in one particular class, which could slightly increase the number of random classifications in the other class. In other words, if a measure does not capture either of false or true positives or negatives the removal of a trial with a low quality hint using such a measure does not necessarily hint a training data set that will result in a classifier with improved control from the subject's point of view. I.e. the control may be improved for one class, but may also become worse for the other.

For a trial i the quality q_i is then defined by

$$q_i := \begin{cases} p(C_0) - p(C_i) & ;\text{if } p(C_o) \geq p(C_i) \\ p(C_i) - p(C_0) & ;\text{else} \end{cases}$$

q_i measures whether the classifier would be better in terms of the performance measure with the trial or without. Explicitly, it says, that if I leave out exactly only the trial i the performance of the classifier on the same test data using the same windows would increase by q_i. So, it should give a hint, whether a certain trial i should be used or not to train the classifier. Since the windows are different each training and the test data also is different each time, this measure is merely a hint and no definite way to improve quality. Nonetheless, it should be provide enough hints to reasonably assist a subject's training towards a better quality classifier.

Procedure The procedure is then divided into a sequence of runs, where each run consists of 10 trials. I do not require to do any special task like solving math problems or counting. As orientation for one action the subject should strongly concentrate, for the other she should be relaxed and possibly distracted from the training. During the run the previously described data collection method is used.

If there is a trained classifier, the subject, also, is provided with a "power output". That "power output" is computed from an impulse with two properties:

(1) It is incremented by a certain velocity, which itself is incremented by the scaled probability output from the LDA-classifier. For probability output $p \in [0.5, 1]$, class $c \in \{0, 1\}$, and some weight w the scaling is given as $(-1)^c w(1 - 2p)$. Note, that this scaling does not take into account a classifier threshold other than 0.5. For a scaling aware of threshold $0 < t < 1$ use $w\frac{p-t}{1-t}$ for class 1 and $-w\frac{p}{t}$ for class 0.

(2) It is always within the interval $[-1, 1]$

The weight w is chosen according to the classification rate which is classifications per second. It determines how fast the "power output" reacts on the subject holding a certain state.

The "power output" serves as feedback to the subject and thus is intended to encourage her to hold her mental state longer, if she succeeds, or adjust it, otherwise.

After one training run is finished the collected data is analysed and used to create a number of LDA-classifiers. These can then be tested through a test run that works the same as a training run. That allows to determine the classifiers' performances and quality hints for trials. For a sufficient classifier performance at least two training runs should be done. With test runs the data can be filtered further to improve it slightly. That will be explained later under "Guided training".

Result With this classifier model, I could create a simple training program, that includes formal data collection for supervised learning and a formal testing method to provide some quality feedback and thus to assist the classifier's training.

Device control

As the 0–1–classifier has quite limited possibilities to control a device, I oriented myself towards BCI-spellers and created a simple menu-based BCI, that takes the classifier's output and allows a user to give commands from a set of arbitrary ones. That menu-based BCI application is discussed in more detail in section 4.1.

Problems and solutions

During development of the system I encountered some problems that blocked the way to my goals. The three major problems were user performance feedback, guided training, and finding "good data".

Performance Feedback The performance feedback should provide a user with a hint on "how well" her mental state is classified as output by the classifier. It serves two purposes:

(1) To show whether and how well the classifier reflects the user's intent.

(2) To encourage the user to try harder towards the intended action or to hold her state with more concentration.

The Emotiv Cognitive Classifier outputs a bar, which reflects percentage of "power" of the classified action. I could not find out how the Emotiv software determines the power to output. For us, one reasonable solution would be to use the posterior probability from the LDA classifier. The main problem with that approach is the extremely fluctuation in that output, where even the class output alternates quickly (see figure 9). This fluctuation is not well-suited as feedback, since it is hardly understood by a human. Therefore, I smoothed the output by using it for an impulse. So, the power is limited within [-1, 1] and it is increased by the impulse after every output from the classifier. The impulse itself is increased by the scaled posterior probability, as explained above. So, in fact, the "power output" accelerates by the scaled posterior probability and also is limited. This results in a very smooth change in "power", which is far different from the Emotiv software "power". The concept of my feedback is visualised in figure 10. The reason behind the decision to use a smoothly changing output is that for my BCI prototype I intended to use holding of an action as trigger. If that is the optimal strategy to provide control probably differs from subject to subject. Unfortunately, I was not able to already research this part of the whole BCI system.

Guided Training One main problem with a BCI is the necessary training. Here, I aim to make the training as efficient as possible, since it highly likely needs to be done often before use and can easily be quite exhausting for a user. There are two types of guidance that can be provided. One is to find and choose an appropriate mental task for the current user, that provides best possible performance and is well-suited. Such a guidance requires knowledge about many possible strategies and probably user-specific analysis for each strategy. Such knowledge could not be acquired by me during that present work, so I left this as open as possible by not specialising on certain strategies or mental tasks. The second is to hint the user, whether the data she provided is well-suited to reflect her intent. I oriented myself by the idea of Fails and Olsen [37], which I stripped down for my use. The idea is very simple: I use multiple differently trained classifiers to decide, whether a certain portion of the training data makes the performance better or worse, when used for during testing. Then, one can choose to not use that portion of the training set for retraining. My implementation covers that on trial-level, since that something familiar to the user, as she performed on a per-trial basis.

"Good data" "Good data" is a data that, when labelled with a certain action really belongs to that action. So, it is the correct labelling problem. Although, during training I try to minimise the collection of wrongly labelled data, it is inevitable to receive such data. Let us consider a first simple scenario in which I have a "neutral" action, that is "do nothing", and a "non-neutral" action, i.e. "do something". The training program instructs the user to "do nothing". In that case, I can quite safely assume that the collected data will be for "do nothing" at least after a short period, when the signal change due to receiving the command and changing state is done by the user. However, if the user should perform "do something", it is highly likely that she is not fully concentrated to hold her mental state for that action over the whole trial time. So, in between there probably will be data

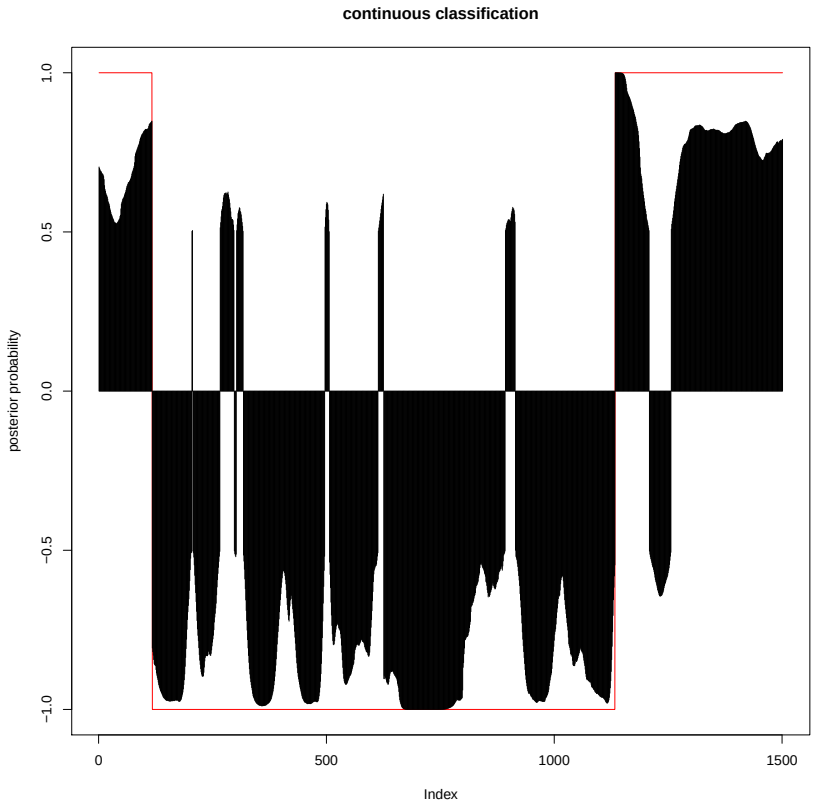


Figure 9: The LDA class output on continuous input. That is, consecutive windows that are input partly overlap.

for “do nothing”, as I only have two classes of actions. So, while all samples in such a trial are labelled “non-neutral” some do not belong to that action. A first approach to detect stemmed from the fact, that I could quite safely detect “neutral” action. In some kind data of that action will follow some distribution fixed over at least the session, if I assume the user to not be extremely unstable in terms of her brain signals. I tried to use the total power of each window in the data as a distinguishing feature. The total power is the sum of the power for each frequency within the window. And, indeed, when I looked at the distributions of the total power, even assuming correct labelling, both actions had different distributions as displayed in figure 11. Unfortunately, in the total power they differed mainly in variance and hardly in mean. So, simply removing samples from “no something” trials that look like “neutral” samples usually removes nearly everything non-neutral. One may consider to use the assumed distribution from “non-neutral” data to remove outliers, however, that is not safe due to the wrong assumption of wrong labelling and, in fact, were obsolete, if I already assume correct labelling. Another approach could be the Mahalanobis distance to detect outliers and remove them, although it follows the same assumption and therefore only increases the bias. Therefore, I decided as a “solution” to just use randomly chosen windows, since I avoid systematically choosing

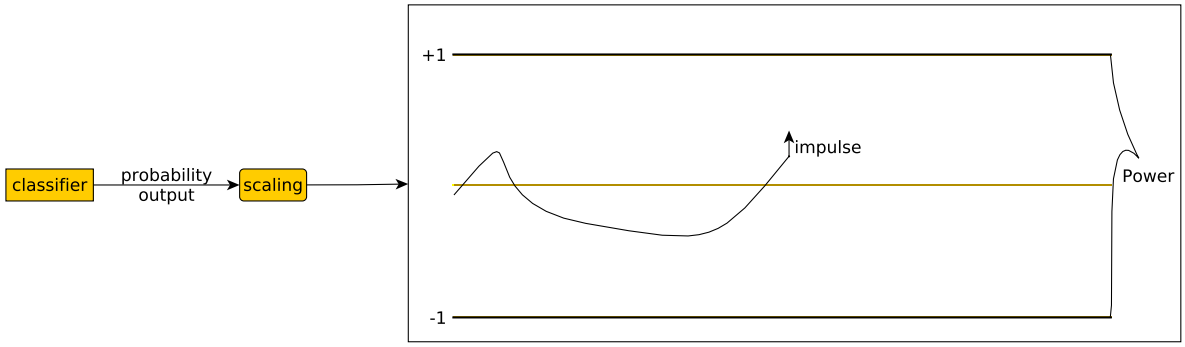


Figure 10: The concept of “power output” in my prototype.

windows wrong. Additionally to that, one could score the windows during testing according the test performance. And so in the long run weighting windows as more or less useful for training. However, this process of weighting windows reliably would take many retraining and testing steps due to the vast mass of possible windows, so that seems not useful. The problem stays when attempting to train and test different window sets in parallel, since that would cost an extreme amount of resources due to the mass of possible windows.

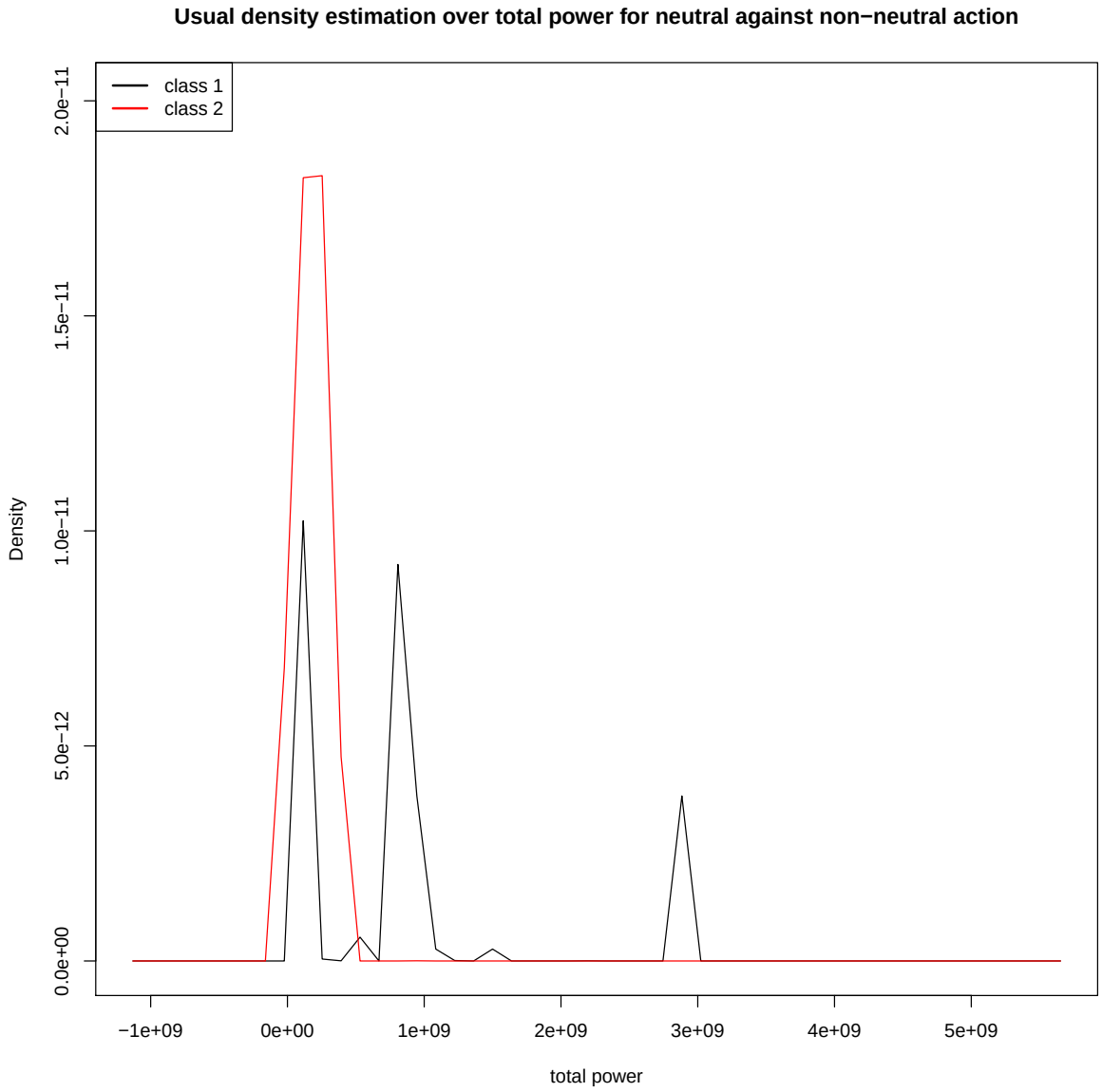


Figure 11: The estimated density functions for "neutral" and "non-neutral" actions from data for one subject.

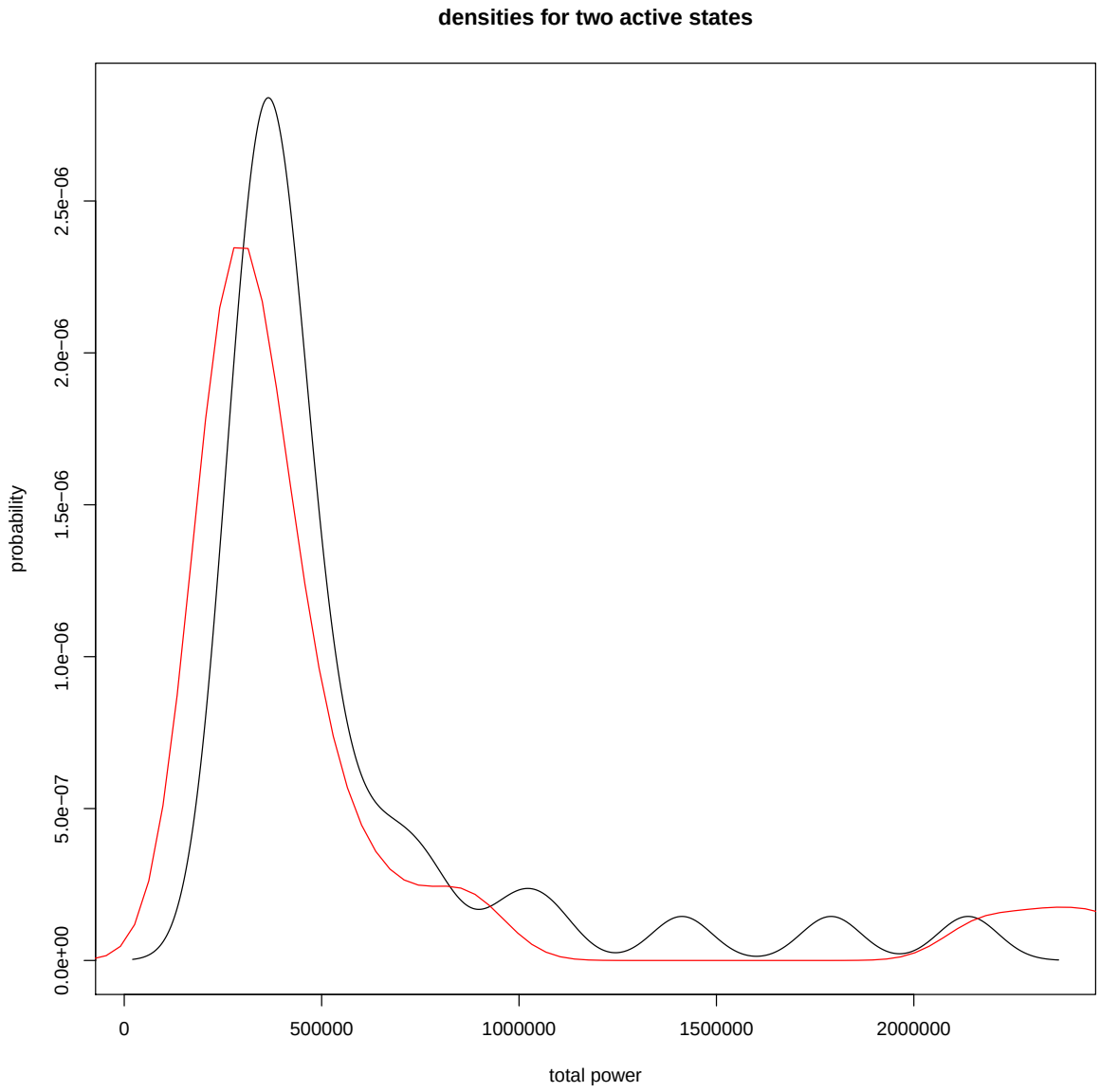


Figure 12: The estimated density functions for two different “non-neutral” actions from 1 subject. As one can see, with these the described filter method suffers from the same problem. Also, from in terms of total power, it is possible to determine distinguishability using a statistical test.

4.1 Implementation

During course of the prototype implementation I created a simple data collection program, a simple menu-based BCI-application, and a training application that realises data collection, classifier testing, and its improvement by making use of my R-based data analysis program. For the analysis program, the power spectral density (psd [27]), the Mann-Whitney-Wilcoxon test (stats::wilcox.test [38]), and linear discriminant analysis (MASS, [39]), which outputs a posterior probability calculated from the discriminant are taken from libraries. The other two depend on Emotiv's software development kit and were both written to run at least on Microsoft Windows. The Cognitive Trainer is based on Qt[5].

Cognitive Experiment

This simple console program was my first data collection attempt. It executes the following simple experiment: Given two actions, choose one at random, and start a trial of 5 seconds. Give the action as command to the subject, who should try to adjust her mental state to represent the given action. Record the EEG data. After 5 seconds, pause for 5 seconds, then start a new trial as before. Since the subject does not know which action will come next, she is not able to adapt herself to the next action. The pause is done, so she can go back into a relaxed state in order to be "fresh" for the next trial. "Fresh" refers to a mental state that is comparable for each new trial, i.e. at the start of a new trial the subject's state should be in a way unbiased.

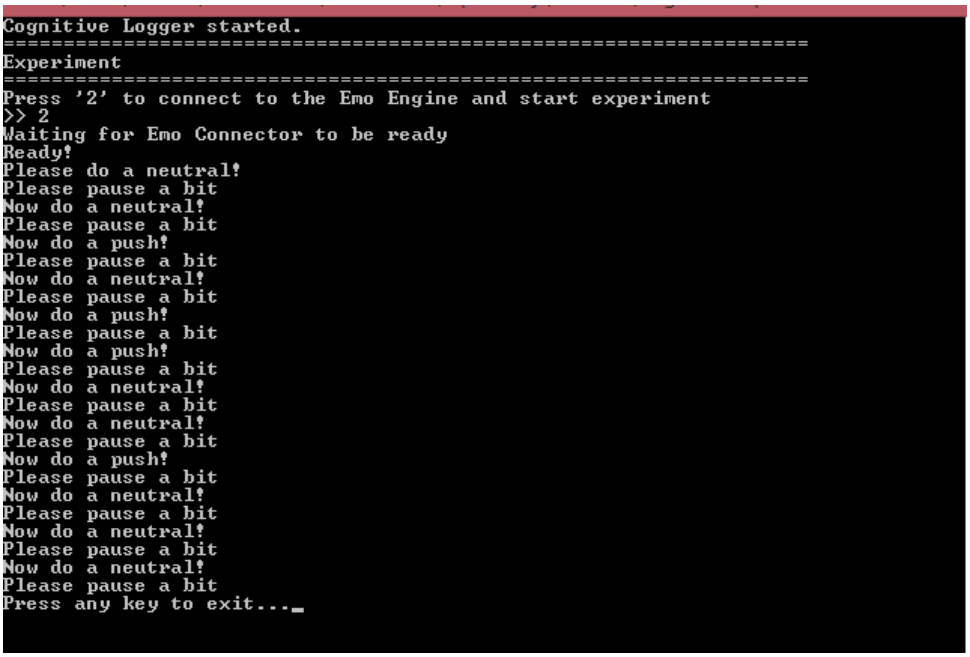


Figure 13: My simple first data collection program.

[5] *http://qt-project.org/*

Cognitive Menu

During RoboCup 2014 German Open, for demonstration purposes, I created a simple menu-based BCI on the idea of selection similar to BCI-Spellers as already discussed in section 4. It takes the certainty value output by a classifier and translates it to a trigger. While rotating through a number of options, if the certainty value exceeds a certain threshold, it holds the option. If the threshold is exceeded often enough within a period, it triggers the option. Since this method only requires a 0–1–classifier it is well-suited to test the application of my classifier. I also tested it using the Emotiv classifier and clearly felt the at least 1 second reaction time. Also, it occurred that I was accidentally activating the menu and choosing some random option, when not paying attention to the system. That is one of the main problems BCIs have as identified by Moore: BCIs do not allow to be turned on or off by the user through her brain signals [8]. Therefore, I spent a lot of time in manually adjusting the triggering and menu parameters to fit at least my own needs and prevent such accidents as well as possible. During "Lange Nacht der Wissenschaften 2014" two random subjects from the visitors willingly tried to use Cognitive Menu after training a few minutes with the Emotiv Cognitive Suite. They both succeeded in reaching enough control to intentionally select a certain one of four options. And, since Cognitive Menu only relies on the classifier's power output I concentrated during my later tests on checking, whether a subject can reach a certain level of control over the power output of my own classifier.

Parameters of Cognitive Menu

Cognitive Menu allows to configure the images shown for each action and the command that is sent to the wheelchair's control program. Also, it is possible to set some trigger parameters. These are: the length of a trigger period, the threshold for the trigger, how long a trigger must be hold. Further, it is possible to set how long the menu will wait until it rotates to the next command.

During creation and first tests, I noticed that with a lot of tweaking of parameters in the command output, weaknesses of the employed classifier can greatly be compensated. At the same time, I noticed that an insufficient classifier performance will also render the BCI uncontrollable. A special consideration is the stability of its output regarding distractions and changes in the environment. It is highly desirable that a user never loses her control unexpectedly due to changes in the environment.

Assumptions of Cognitive Menu

This application assumes that the subject is able to mostly intentionally let the classifier output 1, the so-called non-neutral action. And, most of the time the output is 0, the neutral action. The classifier will from time to time output a wrong class. However, the menu will only react, if the non-neutral action is output sufficiently often. So, I assume, that the subject really intends to trigger, if she holds her non-neutral state for a while.

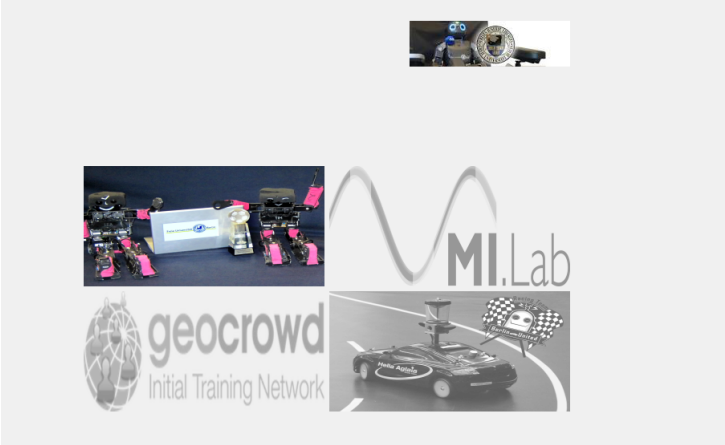


Figure 14: My menu-based BCI prototype as used during “Lange Nacht der Wissenschaften 2014”.

The Analysis program

As already stated, the analysis program is an R-based prototype for the analysis of EEG data and it contains the classifier’s implementation. It handles at least the data transformation using power spectral density, the feature extraction, the classification, the measuring of the classifiers’ performances, user sessions, and the quality measurement of trials. Aside from a collection of functions to do all the above tasks, it contains TCP-server-based interface to allow being used by other applications in a way similar to the Cognitive Trainer’s.

During the operation with Cognitive Trainer, the analysis program showed to perform a bit poorly, slowing down the application when it classifies. Due to that problem, I decided to make Cognitive Trainer’s testing phase twice as long, so I am able to provide enough testing data during one test run for a more reliable performance measure. I could also have done the performance measurement offline, but wanted to concentrate more on an online application, since this has been one goal. As the application is split into two separate processes of which the crucial one is an R interpreter, the run time performance issues should be solved when it is replaced by a C++ implementation of the relevant analysis methods.

The interaction is realised through a TCP connection through which the analysis program provides an interface that supports all commands necessary for the training process, like starting a new trial, classifying, or training the classifiers from collected data. The data handling is totally done by the analysis program similar to the offline case.

The Analysis program allows a lot of configurations to fit different kinds data. It allows among others to change the length of the window to use, which frequencies should be considered when extracting features, the threshold of the classifier for the positive class. These parameters are especially important as they influence the classification performance and the system’s reaction latency and are highly dependent on whether one decided to use for example motor imagery or certain stimuli-based signals like SSVEPs.

As last point, the analysis program assumes to receive data in the following format:

- It needs to be comma separated with header,
- each row should contain one sample, i.e. one value for each channel and a label,
- each row needs to contain the label of the class that row should be in.

Offline Analysis flow

In the offline analysis as seen in figure 15 I evaluate the classifier's performance either through cross-

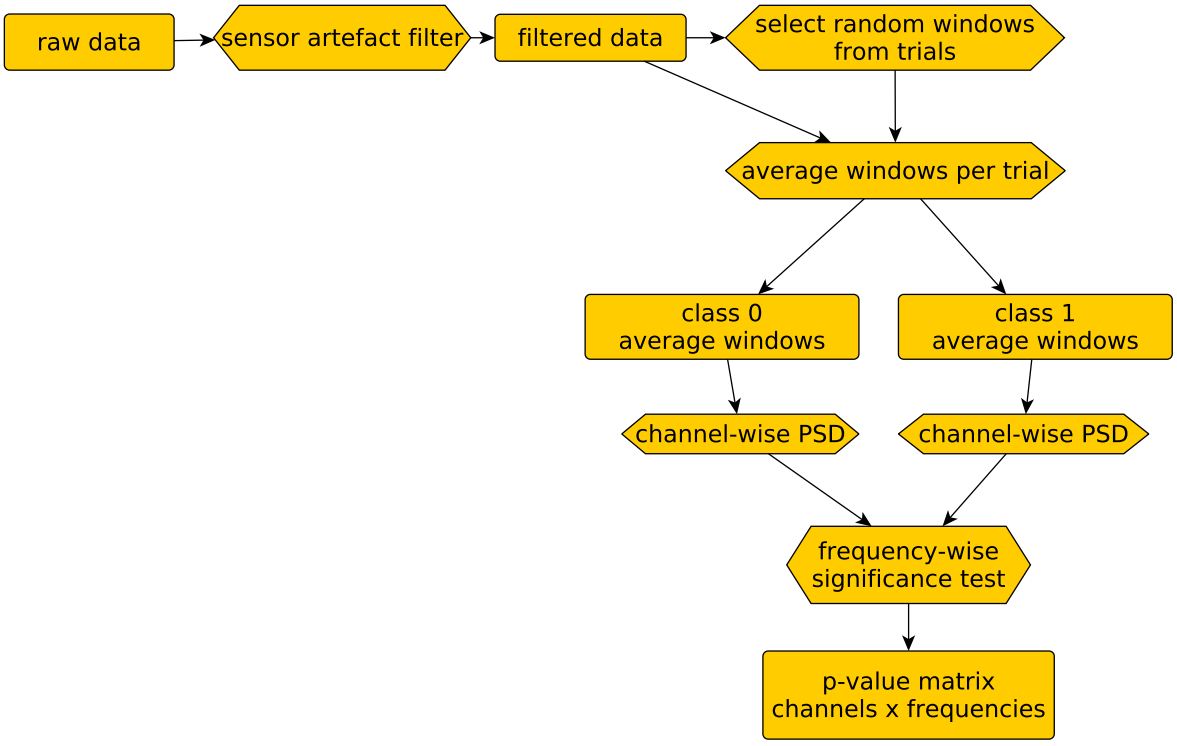


Figure 15: The offline analysis.

validation or using additional test data. As first step from the raw data, sensor artefacts are removed for each possible window, i.e. all windows are checked whether they are valid for classification. Then, in order to be able to classify, the "most important" features are determined from a number of average windows for each class using a significance test. Note, that the significance test is repeated over many different sets of averaged windows and the most often occurring significant features are taken. This should avoid that the test yields significance which resulted from chance. Also note, that the significance test is only univariate, which means that it does not take into account

possible dependencies in different features that may be exploited to distinguish both classes. When the features-to-use are determined, from each trial a number of windows is selected randomly. When cross-validating 90% of the trials are taken randomly for training, the rest for testing. The cross-validation step is repeated a number of times with the same windows but different selections of trials for training and testing. With additional test-data, the classifier is trained once with the training data and then applied to the chosen windows from the test data. In both cases the result is a confusion matrix, from which performance measure can be computed. Also, the probabilities are output, so a ROC graph can be constructed. See section 5.2 for more details to that.

Cognitive Trainer

This application realises my idea of "guided training". It provides two interfaces. One as a bidirectional channel to the underlying analysis program, which receives the data and provides certain analysis output. The second as unidirectional channel from the EEG headset through Emotiv's software development kit. So, it provides the analysis program with EEG data and the user with a way to interact with the analysis program such that it trains a classifier and provides hints on which data to use and which not. The "guided training" idea is realised exactly through the feedback to the user on which trial is of good or bad quality, and the mechanism to remove or reconsider certain trials to use for classifier training.

Parameters of Cognitive Trainer Cognitive Trainer may be compiled to use a dummy classifier for testing purposes that produces high-variance data for one class and low-variance data for the other. It also may be compiled to use the Cognitive Suite classifier from Emotiv so that may be tested using the same procedure as my classifier. However, usually it is compiled without any of both options. For its normal actions it provides some parameters to customise its feedback output. That is, how fast the power is incremented and decremented. Also, the trial lengths and pause lengths may be configured.

Assumptions of Cognitive Trainer Since Cognitive Trainer still incorporates a "power output" built upon the classifier to provide a user with a means of feedback, I put some assumptions into the computation of that power, so it looks reasonable. First, since the classifier has strongly fluctuating class output, it is not possible to just use the rescaled posterior probability it provides. In order to create a smoother output, the probability is used to increase and decrease some sort of impulse, which then determines the power. The power is increased, if the subject is able to hold her state and decreased otherwise. And it stays within the range of $[0, 1]$. So, the desire to increase the power should make the subject try harder, when she sees the power fall or slow down despite her attempt. Also, the parameters are adjusted in a way, that the power does hardly increase, when the subject is in a neutral state. Similar to Cognitive Menu, the aim was to prevent unintentional triggering of a command, although the trainer itself does not output commands to external devices. Cognitive Trainer also assumes that second suffices after the subject reads a command during training before data is recorded. As already explained this serves avoidance of bad data. As a remark, one subject

supposed to use audio feedback to give the training commands, since the currently written commands force a certain concentration on the screen. Therefore, my assumption, that pauses of even lengths and even trial lengths are enough, is not ideal.

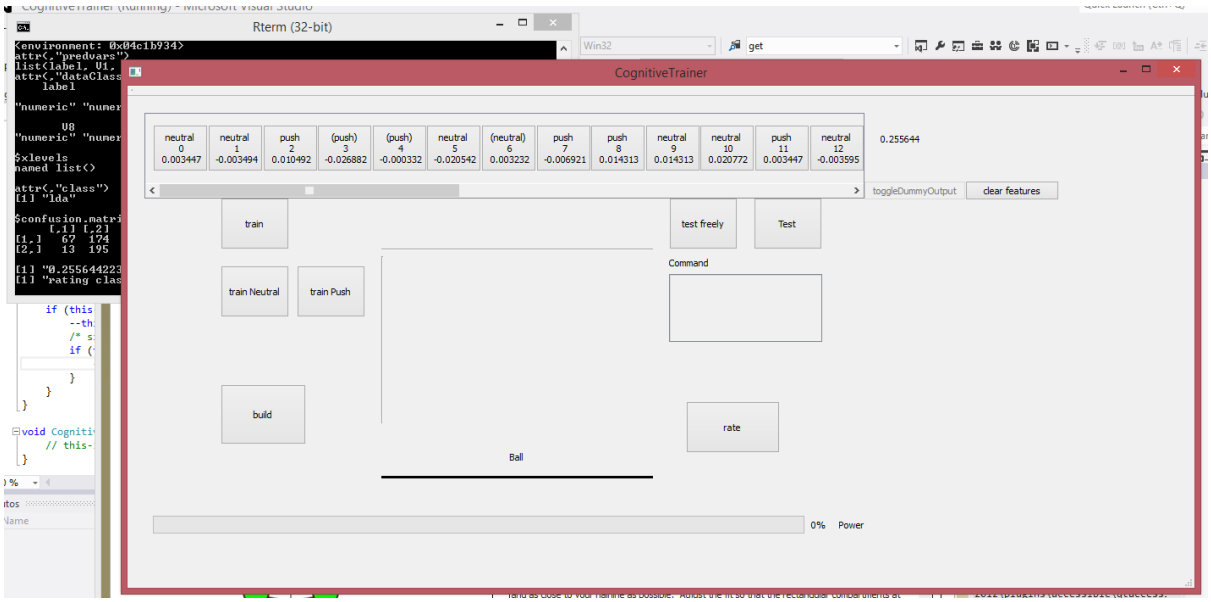


Figure 16: My prototype of a “guided” training.

Process flow

First, I explain the more abstract online process as seen in figure 17 that is implemented by Cognitive

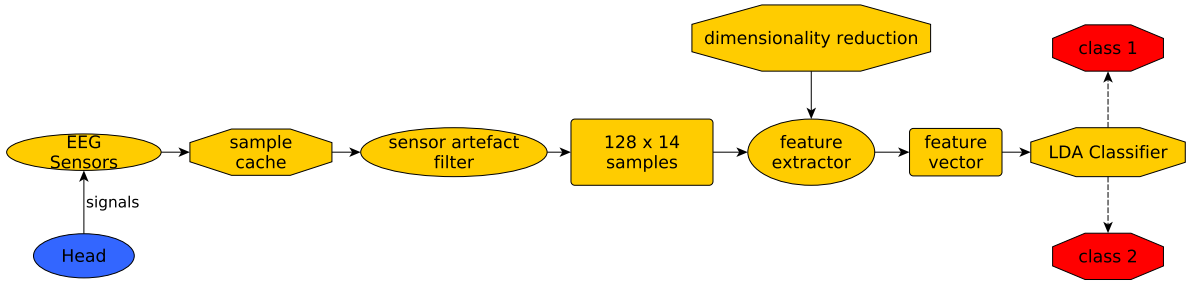


Figure 17: The online process.

Trainer. In the beginning, there is a subject’s head, that creates the signals recorded through EEG sensors. The headset provides me with about 128 measurements at each of the 14 sensors per second. One measurement of 14 values is a sample. Such a sample is cached for 1 second, so I have something like a 1 second sliding window over the recorded data. On classifying such a window, I first check

for sensor artefacts as discussed in section 4. If such a window contains artefacts, it cannot be classified. Otherwise, I can extract its features by first computing power spectral density for each channel on the window. From the resulting power estimations for each frequency up to half of the sample frequency for each of the 14 channels, I take only some specific values. Which values I take has been computed previously from training data using a statistical significance test as described in section 4. The resulting extracted features are then fed into an LDA, which will output a class and a probability for each of the two classes according to its training.

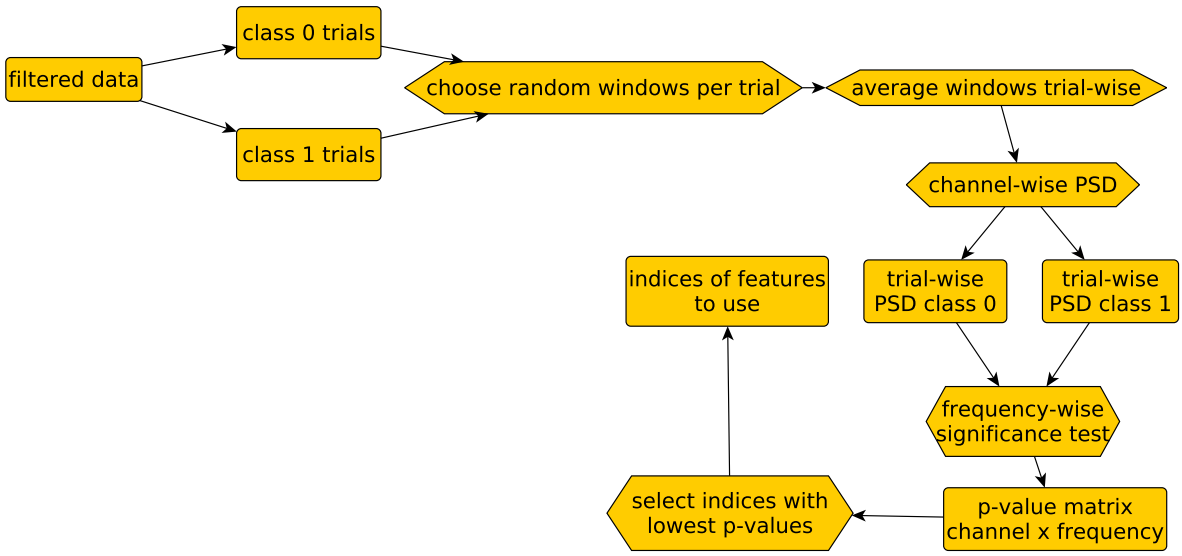


Figure 18: The offline part of feature extraction.

5 Evaluation and Results

After having finished the Cognitive Trainer, I tested it with different subjects. I trained my classifier and measured its performance after a short training phase assisted by me. After that training I also asked for the subjective opinion about the subject's control using that classifier after that training. For that, I used a 5-level rating scale.

Another test was to apply my classifier to the BBCI competition 2014 data set 2B (the two-class case). I wanted to see, whether my scheme yields reasonable results, so that I am highly likely not totally wrong. The reasoning is as follows: Since this data set contains reliably recorded motor imagery data, I can determine, whether my method retrieves features that are typical for motor imagery. That provides me with a justification to use my exact way of feature extraction through a statistical test.

5.1 Tests

BCI Competition data

In order to check, whether my classifier performs reasonably, I took the dataset 2b of the BCI Competition IV [6]. That data set was the only one of that competition with only two classes, therefore I selected it. It contains data from 3 EEG channels for 9 subjects, which performed motor imagery. The recording and filter procedure is described in [40]. Since it is not provided in the data format I used so far, I needed to convert the data. As I wanted to compare my classifier to the results[7] of the competition, I tried to simulate the performance measuring procedure. Cohen's kappa statistics κ has been used as performance measure, which is defined by

$$\kappa := \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)},$$

where $\Pr(a)$ is the rate of correct predictions from all predictions (i.e. $\frac{TP+TN}{\#predictions}$) and $\Pr(e)$ the probability that randomly the correct prediction is made (i.e. $\frac{TP+FP}{\#predictions} \cdot \frac{TP+FN}{\#predictions} + \frac{FN+TN}{\#predictions} \cdot \frac{FP+TP}{\#predictions}$)
with $(TP, FP, TN, FN) =$ (true positive, false positive, true negative, false negative). For each subject there were 5 data sets of which 3 are meant for training and 2 for evaluation purposes, so 5 sessions. The results are given per subject and an average performance. After transforming the data, I preprocess it as always by removing sensor artefacts and then train my classifier with data from all 3 training sessions and evaluate it for every eight window from the two evaluation sessions. I also repeated the training and evaluation process 10 times with different training selections, since during analysis only random training windows are taken. Also, this evaluation is not intended to prove competitiveness, but to give me an intuition, whether my method seems reasonable. Although

[6] *http://www.bbci.de/competition/iv/#datasets*

[7] *http://www.bbci.de/competition/iv/results/index.html#dataset2b*

it has not been prepared sufficiently for that data set. Also note, that the competition data contains additional eye movement sensor data, which can and should be used to filter out eye movement artefacts in the EEG. I did not prepare anything to do so.

Training Sessions

Also, for a real online application test, I made one session of 45 to 60 minutes for a subject and let it train the classifier using Cognitive Trainer. During that, the subject was instructed by me and I also operated Cognitive Trainer itself. After some training runs and test runs, I took the last test performance as result. Also, after the session, I asked the subject how well she felt to have control over the classifier's output within a scale of 1 to 5 with 1 as worst control. I.e. how well the classifier reflected the subject's intention. I repeated that for 4 subjects.

Emotiv comparison session

For only 1 subject I, also, compared the performance to the Emotiv Cognitive Suite classifier. Therefore, I used the same test procedure of Cognitive Trainer. Although, due to different run time performances the Emotiv classifier allows to test more samples in the same time, the results should be somewhat comparable. The procedure was to first complete the Online Training Session from above, then train the Emotiv classifier in one action within a couple of minutes (less than 10, since more data usually led to worse results), and then the evaluation. As I spontaneously tested that with the one person that has the best known control over Emotiv Cognitive Suite, I unfortunately cannot reconstruct the data. At that time I used Matthews Correlation Coefficient as feedback in the CognitiveTrainer. First using my classifier it resulted in about 0.4 as overall performance after a test run. Then using the shortly trained Emotiv classifier, the result was slightly better in an equal test run within CognitiveTrainer. It was about 0.5. As this is not much data, I cannot really conclude much from that. I and the subject think that control over both classifiers is about the same and the slightly better performance comes from far more experience in using Emotiv's classifier. Also, 3 subjects had some longer experience with the Emotiv Cognitive Suite and each said the control over the power output my classifier felt about the same as with Emotiv's, even after only a short time (less than 1 hour) working with mine. Also, the last subject easily became very confident in the control over the power output and in the comprehension of when and why the classifier reacts or not.

5.2 Results and Interpretation

BBCI Competition data

The classifier itself is not competitive in terms of performance, as expected. Nonetheless, that is not what I wanted to check, since for high performance results it is necessary to use highly sophisticated preprocessing and feature extraction operations that are dependent on the task used for training. The check I wanted to do, is to find out, whether given the mental task motor imagery, does my feature extraction method select features from bands, that are related to the sensorimotor rhythm, which is

considered to be in the 12–15 Hz band for most people, after Omar [30], or 13–15 Hz, if you look it up in Wikipedia[8].

I checked the frequencies for the best 4 subjects in terms of kappa for the best research group and restricted the frequency band to 1–64Hz. The results are found in table 2.

Subject	list of selected (channel, frequency)-feature
B4	(C3, 11), (C3, 12), (C3, 13), (C4, 13), (C3, 14), (C4, 14), (C3, 21), (C4, 21), (C3, 20), (C3, 22)
B5	(C3, 3), (C4, 13), (C4, 14), (C4, 15), (C3, 4), (C4, 20), (C4, 18), (C4, 12), (C3, 23), (C4, 21)
B8	(C4, 11), (Cz, 12), (C4, 12), (C3, 13), (C4, 13), (Cz, 23), (C4, 23), (C4, 24), (C4, 34), (C3, 14)
B9	(C3, 13), (C4, 13), (C3, 14), (C4, 14), (C3, 15), (C3, 22), (C3, 23), (C3, 12), (C3, 18), (C4, 20)

Table 2: Results from the analysis of selected features from the BBCI competition data.

As stated the desired band to select is about 12–15 Hz. As one can see, for each subject about half of the features selected are from that band. Even with the more narrow definition of the sensorimotor rhythm band, there are at least 3 features each. Also, after Omar [30], the selected channels should be near C3 or C4. And from the 20 selected features from the desired band, only 1 is from Cz and not from C3 or C4. Of course, there were only 3 channels to select from so, the probability to select the correct one given a frequency was already $\frac{2}{3}$, but in the result only $\frac{1}{20}$ was chosen wrongly, which is only 15% of $\frac{1}{3}$, so I conclude the channel selection to not be random. I also tried tried to reproduce the above results a few times, since the process relies an randomness. For subjects B5, B8, and B9 it appeared that the above results are above average, which means that very often less that 3 features from the desired band are selected. For subject B4 however it turned out to be always the same. I computed a ratio of how often the selected features have been considered as best features. For subject B4 these values lied in 60–100% of the cases. I.e. a certain feature as (C3, 11) has been considered as most significant each time. For the other subjects these ratios lied around only 40%. Altogether, that still means, that, as subject $B4$'s data gave by far the best overall performance for all groups that participated in that competition, it most likely contained the best quality motor imagery data. Therefore, I can conclude, that if the data is such that both classes are best distinguished through signals stemming from motor imagery, my method will find features in a frequency band that connected to motor imagery.
Lastly, only half of the selected features were from desired band. That means, either the method is not reliable enough, the data contains artefacts that make the desired bands much more noisy — which is the case, and which is why Electrooculography data (recording of eye-movements) is provided to reduce such artefacts — and I did not remove those artefacts explicitly, or that there are merely not enough relevant features.

To the last point that there are insufficiently many relevant features, when I checked the p-values for the selected features, they were below 0.05 mostly only in 5–7 features. Also, the selection ratio fell drastically after 5–7 features, which means, there are, indeed, not more than 7 features that can

[8] *http://en.wikipedia.org/w/index.php?title=Sensorimotor_rhythm&oldid=639330429*

be considered relevant from my method. The outcome for the last 3–5 is therefore quite random. To the first point of reliability of my method, for subject $B4$ it turned out to be highly stable and reliable in terms of selecting features as expected. For the rest of the subjects the data has been too noisy such that, considering the subjects performed motor imagery, the desired band is not necessarily used as far as possible. The second point, the missing artefact removal may also explain this noise. After Omar [30] such artefacts have the highest impact on motor imagery signals.

All in all, the method yields reasonable results. However, the preprocessing needs to be more sophisticated in reference to the data to classifier.

Training Sessions

From the experiments, I computed accuracy, precision, recall, and Cohen's Kappa. Also, I made a cross-validation for each subject's data to determine a ROC graph, that relates true positive rate to false positive rate, and the projection of the test data as the LDA creates it. The projection visualises how well the LDA is able to distinguish both classes. The ROC graph provides hints to the threshold to choose for classification. I.e. at which posterior-probability the LDA should classify a sample as member of class 1. The ROC (Receiver operating characteristic) graph visualises the true positive rate against the false positive rate for different thresholds for the positive class. So, it shows which threshold should be used for the desired trade-off between correct predictions against false positives. It can only be used for binary classifiers. It can be used to determine an optimal threshold for that measure. Accuracy shows the size of the correctly classified test data portion. Precision and recall are usually used in a pair to show how well a classifier works on finding the positive class. Precision gives the portion of correctly classified positives from all classified positives. While recall gives the portion of true positives from all samples classified as positive. Cohen's Kappa has also been used in the BBCI competition, and gives hints to overall performance in respect to both correct classifications and false classifications. It combines the performance hints of accuracy, precision, and recall to some extent. A high kappa value hints to good overall performance similar to the MCC mentioned earlier. In fact, both values differed in at most 0.01 during the evaluation. At 0 it represents total randomness in classification.

The results are listed in table 3.

For me, the subjective control plays a very important role, as a BCI should be controllable by a user. Each subject except for 1 rated her control with 3 of 5. The best rating was 4 of 5. And each subject said, it would probably increase her control after some more training. Mainly, since the subject started to understand how the classifier reacts and how she could manipulate the power output at will. She would adapt more to the system. Ideally, the system should also adapt to the user, however, for now, it does so only during training. Besides that, everyone noticed, that the classifier is vulnerable to some distractions and to potentials from actions like breathing or eye movements. For instance, if a subject trains the concentrated action while always holding breath during the trials, the classifier does not output that action sufficiently often when breathing. Some of these effects may be avoided by altering the training method, however, the better alternative would be to extend

Session	#trials	#samples for testing	accuracy	precision	recall	kappa	subjective control
Subject 1	30	449	0.57	0.26	0.83	0.19	3 to 4 with training
Subject 2	28	434	0.54	0.19	0.65	0.08	3 to 4 with training
Subject 3	20	364	0.76	0.86	0.72	0.51	4 to 5 with training
Subject 3 b^*	21	251	0.68	0.45	0.89	0.38	- - - - -
Subject 3 c^*	20	258	0.54	0.40	0.52	0.06	- - - - -
Subject 3 d^*	19	278	0.70	0.71	0.77	0.39	- - - - -
Subject 4	20	320	0.77	0.65	0.90	0.55	3 to 4 with training

(* — these results have been determined using cross-validation, in contrast to realtime test data)

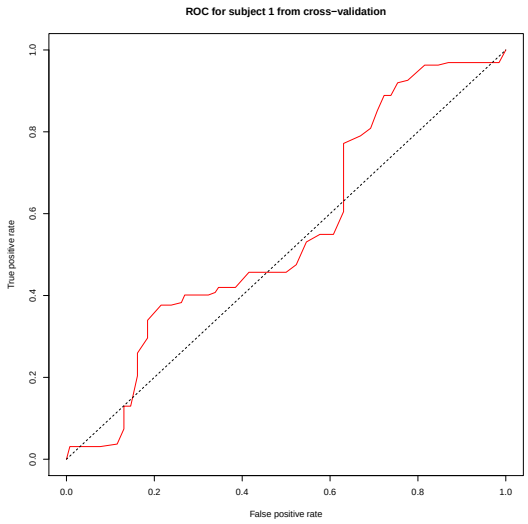
Table 3: The results from the training and test sessions I did with some subjects.

the preprocessing to remove such artefacts. Also, when looking at the LDA—projections, one can see, that the LDA has many problems in distinguishing both classes with the features I employed. The distance between the classes is really small and they are partly overlapping. Overlapping partly stems from the labelling problem, i.e. trials may contain data that does not represent the desired action. Although the impact of that problem does not seem to be large. Additionally, there are some outliers. I computed for each subject the distance of the farthest member from the mean in the projection in terms of standard deviations. These distances were between 2 and about 10 standard deviations. Only a few members were far away, if I consider more than 3 standard deviations as far. So, removing outliers may have an effect on the performance, but it should also not be that large. With the comparison I have from the BBCI data evaluation, I assume that an improvement in artefact detection and removal and in feature extraction should have the greatest impact. The classifier model itself should not play an important role here. I would even prefer to stick with the LDA because of its stability and application and training speed. Nonetheless, even with my quite primitive prototype a user can get some decent control over a single trigger. However, before extending the classifier to multi-class some performance improvements are necessary. And, as already stated, these can mostly be made in altering preprocessing and feature extraction steps.

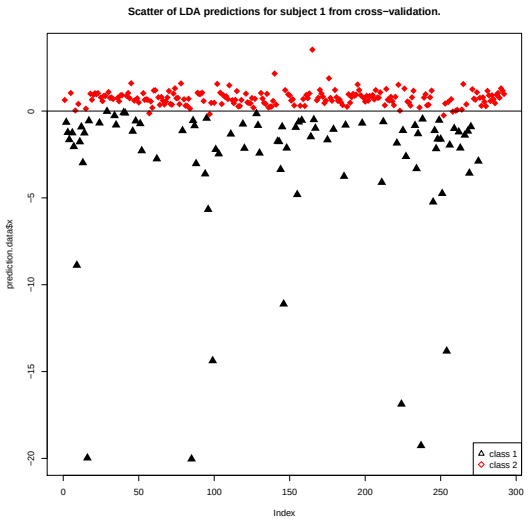
Graphical representation of the results From the results in figure 19, one can see that it would be possible to set the classifier's threshold to about 0.7, if the neutral action is more favourable, or to about 0.4 to favour the non-neutral action and with it to make increasing the power output easier. In (b) and (c) the distinguishability of both actions is displayed from the point of view of the classifier. The red line and the red squares represent the non-neutral action, which is for more dense than the neutral action. This difference in density is expected, as during training the subject was very concentrated on the non-neutral action, while being very quite distracted during neutral action. This distraction should have made the produced data more noisy. And that can be seen in these two graphs.

Subject 2's data (figure 20) is more noisy as one can see. It also was harder to classify as expected. The performance display during training was already quite low. Through the quality hints, however,

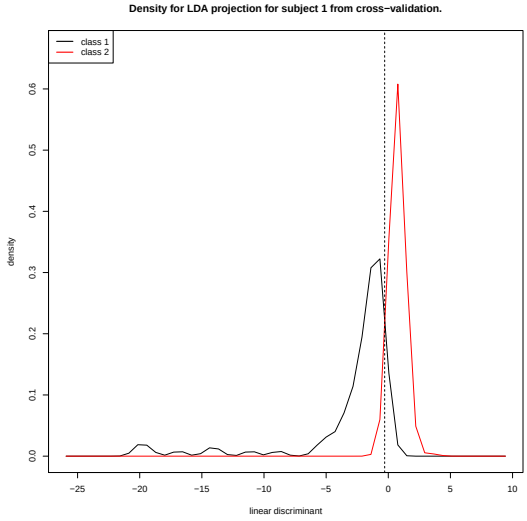
it could be raised to the presented performance. This subject had many problems maintaining a stable concentrated state. Nonetheless, subjectively spoken, the classifiers performance was decent enough as it has been rated with a 3 of 5 possible points. In fact, that subject had got the most benefit out of the guided training of all subjects.



(a) ROC graph for subject 1 from cross-validation.

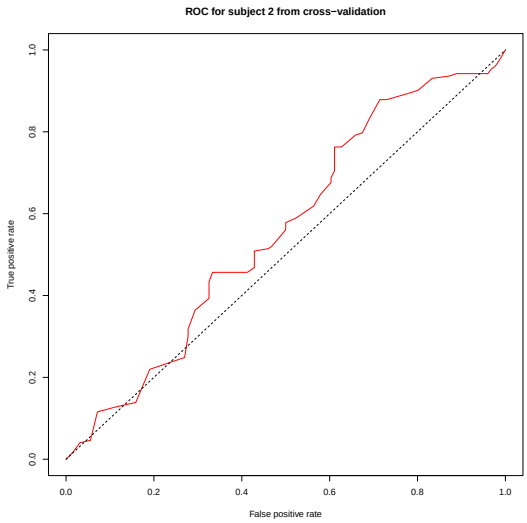


(b) Projection of the test data by the LDA from cross-validation. It can easily be seen that class 1 is widespread (the neutral action), while class 2 is quite concentrated (the concentrated action). Nonetheless, it is apparently hard to distinguish both classes for the LDA.

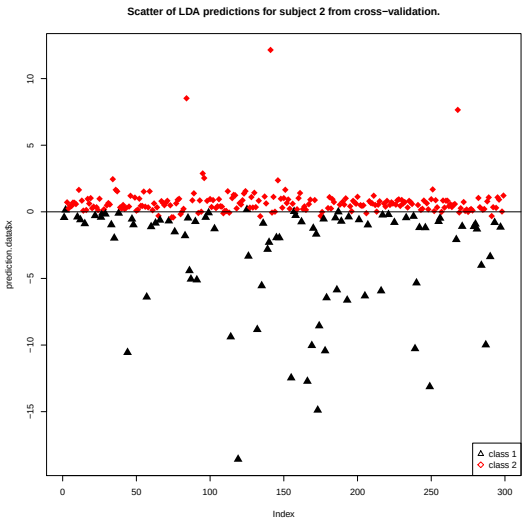


(c) Projection of the test data by the LDA expressed as density functions.

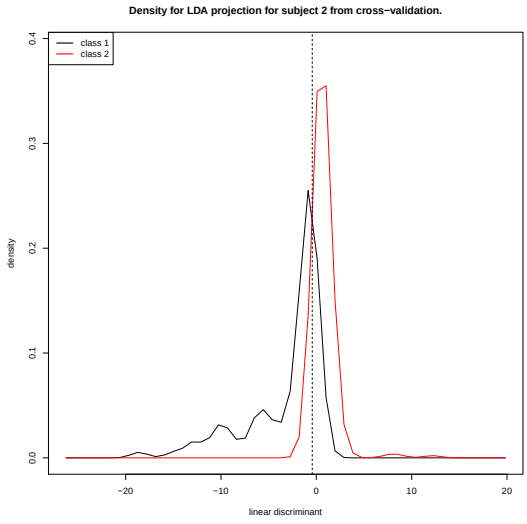
Figure 19: Graphs from the results for subject 1.



(a) ROC graph for subject 2 from cross-validation.



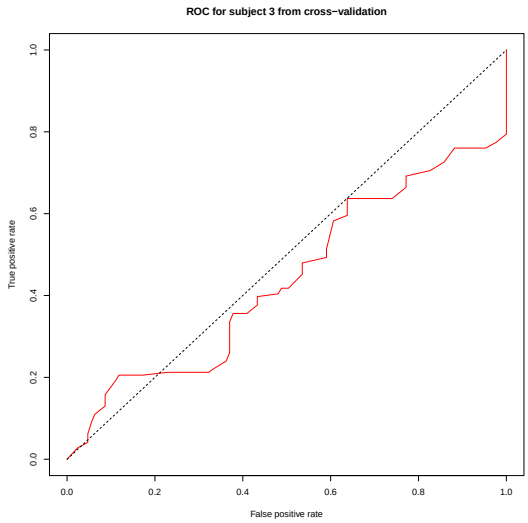
(b) Projection of the test data by the LDA from cross-validation for subject 2.



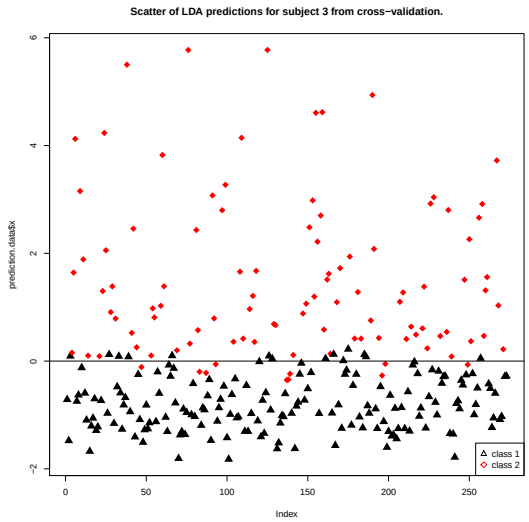
(c) Projection of the test data by the LDA expressed as density functions.

Figure 20: Graphs from the results for subject 2.

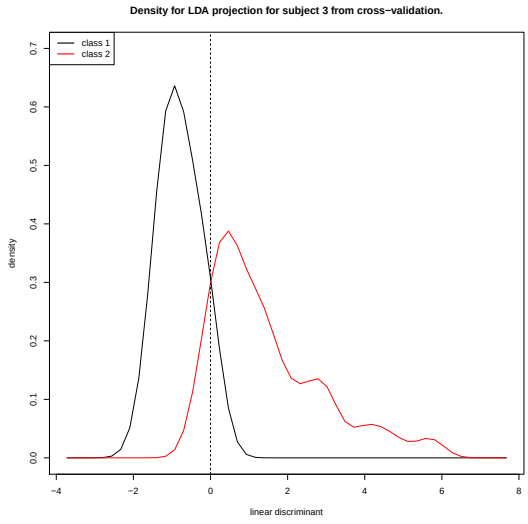
Subject 3 (figure 21) had the most experience with this headset. While viewing the control over the power output, it looked responding. During the comparison with the Emotiv classifier, my classifier provided about the same control. Nonetheless, the graphs are really surprising. And I must admit, I do not really understand the graphs. The performance from the real-time test data is very good, however, these cross-validation results look worse than subject 1's. The kappa is with 0.37 also far worse than during the test session (where it was 0.51). So, maybe the training data was insufficient during cross-validation, which may stem from the fact that the classes are not very dense.



(a) ROC graph for subject 3 from cross-validation.

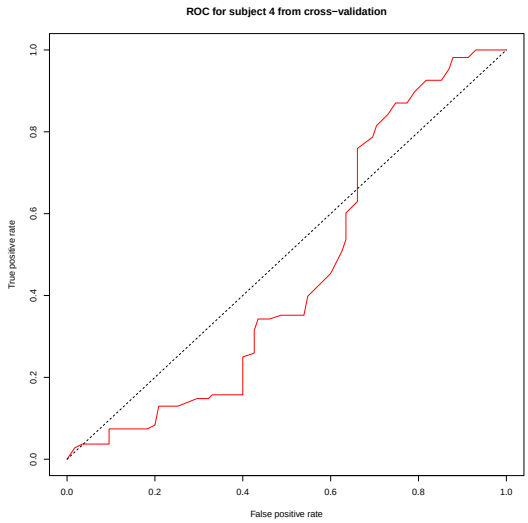


(b) Projection of the test data by the LDA from cross-validation for subject 3.

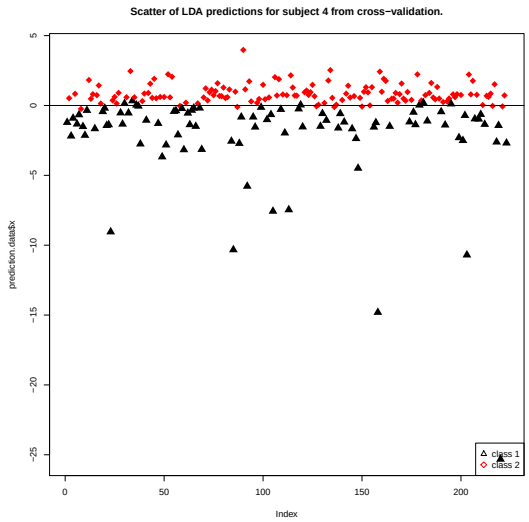


(c) Projection of the test data by the LDA expressed as density functions.

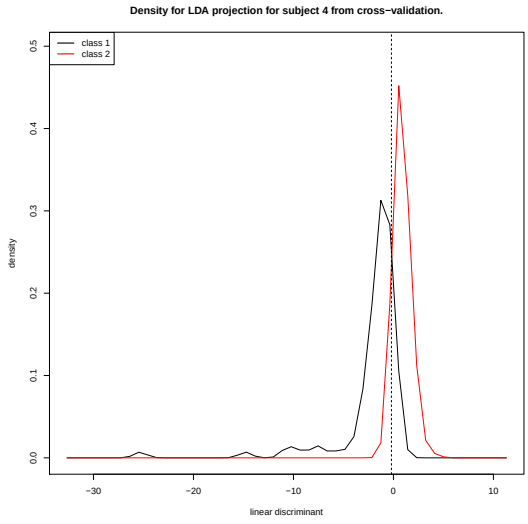
Figure 21: Graphs from the results for subject 3.



(a) ROC graph for subject 4 from cross-validation.



(b) Projection of the test data by the LDA from cross-validation for subject 4.



(c) Projection of the test data by the LDA expressed as density functions.

Figure 22: Graphs from the results for subject 4.

Subject 4 (figure 22) performed surprisingly well. The classes are similar distributed as in the results of subject 1. They are both quite dense and well-distinguishable. Also the control over the power output I could watch during the test session was outstanding. It seems that subject 4 has quite a talent for using EEG-based systems.

6 Conclusion

During course of this work the question, whether it is possible to create a brain computer interface upon the affordable Emotiv EPOC EEG headset, such that it enables a user to command an arbitrary computerised device has been examined. A proof-of-concept system has been designed, implemented, and evaluated whether it works. Employing linear discriminant analysis as classifier, the Mann-Whitney-Wilcoxon test to assist feature extraction, and the power spectral density as feature space, a working system prototype has been created, that use merely 10 features. Due to the fast processing that linear discriminant analysis allows, a form of guidance could be provided during training that enables a user to improve the classifier's performance and let it reflect her intents more often. This guidance was very useful for one of the four subjects.
Also, the statistical test was examined on whether it performs in a way that can be expected. Therefore, some of the BBCI competition data has been employed.
The control provided by the developed system has been sufficient to conclude that it provides enough control that a user can command an arbitrary computerised device. Also, it showed to be easily trainable.

Discussion Although, the overall performance measures are shown to be not very good, the system worked quite well for all subjects in terms of control it provides. Its reactions are also quite stable and predictable. However, due to the signal source of BCIs — the brain — between tests in a laboratory and in a crowded, distracting outside world there is a very huge difference. During tests using the Emotiv classifier at the crowded hall of RoboCup 2014, I noticed clearly that such distractions may render a BCI totally uncontrollable. In that sense, the results from my laboratory tests above are still insufficient.
Nonetheless, before considering more extreme tests, it is highly desirable to improve the classifiers performance. To that extent, improvements can be made at preprocessing level, feature extraction level, the feature translation, the device control level, or the training method level.
As for more sophisticated feature extraction the statistical test may be improved to be better suited for the encountered distributions. Maybe a combined test can yield better results. However, especially during preprocessing many sources of noise need to be reduced. Therefore, more specific signals should be used in a specific training method. From that, it is necessary to determine noise sources and ways to remove them. One method that is a quite general approach for that case, is Common spatial patterns, which might be applicable using this headset. Also, multivariate analyses for feature extraction could be considered.
Furthermore, to further clean the training data it may be successful to remove outliers using the Mahalanobis distance.
Considering the type of classifier, linear discriminant analysis seems to have been a good decision. Although, more complex models could have abstracted from certain noise, their output would have become less predictable and less comprehensible. However, that requires specific noise reduction during preprocessing.

A lot of filter methods and transformations are found in the literature, that might improve the classifier's performance.

Outlook All in all, the system needs more sophistication in the choice of signals to use, and therefore in preprocessing steps. With that it is possible to examine the best ways to reduce noise and extract the most useful features. A first specialisation to try could be motor imagery, since from the BBCI competition data alone one would have a lot of high quality data as reference, as well as, competitors' performance measures.
A different approach would be to further test and design the system using data recorded through a higher quality equipment and from that adapt to the affordable one.
Also, as there exist many tools that could also be used in conjunction with the present system. It would be desirable to integrate these. I already considered using the tools of bci2000[9] or OpenVibe[10], however, I could not bring either in a state working for us. That may be a topic to concentrate on, very soon, since such a tool set would not only make work easier, but also make the present system easier to integrate for others. Also, they both seem not to be suited for offline analysis. Unfortunately quite late, I found one Emotiv-based OpenVibe application under [41].
Regarding the performance issues, these could be solved using the already existing C++-based analysis library created by Omar to implement the prototype's methods.
Especially important are steps for supporting multiple kinds of signals, preprocessing types, feature selection, and training methods in order to allow many different subjects to be able to use the system as well as possible. Since, the performance of a BCI has been shown to be highly dependent on the subject [28].

As a last point, I want to point out that no matter how well the BCI works, it will not always be the case that its output reflects the user's intent. At each point unintended commands may be given. Therefore, each device that shall be controlled using a BCI needs to be prepared for that with appropriate safety measures.

[9] *http://www.schalklab.org/research/bci2000*
[10] *http://openvibe.inria.fr/*

7 Bibliography

[1] L. F. Nicolas-Alonso and J. Gomez-Gil, "Brain computer interfaces, a review," *Sensors*, vol. 12, no. 2, pp. 1211–1279, 2012.

[2] D. Göhring, D. Latotzky, M. Wang, and R. Rojas, "Semi-autonomous car control using brain computer interfaces," in *Intelligent Autonomous Systems 12*. Springer, 2013, pp. 393–408.

[3] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain–computer interfaces for communication and control," *Clinical neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.

[4] S. Waldert, T. Pistohl, C. Braun, T. Ball, A. Aertsen, and C. Mehring, "A review on directional information in neural signals for brain-machine interfaces," *Journal of Physiology-Paris*, vol. 103, no. 3, pp. 244–254, 2009.

[5] D. J. McFarland, C. W. Anderson, K. Muller, A. Schlogl, and D. J. Krusienski, "Bci meeting 2005-workshop on bci signal processing: feature extraction and translation," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 14, no. 2, p. 135, 2006.

[6] M. Teplan, "Fundamentals of eeg measurement," *Measurement science review*, vol. 2, no. 2, pp. 1–11, 2002.

[7] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, B. Arnaldi *et al.*, "A review of classification algorithms for eeg-based brain–computer interfaces," *Journal of neural engineering*, vol. 4, 2007.

[8] M. M. Moore, "Real-world applications for brain-computer interface technology," *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 2, pp. 162–165, 2003.

[9] P. W. Ferrez and J. del R Millan, "Error-related eeg potentials generated during simulated brain–computer interaction," *Biomedical Engineering, IEEE Transactions on*, vol. 55, no. 3, pp. 923–929, 2008.

[10] C. Vidaurre, C. Sannelli, K.-R. Müller, and B. Blankertz, "Machine-learning-based coadaptive calibration for brain-computer interfaces," *Neural computation*, vol. 23, no. 3, pp. 791–816, 2011.

[11] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial eeg during imagined hand movement," *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 4, pp. 441–446, 2000.

[12] B. Blankertz, M. Kawanabe, R. Tomioka, F. Hohlefeld, K.-r. Müller, and V. V. Nikulin, "Invariant common spatial patterns: Alleviating nonstationarities in brain-computer interfacing," in *Advances in neural information processing systems*, 2007, pp. 113–120.

[13] T. Carlson and J. d. R. Millán, "Brain-controlled wheelchairs: a robotic architecture," *IEEE Robotics and Automation Magazine*, vol. 20, no. EPFL-ARTICLE-181698, pp. 65–73, 2013.

[14] B. Rebsamen, "A brain controlled wheelchair to navigate in familiar environments," Ph.D. dissertation, National University of Singapore, 2009.

[15] C. Mandel, T. Luth, T. Laue, T. Rofer, A. Graser, and B. Krieg-Bruckner, "Navigating a smart wheelchair with a brain-computer interface interpreting steady-state visual evoked potentials," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 1118–1125.

[16] 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, p. 299, 2006.

[17] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[18] L. Schwardt and J. du Preez, "Linear discriminant analysis (lda)," last visited on 2014-12-11. [Online]. Available: http://courses.ee.sun.ac.za/Pattern_Recognition_813/lectures/lecture01/node6.html

[19] C. M. Bishop *et al.*, *Pattern recognition and machine learning*. springer New York, 2006, vol. 1.

[20] R. Rojas, "The secret life of the covariance matrix," *Freie Universität Berlin [online], URL: http://www. inf. fu-berlin. de/inst/ag-ki/rojas_home/documents/tutorials/secretcovariance. pdf [cited 1 October 2012]*, 2009.

[21] Y. K. Cheung and J. H. Klotz, "The mann whitney wilcoxon distribution using linked lists," *Statistica Sinica*, vol. 7, no. 3, pp. 805–813, 1997.

[22] P. Pittner, "An algorithm for the mann-whitney u-test," *Biometrical Journal*, vol. 23, no. 1, pp. 105–107, 1981.

[23] K. Yatani, "Mann-whitney's u test," last visited on 2014-11-12. [Online]. Available: http://yatani.jp/teaching/doku.php?id=hcistats:mannwhitney

[24] Y. Shmaliy, *Continuous-time signals*. Springer, 2006.

[25] http://en.wikibooks.org, "Electronics/voltage, current, and power," last visited on 2014-12-07. [Online]. Available: http://en.wikibooks.org/wiki/Electronics/Voltage,_Current,_and_Power

[26] H. Blanton, "Power spectral density - the basics," last visited on 2014-12-07. [Online]. Available: http://faculty.etsu.edu/blanton/lab_3_psd.doc

[27] A. J. Barbour and R. L. Parker, "psd: Adaptive, sine multitaper power spectral density estimation for R," *Computers and Geosciences*, vol. Accepted, 2013, in revision.

[28] M.-C. Dobrea and D. M. Dobrea, "The selection of proper discriminative cognitive tasks—a necessary prerequisite in high-quality bci applications," in *Applied Sciences in Biomedical and Communication Technologies, 2009. ISABEL 2009. 2nd International Symposium on*. IEEE, 2009, pp. 1–6.

[29] Computers and I. Structures, "Can i use power-spectral-density analysis to evaluate human-induced vibrations?" last visited on 2014-12-26. [Online]. Available: https://wiki.csiamerica.com/display/kb/Power-spectral-density+FAQ

[30] O. Mendoza Montoya, private conversations.

[31] E. Kasuya, "Mann–whitney u test when variances are unequal," *Animal Behaviour*, vol. 61, no. 6, pp. 1247–1249, 2001.

[32] S. J. Raudys and A. K. Jain, "Small sample size effects in statistical pattern recognition: Recommendations for practitioners," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 13, no. 3, pp. 252–264, 1991.

[33] H. Ekanayake, "Scalp locations covered by emotiv epoc," last visited on 2014-12-01. [Online]. Available: http://neurofeedback.visaduma.info/images/fig_emotivpositions.gif

[34] D. J. McFarland, W. A. Sarnacki, and J. R. Wolpaw, "Brain–computer interface (bci) operation: optimizing information transfer rates," *Biological psychology*, vol. 63, no. 3, pp. 237–251, 2003.

[35] S. Matsumoto, Y. Kamei, A. Monden, and K.-i. Matsumoto, "Comparison of outlier detection methods in fault-proneness models," in *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*. IEEE, 2007, pp. 461–463.

[36] B. W. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[37] J. A. Fails and D. R. Olsen Jr, "Interactive machine learning," in *Proceedings of the 8th international conference on Intelligent user interfaces*. ACM, 2003, pp. 39–45.

[38] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2013. [Online]. Available: http://www.R-project.org/

[39] W. N. Venables and B. D. Ripley, *Modern Applied Statistics with S*, 4th ed. New York: Springer, 2002, iSBN 0-387-95457-0. [Online]. Available: http://www.stats.ox.ac.uk/pub/MASS4

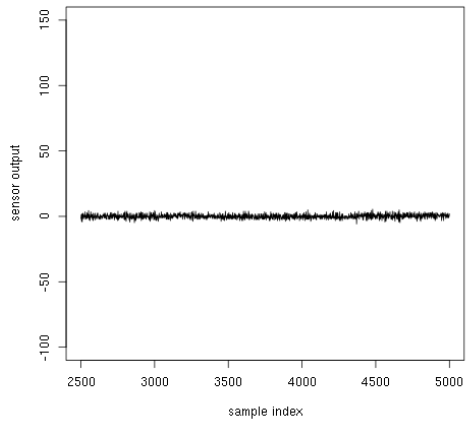
[40] R. Leeb, C. Brunner, G. Müller-Putz, A. Schlögl, and G. Pfurtscheller, “Bci competition iv data set 2b description,” last visited on 2014-11-30. [Online]. Available: http://www.bbci.de/competition/iv/desc_2b.pdf

[41] H. Ekanayake, “Research use of emotiv epoc,” last visited on 2014-12-01. [Online]. Available: http://neurofeedback.visaduma.info/emotivresearch.htm

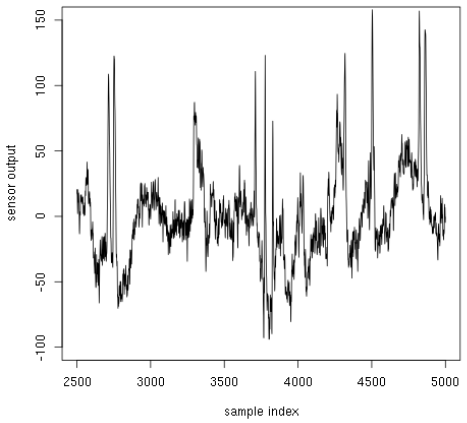
8 Appendix

8.1 Signal-to-noise ratio estimation

Since the EEG signals itselves are already quite noisy, it is important to have high-quality equipment that prevents environment noise. To see how the Emotiv EPOC EEG headset performs on environmental noise I also estimated its signal-to-noise ratio by first recording for a while without any head and then a while wearing it. Since there is only a single sensor at each location, I could not compare both signals at the same time.



(a) The raw signal without any wearer.



(b) And the signal with wearer.

Figure 23: Signals from recordings for noise estimation.

The signals looked about as in figure 23.
This resulted in the following ratios per channel:

AF3	F7	F3	FC5	T7	P7	O1
17.28	22.83	17.37	19.34	17.76	17.51	19.26
O2	P8	T8	FC6	F4	F8	AF4
15.61	17.46	17.35	16.32	17.04	19.21	15.35

Table 4: Results from the signal-to-noise ratio estimation regarding sensor noise from the environment.

Which results into an average signal-to-noise ratio of about 17 to 18 dB. That is really poor in general. Unfortunately, I did not find a reference that provides signal-to-noise data for more advanced equipment.

8.2 Taxonomy of feature extraction methods in context of BCI development

After [5]:

(1) Time

- (a) Fourier transform
- (b) Wavelets
- (c) Autoregressive models
- (d) Bandpass filtering
- (e) Template Matching
- (f) Kalman filter
- (g) Spike detection

(2) Space (refers to the origin of the signal)

- (a) Laplacian filter
- (b) Principal components analysis
- (c) Independent components analysis
- (d) Common spatial patterns
- (e) Amplitudes
- (f) Ratios and differences

(3) Time-Space

- (a) Components analysis in time and space
- (b) Multivariate autoregressive models
- (c) Coherence

(4) Inverse models

- (a) EEG to ECoG
- (b) EEG to source dipoles

8.3 Taxonomy of feature translation methods in context of BCI development

After [5]:

(1) Linear

  (a) Linear discriminant analysis

  (b) Perceptron

  (c) Regression, regularised or adaptive

(2) Nonlinear

  (a) Fixed structure

    i. Quadratic discriminant analysis

  (b) Modifiable structure

    i. Memory-based

      A. k-nearest-neighbours

      B. Support vector machines

      C. Partial least squares

    ii. Combinations of simple nonlinearities

      A. Neural Networks

      B. Decision trees

      C. Learning vector quantization

    iii. Generative models

      A. Mixture of Gaussians

      B. Hidden Markov models

8.4 List of figures