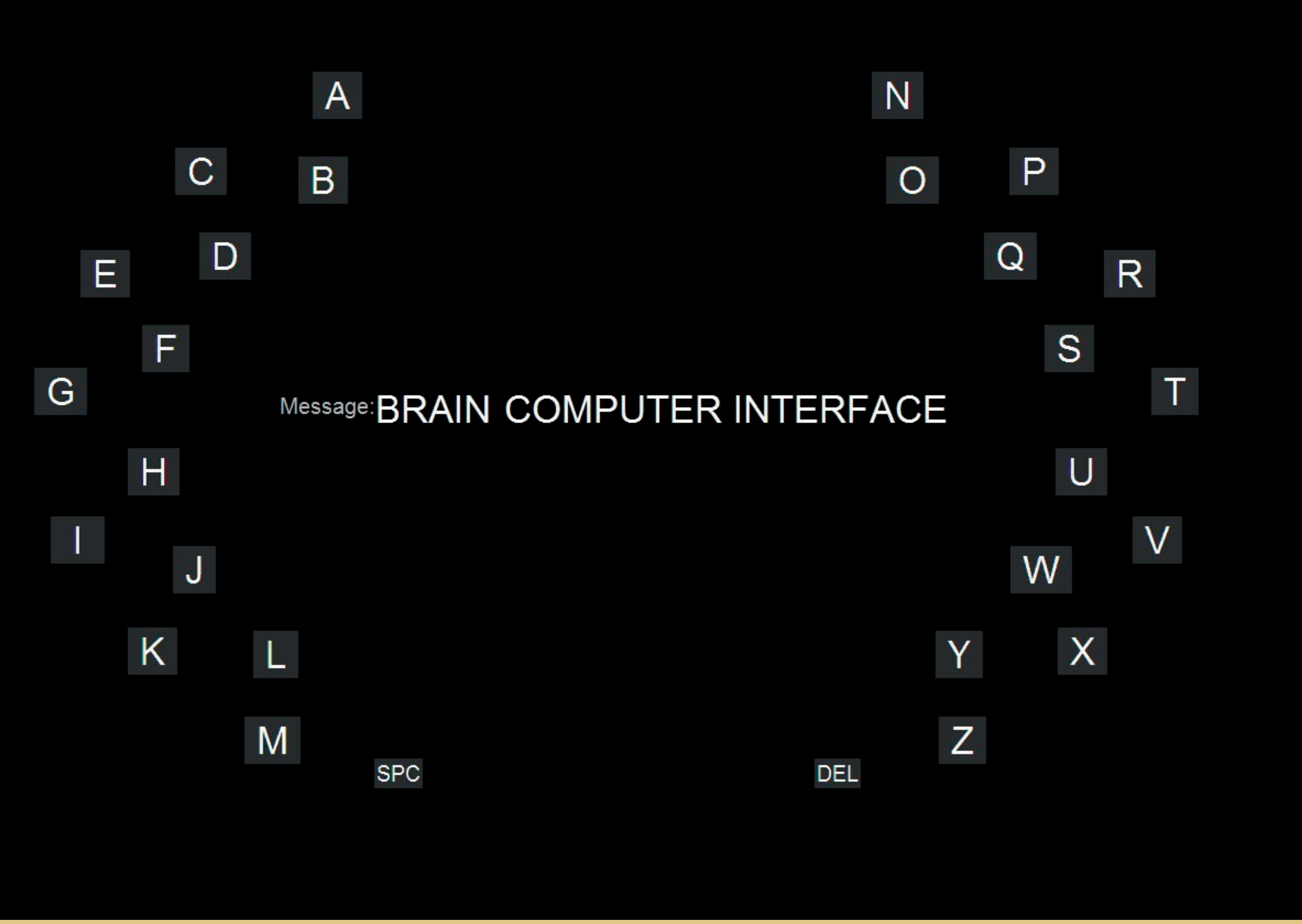


Joana Henriques de Figueiredo

RESEARCH OF ADAPTIVE CLASSIFICATION TECHNIQUES TOWARDS ONE-TIME CALIBRATION IN BRAIN-COMPUTER INTERFACES

Setembro de 2014

• U C •

Universidade de Coimbra

· U C ·

Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Eletrotécnica e de Computadores

Research of adaptive classification techniques towards one-time calibration in brain-computer interfaces

Joana Henriques de Figueiredo

Coimbra, 2014

Research of adaptive classification techniques towards one-time calibration in brain-computer interfaces

Joana Henriques de Figueiredo

Departamento de Engenharia Eletrotécnica e de Computadores

Faculdade de Ciências e Tecnologia, Universidade de Coimbra

Setembro, 2014

Agradecimentos

Em primeiro lugar gostaria de agradecer aos meus orientadores, Professor Doutor Urbano Nunes e Professor Doutor Gabriel Pires, por me darem a oportunidade de conhecer e trabalhar numa área tão fascinante e complexa como é a interface cérebro-computador, pelo tempo que dedicaram a completar e solidificar a minha formação académica, e pela orientação e conselhos que me permitiram realizar um trabalho mais completo e fundamentado.

Ao Sirvan Khalighi queria agradecer a forma paciente como simplificou e me ensinou complexos assuntos indispensáveis para a elaborar este trabalho.

À Sara Barbosa gostaria de agradecer a colaboração e constante disponibilidade.

Devo também um agradecimento ao ISR pelos excelentes recursos e condições disponibilizadas para a realização do meu trabalho. Esta dissertação foi realizada e suportada no âmbito dos projectos "RECI/EEI-AUT/0181/2012 - AMS-HMI12: Apoio à Mobilidade Suportada por Controlo Partilhado e Interfaces Homem-Máquina Avançados" e "Centro-07-ST24-FEDER-002028: Diagnosis and Assisted Mobility for People with Special Needs" financiados pela Fundação para a Ciência e Tecnologia (FCT), FEDER, e programas QREN e COMPETE.

Aos meus amigos, agradeço estes anos maravilhosos cheio de boas memórias, a amizade e apoio a cada fase deste percurso.

Um grande e sincero agradecimento à minha família, pelo apoio incondicional e incansável, pela motivação ao longo de toda a minha formação académica. Aos meus irmãos, que se ofereceram com entusiasmo para participarem em todas as experiências necessárias, quero agradecer a amizade incondicional e insubstituível. Ao João, o meu especial obrigada, pelo apoio, pela entrega e dedicação, pela paciência e pela motivação constante.

Obrigada.

Abstract

Brain-computer interface (BCI) is a communication channel between a human being and a machine, and has surfaced as the last communication method available to patients suffering from locked-in syndrome. However, its usage requires a long setup because of the need to place several electrodes in order to record brain activity and, after that, the calibration of the system. This calibration is necessary for the computer to learn how to model the neuronal patterns associated with the user's intention.

In this work, several classification techniques were applied to a BCI that allows the user to write on a computer screen. This BCI is controlled through the recognition of an encephalographic evoked potential neuromechanism called P300. This work sought the implementation of supervised and semi-supervised methods as well as linear and non-linear methods always with the goal of improving classification rates (and so, also the bit rate) and eliminating the necessity of calibrating the system in every session.

Supervised methods are effective only when the data used to train the classifier is similar to the one extracted by the BCI system at the time of its usage. Since brain signals are commonly described as non-stationary, this may not be the case, rendering the classification process more difficult. This is why the calibration of the system is needed every time it is going to be used is usually assumed, making the usage of BCI systems an often tedious affair. This work evaluates the need of this recurrent calibration through the analysis of the non-stationarity encountered in the P300 evoked potential. In an attempt to reduce the effects of the recorded brain activity's non-stationarity, domain adaptation methods were also implemented. These techniques surface as a method of mitigating the differences between the train and the test domains.

The Fisher's discriminant analysis (FLD) and the kernel logistic regression (KLR) methods were implemented. Both presented higher classification rates when compared to the one originally used (Bayes classifier). However, since both presented very similar results, it was concluded that there is no advantage in introducing the inherent complexity of a non-linear classifier to this particular system.

An adaptative and semi-supervised method called importance weighted KLR (IWKLR) and a domain adaptation method called covariate shift minimization (CSM) were also tested. The results shown by the IWKLR were inconclusive about its efficiency. Relative to the CSM, improvements in the offline experiment were observed though it did not produce any in the online experiment. All methods were implemented offline and two of the supervised methods were also applied on online experiments performed by three different subjects. Online results showed an increased performance when compared with the original classifier.

Key words: BCI, P300, domain adaptation, non-stationarity, FLD, KLR, IWKLR, CSM

Resumo

A interface cérebro-computador (ICC) é um canal de comunicação entre o ser humano e o computador e tem surgido como último modo de comunicação disponível para pacientes que sofrem de "locked-in syndrome". No entanto, a sua utilização carece de uma preparação morosa devido à necessidade de colocação de eléctrodos para registar a actividade cerebral e à necessidade de calibração do sistema. Esta calibração é necessária para que o computador aprenda a modelar os padrões neuronais associados à intenção do utilizador.

Neste trabalho, diferentes técnicas de classificação foram aplicadas a uma ICC que permite ao utilizador soletrar ("spelling board"). Esta ICC é controlada através de um neuromecanismo chamado P300 que corresponde a um potencial electroencefalográfico evocado por eventos relevantes. Foram implementadas técnicas de classificação supervisionada e semi-supervisionada bem como técnicas lineares e não lineares com o objetivo de melhorar as taxas de classificação (aumentando o débito de comunicação) e eliminar a necessidade de calibração em cada sessão.

Os métodos de classificação supervisionada apenas são eficazes quando a informação utilizada pelo classificador para aprender a distinguir as caracteristicas do sinal recolhido é semelhante à informação recolhida no momento em que o sistema ICC é utilizado. Uma vez que os sinais cerebrais são descritos como não estacionários, isto pode não acontecer, dificultando a sua classificação. Assim sendo, é assumida a necessidade de calibração dos sistemas ICC de cada vez que o paciente os utiliza, tornando o seu uso entediante. Neste trabalho, foi avaliada a necessidade desta recorrente calibração através da análise da não estacionaridade do potencial evocado P300. Métodos de adaptação dos domínios foram utilizados na tentativa de mitigar estes efeitos. Estas técnicas surgem com o objectivo de aliviar as diferenças entre o dominio no momento do treino do classificador e o momento da sua utilização.

Implementou-se o método linear "Fisher's discriminant analysis" (FLD) e o método não linear "kernel logistic regression" (KLR). Ambos apresentaram taxas de classificação superiores ao método originalmente implementado (classificador de Bayes). No entanto, sendo que os dois classificadores verificaram desempenhos muito idênticos, concluiu-se que não advém nenhuma vantagem na introdução da complexidade inerente a um classificador não linear.

Foi também testado um método adaptativo semi-supervisionado chamado "importance weighted KLR" (IWKLR) e analisado um método de adaptação de domínio das característica ("covariate shift minimization" - CSM). Os resultados do IWKLR não foram conclusivos relativamente à sua eficiência. No caso do CSM houve melhorias no desempenho para a experiência offline, não trazendo vantagens na experiência online. Todos os métodos foram implementados e testados numa experiência offline e os dois métodos supervisonados foram também implementados e testados numa experiência online efectuada por três participantes. Os resultados obtidos mostram um aumento na performance quando comparados com o classificador usado originalmente.

Palavras-chave: ICC, P300, adaptação de domínimo, não estacionaridade, FLD, KLR, IWKLR, CSM

Contents

List of Figures

List of Tables

List of Abbreviations

BCI	Brain and Computer Interface
CSM	Covarite Shift Minimization
ECoG	Electrocorticographic
EEG	Electroencephalography
FLD	Fisher's Linear Discriminant
GSVM	Gaussian Kernel Support Vector Machine
ISR	Institute for Systems and Robotics
ITI	Inter-stimulus Interval
ITR	Information Transfer Rate
IWCV	Importance weighted cross validation
IWKLR	Importance Weighted Kernel Logistic Regression
KLIEP	Kullback-Leibler Importance Estimation Procedure
KLR	Kernel Logistic Regression
PCM	Pearson's Correlation Method
SC	Single-character
SPM	Symbols Per Minute
SSVEP	Steady State Visual Evoked Potential
SVM	Support Vector Machine
LSVM	Linear Support Vector Machine
SWLDA	Stepwise Linear Discriminant Analysis

Chapter 1

Introduction

Brain-computer interface (BCI) is no longer an object of science fiction movies, but it is also far from being a plug-and-play device ready for everyday use. It is an emerging and exciting field of study that, despite the undeniable progress, is still facing several challenging setbacks. The term BCI was introduced by Jacques Vidal who presented the first BCI system in 1973 [4]. His work was the first demonstration of proof-of-concept, and since many other BCI applications have been developed.

Being a direct communication channel between the brain and an external device, its goal is to convert brain activity signals into computer commands. This is done by detecting patterns in the recorded brain activity and matching these patterns with specific commands.

Allowing a new way of establishing communication between a human being and a machine, BCI may be the only and last hope of subjects in locked-in state. Subjects in this state lose the ability to speak and move due to complete paralysis, while remaining completely aware. It can be caused by amytrophic lateral sclerosis (neurodegenerative disease that causes the death of motor neurons), by multiple sclerosis, stroke or any other cerebrovascular incidents that damage parts of the brain.

BCI has been used for many different applications. Some examples are: the control of external devices such as spelling devices, environment control, wheelchair control and gaming. There is also the possibility of feeding electric impulses to the brain in order to restore some lost sensory capacity such as hearing and vision. However, all the applications to control external devices are just proof of concept and are still not commercially available.

It is thrilling to learn and study an area that can change the way we interact with a machine. Many of the BCI applications are designed to improve the life of people suffering from diseases that cause brain damage.

The brain is an exciting structure and there is still much to discover about this amazing organ. Allied with the progress of engineering, discoveries made on the brain may revolutionize technology and medicine.

1.1 Context

To control a BCI system, electroencephalography (EEG) is used to record the electrical activity along the scalp. EEG measures voltage fluctuations resulting from ionic current flows within the neurons of the brain.

This work is about a speller device (a "mental writing machine" that allows to spell letters sequentially) based on evoked potentials. These devices allow the communication with the outside world without any movement of the body using only the brain response to an external stimulus to control the system. The neuromechanism used to control the BCI is based on an event related potential (ERP) that is generated as a reaction to a relevant external stimulus and it is elicited in the process of decision making [5].

To elicit a P300 ERP, the interface must be designed as an oddball paradigm, i.e., a rare target stimulus that occurs among a series of more common non-target events. It is possible to detect which stimulus a user is focusing his/her attention on, because a P300 waveform will appear in the EEG signal recorded during the presentation of that stimulus.

To automatically detect which letter the user wants to write, a machine learning algorithm (classifier) has to detect the presence of a P300 waveform within the ongoing EEG. It can learn patterns using supervised, semi-supervised or unsupervised learning. In supervised learning, the classifier learns patterns using a training dataset containing samples of labelled EEG epochs (an epoch is an EEG data segment, typically of 1 second, associated to each event). This label can be "target" if the epoch contains a P300 waveform or "non-target" if the epoch contains ongoing EEG. Using semi-supervised learning, labeled and unlabeled data is used to train the classifier, and in unsupervised learning only unlabeled data is used to train the classifier.

The non-stationarity of the EEG signal is pointed out as one of the biggest challenges to overcome in most BCI applications. It can have multiple causes such as: fatigue, changes in recording conditions, effects of feedback training, different mental activity, etc. Due to the non-stationarity of the signal, the distribution of the samples may change from the training to the test session. Sometimes this effect is so visible that the trained classifier becomes unpredictable. Therefore, adaptation of the system to those changes is necessary. In this work we want to evaluate the non-stationarity of the P300 waveform within-subjects in order to understand if there is a need to calibrate the system each time a session is performed.

1.2 Main contributions

This work is mainly focused on the classification problem. Linear and non-linear classifiers, following supervised and semi-supervised approaches, were analysed and validated on a P300-based BCI framework called lateral single character (LSC) speller, which was developed at the Institute for Systems and Robotics (ISR) of the University of Coimbra ([6]). A detailed theoretical explanation and analysis of the performance of the classifiers is presented.

The effects of the non-stationarity of the EEG signal were analyzed in datasets collected from different sessions to determine how robust the P300 detection really is. In a first approach, a method was implemented in the feature extraction step to correct the change in the distribution of the feature vector before classification using the covariate shift minimization (CSM) method .

In a second approach, non-stationarity was tackled by semi-supervised methods, where labeled data were used for training the classifier and some unlabeled data were used to adapt the system to the changes.

The main contributions of the presented work, as shown in Figure 1.1, are:

- The implementation and analysis of the supervised Fisher's linear discriminant (FLD) classifier on the P300 speller.
- The implementation of an application that uses the supervised kernel method called kernel logistic regression (KLR) (using the toolbox for Matlab by Makoto Yamada [7]) and analysis of its performance on the P300 speller.

- Analysis of the effects of the non-stationarity of the EEG signal.
- The implementation of an application that uses an adaptative method called importance weighted kernel logistic regression (IWKLR) (using the toolbox for Matlab by Makoto Yamada [7] and the toolbox developed by Sugiyama [8]) and analysis of its performance when applied to the P300 speller.
- Application and analysis of the CSM to the feature vector.
- Offline validation of the methods on datasets collected from three subjects gathered in different sessions.
- Online validation of the methods performed with three subjects.

The diagram presented in Figure 1.1 shows a summary of the implemented work.

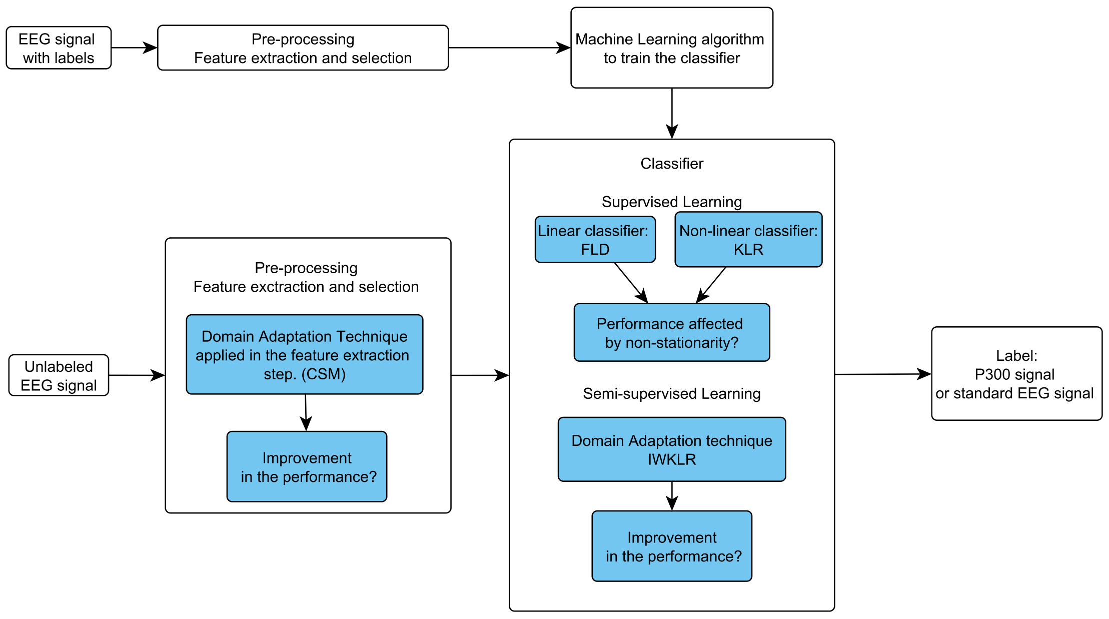

EEG signal with labels
Pre-processing Feature extraction and selection
Machine Learning algorithm to train the classifier
Classifier
Supervised Learning
Linear classifier: FLD
Non-linear classifier: KLR
Performance affected by non-stationarity?
Semi-supervised Learning
Domain Adaptation technique IWKLR
Improvement in the performance?
Unlabeled EEG signal
Pre-processing Feature exctraction and selection
Domain Adaptation Technique applied in the feature extraction step. (CSM)
Improvement in the performance?
Label: P300 signal or standard EEG signal

Figure 1.1: Main contributions

Chapter 2

State of the Art and Neurophysiologic Background

2.1 Neurophysiologic Background

In order to use a BCI system, we first need to record the electrophysiological signal. This can be done by using invasive methods (electrocorticographic (ECoG) and intracortical) or by using non-invasive methods such as the electroencephalography (EEG). EEG can be recorded with wet or dry electrodes. The former requires skin cleaning and a conductive gel that must be applied by an assistant making the whole system setup a tedious and time consuming task. Dry electrodes allow for a faster setup because they are put directly on the user's scalp without requiring skin preparation. However, this option has a smaller signal-to-noise ratio and so produces weaker results. These inconveniences limit BCI usability.

The EEG signal is obtained from the surface of the scalp using multiple wet electrodes and the recorded signals are the potential differences between each electrode and the reference electrode. These can range from 0 to 100µV. The collected signals are often corrupted by artifacts such as: physiological artifacts (eye movement, heart beating, muscular activity) or nonphysiological artifacts (e.g. power supply noise, noise due to EEG amplification). Those corruptions may sometimes have larger amplitude than the signals of interest.

BCI can be operated through different EEG neural mechanisms. They can be internally induced by the user or externally evoked by external stimuli. In induced neural mechanisms, users try to modulate their rhythms by focusing on mental tasks. An example of a mental task is the motor imagery, in which the user imagines that he/she is performing a movement, and thereby modulates sensorimotor rhythms [9]. The imagination of different movements generates changes in mu and beta rhythms in different regions of the motor cortex which can be associated to different commands. The other two neuromechanisms, P300 and steady state visual evoked potential (SSVEP), use external stimuli to evoke brain activity.

The SSVEP is evoked by repetitive visual stimulation (e.g., flickering stimulus at a given frequency). The repetition of a stimulus in a specific frequency evokes an EEG rhythm with the same frequency in the visual cortex. Users can select one stimuli by gazing and focusing their attention on it. The frequency bands corresponding to the flickering frequency of the stimulus increases and can then be associated to commands to control the BCI.

The P300 ERP is elicited whenever a relevant target event occurs. This corresponds to a peak in the amplitude of the EEG signal which occurs with a latency ranging from 300 ms to 500 ms after the stimulus occurs. Other ERP are also evoked by the event. Early components such as the N100 are mostly sensorial responses, while late components such as P200 and N200 are perceptual responses associated with selective attention and decision making (P300). The usual waveform of the P300 event can be seen in Figure 2.1.

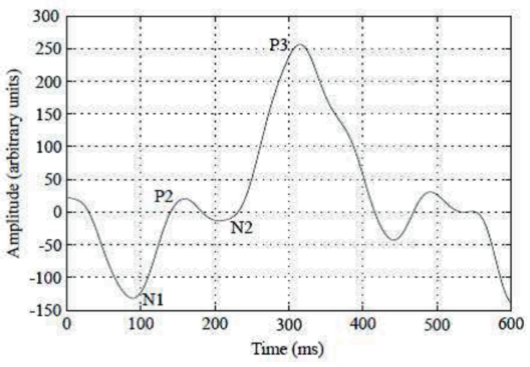


Figure 2.1: P300 waveform. [1]

The P300 is usually evoked in oddball tasks where the subject is asked to react to a rare target stimulus that occurs among a series of more common non-target events.

This work uses a P300-based BCI to control a speller device. In the P300 speller, the symbols of the spelling board flash randomly and every time a flash occurs, the EEG signal is recorded for the span of one second and labeled with the code associated with the flashing symbol (letter). The user is asked to focus his/her attention on the letter he/she intends to write on the screen. This letter is called the target, and the recorded signal associated with this letter is supposed to contain the P300 waveform. All the other letters contain standard EEG signal and are called non-target. The most used paradigm is the row-column (RC) speller first proposed by Farwell-Donchin in 1988 [2]. This spelling board, consists of a 6x6 matrix containing the letters of the alphabet, numbers from 0 to 5 and some useful symbols ('del', 'spc' , '.' and ',') as seen in Figure 2.2. The rows and columns flash randomly. The letter selected by the user corresponds to the intersection of the target raw and target column.

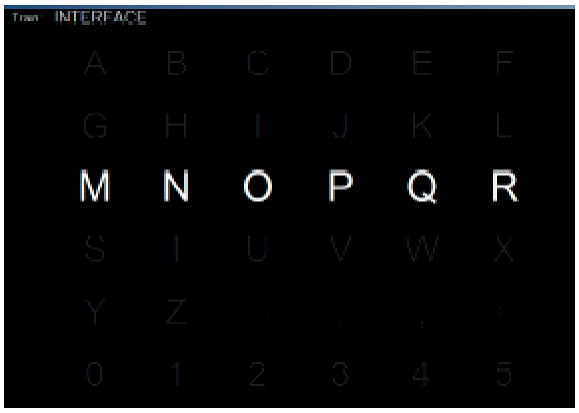


Figure 2.2: RC speller [2]

2.2 State of the Art

There are many different types of P300 spellers. The two more commonly used are the RC speller developed by Farwell and Donchin (1988), and the Single-character (SC) speller by Guan (2004). The second one is similar to the RC speller but instead of highlighting a whole row or colummn, only one letter or one symbol is highlighted at any given time. By doing this, the amplitude of the P300 peak is larger than with the RC speller since the target probability decreases [10]. However, this diminishes the bit/rate in which the user can write on the screen. The Lateral Single Character (LSC) speller developed by G.Pires [6] uses a different letter placement on screen, half in the right side and half on the left side, which simultaneously overcomes some limitations found on the RC speller and on the SC speller and increases the bit rate. The LSC speller has a layout that reduces the effects of local distractors. This effect is caused by the surrounding stimuli and depends on their number and distance to the target stimulus. Since, as in the SC speller, only one symbol is highlighted at any given time, the target probability is smaller than the one observed on the RC speller and the amplitude of the P300 peak is larger. However, this new layout allows a single letter to always be highlighted, alternating between the right and left side, increasing the bit/rate when compared with SC speller.

Several efforts in BCI research have been devoted to improve the supervised classification methods in order to increase classification rates and, consequently, the bit/rate. Recently, the semi and unsupervised techniques have caught the attention of a several researchers. Krusienski et. al [11] made a comparison between the supervised classifiers mainly used in RC speller systems is presented. Four linear classification methods were used: Pearson's Correlation Method (PCM), Fisher's linear discriminant (FLD), stepwise linear discriminant analysis (SWLDA) and a linear support vector machine (LSVM); and one nonlinear classification method: Gaussian kernel support vector machine (GSVM). In this work it was concluded that all the implemented algorithms were able to adequately classify the data and that there was no need for the added complexity of the nonlinear method. The PCM classifier was the one that achieved worst performances, caused by the fact that it does not use the covariance between the features. The FLD and SWLDA provided the best overall combination of training effort and performance achieved. However, the FLD has deteriorated performance if the feature vector is large and the number of samples is small. The SWLDA solves this problem since it has automatic feature extraction, that is, insignificant features are removed from the model. SVM provides a good generalization using less data for training the classifier, but it is a very complex algorithm and its training is slower when compared to the other methods. Moreover it does not provide an evident performance advantage over the other methods. The GSVM algorithm presents an inferior performance when compared to the SVM, this is most likely due to overfitting that is a commom problem for nonlinear classifiers. The nonlinear classifiers are often able to model the training data very accurately but fail if the training data is not representative of the test data.

In this dissertation we will apply the linear FLD classifier and a non-linear classifier called Kernel Logistic Regression. This classifier was previously used in speaker identification [7] and sleep stage classification [12]. This classifer has the advantage of providing estimates of the class probabilities and it can be easly used in a multiclass classification problem.

The EEG signal is described as non-stationary within-subjects in many different works [13–15] and several researchers are applying adptative BCI in order to overcome the problems that arise with this non-stationarity especially in motor imagery. These adaptation techniques analize the resemblance between the test data and the training data to readjust the classification algorithm based on it. In [14], Satti et al. applies a feature adaptation technique called covariate shift minimization (CSM). This method tries to eliminate the diferences in the distribution from the training of the classifier to its test and it was applied to motor imagery using only one feature. In the presented work this technique was applied to the P300 speller. Liand et. al [16], applied an extension of linear discriminant analysis based on importance sampling is applied to motor imagery, that is, each training sample is weighted according to their importance in the test distribution. In the presented work an extension of KLR is applied based on importance sampling, a method called importance weighted KLR (IWKLR). This adaptation technique was also used in both speaker identification and sleep stage classification [7, 12] with promising results.

The P300 neuromechanism has been considered to be stable by some researchers [11, 17], despite the significative changes in its peak amplitude and latency. Other works, such as the one developed by Dauce et. al [18] refer the P300 as a non-stationary signal and it points out that the separation between the moment of the training of the classifier and the use of the system can arise as a problem due to this non-stationarity.

In what concerns domain adaptation techniques in the P300 speller system, Li et al. [19] developed a semi-supervised version of the Support Vector Machine (SVM) classifier. It used a small number of labeled samples to train the classifier and a much larger number of unlabeled samples to adjust the operation. This reduces the effort required by the classifier calibration. However, the presented results only show the performance when the number of epochs is equal to 10, which represents a low bit/rate. Lu et al. [20] made an effort to completely eliminate the need of calibration in a P300 system. First, a generic subject-independent model using a pool of EEG signals was created and then the subject-specific EEG characteristics were learned when the user started using it in the online session. This requires a long time to achieve an acceptable accuracy rendering the first moments of the system's usage unreliable.

Chapter 3

Methods

To construct the LSC speller the main task is to identify the presence of the P300 ERP in the recorded EEG signal. The presence of this ERP means that the letter that flashed was most likely the one the user was looking at. The automatic detection of the letter selected by the user follows two steps of classification. First, a binary classifier (target vs. non-target) is applied to EEG data associated with each flashing letter, and then, in the second step, the letter with the highest classification score is chosen.

The overall classification process consists of 4 different stages: pre-processing of the EEG signal, feature extraction, feature selection and classification (Figure 3.1).

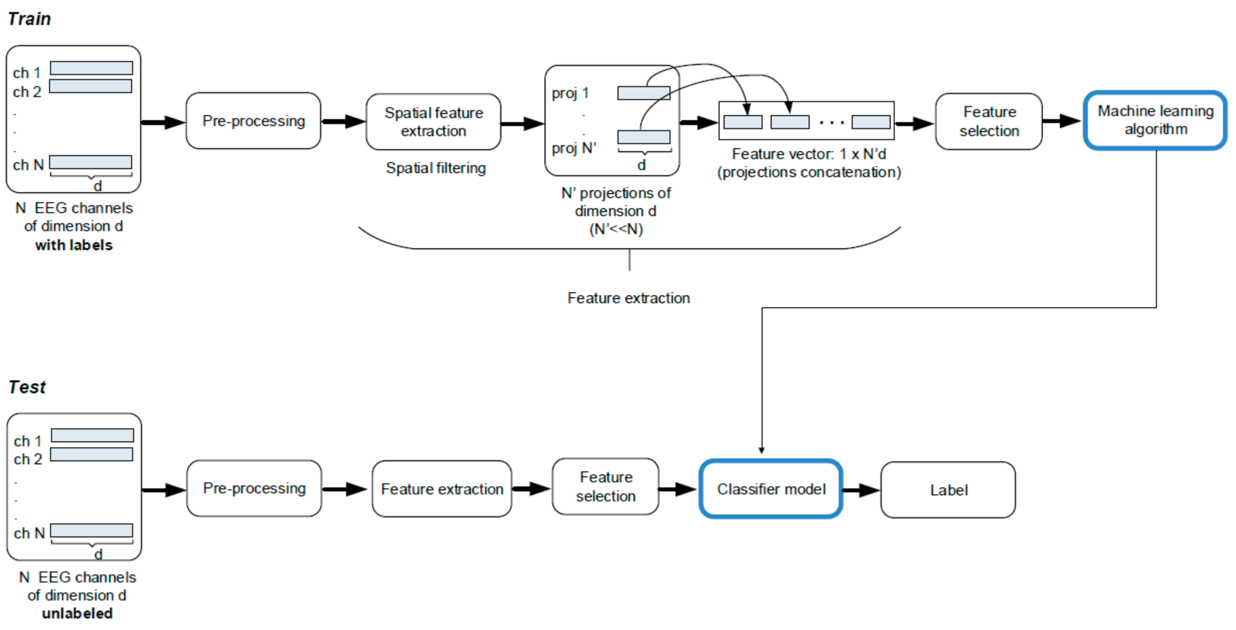

Train
ch 1
ch 2
ch N
d
N EEG channels of dimension d **with labels**
Pre-processing
Spatial feature extraction
Spatial filtering
proj 1
proj N'
d
N' projections of dimension d (N'<<N)
Feature vector: 1 x N'd (projections concatenation)
Feature selection
Machine learning algorithm
Feature extraction
Test
ch 1
ch 2
ch N
d
N EEG channels of dimension d **unlabeled**
Pre-processing
Feature extraction
Feature selection
Classifier model
Label

Figure 3.1: Classification architecture

The core contribution of this thesis is the implementation and comparison of different classification approaches, most specifically, their ability to increase the classification accuracy and ability to adapt to EEG changes. The following classifiers were tested: the Fisher's Linear Discriminant (FLD), the kernel logistic regression (KLR) classifier and later on, an importance weighted KLR (IWKLR) to adapt the classifier to the non-stationarity of the EEG signal. In the feature extraction stage a feature adaption technique was applied. The original LSC framework used a Naive Bayes classifier [6].

The methodological background of these methods is presented here, and implemetation details specific to the P300 classification problem are described in chapter 5.

3.1 Pre-processing and Feature Extraction

The EEG signal recorded for the span of 1 second and labeled with the code associated with the flashing symbol is used to perform the classification. This span is sampled at a frequency of 256 Hz, this means that for the recorded activity, 256 samples will be available. Since 12 electrodes were used and the brain activity was recorded for each electrode, this means that the dataset used to perform classification was initially a matrix $N \times T$ where $N = 12$ channels and $T = 256$ samples. That is, the features used to perform classification consist in the amplitude of the recorded brain activity trought time. Before classification, there is a pre-processment, a feature extraction, and a feature selection module that will choose the information in this dataset that better discriminates the two classes.

The pre-processing phase aims to eliminate signal frequencies outside the range of interest and interferences that the signal may have suffered from the surroundings.

The feature extraction consists of an extraction of the features that enable us to distinguish the two different classes (target and non-target). The feature extraction methods implemented in the LSC framework are based on statistical spatial filters, applied in the spatial domain. The one used in this thesis was the Fisher criterion beamformer (FCB) spatial filter proposed in [6] . This filter builds a linear combination of the signals measured at the 12 electrodes, then is used to project the original data space into one projection that maximizes the discrimination between the two classes. The signal-to-noise ratio of P300 components increases drastically as well as the discrimination power.

3.2 Feature Selection

After feature extraction, the features that best distinguish one class from another are chosen. This is called feature selection. For this purpose the r^2 correlation method is used. This allows the determination of the inter-class and within-class variances. The r coefficient varies between 0 and 1, a high value of this coefficient means a large inter-class variance and a small within-class variance, and vice versa. Applied to each feature this coefficient enables the ranking of the features according to their discriminative power.

Considering samples from two different classes (X and Y), the r^2 correlation is obtained from the square of the equation 3.1.

$$r(X,Y) = \frac{\sigma_{X,Y}}{\sigma_X \sigma_Y} = \frac{\sum_{k=1}^{K}(X_k - \bar{X})(Y_k - \bar{Y})}{\sqrt{\sum_{k=1}^{K}(X_k - \bar{X})^2}\sqrt{\sum_{k=1}^{K}(Y_k - \bar{Y})^2}} \tag{3.1}$$

3.3 Classification Methods

The difficult task of labeling an EEG signal as target or non-target is assigned to the classifier and this is the last step in the classification process where different classification methods can be applied.

3.3.1 Supervised learning

In supervised learning, the training set that consists of several labeled feature vectors is used to train the system how to distinguish patterns from different classes (each class has a label associated with it). The learning algorithm's goal is then to predict which class a given unseen feature vector belongs to.

The parametric learning scenario is considered. This means that the model used to describe the mapping between features and labels depends on parameters and the goal of supervised learning is to select these parameters to minimize the structural risk.

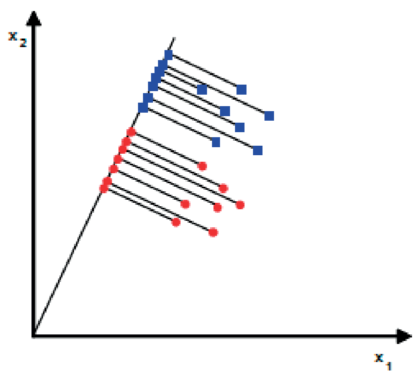
A linear (FLD) and a non-linear (KLR) classifier were applied to compare their performances in the P300 speller.

3.3.1.1 Fisher's linear discriminant

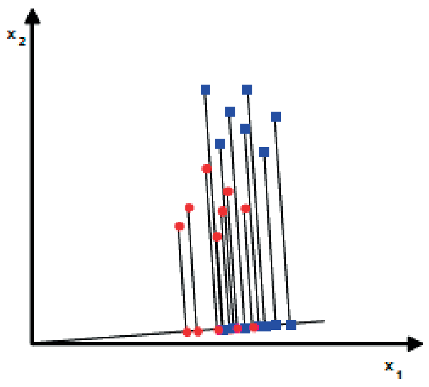
The FLD seeks the reduction of the dimensionality of the feature space. This is done in a way that maximizes the discrimination between classes. Lets assume that each sample corresponds to a feature vector with dimension d, we have N labeled samples ($[\mathbf{x}_i^1, x_i^2 ... x_i^N]$) available to train the classifier, and that N_1 samples belong to class 1 and N_2 samples belong to class 2.

The FLD consists in obtaining a scalar y by projecting d-dimensional data onto a line (3.2).

$$y = \mathbf{w}^T \mathbf{x} \tag{3.2}$$



(a) Projection with good separability



(b) Projection with bad separability

Figure 3.2: Examples of different projections

This projection must be chosen in order to maximize the separability between the two classes. In this explanation the problem will be shown with a two dimentional feature vector because it is easier to understand and visualize how the classifier works while still allowing for the application of the solution in any number of dimension. In Figure 3.2 we have an example of two different projections. We can see that in 3.2a we can easily distinguish one class from another, but in the projection shown in 3.2b it is impossible to separate the two classes. It is easy to conclude that the main goal is to choose a good projection vector and to do so we need to define how the separation between the two classes should be measured.

The measure of the separability between the two classes is done by maximizing the difference between the two projected means, normalized by a measure of the within-class scatter. The scatter is an equivalent of the variance and it is equal to (3.3):

$$\tilde{s_i}^2 = \sum_{y \in w_i} (y - \tilde{\mu}_i)^2 \tag{3.3}$$

$$\text{where} \quad \tilde{\mu}_i = \frac{1}{N_i} \sum_{y \in w_i} y = \frac{1}{N_i} \sum_{y \in w_i} \mathbf{w}^T \mathbf{x} = \mathbf{w}^T \mu_i \tag{3.4}$$

$$\text{and} \quad \mu_i = \frac{1}{N_i} \sum_{x \in w_i} \mathbf{x} \tag{3.5}$$

μ_i is the d-dimensional sample mean of class i and $\tilde{\mu}_i$ is the d-dimensional projected mean of class i. The quantity $\tilde{s_1}^2 + \tilde{s_2}^2$ is called within-class scatter of the projected samples. The distance between the projected means is given by (3.6).

$$\mid \tilde{\mu}_1 - \tilde{\mu}_2 \mid^2 \tag{3.6}$$

FLD is defined as the linear function $\mathbf{w}^T x$ that maximizes the criterion function (3.7) .

$$J(w) = \frac{\mid \tilde{\mu}_1 - \tilde{\mu}_2 \mid^2}{\tilde{s_1}^2 + \tilde{s_2}^2} \tag{3.7}$$

Therefore, this criterion will give us the solution in which the projected samples from each class are closer to each other and their correspondent projected means are furth apart. This means that to obtain good separation of the projected data we want the difference between the means to be larger, in relation to some measure of the standard deviations for each class. To find the optimum value of $\mathbf{w}$ the criterion must be expressed as a function of $\mathbf{w}$. The measure of the scatter in feature space $\mathbf{x}$ is given by (3.8)

$$\mathbf{S}_i = \sum_{x \in w_i} (\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T \quad \text{and} \quad \mathbf{S}_1 + \mathbf{S}_2 = \mathbf{S}_w \tag{3.8}$$

$\mathbf{S}_w$ is the within-class scatter matrix. The scatter of the projection y can be written as function of $\mathbf{S}_w$ (3.9)

$$\tilde{s_1}^2 + \tilde{s_2}^2 = \mathbf{w}^T \mathbf{S}_w \mathbf{w} \tag{3.9}$$

where $\tilde{s_i}^2$ is given by

$$\tilde{s_i}^2 = \sum_{y \in w_i} (y - \tilde{\mu}_i)^2 = \sum_{x \in w_i} (\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \mu_i)^2 = \sum_{x \in w_i} \mathbf{w}^T (\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T \mathbf{w} = \mathbf{w}^T \mathbf{S}_i \mathbf{w} \tag{3.10}$$

The difference between the projected means can also be written in terms of the means in the original feature space (3.11)

$$(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = (\mathbf{w}^{\mathbf{T}} \mu_1 - \mathbf{w}^T \mu_2)^2 = \mathbf{w}^T (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T \mathbf{w} = \mathbf{w}^T \mathbf{S}_B \mathbf{w} \tag{3.11}$$

The matrix $\mathbf{S}_B$ is the between-class scatter. The criterion can be rewritten in terms of $\mathbf{S}_w$ and $\mathbf{S}_B$ (3.12)

$$J(w) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}} \tag{3.12}$$

To find the maximum of J_w, the zeros of it's derivative must be found (3.13) .

$$\frac{d}{dw}[J(w)] = 0 \Leftrightarrow \mathbf{S}_w^{-1}\mathbf{S}_B\mathbf{w} - \lambda\mathbf{w} = 0 \tag{3.13}$$

Solving the eigenvalue problem $\mathbf{S}_w^{-1}\mathbf{S}_B w = \lambda w$ yields (3.14).

$$\mathbf{w}^* = \mathbf{S}_w^{-1}(\mu_1 - \mu_2) \tag{3.14}$$

After finding the direction of $\mathbf{w}$ the classification has been converted from a d-dimensional problem to a hopefully more manageable one-dimensional problem. After mapping, all that is left is to find the bias, the point along the one-dimensional subspace separating the projected points.

The bias must be selected in a way that increases the success rate of the decision making process. It was calculated with the projected samples by using the mean and the standard deviation of each class. If $\tilde{\mu}_1 > \tilde{\mu}_2$ the bias should be found using

$$b = -\frac{(\tilde{\mu}_1 - \tilde{\sigma}_1) + (\tilde{\mu}_2 + \tilde{\sigma}_2)}{2} \tag{3.15}$$

and otherwise the bias is given by

$$b = -\frac{(\tilde{\mu}_1 + \tilde{\sigma}_1) + (\tilde{\mu}_2 - \tilde{\sigma}_2)}{2} \tag{3.16}$$

The bias will cause a shift in the projected samples. Applying this shift will allow us to decide which class the new feature vector belongs to. The discriminant function gives us the rule used to classify each feature vector and it is given by

$$sign(\mathbf{w}^T\mathbf{x} + b) \tag{3.17}$$

The decison of which class a new feature vector belongs to is done by analyzing the signal of the discriminant function.

3.3.1.2 Kernel logistic regression

Logistic regression is a classifier that predicts multiclass posterior probabilities allowing us to measure the confidence in the predicted label. This can be modeled by the softmax function

$$P(y = c|\mathbf{X}) = \frac{e^{w_c^T x}}{\sum_{k=1}^{K} e^{w_k^T x}} \tag{3.18}$$

where P(y = c|X) indicates the probability of a given feature vector X belonging to class c, w_c is the parameter tuned to fit the training data belonging to class c, w_k is the parameter tuned to fit the training data belonging to class k, K is the number of different classes and $\mathbf{X}$ is the feature vector.

Since this formula returns values ranging between 0 and 1, and the sum of $\sum_{c=1}^{K} P(y = c|\mathbf{X}) = 1$, it is used in various probabilistic multiclass classification methods. However, this method only gives us a linear decision boundary. In order to get a nonlinear decision boundary a kernel must be applied.

A kernel method makes it possible to capture a nonlinear pattern using a linear model. This is done by mapping data into a higher dimension, known as the feature space, where the data exhibits a linear pattern. After that a learning algorithm should be applied to discover linear patterns in that space. This mapping is done using kernel functions which enable the operations in the feature space to occur without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space. This allows us to identify the similarities between two feature vectors. Any linear model can be turned into a non-linear model by applying the "kernel trick", meaning that its features are replaced by a kernel function [21].

The Kernel Logistic Regression (KLR) [7] is a kernelized variant of logistic regression and it models the class-posterior probability ($P(y = c|X)$) using the softmax function. However, instead of using the feature vector it uses the kernel trick. Suppose a labeled feature vector $X_i = [X_1, ..., X_N] \in \Re^{d \times N}$ with d dimension containing N samples is used to train the classifier. The training dataset is given by

$$\mathcal{Z}^{tr} = \{X_i, y_i\}_{i=1}^{N} \tag{3.19}$$

where $y_i \in \{1, ..., K\}$ denotes the class (label) associated with the feature vector i and K corresponds to the number of existent classes (number of labels). The prediction of this associated label is done following the Bayes decision rule, that is, if $P(y = c|X) > P(y = k|X) \forall k \neq c$ then we can predict the class of the test sample X as being equal to c.

For approximating the class-posterior probability we use the parametric model (softmax function and the kernel trick) (3.3.1.2)

$$p(y = c|X, V) = \frac{e^{f_{v_c}(X)}}{\sum_{k=1}^{K} e^{f_{v_k}(X)}} \tag{3.20}$$

where $V = [v_1, ..., v_K]^T \in \Re^{K \times N}$ is the parameter to be learned and f_{v_k} is the discriminant function of the class k. The discriminant function used is the kernel regression model (3.21):

$$f_{v_k}(X) = \sum_{i=1}^{N} v_{k,i} \kappa(X, X_i) \quad k = 1, .., K \tag{3.21}$$

where $v_k = (v_{k,1}, ..., v_{k,N})^T \in \Re^N$ are the parameters corresponding to the class k and $\kappa(X, X_i)$ is a Gaussian kernel function, given by (3.22)

$$\kappa(X, X_i) = exp\left(\frac{-||X - X_i||^2}{2\sigma^2}\right) \tag{3.22}$$

To use the KLR the parameter V must be learned. The maximum likelihood is the method used to find the value for this parameter that best fits the statistic model with the observed data. To estimate V the negative log-likelihood function $\mathcal{P}_{\delta}^{log}(V; \mathcal{Z}^{tr})$ is used. This function is a convex function meaning that only one unique minimizer exists and it is given by (3.23)

$$\mathcal{P}_{\delta}^{log}(V; \mathcal{Z}^{tr}) = -\sum_{i=1}^{N} logP(y_i|X_i, V) + \frac{\delta}{2} trace(VKV^T) \tag{3.23}$$

where $trace(VKV^T)$ penalizes complex solutions to avoid overfitting, δ controls the strength of the regularization and $K = |\kappa(X_i, X_j)|_{i,j=1}^{N}$ is the kernel Gram matriz. The value of V that minimizes the value of the negative log-likelihood can be found using Newton's method. This is an iterative method that finds the minima by finding the zeros of the derivative of the negative log-likelihood function, for more details consult the work [7].

It is also necessary to tune two different parameters: the gaussian width σ and the regularization parameter δ. To select the best parameters for the KLR model we used cross validation (CV). This consists in dividing the training set $\mathcal{Z}^{tr}$ into L subsets $\{\mathcal{Z}_l^{tr}\}_{l=1}^{L}$. Let $\hat{y}_{\{\mathcal{Z}_j^{tr}\}}(X)$ be the estimated class for the test sample X obtained from $\{\mathcal{Z}_l^{tr}\}_{l=j}$. The score of the k-fold cross validation [7] is given by 3.24

$$\widehat{R}_{kCV}^{\mathcal{Z}^{tr}} = \frac{1}{L}\sum_{j=1}^{L}\frac{1}{|\mathcal{Z}_j^{tr}|}\sum_{(X,y)\in\mathcal{Z}_j^{tr}} I(y = \widehat{y}_{\mathcal{Z}_j^{tr}}(X)) \tag{3.24}$$

where $|\mathcal{Z}_j^{tr}|$ is the number of samples in each subset and $I()$ denotes the indicator function (is equal to 1 if the predicted label is well assigned and zero otherwise).

3.3.2 Domain adaptation

As mentioned previously the EEG signal is often considered non-stationary. There are two types of non-stationarity: long-term changes caused by fatigue or different electrodes positioning and short-term changes caused by different mental activities. The short-term drifts can be handled in the feature extraction step and long-term changes in the classification step.

Classical learning methods are only able to predict the true label of a test sample if there is a relationship between the data used to train the classifier and the data collected to test the classifier. However, in the actual application, this might not happen. For example, the existence of non-stationarity in the EGG signal can make the assumption that the training set follows the same probability distribuition as the test set, which is not always valid. This change in probability distribution may cause a mismatch between the training and test domains. Domain adaptation handles this mismatch by building a classifier using the labeled training data in the old domain that performs well on the test data in the new domain.

Domain adaptation methods assume that probability distributions of training and test datasets are related to each other in some sense, and because of that it is possible to learn something about test probability distribution from the training set. An assumption commonly made about the connection between the training and the test domains is that given the same observation $X = x$, the class-posterior probability is the same in both domains. This means that we assume that $P_{train}(Y|X = x) = P_{test}(Y|X = x)$ for all $x \in \mathcal{X}$, but $P_{train}(X) \neq P_{test}(X)$. This difference between the two domains is called covariate shift.

3.3.2.1 Covariate shift minimization

The covariate shift minimization [14] is an adaptation method that performs feature adaptation. As stated before, the probability distribution of each feature is different from the training to the test domain and this may affect the classification accuracy. To avoid the problems caused by the shift of each feature distribution, the estimation of this shift must be done and the features should be adapted to minimize its effects. The least square fitting polynomial is used to estimate and predict the features covariate shift over time. This polynomial allows us to evaluate how much the feature distribution changes.

Suppose that we have N different trials and for each trial we extract d different features resulting in the feature vector: $f_i = [f_1, ..., f_d]^T$ where T denotes the transpose of the matrix . A polynomial with order h is fitted for each feature i over $T-1$ different trials (3.25).

$$\hat{f}_i(t) = a_0 + a_1 t + a_2 t^2 + ... + a_h t^h \tag{3.25}$$

where $a_0, ..., a_h$ are the polynomial coefficients, $t = 1, ..., T-1$ are the number of each trial and $\hat{f}_i(t)$ is the estimated value of the feature for the trial t.

The estimated polynomial value at the current time T is equal to $\hat{f}_i(T)$. The difference between the i-th feature value for the respective time segment T $(f_i(T))$ and the calculated value using the polynomial $\hat{f}_i(T)$ represents the shift in the i-th feature: $shift_i(T) = f_i(T) - \hat{f}_i(T)$.

Adding the calculated shift to the common mean of the training feature distribution gives the readjusted feature for T-th time segment (3.26):

$$Reajusted\,feature_i = f_i(T) - \hat{f}_i(T) + \mu_{0i} \tag{3.26}$$

where the common mean of the training feature distribution is given by $\mu_{0i} = \frac{1}{M}\sum_{j=1}^{M} f_{ij}$ where M is the number of samples in the training dataset.The polynomial coefficient $a_0, ..., a_h$ (Equation 3.25) must be continuously updated and, in each trial, all the features should be readjusted using the polynomial containing information from the $T-1$ previous trials. The selection of the order of the polynomial (h), and the number of previous samples $(T-1)$ used for recalculating the polynomial coefficients is one of the largest challenges when using this method.

3.3.2.2 Importance weighted kernel logistic regression IWKLR

A possible way to address the differences between the training and test domain is to assing intance-dependent weights. The importance weighted kernel logistic regression (IWKLR) [7] is an instance based adaptation method that re-weigh the importance of each training sample in the process of decision making. This is done with the attempt of matching the new domain (test domain) with the training domain. More weight is assigned to the training samples that are in a more populated region of the new domain.

The KLR and CV can be used under the assumption that the test and train features follow the same probability distribution. However, as stated before, this may not be true in the real world. Under covariate shift the input distributions change between the training and test phases $(P_{train}(X) \neq P_{test}(X))$, but the conditional distribution remains unchanged $(P_{train}(Y|X = x) = P_{test}(Y|X = x))$. The classification is interested in the probability $P(Y|X)$, however, the optimal model selected in the training depends on $P(X)$ and since this distribution changes from train to test the optimal model for the train will differ from the optimal test domain model.

To compensate for the distribution change, importance sampling is used. This allows us to weight the training samples according to their importance in the test distribution (3.27).

$$w(X) = \frac{p_{te}(X)}{p_{tr}(X)} \tag{3.27}$$

This weight plays an important role in the adaptation. However, it is usually unknown so its value must be estimated. A direct way of learning the importance weight function is shortly explained in section 3.3.0.1.

Applying the sample's importance to the KLR we can obtain the importance weighted KLR (IWKLR) [7], the negative log-likelihood function is now given by (3.28)

$$\mathcal{P}_{\delta}^{log}(V; \mathcal{Z}^{tr}) = -\sum_{i=1}^{N} w(X_i) logP(y_i|X_i, V) + \frac{\delta}{2} trace(VKV^T) \tag{3.28}$$

The CV should also be applied to the IWKLR in order to find the best parameters. However, ordinary CV is no longer unbiased due to covariate shift, and is not reliable as a model selection method. To cope with this problem an importance weighting technique is used called importance weighted cross validation (IWCV) [8] (3.29).

$$\widehat{R}_{kIWCV}^{\mathcal{Z}^{tr}} = \frac{1}{L} \sum_{j=1}^{L} \frac{1}{|\mathcal{Z}_j^{tr}|} \sum_{(X,y) \in \mathcal{Z}_j^{tr}} w(X) I(y = \hat{y}_{\mathcal{Z}_j^{tr}}(X)) \tag{3.29}$$

3.3.2.3 Kullback-Leibler importance estimation procedure

The Kullback-Leibler importance estimation procedure (KLIEP) is used to estimate the values of the weights (Equation 3.27). Using this method there is no need to use the naive approach for estimating the importance. In this approach the training and test sample densities were estimated separately and the importance weight was obtained using the ratio of the estimated desnsities. However, density estimation is a hard problem to solve and an appropriated parametric model may not be available. KLIEP arises as a solution to this problem since it is a direct way of estimating the importance weight. The parameters in the KLIEP method are learned by minimizing the Kullback-Leibler divergence between the test and the reweighted training distribution. More information about this method can be found in [8].

Chapter 4

Framework and Experimental Setup

4.1 LSC speller

The LSC speller has 28 symbols: the 26 letters of the alphabet, the space ('spc') and the delete ('del') button. The symbols on the screen are divided symmetrically, left and right. They flash randomly, alternating between left and right and there is always one symbol highlighted (see Figure 4.1 and 4.2a). There is no need for an inter-stimulus interval (ISI), as there was in the RC speller. The ISI corresponds to the time when no symbol is highlighted. In the case of the LSC speller, the ISI is virtually created because the user focuses his/her attention exclusively on one side of the screen ignoring what is happening on the other side. Since the letters are highlighted alternating between sides, the user perceives an ISI when a letter from the other side of the screen is highlighted. The ISI is then equal to the time that each of the speller board's half is completely unhilighted. The interval between the onset of two consecutive stimuli (SOA - stimulus onset asychrony) is equal to the perceived ISI and it corresponds to the time that each symbol is highlighted, in this work this time is 75ms. The flash of each symbol is done by changing the symbol color from white to red, the background from grey to green and by increasing the size of the symbol. The temporal diagram expressing these events is shown in Figure 4.2a. Figure 4.2b shows how the EEG epochs associated to each event are recorded.

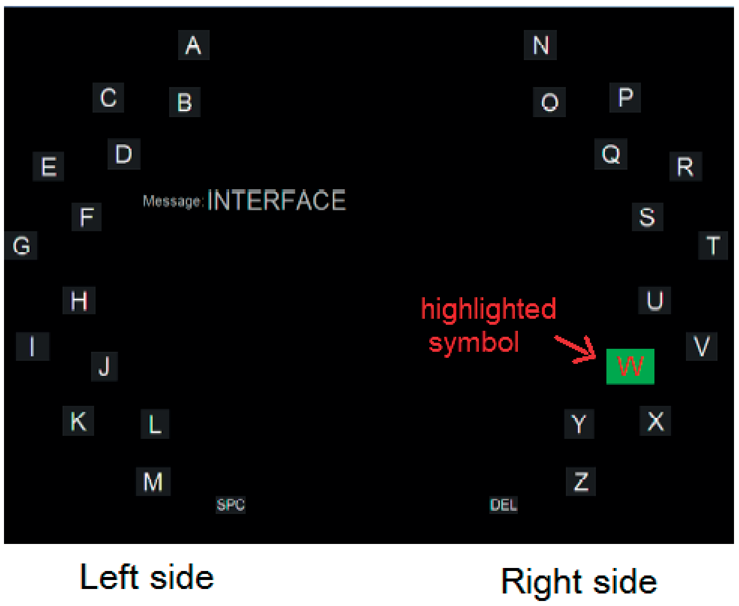
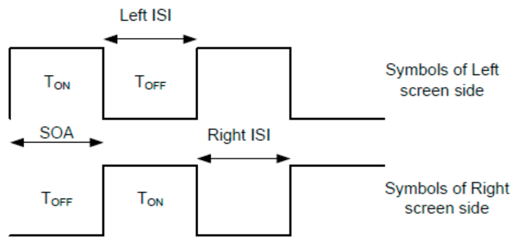
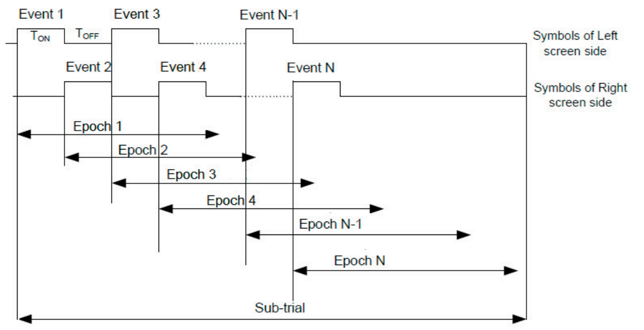


Figure 4.1: LSC speller where the right side of the screen contains letters from A to M and the 'spc' symbol and the left side of the screen contains letters from N to Z and the 'del' symbol. The highlighted symbol is w.



(a) Temporal diagram of the LSC stimuli/events.



(b) Temporal diagram showing epochs extracted from the continuous EEG data stream.

Figure 4.2: Temporal diagram of the LSC speller

4.2 Calibration and Collected Datasets

The experiment consists off two phases: a calibration/training session and an online session.

During the calibration part the user is asked to focus his/her attention on a specific event and to mentaly count every occurrence of that event. (In this case an event corresponds to a letter being highlighted). This offline training is done by asking the user to focus his/her attention on each letter of the word "INTERFACES", each letter is highlighted 9 times per trial. After the training session, labeled data (ground truth) is available to teach the classifier the patterns that allow the distinction between standard EEG signal and the P300 signal.

The number of samples used to train the classifier depends of the number of epochs used. The number of epochs corresponds to the number of times a letter is highlighted per trial. The signals collected for each letter are then averaged to find their mean value. For example, if the number of epochs is equal to four, it means that each letter of the keyboard is highlighted four times per trial and the signal used to do the calibration is equal to the mean of the four signals collected for each letter. Calculating the mean of several signals improves the signal-to-noise ratio and improves the classifier accuracy.

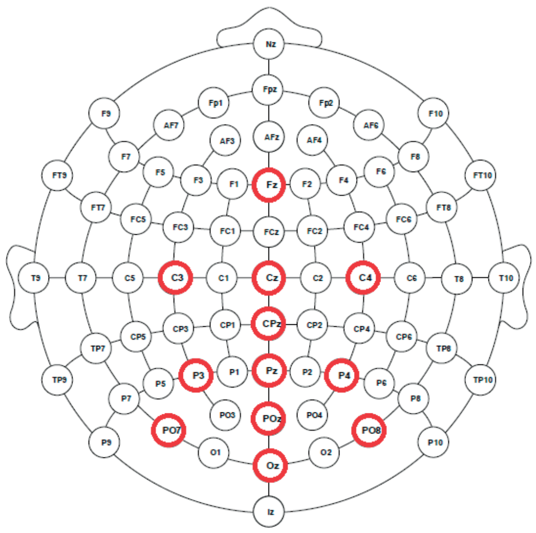
In the LSC calibration, 90 target and 840 non-target events are collected in the calibration period. The number of samples used to train the classifier depends on the intended number of epochs. For a number of epochs equal to n_{epochs}, the number of target events avaliable to train the classifier is $\frac{90}{n_{epochs}}$ and non-target events is $\frac{840}{n_{epochs}}$. The EEG signal recorded from 12 channels is pre-processed, projected with the statistical spatial filter, and then the 200 best features are used to obtain the classification models for online operation.

After the training of the classifier, the system is ready to be used and the online session can begin. In the online session the user can freely write on the screen.

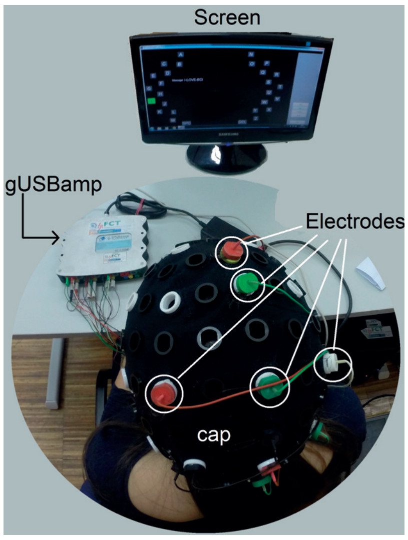
4.3 Experimental Setup

The EEG signal is recorded and labeled in strict synchronization with the event triggers. The 12 electrodes are setup in the g.EEGcap using a conductive gel to improve the conductivity with the scalp.

The electrodes are connected to gUSBamp amplifer and acquistion system (gtec). The 12 electrodes [Fz, Cz, C3, C4, CPz, Pz, P3, P4, PO7, PO8, POz and Oz] were placed with the configuration shown in Figure 4.3a according to the international 10-20 extended system, of the 65 channel g.EEGcap. The reference was placed at the right ear lobe and ground was placed at AFz position in the g.EEGcap. During the experiments, subjects sat on a chair at a distance of about 50 cm from the screen. The experimental setup is presented in Figure 4.3b.



(a) Fig. 3.1: EEG signals according to the 10-20 extended system.



(b) Picture of the experimental setup at ISR.

Figure 4.3: Experimental setup

The BCI software framework was implemented by G.Pires [3] in a Simulink framework. There are four main blocks that compose this implementation as shown by the generic Simulink diagram in Figure 4.4 :

- **'Acquisition driver':** where the Simulink driver provided by g.tec was used. This driver reads the flow sent from gUSBamp vias USB connection. The gUSBamp block allows the configuration of some hardware such as filter selection and sampling rate.

- **'Preprocessing':** performs temporal filtering and other basic preprocessing;

- **'Event synchronization and classification'** : implements the event generation and synchronization, data buffering, epoch segmentation and implements the algorithms for EEG signal processing and classification; it sends the control output to the 'Visual paradigm' block.

- **'Visual paradigm'** : implements the visual paradigms - LSC speller.

The classification algorithms are directly embedded in the 'Event synchronization & classification' block after they have been trained offline in Matlab. This means that no significant changes have to be made between offline training and online implementation.

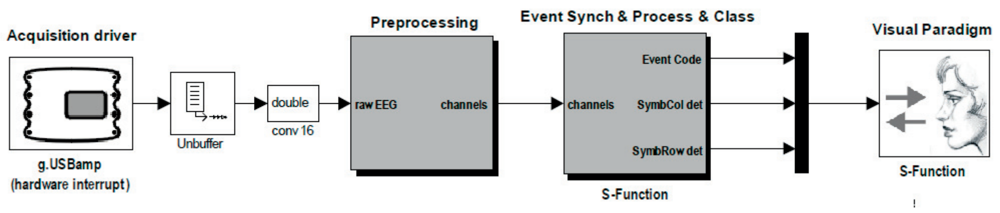


Figure 4.4: Main blocks of a general Simulink implementation of the BCI system [3]

For the purpose of this work only the 'Event synchronization & classification' block was changed.

The FLD and the CSM algorithm were implemented from scratch using Matlab. The KLR and IWKLR were implemented and adapted using the toolbox for Matlab by Makoto Yamada [7] and the KLIEP using the toolbox developed by Sugiyama [8].

Chapter 5

Classification Methods Applied to P300 Speller System

As stated before, two supervised learning techniques were tested in the LSC P300 speller. The implementations/changes of the different classifiers are presented and explained in this chapter.

5.1 Supervised Learning

5.1.1 Application of the Fisher's linear discriminant classifier to the P300 speller

As presented in the theoretical explanation in subsection 3.3.1.1, the FLD classifier finds the direction of **w** that better separates the two classes. That is, we want to find **w** so that the diference between the two means is larger when compared to a measure of the standard deviation of the two classes.

In order to do so, a labeled dataset containing samples from the two classes collected during the calibration period is used. This dataset contains several samples, each sample corresponds to a feature vector that contains either the P300 signal or the standard EEG signal.

The FLD algorithm was implemented from scratch following the pseudocode presented in algorithm 5.1.

Algorithm 5.1 Fisher's Linear Discriminant - Pseudo-code

```
Require:
  X[1..d][1..N] //X= samples; d=nºfeatures; N= nº labels
  C= {1, 2}   // C=label associated with each sample
Ensure:
  w // direction of the projection vector
  b //bias

  Finding the number of samples belonging to each class
  Initialize N1=0 and N2=0
  for k = 1 to N do
    if C == 1 then
      N1 = N1 +1
    else
      N2 = N2 +1
    end if
  end for

  // Feature vector of each class
  c1[1..d][1..N1] = X[1..d][1..N1]
  c2[1..i][1..n2] = data.X[1..i][N1+1..end]
```

```
// Finding the d-dimensional sample mean for each class (Equation 3.5)
for j=1 to d do
    $\mu_1$[j] = mean(c1[j][1..N1])
    $\mu_2$[j] = mean(c2[j][1..N2])
end for

//Measure of the scatter matrix of each classe ($S_i$) in the feature space (Equation 3.8)
Initialize S1=0 and S2=0

for k = 1 to N1 do
    S1 = S1+(c1[1..d][k]-$\mu_1$[1..d])*(c1[1..d][k]-$\mu_1$[1..d])'
end for
for k = 1 to N2 do
    S2 = S2+(c2[1..d][k]-$\mu_2$[1..d])*(c2[1..d][k]-$\mu_2$[1..d])'
end for

// Finding the withing-class scatter matrix (Equation 3.8)
Sw = S1+S2

// Use the diagonal of Sw to construct a diagonal matrix (dSw)
dSw = diagonal(Sw)

// Finding the direction of w
w = inv(dSw)*($\mu_1$-$\mu_2$)

// projection of the samples X into w
proj_data = W'*X;

// Finding the d-dimensional projected mean of each class (Equation 3.4)
$\tilde{\mu_1}$ = mean(proj_data[1..N1]);
$\tilde{\mu_2}$ = mean(proj_data[N1+1..end]);

// Finding the bias
if $\tilde{\mu_1} < \tilde{\mu_2}$ then
    b1 = $\tilde{\mu_1}$ +sd(proj_data[1..N1]); //sd is the standard deviation
    b2 = $\tilde{\mu_1}$ -sd(proj_data[1..N2]);
else
    b1 = $\tilde{\mu_1}$ -sd(proj_data[1..N1]);
    b2 =$\tilde{\mu_1}$ +sd(proj_data[1..N2]);
end if
model.b = -(b1+b2)/2; (Equation 3.15 or 3.16)
```

It should be noticed that instead of using the S_w matrix, a diagonal matrix containing the values of the diagonal of S_w was used. S_w corresponds to the within-class scatter matrix which represents a measure of the variance of the feature vector. By only using the values of the diagonal of S_w, we are assuming that the features are independent from each other. This assumption greatly improves the performance of the FLD classifier. An analysis and comparison of the method using the whole S_w matrix and using only its diagonal is presented in section 6.1.1.

After obtaining the parameters $\mathbf{w}$ and b, the label can be predited for any other new samples. This is done by mapping the sample to a one-dimensional space and evaluating the signal of the projected sample. If $sign(\mathbf{w}^T \mathbf{x_{test}} + b) > 0$ than the sample x belongs to the class with a greater projected mean, otherwise it belongs to the class with the smaller projected mean.

5.1.2 Application of the kernel logistic regression to the P300 speller

In this section it is explained how the KLR was applied to the P300 speller. As stated in Section 3.3.1.2, the first step to use the KLR classifier is to tune the parameters σ, which is the width of the gaussian kernel used (Equation 3.22), and δ, the regularization strength (Equation 3.23). This is usually done by using cross-validation. However, instead of using cross-validation to tune the parameters σ and δ as suggested and implemented by Yamada [7], a new method to find σ was applied. This method is an automatic parameter selection method to find σ applied to SVM developed by Cheng-Hsuan Li et al. in [22] . This change in approach was due to the unpredictable results of the classifier when both parameters were tuned using cross-validation. Moreover, cross-validation is a time consuming process and by choosing the σ with this automatic process, the time consumed performing cross validation decreases drastically.

The Gaussian Kernel has the goal of expressing the similarities between two samples and it returns values ranging from 0 to 1. If the two samples belong to the same class, then the Kernel value should be close to 1 and if they belong to different classes the Kernel values should be close to 0. In this way, we want to find the value of σ that best makes the distinction between samples belonging to the same class or to different classes.

KLR is a supervised method and therefore each sample has a label indicating its class. To decide which σ value is the best, the training kernel is computed for several values of σ. The mean kernel value for samples belonging to the same class ($w(\sigma)$) in function of σ should be calculated and should be as close as possible to 1. The mean kernel value for samples belonging to different classes ($b(\sigma)$) in function of σ are also computed and this value should be as close as possible to 0. So we want to find $w(\sigma) \to 1$ and $b(\sigma) \to 0$. To find the best sigma value we need to minimize the following objective function (5.1):

$$J(\sigma) = 1 - w(\sigma) + b(\sigma) \qquad (5.1)$$

The relation between the σ and the $J(\sigma)$ function is a curve similiar to the one presented in Figure 5.1. The best sigma value is presented in the figure by a red dot that represents the minimum value of the function $J(\sigma)$. This result was obtained experimentally using one of the datasets collected.

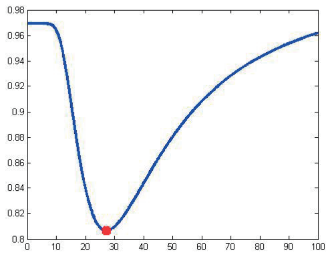


Figure 5.1: $J(\sigma)$ vs σ

Cross validation was used to find the best δ value by dividing the train dataset in 5 different subsets. Then, the KLR classifier was trained using only four of the subsets leaving one out to test the performace. This training was done for different values of δ and consists of finding the parameter V using the Newton's method through 3 iterations. Then the trained KLR classifier is applied to the remaining subset and a score is given by analizing the number of well predicted labels for both classes for each δ value. This is done 5 times, always leaving a different subset out to test the trained classifier. For each tested subset, a sucess rate is measured (number of trials well classified divided by the number of trials in the tested dataset is found) and the final k-fold score correponding to the mean value of the sucess rate trought the 5 subsets (Equation 3.24). However, since for many different values of δ the k-fold score was equal to one (indicating that no mistakes were made in the classification process), the predicted value of the class-posterior probability was also taken into acount, meaning that if we have many different combinations of parameters which reach a k-fold score equal to one, we should look to the class-posterior probability and choose the parameters that have more confidence in the predicted labels. The δ parameter that reaches the larger score is considered to be the optimal one. A diagram explaining the tuning process of the parameters σ and δ is presented in Figure 5.2 .

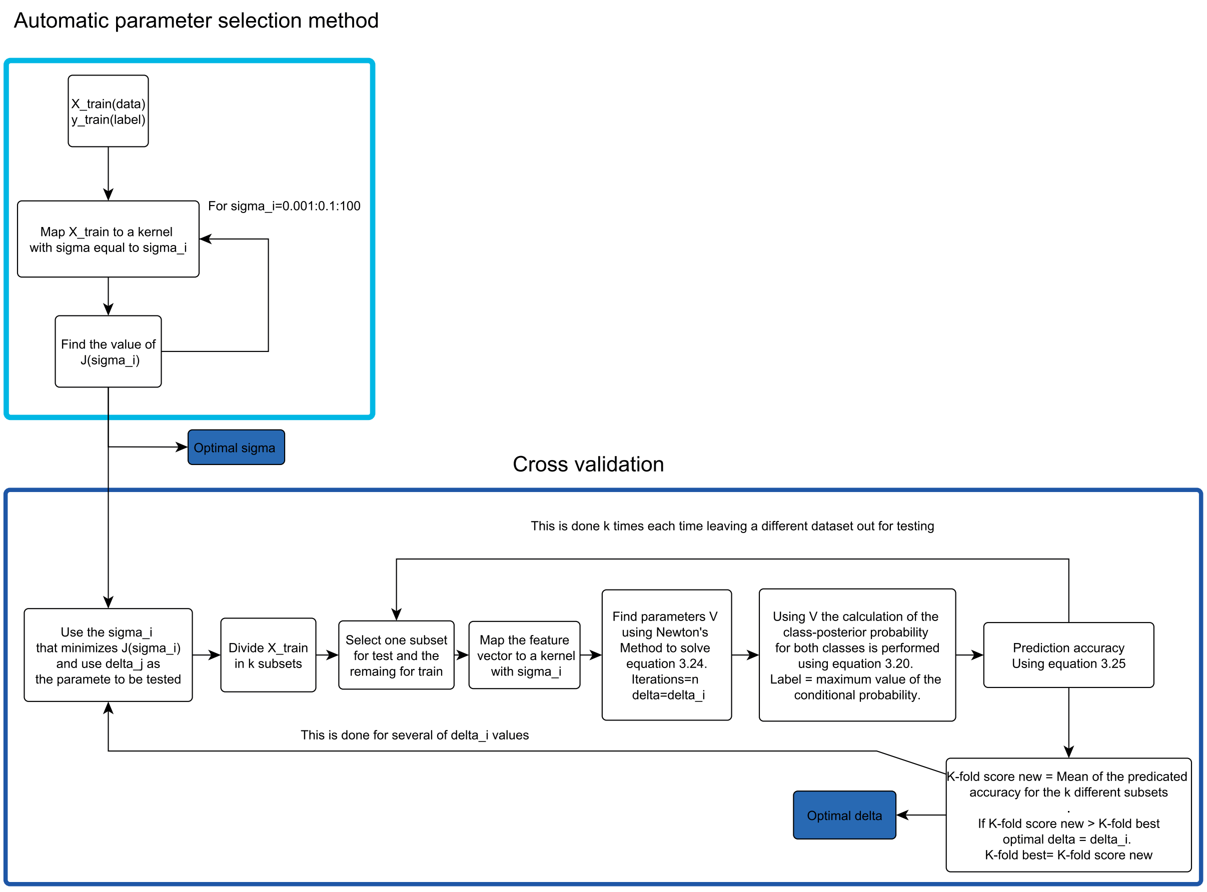

Automatic parameter selection method
X_train(data)
y_train(label)
Map X_train to a kernel with sigma equal to sigma_i
For sigma_i=0.001:0.1:100
Find the value of J(sigma_i)
Optimal sigma
Cross validation
This is done k times each time leaving a different dataset out for testing
Use the sigma_i that minimizes J(sigma_i) and use delta_j as the paramete to be tested
Divide X_train in k subsets
Select one subset for test and the remaing for train
Map the feature vector to a kernel with sigma_i
Find parameters V using Newton's Method to solve equation 3.24. Iterations=n delta=delta_i
Using V the calculation of the class-posterior probability for both classes is performed using equation 3.20. Label = maximum value of the conditional probability.
Prediction accuracy Using equation 3.25
This is done for several of delta_i values
K-fold score new = Mean of the predicated accuracy for the k different subsets
If K-fold score new > K-fold best optimal delta = delta_i. K-fold best= K-fold score new
Optimal delta

Figure 5.2: How to tune the KLR parameters σ and δ

After finding the best parameters, we can build the KLR model that best suits the training data. As is outlined in Figure 5.3 this is done by finding the parameter V that minimizes the value of the negative log-likelihood. This parameter will model the class-posterior probability. To test the classifier it is necessary to map the feature vector to a kernel that will compare the new samples with the train dataset. By using the parameter V found in the training, the class-posterior probability is calculated. A label is assigned to each sample corresponding to the largest class-posterior probability.

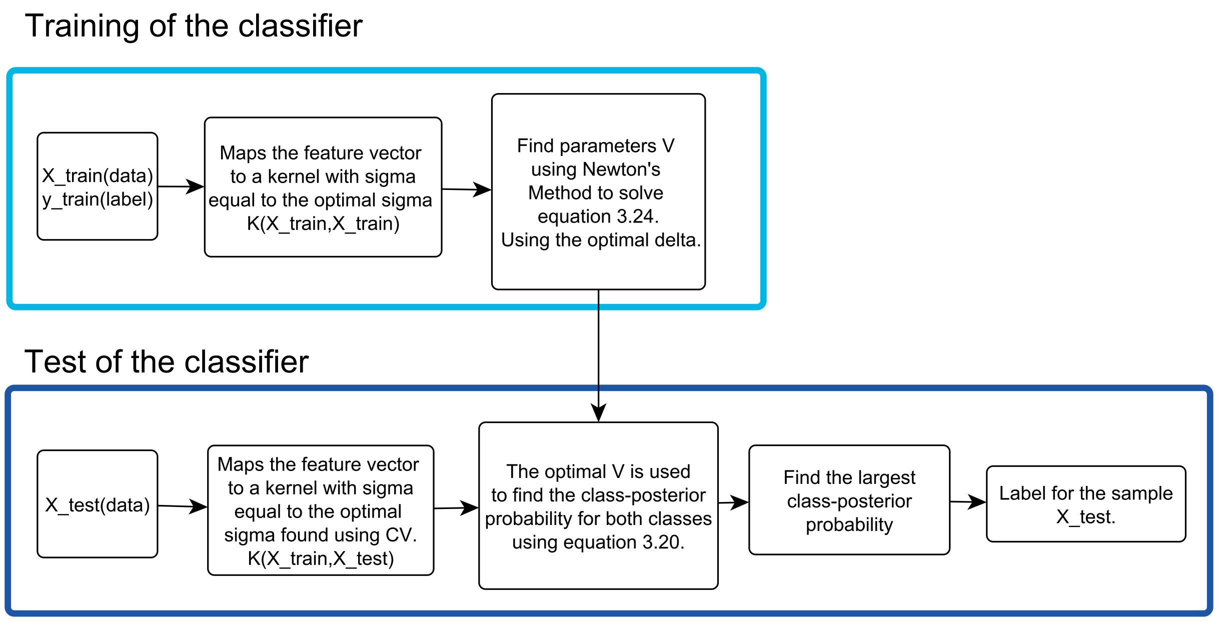

Training of the classifier
X_train(data) y_train(label)
Maps the feature vector to a kernel with sigma equal to the optimal sigma K(X_train,X_train)
Find parameters V using Newton's Method to solve equation 3.24. Using the optimal delta.
Test of the classifier
X_test(data)
Maps the feature vector to a kernel with sigma equal to the optimal sigma found using CV. K(X_train,X_test)
The optimal V is used to find the class-posterior probability for both classes using equation 3.20.
Find the largest class-posterior probability
Label for the sample X_test.

Figure 5.3: Train and test of the KLR classifier

5.2 Semi-supervised Domain Adaptation Techniques

Two different domain adaptation techniques, IWKLR and CSM, were tested on the the P300 speller to overcome the problems caused by the non-stationarity of the EEG signal.

5.2.1 Application of the importance weight kernel logistic regression to the P300 speller

The implementation of IKWLR is very similar to the one of KLR except that, in the process of decision making, an importance is given to each train sample according to their similarity with the new collected samples.

The first step estimates the importance weight w of each training sample, by using tthe KLIEP algorithm as described in Section 3.3.2.3. The σ used to estimate the weight is the same as the one found using the automatic method applied in the KLR algorithm. After having the importance weight, the IWCV is performed to estimate the best parameter (δ) necessary to implement the IWKLR. The IWCV follows the same approach as the one presented in Figure 5.2, however in the IWCV the importance given to each sample in the train dataset is also considered, so instead of using Equation 3.23 we use 3.28, and instead of Equation 3.24 we use the Equation 3.29. In Equation 3.28, however, instead of using the indicator function the posterior probability is used.

The parameters found using IWCV can now be used to train the classifier. A schematic explanation of how the IWKLR is implemented can be seen in Figure 5.4.

Each time a new trial is collected the respective importance weight is estimated and the IWCV is performed to find the best parameters for the IWKLR algorithm. After finding those parameters, the training of the classifier is performed. The collected signal is mapped to a kernel and the classification takes place.

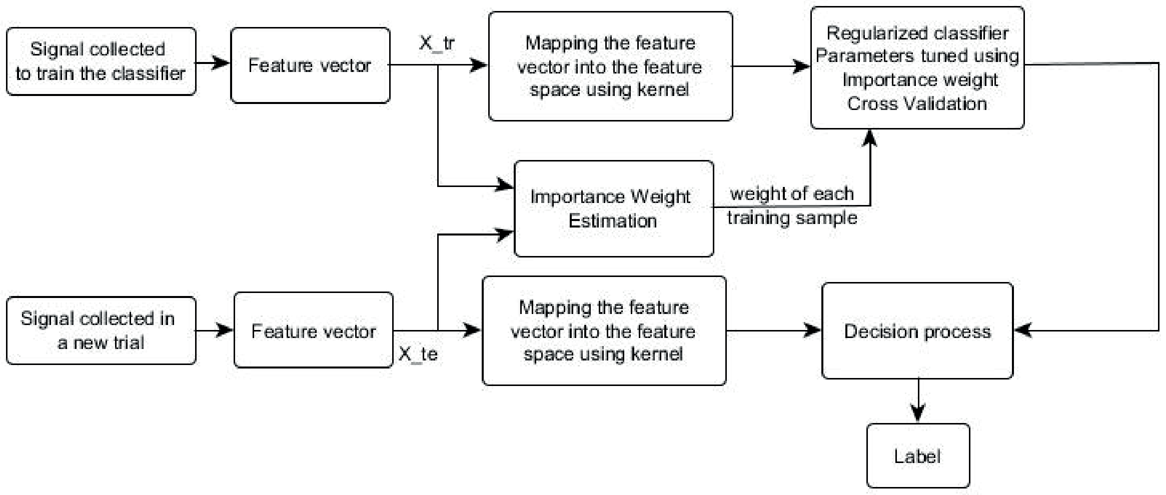

Signal collected to train the classifier
Feature vector
X_tr
Mapping the feature vector into the feature space using kernel
Regularized classifier Parameters tuned using Importance weight Cross Validation
Importance Weight Estimation
weight of each training sample
Signal collected in a new trial
Feature vector
X_te
Mapping the feature vector into the feature space using kernel
Decision process
Label

Figure 5.4: Adaptative architecture of the IWKLR classifier

5.2.2 Application of a method derived from the covariate shift minimization to the P300 speller

The Covariate Shift Minimization (CSM) is a feature adaptation method applied before the classification step. This adaptation is performed individually to each feature of the feature vector separately.

The method implemented for the P300 speller is adapted from the CSM method presented in the previous chapter. Instead of using a polynomial as suggested in [14] the mean value of the feature from the new trial is used to correct the shift that the features suffered from the training dataset to the testing dataset.

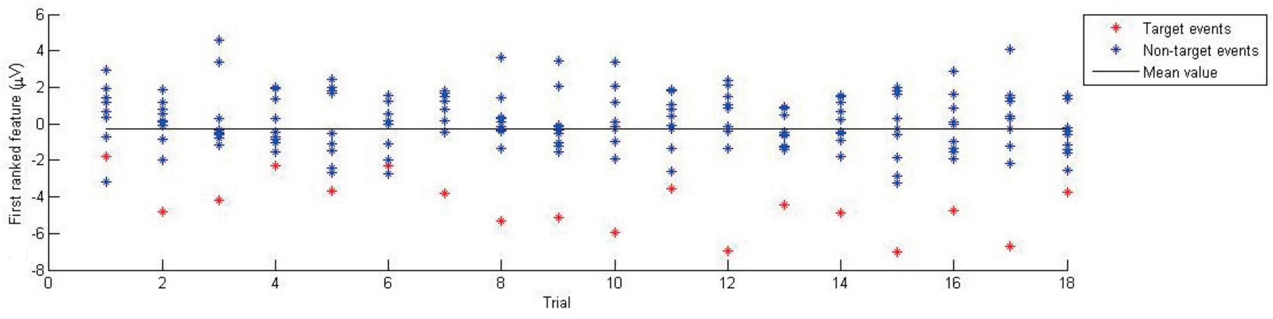
Initially, the training dataset is used to find the mean value of each feature in the training domain (μ_{0i}) and for each feature i in the new trial (T), the corresponding mean value is calculated μ_{Ti} and the value of the feature is updated using the following expression (5.2):

$$Reajusted\,feature_i = f_i(T) - \mu_{Ti} + \mu_{0i} \tag{5.2}$$

where $f_i(T)$ is a vector containing the values of the features i for each sample in the new trial.

This readjustment causes a shift on all the features i collected in the new trial in a way that the mean value of the readjusted features vector is equal to μ_{0i}.

In Figure 5.5, an example of the effects of the implementation of the CSM method for the best ranked feature are presented. On the top figure, the train dataset used to find μ_{0i} is shown, this value is represented by a black line. On the bottom, the values of the best ranked feature for each trial and the readjusted feature are plotted. As it is shown in the figure the mean value of each trial (represented by a continuous black line) is readjusted to be equal to μ_{0i}.



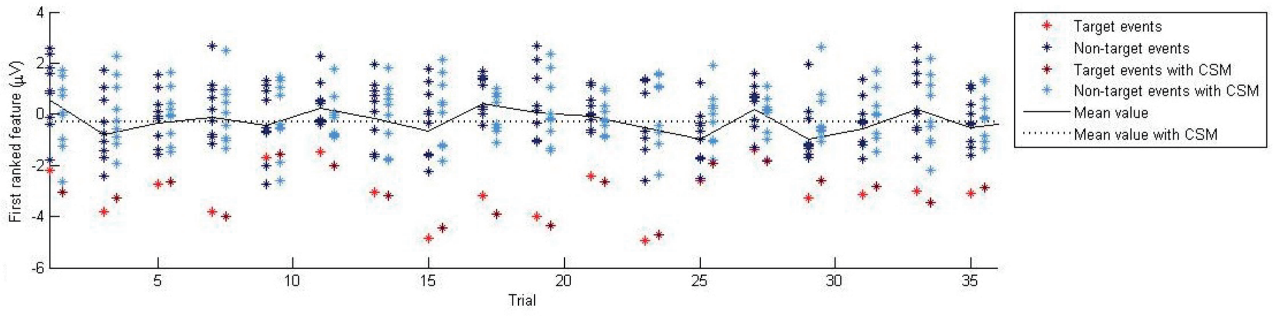


Figure 5.5: On top: the train dataset and the mean value of the best ranked feature. On bottom: The value of the best ranked feature for several trials and the readjusted feature as well as the mean value of the feature per trial.

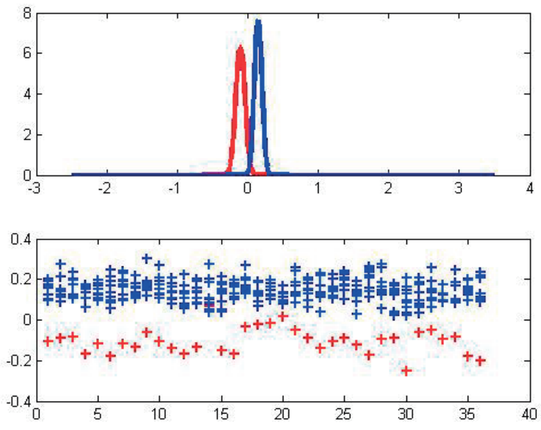
Chapter 6

Results and Analysis

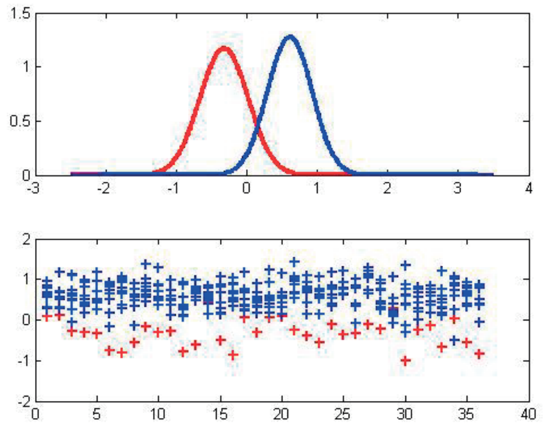
6.1 Supervised-learning

6.1.1 FLD: full or diagonal Sw matrix

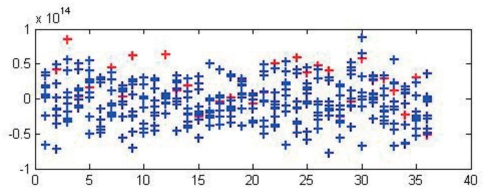
The S_w matrix measures the covariance between the features. Several experiments were performed to evaluate the effects of using the full or the diagonal S_w matrix in the classification using the FLD classifier. The matrix diagonal assumes that the features follow an independent distribution. An example of the projections obtained using the FLD for the full and the diagonal S_w matrix is shown in Figures 6.1 and 6.2 respectively. These projections where made using the average of groups of 5 epochs. The class "target" is plotted in red and the class "non-target" in blue. Looking at Figure 6.1 it can be concluded that, when using the full matrix, it is only possible to distinguish both classes when a small number of features is used. In Figure 6.2 it can be seen that using the diagonal matrix for any number of features always produces better separability. Computation problems of the inverse full-covariance may explain these poor results, which worsen with the increase of the dimensionality. An analysis of the singularities and the application of the pseudo-inverse were inconclusive.



(a) Using full Sw with 50 features

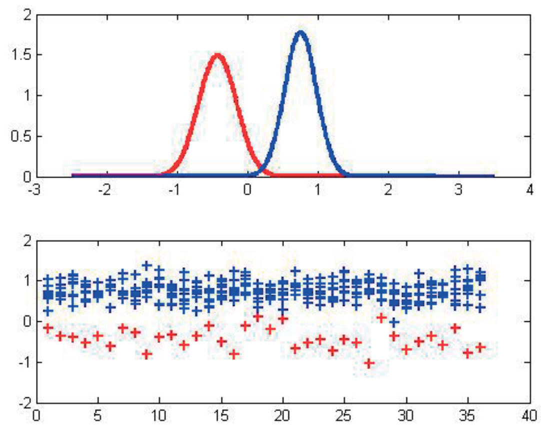


(b) Using full Sw with 125 features

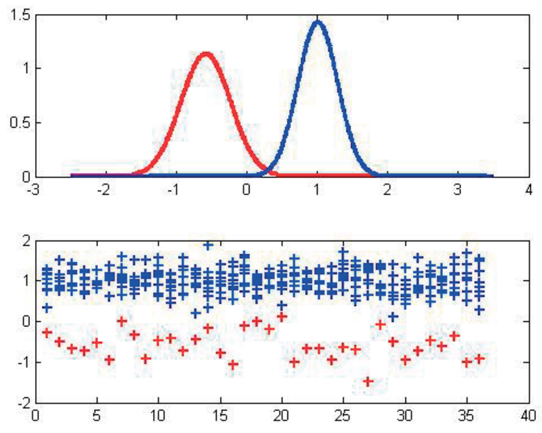


(c) Using full Sw with 200 features

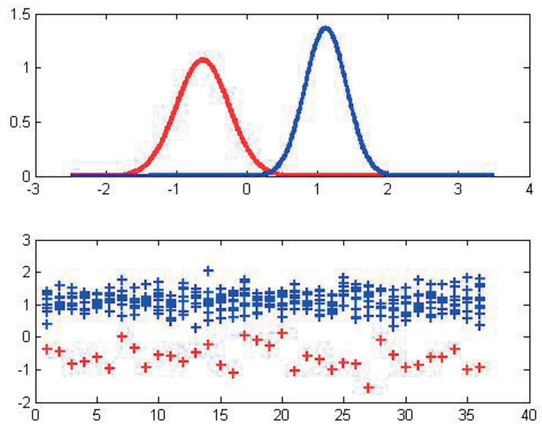
Figure 6.1: Projection of the test samples using full Sw



(a) Using diagonal Sw with 50 features



(b) Using diagonal Sw with 125 features



(c) Using diagonal Sw with 200 features

Figure 6.2: Projection of the test samples using diagonal Sw

Since the usage of the diagonal matrix seems to always produce better results, that was the elected method for this experiment. Looking at the different projections obtained for a different number of features we chose to use 200 features since the mean of the two classes is farther apart.

6.1.2 Datasets collected and experiments

The analysis was performed by using datasets collected in 3 different sessions for 3 different subjects. Each session was performed with more than a week apart. In each session 3 labeled datasets where collected which corresponds to recording 270 target events and 2520 non-target event. Two tests were performed:

- **Test 1** - where all the samples are classified considering only the binary classification (i.e., independent of the application), that is, for each sample a label is assigned using the discriminant function.
- **Test 2** - where we want to simulate the classification problem that occurs in the P300 speller. In this second test the dataset is organized in trials, each trial containing 1 target event and

9 non-target events. By doing this we are simulating the online experiment where each trial should contain 1 target event and 27 non-target events, performing the classification under the assumption that only sample can be classified as target, that is, only one letter can be written on the screen . In this test what is evaluated is which letter is most likely to be a P300 waveform.

- **Test 3** - Online experiment. The user is asked to write a sentence on the screen.

For each test the KLR classifier was trained using two different approaches:

- Balanced (b) - where a dataset containing the same number of target and non-target events is used for training the classifier.
- Unbalanced (u) - where a different number of target and non-target events is used for training the classifier.

For the FLD only an unbalanced training was performed.

The training of the FLD classifier and the unbalanced KLR classifier was performed with the first dataset collected in each session, meaning that, an unbalanced dataset with $\frac{90}{n_{epochs}}$ target events and $\frac{840}{n_{epochs}}$ non-target events was used, and for the balanced KLR the number of target and non-target events used for training the classifier is $\frac{90}{n_{epochs}}$. The test of all the classifiers is performed using the second and the third dataset collected in the session and $\frac{180}{n_{epochs}}$ target events and $\frac{1620}{n_{epochs}}$ non-target events are used. This means that not all the collected non-target samples are used to test the performance of the classifiers. It should be noted that the training of the classifier is always performed with a different dataset than the one used to test the classifier, even if the test and training are performed for the same session.

In order to evaluate the performance of the classifier for Test 1 a balanced error rate was used (6.1):

$$e_b = \frac{FPR + FNR}{2} \tag{6.1}$$

where FPR is the rate of false positives (wrong classified target events rate) and FNR is the rate of false negatives (wrong classified non-target events rate).

For Test 2 and for the online experiment the accuracy of the classification is given by (6.2):

$$P_{ac} = 1 - \frac{N_e}{N_c} \tag{6.2}$$

where N_e is the number of misspelled letters and N_c is the number of characters in the sentence.

In order to evaluate the online performance we also measured the information transfer rate (ITR) introduced by Wolpaw et. al [23] (6.3):

$$ITR = r_s B \tag{6.3}$$

where B gives us the number of bits per symbol and is equal to (6.4):

$$B = log_2(N_s) + P_{ac}log_2(P_{ac}) + (1 - P_{ac})log_2\frac{1 - P_{ac}}{N_s - 1} \tag{6.4}$$

where P_{ac} is the online accuracy and N_s is the number of possible choices (number of symbols = 28) and r_s gives us the number of possible choices per minute (symbols per minute, SPM) (6.5)

$$r_s = \frac{60}{N_{rep} \times (N_{ev} \times SOA) + ITI} \tag{6.5}$$

In the online experiment $N_{rep} = 5$, $N_{ev} = 28$, $SOA = 75ms$ and $ITI = 4s$ (ITI - Inter-stimulus interval). This gives us $r_s = 4.13spm$.

6.1.3 KLR and FLD performance

This section assesses the classification accuracy of FLD and KLR, the training was performed with the first dataset collected in the session and testing with 2 other datasets gathered in the same session.

Test 1 was made to evaluate on a first phase the general performance of each classifier. The results are presented in Table 6.1.

		Nepochs=1			Nepochs=2			Nepochs=3			Nepochs=4			Nepochs=5		
	Session	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)
Subject 1	1	**90.99**	84.48	89.32	**95.68**	84.88	93.15	**96.67**	86.67	96.57	**97.78**	92.22	96.67	**95.83**	87.50	93.06
	2	**89.01**	78.27	87.22	**92.41**	86.67	91.91	**96.48**	89.91	95.28	**96.67**	93.33	96.54	**97.22**	93.06	95.83
	3	**89.38**	82.69	89.10	**96.30**	88.27	94.01	95.65	92.50	**97.13**	**97.65**	92.22	97.53	**98.61**	94.44	98.46
	Mean	**89.79**	81.81	88.55	**94.79**	86.60	93.02	96.27	89.69	**96.33**	**97.37**	92.59	96.91	**97.22**	91.67	95.78
Subject 2	1	91.08	82.13	**91.27**	**97.53**	90.93	97.16	98.15	94.91	**98.98**	**98.27**	96.54	**98.27**	99.69	98.61	99.85
	2	**92.72**	85.65	92.13	96.42	95.00	**96.48**	99.54	97.50	**99.91**	**100.00**	97.78	99.63	**100.00**	**100.00**	99.85
	3	**95.15**	89.04	94.94	98.70	95.56	**99.07**	99.07	95.00	**99.17**	**100.00**	98.89	**100.00**	**100.00**	**100.00**	**100.00**
	Mean	**92.98**	85.61	92.78	97.55	93.83	**97.57**	98.92	95.80	**99.35**	**99.42**	97.74	99.30	**99.90**	99.54	**99.90**
Subject 3	1	86.17	81.05	**87.56**	91.60	85.25	**92.22**	92.50	85.19	**96.48**	96.17	88.77	**96.91**	98.15	95.83	**98.46**
	2	81.36	71.70	**82.10**	89.07	74.44	**89.75**	92.22	77.31	**93.52**	92.84	88.64	**94.20**	95.83	86.11	**96.30**
	3	**83.80**	73.80	83.46	**91.60**	83.58	91.54	**93.24**	81.57	93.06	**97.28**	89.88	96.54	95.68	83.33	**96.30**
	Mean	83.78	75.51	**84.37**	90.76	81.09	**91.17**	92.65	81.36	**94.35**	95.43	89.09	**95.88**	96.55	88.43	**97.02**
Overall Mean		**88.85**	80.98	88.57	**94.37**	87.17	93.92	95.95	88.95	**96.68**	**97.41**	93.14	97.37	**97.89**	93.21	97.57

Table 6.1: Sucess rate for Test 1 - results for 3 different subjects in sessions performed in 3 different days, for $n_{epoch} = 1...5$. Best results per session and per number of epochs for each subject are presented in bold. (KLR (u) - KLR unbalanced and KLR (b) - KLR balanced)

Looking at the results we can conclude, as expected, that using a larger number of epochs in the test improves the obtained results.

In what concerns the perfomance of the three different classifiers we can easily conclude that the train of the KLR using a balanced dataset increases the accuracy of the classifier drastically when compared to the unbalanced KLR, with the mean difference between them being about 7 percentual points. The results obtained with FLD and the KLR trained with a balanced dataset are very similar and the difference between the two is less than 2 percentual points.

	Nepoch = 1	Nepochs = 2	Nepochs =3	Nepochs = 4	Nepochs = 5
FLD	0.027	0.019	0.010	0.014	0.009
KLR (u)	414.17	59.19	22.44	11.17	6.58
KLR (b)	5.43	1.56	0.87	0.63	0.59

Table 6.2: Training time for each classifier for one run in seconds

Besides that, if we look at the time it takes for each classifier to find the model that best suits the training dataset (Table 6.2) we can conclude that the training of the unbalanced KLR is the most demanding and the FLD is almost immediate for all the number of epochs. Both balanced and unbalanced KLR take more time than the FLD since they demand the use of two parameters. The diference between the unbalanced and balanced KLR is related with the amount of data used to train each classifier. This means that there is no real advantage in using a non-linear classifier in the classification process.

The results obtained for the second test, simulating the online operation, are presented in Table 6.3.

		Nepochs=1			Nepochs=2			Nepochs=3			Nepochs=4			Nepochs=5		
	Session	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)
Subject 1	1	83.89	**90.00**	80.56	**97.78**	94.44	95.56	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	2	**82.22**	65.56	76.11	92.22	**94.44**	90.00	95.00	95.00	95.00	**97.78**	95.56	95.56	**100.00**	**100.00**	97.22
	3	81.11	**84.44**	80.56	94.44	94.44	**95.56**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	Mean	**82.41**	80.00	79.07	**94.81**	94.44	93.70	**98.33**	**98.33**	**98.33**	**99.26**	98.52	98.52	**100.00**	**100.00**	99.07
Subject 2	1	88.89	71.11	**86.67**	**98.89**	**98.89**	97.78	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	2	**94.44**	93.89	90.56	98.89	**100.00**	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	3	93.33	**98.33**	96.11	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	Mean	**92.22**	87.78	91.11	98.89	**99.63**	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Subject 3	1	75.00	65.56	**76.11**	87.78	**90.00**	83.33	88.33	88.33	**95.00**	95.56	**97.78**	**97.78**	94.44	**97.22**	**97.22**
	2	63.89	42.78	**68.33**	80.00	74.44	**84.44**	86.67	91.67	**93.33**	86.67	**95.56**	**95.56**	94.44	**100.00**	94.44
	3	**75.00**	47.78	73.89	**87.78**	**87.78**	**87.78**	**96.67**	90.00	93.33	**100.00**	97.78	95.56	97.22	**100.00**	97.22
	Mean	71.30	52.04	**72.78**	**85.19**	84.07	85.19	90.56	90.00	**93.89**	94.07	**97.04**	96.30	95.37	**99.07**	96.30
Overall mean		**81.98**	73.27	80.99	**92.96**	92.72	92.59	96.30	96.11	**97.41**	97.78	**98.52**	98.27	98.46	**99.69**	98.46

Table 6.3: Sucess rate for Test 2 - results for 3 different subjects in sessions performed in 3 different days, for $n_{epoch} = 1...5$. Best results per session and per number of epochs for each subject are presented in bold. (KLR (u) - KLR unbalanced and KLR (b) - KLR balanced)

For one epoch the unbalanced KLR gives us inconsistent results with a low accuracy when compared with both FLD and balanced KLR. This unpredicatable behaviour is well visible in subject 3 where the difference in performance is quite remarkable (20 percentual points when compared with the balanced KLR).

When the number of epochs increases the results obtained with the three used classifiers are very similiar. It should, however, be noticed that the overall mean accuracy is larger for the unbalanced KLR when the number of epochs is larger than 3. We can conclude that all three classifiers can be applied and all will perform well in this application for a large number of epochs.

6.1.4 Assesment of P300 stationarity: session-dependent calibration vs. single calibration

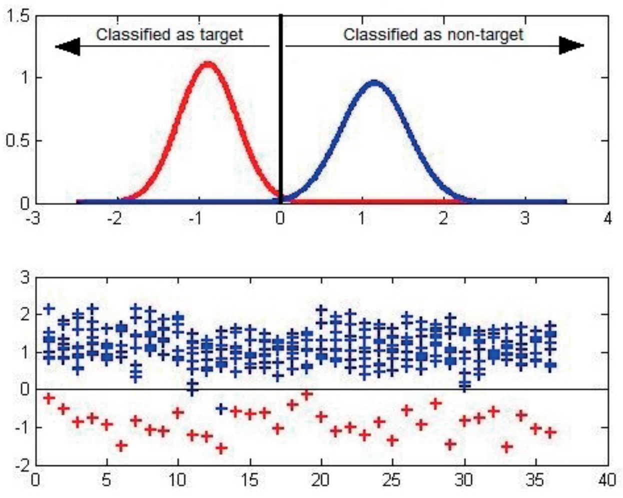
In order to analyze the stationarity of the P300 patterns and robustness of classification models, the performance of the classifiers is tested for the same dataset when the training is performed using datasets collected from the three different sessions. For example, if the test dataset is collected in Session 1 the performance is evaluated when the training is performed using a dataset collected in Session 1, Session 2 and Session 3. This means that one of the datasers used for training the classifier was collected in the same day as the dataset used to test the classifiers and the other two datasets were collected in different sessions. The main goal was to analyse whether a calibration session is needed prior to each session or not.

The results obtained for Test 1 are presented in Table 6.4. Looking at the results obtained in Test 1 we can conclude that the accuracy of the classifier is affected by the non-stationarity of the EEG signal. The difference between the accuracy of the classifier when the train and test are performed in datasets collected in the same session with the accuracy of the train and test perfomed in different sessions in the overall test is around 4.5 percentual points for the balanced KLR and the FLD classifier and around 9 percentual points for the unbalanced KLR.

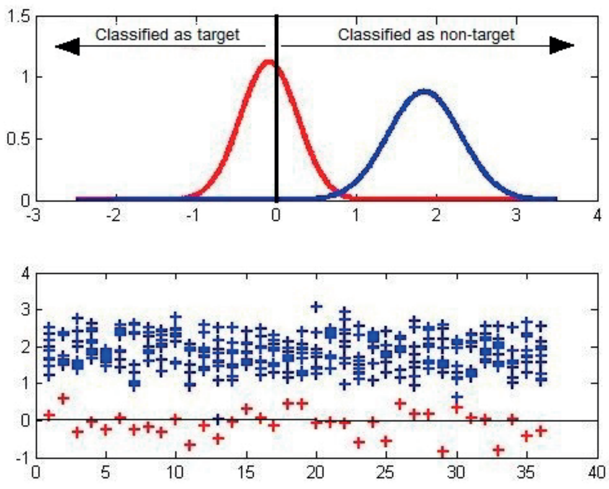
			Nepochs=1			Nepochs=2			Nepochs=3			Nepochs=4			Nepochs=5		
	Test	Train	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)
Subject 1	1	**1**	**90.99**	84.48	89.32	**95.68**	84.88	93.15	**96.67**	86.67	96.57	**97.78**	92.22	96.67	**95.83**	87.50	93.06
		2	**83.06**	76.98	79.97	88.95	74.94	**87.59**	92.22	78.24	**93.80**	94.32	85.56	**95.31**	**94.44**	86.11	94.29
		3	86.27	75.93	**86.48**	90.80	82.04	**91.30**	92.87	88.06	**93.33**	**93.95**	86.54	92.35	**95.52**	88.73	95.37
	2	1	84.10	66.48	**75.68**	87.90	65.43	**88.64**	**88.15**	65.00	85.56	**91.11**	65.56	90.99	87.50	70.83	**91.67**
		2	**89.01**	78.27	87.22	**92.41**	86.67	91.91	**96.48**	89.91	95.28	**96.67**	93.33	96.54	**97.22**	93.06	95.83
		3	81.64	66.76	**81.94**	88.40	75.31	**90.49**	89.72	77.31	**89.81**	93.33	74.44	**94.94**	93.06	79.17	**95.52**
	3	1	**87.96**	75.86	86.48	**93.02**	73.89	91.79	92.41	79.17	**95.83**	**95.56**	84.44	**95.56**	**95.56**	77.78	94.44
		2	**82.16**	71.23	81.02	**85.43**	68.27	85.25	85.83	73.33	**91.57**	92.22	72.22	**94.32**	**91.67**	69.44	88.89
		3	**89.38**	82.69	89.10	**96.30**	88.27	94.01	95.65	92.50	**97.13**	**97.65**	92.22	97.53	**98.61**	94.44	98.46
	μ_{ss}		**89.79**	81.81	88.55	**94.79**	86.60	93.02	96.27	89.69	**96.33**	**97.37**	92.59	96.91	**97.22**	91.67	95.78
	μ_{ds}		**84.20**	72.21	81.93	89.08	73.31	**89.18**	90.20	76.85	**91.65**	93.42	78.13	**93.91**	92.96	78.68	**93.36**
Subject 2	1	**1**	91.08	82.13	**91.27**	**97.53**	90.93	97.16	98.15	94.91	**98.98**	**98.27**	96.54	**98.27**	99.69	98.61	**99.85**
		2	92.62	84.23	**93.49**	97.22	92.22	**97.41**	99.72	95.00	**99.91**	98.77	98.89	**100.00**	99.85	98.61	**100.00**
		3	75.59	64.91	**79.66**	**77.65**	65.00	77.10	**79.07**	63.33	**79.07**	**80.99**	57.78	76.67	**79.17**	58.33	75.00
	2	1	89.51	82.16	**90.59**	95.68	93.02	**96.73**	96.67	97.31	**99.07**	97.04	**97.78**	96.91	**98.15**	94.44	97.99
		2	**92.72**	85.65	92.13	**96.42**	95.00	96.48	99.54	97.50	**99.91**	**100.00**	97.78	99.63	**100.00**	**100.00**	99.85
		3	78.80	66.17	**80.80**	81.23	67.22	**85.49**	85.65	65.00	**88.24**	86.54	68.89	**86.67**	**88.89**	63.89	**88.89**
	3	1	**94.63**	90.93	93.67	97.96	97.65	**98.58**	99.35	**99.91**	98.89	99.75	**100.00**	99.51	**100.00**	**100.00**	99.85
		2	**94.41**	90.43	92.10	**98.27**	95.49	97.16	98.70	99.17	**99.63**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	99.85
		3	**95.15**	89.04	94.94	98.70	95.56	**99.07**	99.07	95.00	**99.17**	**100.00**	98.89	**100.00**	100.00	**100.00**	**100.00**
	μ_{ss}		**92.98**	85.61	92.78	97.55	93.83	**97.57**	98.92	95.80	**99.35**	**99.42**	97.74	99.30	**99.90**	99.54	**99.90**
	μ_{ds}		87.59	79.80	**88.38**	91.34	85.10	**92.08**	93.19	86.62	**94.14**	**93.85**	87.22	93.29	**94.34**	85.88	93.60
Subject 3	1	**1**	86.17	81.05	**87.56**	91.60	85.25	**92.22**	92.50	85.19	**96.48**	96.17	88.77	**96.91**	98.15	95.83	**98.46**
		2	**83.40**	72.78	83.18	**90.56**	77.65	90.49	**92.22**	83.15	90.93	92.59	86.67	**93.70**	**96.60**	83.33	92.59
		3	82.10	68.09	**83.33**	85.31	73.70	**87.47**	91.20	69.07	**91.67**	89.75	83.33	**94.81**	91.67	73.61	**92.75**
	2	1	81.05	68.30	**81.23**	**89.81**	80.80	89.26	**91.20**	88.89	90.19	91.60	86.91	**91.85**	93.36	89.97	**93.83**
		2	81.36	71.70	**82.10**	**89.07**	74.44	89.75	92.22	77.31	**93.52**	**92.84**	88.64	**94.20**	95.83	86.11	**96.30**
		3	81.39	67.84	**82.90**	**87.53**	77.41	77.41	**93.24**	77.41	92.50	**93.58**	84.44	91.98	95.52	87.50	**96.14**
	3	1	81.17	70.25	**81.88**	87.90	86.30	**88.89**	88.98	86.11	**90.09**	94.07	85.31	**94.32**	93.98	91.51	**94.44**
		2	**78.40**	62.87	77.35	86.05	70.86	**87.10**	**89.17**	74.54	86.30	**89.51**	82.10	**89.51**	93.06	83.33	**94.91**
		3	**83.80**	73.80	83.46	**91.60**	83.58	91.54	93.24	81.57	**93.06**	**97.28**	89.88	96.54	95.68	83.33	**96.30**
	μ_{ss}		83.78	75.51	**84.37**	90.76	81.09	**91.17**	92.65	81.36	**94.35**	95.43	89.09	**95.88**	96.55	88.43	**97.02**
	μ_{ds}		81.25	68.35	**81.65**	**87.86**	77.79	86.77	**91.00**	79.86	90.28	91.85	84.79	**92.70**	94.03	84.88	**94.11**
Overall μ_{ss}			**88.85**	80.98	88.57	**94.37**	87.17	93.92	95.95	88.95	**96.68**	**97.41**	93.14	97.37	**97.89**	93.21	97.57
Overall μ_{ds}			**84.35**	73.46	83.99	**89.43**	78.73	89.34	91.47	81.11	**92.02**	93.04	83.38	**93.30**	**93.78**	83.14	93.69

Table 6.4: Sucess rate for Test 1 - Test of the performance of the classifier. The train is done using three different datasets one collected in the same session as the dataset used to test the classifier and the other two collected in different days. Where μ_{ss} is the mean accuracy value when the train and test are performed with datasets collected in the same session and μ_{ds} is the mean accuracy value when the train and test are performed with datasets collected in different sessions for each subject.

This difference in classification accuracy should be carefully analyzed. If we look at the projections obtained using the FLD classifier in Figure 6.3 we see that the poor performance obtained using datasets from test and train performed in different days is in fact the effect of a shift in the projection. The separability between the projection of samples belonging to different classes is still as good as it was using the train and test datasets collected in the same day. This indicates that CSM technique will most likely be useful to increase the results of Test 1. However, by comparing the two top figures we can conclude that this shift should not affect the performance of the P300 speller since the classification problem consists in finding the event that is most likely to contain the P300 waveform. Meaning that, the second test is basically immune to this shift. In order to confirm this assumption we performed the simulation of the P300 speller classification problem to see if it was different to train the classifier using a dataset collected in a different session. An illustration of this test is presented in the bottom image of Figure 6.3. The dataset used in this figure is from the Subject 2, the test of the classifier was performed using the dataset collected in Session 1 and the train was done using a dataset collected in the same session and a dataset collected in Session 3. By looking at the Table 6.4 we see that the success rate for the train and test using datasets collected in Session 1 is equal to 99.69% and using the dataset collect in Session 3 to train the classifier the sucess rate was equal to 79.17%.



(a) Projections of samples using FLD. For test and train performed with datasets collected in the same session.



(b) Projections of samples using FLD. For test and train performed with datasets collected in different sessions.

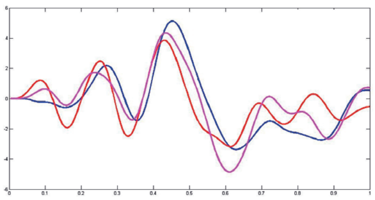
Figure 6.3: Top: Illustration of Test 1; Bottom: Illustration of Test 2

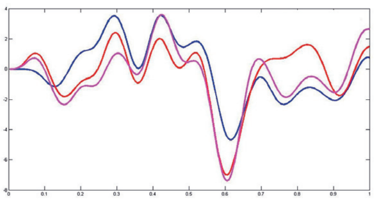
Now, we should analyze the performance of the classifier simulating the P300 application. By the analysis made before, we expect the classifier to suffer no change in the performance even if the train of the classifier is performed in a different day from the test. The results obtained from Test 2 are presented in Table 6.5

By analyzing the results obtained from Test 2, we can conclude that for a small number of epochs the classification accuracy is affected by the non-stationarity of the EEG signal. The overall difference between the train and test from the same session to different sessions is around 9-10 percentual points for one epoch. However, as the number of epochs increases the difference between train and test performed in the same day and performed in different days decreases around 0-2 percentual points for 5 epochs. This means that for a larger number of epochs the system mantains good performance even if the calibration of the system is performed with a dataset collected in a different session than the dataset used to test the classifier. This leads us to conclude that it is possible to use the P300 speller system performing a one-time calibration if the number of epochs per trial is higher than 3 for FLD (where the overall error between the same session and different sessions is smaller than 1 percentual points) and higher than 4 for balanced and unbalanced KLR (where the overall error between the same session and different sessions is around to 2 percentual points). This means that, despite the fact that the ongoing EEG signal suffers from non-stationarity, the mean value of the P300 waveform is similar in every session. To confirm this assumption we plot the mean value of the P300 waveform measured in the CZ channel overtime in the 3 different sessions. This plot is presented in Figure 6.4 and, as we can see, the waveform for the 3 subjects remains very similar in all the three different sessions.

	Test	Train	Nepochs=1			Nepochs=2			Nepochs=3			Nepochs=4			Nepochs=5		
			FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)
Subject 1	1	**1**	83.89	**90.00**	80.56	**97.78**	94.44	95.56	**100.00**	**100.00**	**100.00**	100.00	**100.00**	**100.00**	100.00	**100.00**	**100.00**
		2	71.67	65.00	**72.78**	**91.11**	**91.11**	84.44	**96.67**	91.67	91.67	**100.00**	95.56	97.78	**100.00**	**100.00**	**100.00**
		3	72.78	**80.56**	69.44	90.00	91.11	**97.78**	93.33	96.67	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	2	1	**72.22**	60.56	50.56	**86.67**	74.44	84.44	**90.00**	85.00	85.00	**95.56**	93.33	**95.56**	**97.22**	**97.22**	**97.22**
		2	**82.22**	65.56	76.11	92.22	**94.44**	90.00	**95.00**	**95.00**	**95.00**	**97.78**	95.56	95.56	**100.00**	**100.00**	97.22
		3	**59.44**	53.89	57.78	**84.44**	74.44	83.33	**93.33**	88.33	90.00	**97.78**	88.89	91.11	**100.00**	97.22	97.22
	3	1	**86.67**	80.56	69.44	96.67	91.11	**97.78**	98.33	96.67	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		2	**81.67**	62.78	73.33	**94.44**	88.89	90.00	**100.00**	98.33	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		3	81.11	**84.44**	80.56	94.44	94.44	**95.56**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	μ_{ss}		**82.41**	80.00	79.07	**94.81**	94.44	93.70	**98.33**	**98.33**	**98.33**	**99.26**	98.52	98.52	**100.00**	**100.00**	99.07
	μ_{ds}		**74.07**	67.22	65.56	**90.56**	85.19	89.63	**95.28**	92.78	94.44	**98.89**	96.30	97.41	**99.54**	99.07	99.07
Subject 2	1	**1**	**88.89**	71.11	86.67	**98.89**	**98.89**	97.78	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		2	91.67	**92.22**	90.56	98.89	**100.00**	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		3	77.22	**82.22**	78.33	90.00	**92.22**	88.89	98.33	98.33	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	2	1	**85.56**	71.11	82.78	92.22	95.56	**96.67**	96.67	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		2	**94.44**	93.89	90.56	98.89	**100.00**	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		3	66.67	75.00	**80.56**	90.00	**92.22**	91.11	96.67	95.00	**98.33**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	3	1	**92.78**	85.56	92.22	**100.00**	98.89	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		2	**91.67**	90.56	86.67	**100.00**	**100.00**	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		3	93.33	**98.33**	96.11	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	μ_{ss}		**92.22**	87.78	91.11	98.89	**99.63**	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	μ_{ds}		84.26	82.78	**85.19**	95.19	**96.48**	95.56	98.61	98.89	**99.72**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Subject 3	1	**1**	75.00	65.56	**76.11**	**87.78**	90.00	83.33	88.33	88.33	**95.00**	95.56	**97.78**	**97.78**	94.44	**97.22**	**97.22**
		2	71.67	48.33	**74.44**	81.11	**84.44**	**84.44**	86.67	**93.33**	88.33	**97.78**	**97.78**	93.33	97.22	**100.00**	97.22
		3	69.44	51.11	**71.67**	**85.56**	**85.56**	84.44	**88.33**	**88.33**	86.67	**95.56**	**95.56**	**95.56**	**97.22**	94.44	91.67
	2	1	60.00	34.44	**60.56**	**81.11**	76.67	80.00	83.33	80.00	**86.67**	**84.44**	**84.44**	80.00	91.67	**94.44**	86.11
		2	63.89	42.78	**68.33**	**80.00**	74.44	84.44	86.67	91.67	**93.33**	86.67	**95.56**	**95.56**	94.44	**100.00**	94.44
		3	65.00	45.00	**66.67**	84.44	**85.56**	82.22	**91.67**	**91.67**	85.00	91.11	**95.56**	**95.56**	**100.00**	**100.00**	**100.00**
	3	1	61.67	41.67	**67.22**	77.78	81.11	**83.33**	81.67	78.33	**85.00**	**91.11**	88.89	86.67	**94.44**	**94.44**	86.11
		2	**55.00**	31.11	**55.00**	72.22	72.22	**77.78**	83.33	**76.67**	73.33	**97.78**	91.11	93.33	94.44	**100.00**	97.22
		3	**75.00**	47.78	73.89	**87.78**	**87.78**	**87.78**	**96.67**	90.00	93.33	**100.00**	97.78	95.56	97.22	**100.00**	97.22
	μ_{ss}		71.30	52.04	**72.78**	**85.19**	84.07	**85.19**	90.56	90.00	**93.89**	94.07	97.04	**96.30**	95.37	**99.07**	96.30
	μ_{ds}		63.80	41.94	**65.93**	80.37	80.93	**82.04**	**85.83**	84.72	84.17	**92.96**	92.22	90.74	95.83	**97.22**	93.06
Overall μ_{ss}			**81.98**	73.27	80.99	**92.96**	92.72	92.59	96.30	96.11	**97.41**	97.78	**98.52**	98.27	98.46	**99.69**	98.46
Overall μ_{ds}			**74.04**	63.98	72.22	88.70	87.53	**89.07**	**93.24**	92.13	92.78	**97.28**	96.17	96.05	98.46	**98.77**	97.38

Table 6.5: Sucess rate for Test 2 - Test of the performance of the classifier. The train is done using 3 different datasets one collected in the same session as the dataset used to test the classifier and the other two collected in different days.


(a) Subject 1


(b) Subject 2

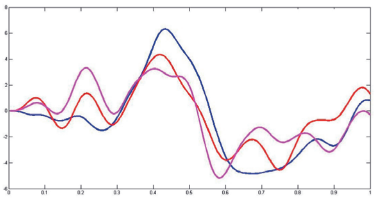
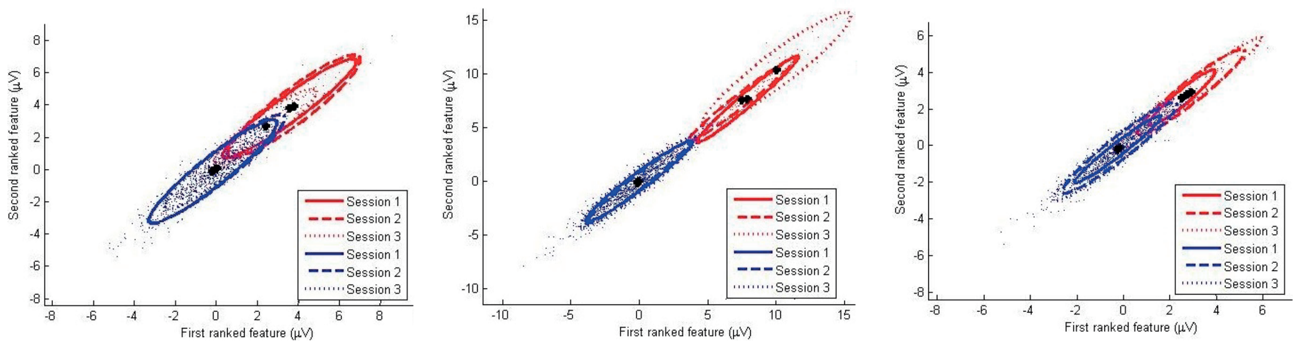

(c) Subject 3

Figure 6.4: P300 waveform measured in channel Cz. Blue - Session 1; Red - Session 2; Magent - Session 3;

Figure 6.5 presents the mean and covariate contour for the two best ranked features when the same trained classifier is applied in different test datasets. Looking at the figure we can conclude that for Subjects 1 and 3 there is almost no difference between the two selected features since the mean value and the covariate contour are almost overlapping. This means that the value of these two features is almost the same in the three sessions for the average of 5 epochs. For Subject 2 when the test is performed with a dataset collected in the same session we can see that the average of the two target best features is farthest apart from the ongoing EEG. However, the separation bewteen the two features is still very good and the covariate contour includes the value of the features in the two other sessions.



(a) Subject 1 (b) Subject 2 (c) Subject 3

Figure 6.5: Mean and covariate contour for the two best ranked features. The training of the classifier was performed with a dataset using the average of 5 epochs collected in session 2 and with test performed with datasets collected in sessions 1, 2 and 3. The blue information referes to the non-target events and the red to the target events.

6.1.5 Online experiments

In order to prove the results shown before, an online test using 5 epochs was performed. In this test (Test 3), the classifier was trained using a dataset collected in a previous session made with at least a week apart. This means that no calibration was performed in that session. This allows us to evaluate the stability of the classifier and wether or not the calibration is mandatory.

It was asked of three different users to write on the screen the sentence: "THESE-AREN-T-THE-DROIDS". One of these users only performed the online test (Subject 4). The unbalanced KLR and the FLD classifiers were used. The accuracy and ITR results obtained in the online session are presented in Table 6.6

	P_{ac}		ITR	
	FLD	KLR	FLD	KLR
Subject 1	95.65	**100.00**	17.93	19.85
Subject 2	91.3	**100**	16.38	19.85
Subject 4	91.3	**95.65**	16.38	17.93

Table 6.6: Online accuracy and ITR for 3 different subjects using a dataset collected in a different session to calibrate the system

As expected, the user was able to write on the screen with great accuracy corroborating the results presented before. However, the performance of the unbalanced KLR was higher than the one obtained with the FLD classifier. This means that using 5 epochs, the effects of the non-stationarity in the P300 recognition are almost non-existent, since the mean value of the P300 waveform is very similiar in all sessions.

This result allow us to conclude that a high number of epochs should be used to mantain a good accuracy for the classification. The drawback is that a small bit/rate is imposed to the users if they don't want to perform a calibration each time they use the system. The bit/rate is an important parameter when evaluating the performance of the P300 speller and we want to keep it as large as possible. Domain Adaptation Techniques are applied in an attempt to improve the accuracy of the classifier for a smaller number of epochs.

6.2 Domain Adaptation Techniques

Since the non-stationarity in the EEG signal prevents us from using the same calibration in different sessions for a small number of epochs, domain adaptation techniques were applied to see if it was possible to reduce the effect of the non-stationarity. Both Test 1 and 2 were performed.

6.2.1 Performance of the KLR and FLD using CSM

The results obtained using the CSM method for Test 1 are presented in Table 6.7. We can conclude that the application of the CSM method increases the accuracy of the classifier for the overall test by around 2 percentual points. This means that the difference between the train and test performed in different days for the test and train, perfomed in the same day is around 2.5 percentual points for the balanced the KLR and FLD and around 7.5 percentual points for the unbalanced KLR.

	Test	Train	Nepochs=1			Nepochs=2			Nepochs=3			Nepochs=4			Nepochs=5		
			FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)
Subject 1	1	2	**84.88**	76.27	82.99	**89.14**	76.54	88.15	92.31	82.50	**93.80**	95.56	84.44	**96.42**	**95.83**	84.72	94.29
		3	87.99	78.33	**88.33**	**93.33**	82.65	91.79	92.78	88.15	**93.98**	**95.43**	88.89	93.70	**98.46**	88.89	97.99
	2	1	**85.83**	67.50	77.99	89.01	67.78	**91.54**	**90.74**	67.50	88.06	91.11	68.89	**92.22**	**90.28**	70.83	**90.28**
		3	**82.35**	66.33	81.54	90.31	73.83	**90.56**	89.63	79.17	**90.83**	**94.32**	77.78	93.70	94.44	79.17	**95.37**
	3	1	**90.56**	77.56	86.54	91.98	75.56	**92.47**	**97.50**	80.83	96.67	97.78	86.67	**98.89**	**97.22**	79.17	**97.22**
		2	**85.12**	74.88	84.48	**88.77**	67.22	87.90	89.17	75.83	**90.83**	95.56	74.44	**96.67**	**95.83**	77.78	93.06
	μ_{ds}		**84.20**	72.21	81.93	89.08	73.31	**89.18**	90.20	76.85	**91.65**	93.42	78.13	**93.91**	92.96	78.68	**93.36**
	μ_{CSM}		**86.12**	73.48	83.65	**90.42**	73.93	90.40	92.02	79.00	**92.36**	94.96	80.19	**95.27**	**95.34**	80.09	94.70
Subject 2	1	2	**93.98**	83.77	93.64	97.47	91.67	**97.65**	100.00	94.17	**100.00**	**100.00**	98.89	99.88	**100.00**	**100.00**	**100.00**
		3	80.49	68.30	**82.50**	**84.32**	65.00	83.21	84.17	63.33	**86.67**	**91.11**	64.44	88.89	**88.89**	59.72	87.50
	2	1	91.91	84.97	**92.78**	96.36	94.38	**97.59**	97.69	96.67	**99.35**	**98.52**	97.78	97.53	**98.30**	97.22	98.15
		3	81.20	69.23	**82.35**	84.44	71.11	**87.16**	89.81	69.17	**89.91**	**90.00**	75.56	87.78	**95.83**	65.28	93.06
	3	1	**94.91**	92.07	94.38	**99.38**	98.15	99.01	99.35	**99.91**	99.26	99.88	**100.00**	99.88	**100.00**	**100.00**	**100.00**
		2	**94.94**	90.49	93.06	**99.69**	97.16	99.01	98.89	**100.00**	99.81	**100.00**	**100.00**	99.88	**100.00**	**100.00**	99.85
	μ_{ds}		87.59	79.80	**88.38**	91.34	85.10	**92.08**	93.19	86.62	**94.14**	**93.85**	87.22	93.29	**94.34**	85.88	93.60
	μ_{CSM}		89.57	81.47	**89.78**	93.61	86.24	**93.94**	94.98	87.21	**95.83**	96.58	89.44	**95.64**	**97.17**	87.04	96.42
Subject 3	1	2	84.81	74.94	**85.40**	91.60	79.38	**91.91**	**93.89**	86.57	92.04	**96.30**	88.89	96.17	97.22	**100.00**	97.22
		3	84.07	69.01	**85.22**	88.15	73.21	**90.19**	92.69	72.31	**94.26**	91.98	84.44	**96.05**	92.90	72.22	**94.14**
	2	1	**83.24**	69.66	82.69	**91.79**	84.20	89.14	90.09	87.87	**90.74**	91.11	87.53	**92.35**	**93.52**	87.35	93.06
		3	81.91	68.43	**82.81**	88.64	75.99	**88.95**	**93.52**	80.74	92.96	94.94	83.33	**95.56**	96.91	91.67	**98.15**
	3	1	**84.29**	70.90	83.09	88.64	85.31	**90.31**	90.09	85.65	**90.93**	94.07	88.64	**94.44**	95.37	92.90	**96.30**
		2	80.77	64.72	81.23	87.16	72.04	**89.07**	**92.87**	76.11	89.17	**94.69**	78.89	93.70	**96.60**	86.11	95.37
	μ_{ds}		81.25	68.35	**81.65**	**87.86**	77.79	86.77	**91.00**	79.86	90.28	91.85	84.79	**92.70**	94.03	84.88	**94.11**
	μ_{CSM}		83.18	69.61	**83.41**	89.33	78.35	**89.93**	**92.19**	81.54	91.68	93.85	85.29	**94.71**	95.42	88.37	**95.70**
Overall μ_{ss}			**88.85**	80.98	88.57	**94.37**	87.17	93.92	95.95	88.95	**96.68**	**97.41**	93.14	97.37	**97.89**	93.21	97.57
Overall μ_{ds}			**84.35**	73.46	83.99	**89.43**	78.73	89.34	91.47	81.11	**92.02**	93.04	83.38	**93.30**	**93.78**	83.14	93.69
Overall μ_{CSM}			**86.29**	74.85	85.61	91.12	79.51	**91.42**	93.07	82.58	**93.29**	95.13	84.97	**95.21**	**95.98**	85.17	95.61

Table 6.7: Sucess rate for Test 1 - Test of the performance of the classifier using CSM.

Now we should analyze the application of the CSM method for changes in the performance in the specific case of the P300 speller. The results obtained for the second test are presented in Table 6.8.

For FLD there is no change in the classification accuracy. For both balanced and unbalanced KLR, small changes in accuracy can be observed. However, sometimes the application of the CSM method damages the classifier accuracy. This change in accuracy for the overall test is around 1 percentual points.

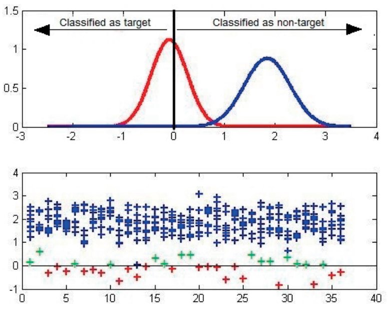
Despite the improvement in the accuracy detected in Test 1, the application of the CSM method in the P300 speller will not increase the online accuracy of the system, which coroborates our conclusions in Section 6.1.4. The CSM method seems to be well suited for motor imagery BCI, where only one feature is used and it represents the band powers associated to mu and beta rhythms. It was revealed to be less suited to ERP classification.

			Nepochs=1			Nepochs=2			Nepochs=3			Nepochs=4			Nepochs=5		
	Test	Train	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)
Subject 1	1	2	71.67	68.89	**73.89**	**91.11**	**91.11**	83.33	**96.67**	91.67	91.67	**100.00**	95.56	**100.00**	**100.00**	**100.00**	**100.00**
		3	72.78	72.78	**73.33**	**90.00**	86.67	88.89	93.33	**95.00**	91.67	**100.00**	97.78	95.56	**100.00**	97.22	**100.00**
	2	1	**72.22**	63.33	51.11	**86.67**	75.56	**86.67**	**90.00**	85.00	86.67	**95.56**	**95.56**	**95.56**	**97.22**	**97.22**	**97.22**
		3	59.44	54.44	**60.00**	**84.44**	76.67	**84.44**	**93.33**	88.33	90.00	**97.78**	86.67	91.11	**100.00**	97.22	97.22
	3	1	**86.67**	81.11	72.78	96.67	93.33	**97.78**	98.33	96.67	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		2	**81.67**	65.00	72.78	**94.44**	91.11	91.11	**100.00**	98.33	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	μ_{ds}		**74.07**	67.22	65.56	**90.56**	85.19	89.63	**95.28**	92.78	94.44	**98.89**	96.30	97.41	**99.54**	99.07	99.07
	μ_{CSM}		**74.07**	67.59	67.31	**90.56**	85.74	88.70	**95.28**	92.50	93.33	**98.89**	95.93	97.04	**99.54**	98.61	99.07
Subject 2	1	2	91.67	**92.78**	91.11	98.89	**100.00**	97.78	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		3	77.22	**82.22**	77.22	90.00	**93.33**	90.00	98.33	98.33	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	2	1	**85.56**	75.00	82.78	92.22	95.56	**96.67**	96.67	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		3	66.67	78.33	**81.67**	90.00	**95.56**	92.22	96.67	98.33	98.33	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	3	1	**92.78**	87.22	91.11	**100.00**	98.89	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
		2	**91.67**	**91.67**	87.78	**100.00**	**100.00**	98.89	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	μ_{ds}		84.26	82.78	**85.19**	95.19	**96.48**	95.56	98.61	98.89	**99.72**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
	μ_{CSM}		84.26	84.54	**85.28**	95.19	**97.22**	95.74	98.61	99.44	**99.72**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Subject 3	1	2	71.67	49.44	**74.44**	81.11	**86.67**	83.33	86.67	**91.67**	88.33	**97.78**	**97.78**	93.33	97.22	**100.00**	97.22
		3	69.44	50.56	**72.22**	85.56	**86.67**	84.44	88.33	88.33	**86.67**	**95.56**	**95.56**	**95.56**	**97.22**	91.67	91.67
	2	1	60.00	36.67	**62.22**	**81.11**	78.89	80.00	83.33	80.00	**85.00**	**84.44**	**84.44**	**84.44**	91.67	**94.44**	88.89
		3	65.00	45.00	**67.22**	**84.44**	**84.44**	80.00	91.67	**93.33**	85.00	91.11	**95.56**	**95.56**	**100.00**	**100.00**	**100.00**
	3	1	61.67	45.00	**65.00**	77.78	80.00	**83.33**	**81.67**	76.67	80.00	**91.11**	88.89	88.89	**94.44**	**94.44**	86.11
		2	**55.00**	36.11	**55.00**	72.22	71.11	**78.89**	**83.33**	75.00	75.00	**97.78**	93.33	93.33	94.44	**100.00**	97.22
	μ_{ds}		63.80	41.94	**65.93**	80.37	80.93	**82.04**	**85.83**	84.72	84.17	**92.96**	92.22	90.74	95.83	**97.22**	93.06
	μ_{CSM}		63.80	43.80	**66.02**	80.37	81.30	**81.67**	**85.83**	84.17	83.33	**92.96**	92.59	91.85	95.83	**96.76**	93.52

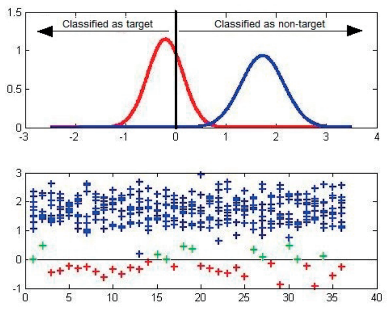
	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)	FLD	KLR (u)	KLR (b)
Overall μ_{ss}	**81.98**	73.27	80.99	**92.96**	92.72	92.59	96.30	96.11	**97.41**	97.78	**98.52**	98.27	98.46	**99.69**	98.46
Overall μ_{ds}	**74.04**	63.98	72.22	88.70	87.53	**89.07**	**93.24**	92.13	92.78	**97.28**	96.17	96.05	98.46	**98.77**	97.38
Overall μ_{CSM}	**74.04**	65.31	72.87	**88.70**	88.09	**88.70**	**93.24**	92.04	92.13	**97.28**	96.17	96.30	**98.46**	**98.46**	97.53

Table 6.8: Sucess rate for Test 2 - Test of the performance of the classifier using CSM.

Figure 6.6 shows the FLD projection with and without appling the CSM method. In the figure we can see that with CSM the number of missclassified samples decreases, because the CSM causes a slightl shift in the final projections of the FLD.



(a) Projections of samples using FLD with no application of the CSM mehtod.



(b) Projections of samples using FLD with application of the CSM method

Figure 6.6: Top: Illustration of Test 1; Bottom: Illustration of Test 2; The blue represents the non-target events, the red the target events and the green represents the missclassified events for Test 1

6.2.2 Performance of the IWKLR classifier

The results obtained from the application of the IWKLR to the system for Test 1 and 2 are presented in both Table 6.9 and 6.10.

			Nepochs = 2	Nepochs = 3	Nepochs = 4	Nepochs = 5
Subject 1	Unbalanced	$\mu_{ds}KLR$	73.31	76.85	**78.13**	78.68
		$\mu_{ds}IWKLR$	**74.83**	**77.42**	77.78	**79.37**
	Balanced	$\mu_{ds}KLR$	**89.18**	**91.65**	**93.91**	**93.36**
		$\mu_{ds}IWKLR$	87.28	90.99	92.26	92.03
Subject 2	Unbalanced	$\mu_{ds}KLR$	85.10	86.62	**87.22**	**85.88**
		$\mu_{ds}IWKLR$	**85.48**	**87.48**	87.04	85.19
	Balanced	$\mu_{ds}KLR$	**92.08**	**94.14**	**93.29**	**93.60**
		$\mu_{ds}IWKLR$	90.91	92.93	90.78	92.05
Subject 3	Unbalanced	$\mu_{ds}KLR$	**77.79**	**79.86**	84.79	84.88
		$\mu_{ds}IWKLR$	75.99	79.48	**86.27**	**85.57**
	Balanced	$\mu_{ds}KLR$	86.77	**90.28**	**92.70**	**94.11**
		$\mu_{ds}IWKLR$	**86.83**	**90.28**	92.20	93.60
Overall	Unbalanced	$\mu_{ds}KLR$	78.73	81.11	**83.38**	83.14
		$\mu_{ds}IWKLR$	**78.77**	**81.46**	83.28	**83.61**
	Balanced	$\mu_{ds}KLR$	**89.34**	**92.02**	**93.30**	**93.69**
		$\mu_{ds}IWKLR$	88.34	91.40	91.75	92.56

Table 6.9: Sucess rate for Test 1 - Accuracy for IWKLR compared with KLR. Where μ_{ss} is the mean accuracy value for training and test performed in the same sessions and μ_{ss} is the mean value when train and test are performed in different days.

			Nepochs = 2	Nepochs = 3	Nepochs = 4	Nepochs = 5
Subject 1	Unbalanced	$\mu_{ds}KLR$	**85.19**	**92.78**	96.30	**99.07**
		$\mu_{ds}IWKLR$	84.44	92.22	**97.04**	**99.07**
	Balanced	$\mu_{ds}KLR$	**89.63**	**94.44**	**97.41**	**99.07**
		$\mu_{ds}IWKLR$	85.93	92.78	97.04	98.61
Subject 2	Unbalanced	$\mu_{ds}KLR$	96.48	98.89	**100.00**	**100.00**
		$\mu_{ds}IWKLR$	**96.85**	**99.17**	**100.00**	**100.00**
	Balanced	$\mu_{ds}KLR$	**95.56**	**99.72**	**100.00**	**100.00**
		$\mu_{ds}IWKLR$	95.00	99.44	**100.00**	**100.00**
Subject 3	Unbalanced	$\mu_{ds}KLR$	**80.93**	**84.72**	92.22	**97.22**
		$\mu_{ds}IWKLR$	73.52	83.61	**92.96**	96.30
	Balanced	$\mu_{ds}KLR$	**82.04**	**84.17**	**90.74**	**93.06**
		$\mu_{ds}IWKLR$	80.00	**84.17**	**90.74**	**93.06**
Overall	Unbalanced	$\mu_{ds}KLR$	**87.53**	**92.13**	96.17	**98.77**
		$\mu_{ds}IWKLR$	84.94	91.39	**96.67**	98.46
	Balanced	$\mu_{ds}KLR$	**89.07**	**92.78**	**96.05**	**97.38**
		$\mu_{ds}IWKLR$	86.98	92.13	95.93	97.22

Table 6.10: Sucess rate for Test 2 - Accuracy for IWKLR compared with KLR. Where μ_{ss} is the mean accuracy value for training and test performed in the same sessions and μ_{ss} is the mean value when train and test are performed in different days.

As we can see the application of the IWKLR can cause the classifier to decrease its performance, and when the accuracy improves it is not significant. In the overall experiment we can only see an improvement in the performance for 4 epochs in the unbalanced KLR and this improvement is not significant (less than 1 percentual points).

There can be several causes for this behaviour, since there is only a small number of features in the new trial that can cause the importance weight estimation to be poor. Each new trial is extremely unbalanced, that is, there are a lot of non-target events compared to the number of target events, and when the instances are weighted, more importance will be given to the non-target events. The IWKLR performance is strongly dependent on the optimal tune of the parameters σ and δ and this may s\omehow be affecting the results.

Chapter 7

Conclusions and further work

After analyzing the accuracy results obtained using the KLR and the FLD classifier it is possible to conclude that there is no real advantage in introducing a non-linear classification method. The linear classifier has a performance similar to the non-linear classifier and it doesn't requires the tuning of any parameters. The non-linear classification system, requiring the tuning of the two parameters, is more time consuming and computationally heavier.

In the analysis of the non-stationarity of the EEG signal, we can conclude that, in fact, the effect of this non-stationarity decreases the performance of the classification system if the number of epochs used is small. In this case the system becomes unreliable if the calibration is not performed at the beginning of each session. However, for a large number of epochs, the classification problem of the P300 speller can still be solved and the performance of the P300 speller remains acceptable, corroborating the conclusions in [17].

In the online experiment we confirmed that the user can use the system for several sessions performing only one calibration, as long as a large number of epochs is being used. Since the P300 speller should allow us to write on the screen with a high bit/rate this solution is not as suitable as it might appear since a large number of epochs results in a small bit/rate.

The CSM technique has proven to be effective in the general classification problem, being able to improve the classification accuracy by about 2 percentual points. However, for the specific case of the P300 speller, the implementation of this adaptation technique can cause a decay in accuracy.

The results obtained with the IWKLR classifier are also disappointing, since most of the times the application of this adaptation techniques cause the accuracy to decrease.

Futher work should be developed since, in the P300 speller, the bit/rate should be kept as high as possible. This is done by decreasing the number of epochs used in the classification process. It has been shown that for a small number of features, there is a decay in the performance of the system when the training and test are performed in different sessions. In order to increase the performance of the classifier, the system can initially be trained using 5 epochs, and new samples can be collected and used to update the classifier with certain confidence. This will adapt the system to the new distribution of the test features. After collecting several samples, the number of epochs can probabily be decreased. This solution is only possible if the classifier has a high accuracy to start with, otherwise wrong labels will be assigned to the new samples.

The detection of error potentials, that are elipted when the subject perceives an erroneous response, can be used to recognize misinterpreted commands in BCI systems and they can be used to update the classifier with more confidence and to increase the bit/rate.

Bibliography

[1] U. Hoffmann, J. Vesin, T. Ebrahimi, and K. Diserens, "An efficient p300-based braincomputer interface for disabled subjects," *Journal of Neuroscience Methods*, 2008.

[2] L. Farwell and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event related brain potentials.," *Electroencephalography Clinical Neurophysiology*, vol. 70, no. 6, pp. 510–523, 1988.

[3] G. P. Pires, *Biosignal Classification for Human Interface with Devices and Surrounding Environment.* PhD thesis, University of Coimbra Faculty of Science and Technology Department of Electrical and Computer Engineering, 2011.

[4] J. J. Vidal, "Toward direct brain-computer communication," *Annual Review of Biophysics and Bioengineering*, vol. 2, pp. 157–180, 1973.

[5] J. Polich, P. C. Ellerson, and J. Cohen., "P300, stimulus intensity, modality, and probability.," *Int. J. Psychophysiology*, vol. 23, no. 1-2, pp. 55 – 62, 1996.

[6] G. Pires, U. Nunes, and M. Castelo-Branco, "Comparison of a row-column speller vs a novel lateral single-character speller: assessment of bci for severe motor disabled patients," *Clinical Neurophysiology (In press)*, 2011.

[7] M. Yamada, M. Sugiyama, and T. Matsui, "Semi-supervised speaker identification under covariate shift," *Signal Processing*, vol. 90, no. 8, pp. 2353–2361, 2010.

[8] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bunau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Annals of the Institute of Statistical Mathematics*, vol. 60, no. 3, pp. 699–746, 2008.

[9] G. Pfurtscheller, N. Christa, A. Scholgl, and K. Lugger., "Separability of eeg signals recorded during right and left motor imagery using adaptive autoregressive parameters.," *IEEE Trans. Rehab. Eng.*, vol. 6, no. 3, pp. 316–324, 1998.

[10] C. Duncan-Johnson and E. Donchin, "On quantifying surprise: The variation of event-related potentials with subjective probability," *Psychophysiology*, vol. 14, pp. 456–467, 1977.

[11] 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*, pp. 299–305, 2006.

[12] S. Khalighi, T. Sousa, and U. Nunes, "Adaptive automatic sleep stage classification under covariate shift," *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2259–2262, 2012.

[13] L. F., C. M., L. A., L. F., and A. B., "A review of classification algorithms for eeg-based brain-computer interfaces," *J Neural Eng*, vol. 4, no. 2, pp. 538–548, 2007.

[14] A. Satti, C. Guan, G. Prasad, and D. Coyle, "A covariate shift minimisation method to alleviate non stationarity effects for an adaptive brain and computer interface," *International Conference on Pattern Recognition*, 2010.

[15] C. Vidaurre, M. Kawanabe, P. von Bunau, B. Blankertz, and K. R. Müller, "Toward unsupervised adaptation of lda for brain-computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 3, pp. 587–597, 2011.

[16] Y. Liand, H. Kambara, Y. Koike, and M. Sygiyama, "Application of covariate shift adaptation techniques in brain computer interfaces," *IEEE Transactions on biomedical engineering*, vol. 57, no. 6, pp. 1318–1324, 2010.

[17] S. EW and D. E., "A p300-based brain-computer interface: Initial tests by als patients," *Clinical Neurophysiology (In press)*, vol. 117, pp. 538–548, 2006.

[18] E. Dauce, T. Proix, and L. Ralaivola, "Fast online adaptivity with policy gradient: example of the bci "p300"-speller," in *ESANN*, 2013.

[19] Y. Li, C. Guan, H. Li, and Z. Chin, "A self-training semi-supervised svm algorithm and its application in an eeg-based brain computer interface speller system," *Pattern Recognition Letters*, vol. 29, no. 9, pp. 1285–1294, 2008.

[20] S. Lu, C. Guan, and H. Zhang, "Unsupervised brain computer interface based on intersubject information and online adaptation," *IEEE Transactions on neural systems and rehabilitation engineering*, vol. 17, no. 135-145, pp. 1285–1294, 2009.

[21] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis.* 2004.

[22] C.-H. Li, H.-H. Ho, Y.-L. Liu, C.-T. Lin, B.-C. Kuo, and J.-S. Tuar, "An automatic method for selecting the parameter of the normalized kernel function to support vector machines," *Journal of Information Science and Engineering*, vol. 28, pp. 1–15, 2012.

[23] J. Wolpaw, N. Birbaumer, W. Heetderks, D. McFarland, P. Peckham, G. Schalk, E. Donchin, L. Quatrano, C. Robinson, and T. Vaughan., "Brain-computer interface technology: a review of the first international meeting," *IEEE Trans. Rehab. Eng.*, vol. 8, no. 2, pp. 164 – 173, 2000.