POLITECNICO DI MILANO

Master of Science in Biomedical Engineering

Department of Electronics, Information and Bioengineering

# Validation of a CNN architecture for Error Potential classification: a novel, ARX-based approach for data augmentation

Supervisor: Prof. Luca Mainardi

Co-supervisor: Dott. Andrea Farabbi

Master Dissertation of:

Vanessa Aloia

921288

Academic Year 2019-2020

# Acknowledgments

*To my family*

# Abstract

In the last decades, Brain Computer Interfaces (BCIs) appear to be promising for clinical applications in the neurorehabilitation field. BCIs decode the electrical activity of the brain and converts it into commands to exert a control on assistive devices such as neural prostheses, wheelchairs or speech synthesizer, in a way that subjects with severe motor disabilities or with speech impairments are able to accomplish the desired action just by thinking of performing it. The use of BCIs in real applications is nowadays still limited: the process to decode the user's intention is error-prone and demands a high level of attention and fatigue to the user.

The Error Potential (ErrP) is an event-related potential detectable in EEG recordings when the user perceives that an error has been committed, either by himself or by an external device during an interaction with it. The activity is localized in the medio-frontal areas of the brain, in particular in the Anterior Cingulate Cortex, and its realization is characterized by a stereotypical shape with presentation of peaks at specific latencies.

The ErrP can provide additional information during the execution of a task with a BCI device: it can be used ad as corrective signal, discarding those BCI's selected outputs that elicit in the user an ErrP or it can be used in reinforcement learning for an intelligent agent which takes low-level decisions. The introduction of an error-correction system in a BCI system can improve

its performances and makes the decision process faster but an efficient ErrP classifier is needed in order to introduce an effective enhancement in the performances.

Convolutional Neural Networks (CNNs) are a class of deep neural networks widely used in many fields, especially in image recognition applications. They have the main advantages to be able to detect spatial-independent features from raw data and have a reduced number of parameters to be trained. This type of architecture has already shown good results in EEG data classification, allowing to reduce the preprocessing and feature extraction processes, usually challenging phases because some biases may be added to the data.

However, when applied to error potential detection, the training of these architectures have to face the problem that event-related potential datasets are usually characterized by a scarcity of instances including the event: the imbalance affects the classification performances and makes the detection of the event difficult: a data balancing method is needed to overcome the bias that may arise.

Eventually, CNNs are characterized by a high number of hyperparameters, which are parameters not learnt during the training session and defined a-priori. A search to identify the best hyperparameters values can improve the network performances.

**The Project**

The aim of this project is to optimize the performances of a CNN already present in literature, EEGNet, for Error Potential classification.

The work is mainly focused on finding the best method to balance the data, since imbalance causes poor detection of the instances including the ErrP. A novel ARX-based data augmentation method is proposed: ARX models are fitted on data and used to generate new signals. Those new signals, being different from the original ones but with the same characteristics that define the ErrP realization, are added to the dataset in order to have an equal representation of ErrP and non-ErrP instances. Different techniques have been identified to generate the new signals, thus different types of ARX-based methods have been implemented. These methods are validated by comparing their performances with traditional balancing methods present in literature: oversampling, undersampling and class weights.

Another focusing area of the project is the hyperparameters optimization: a search strategy to find the best configuration of hyperparameters is implemented and the performances with the new configuration are evaluated.

**Methods**

The dataset used is constituted by EEG recordings from six subjects performing an ErrP-specific experiment.

Data are preprocessed by spatial filtering with a Common Average Reference (CAR) approach, band-pass filtering between 1 and 10 Hz and extracting the epochs (either including the ErrP or not) in the interval $[150,\ 650]\ ms$ after the presentation of the feedback.

Data are randomized and the network is trained with a stratified 5-folds cross-validation strategy, balancing the training set at each iteration with the chosen technique.

The ARX-based data augmentation methods are implemented as follows: an ARX model is generated for each raw ErrP epochs and then used to generate new data through the distortion of the exogenous input which drives the the ARX model. The distortions evaluated are change of amplitude, white noise addition and warping of the signal and they are tested both separately, by creating a dataset including new data obtained with a single technique, or combined, by including new data obtained with different types of transformations.

Network performances are tested both with an inter-subjects analysis, by including all subjects data, and with an intra-subject analysis, by using single subjects datasets separately.

Classification results are evaluated in terms of F1-score, balanced accuracy and Utility gain function. The latter metric quantifies the gain potentially achievable in a BCI device by introducing an ErrP classifier.

The hyperparameters search is performed with a grid search approach identifying four cycles of optimization: hyperparameters with similar characteristics are grouped in cycles and tested together. At each iteration the configuration that results in the highest performances in terms of F1-score for the validation set is chosen. The procedure is repeated until convergence or, in case, after an elevated number of repetitions.

**Results**

In the inter-subjects analysis the best performances are obtained by the ARX methods resulting for the test set in an F1-score of $\sim 78$ % for the ARX amplitude and ARX amplitude+ noise techniques and in a balanced accuracy of $\sim 87$ % for the ARX amplitude + warping. The ARX methods lead to a smaller gap between training and test scores and to a reduction of the number of False Positives respect to the traditional methods.

In the intra-subject analysis a distinction can be done in the results obtained between two subjects groups: Subjects 1, 2 and 5 achieve high scores while Subject 3, 4 and 6 low scores. The best performances are obtained for Subject 2 resulting for the test set in a F1-score of 87.9 % for the ARX mix and in a balanced accuracy of 92.72 % for the ARX amplitude+warping technique. For the second group of subjects the performances are lower and the traditional methods, especially class weights, result in the highest score.

The hyperparameter search leads to an improvement of 2.1 % in terms of F1-score and of 3.7 % in terms of balanced accuracy in the test set. The final hyperparameters configuration differs from the initial one mainly in terms of a differentiation in the pooling, dropout and activation layers between the two main blocks of the EEGNet architecture and in terms of an increment of the number of convolution layers' filters.

**Conclusions**

ARX methods achieve a better capability of generalization, lower number of False Positives and higher performances respect to the traditional methods. This was not verified for the data that result to be quite anomalous respect to the others: for Subjects 3, 4 and 6 data traditional methods worked better. The Utility gain functions show that a high gain could be obtained by adding an ErrP-based correction system in a BCI device. Moreover, the comparison with literature shows that EEGNet combined with the ARX-based data augmentation technique outperforms the more similar studies taken in account. Within these methods, the ARX warping technique is an exception, showing low performances.

Eventually, the hyperparameters search shows an improvement in the performances but needs optimization: as it was implemented, it is very time-consuming and a specific search for the ARX-based methods is required.

# Sommario

Negli ultimi decenni, la ricerca nel campo della neuro-riabilitazione ha portato a considerare le Brain Computer Interfaces (BCI) come un potenziale approccio innovativo. Un dispositivo BCI ha l'obiettivo di decodificare l'azione che un utente vuole eseguire, mediante l'acquizione e l'elaborazione della sua attivitá cerebrale, e generare un segnale di comando per controllare dispositivi come protesi neurali, sedie a rotelle o sintetizzatori vocali. In questo modo, soggetti con gravi disabilitá motorie o con problemi di linguaggio sono in grado di eseguire l'azione desiderata solo focalizzandosi sull'intenzione di compierla. Attualmente l'utilizzo delle BCI é ancora limitato: il processo di decodifica é soggetto a frequenti errori e richiede un elevato livello di attenzione e fatica per il soggetto.

Il potenziale di errore (ErrP) é un potenziale evocato rilevabile nelle registrazioni EEG quando l'utente avverte che é stato commesso un errore, sia da sé stesso sia da un dispositivo con cui sta interagendo. L'attivitá é localizzata nelle aree medio-frontali del cervello, in particolare nella corteccia cingolata anteriore, e la sua realizzazione é caratterizzata da una forma d'onda con picchi a latenze specifiche.

Il potenziale di errore puó fornire informazioni aggiuntive durante l'esecuzione di un task con un dispositivo BCI: puó essere usato come segnale correttivo, escludendo quei segnali di comando per cui un potenziale di errore viene ril-

evato, oppure puó essere usato per reinforcement learning. L'introduzione di un classificatore di ErrP in grado di identificare gli errori in una BCI puó migliorare le sue prestazioni e rendere il processo decisionale piú veloce. Tuttavia, é necessario che il classificatore sia altamente performante per garantire un effettivo miglioramento nelle prestazioni.
Le reti neurali convoluzionali (CNN) sono una classe di reti neurali ampiamente utilizzate in molti campi, specialmente in applicazioni per il riconoscimento di immagini. I principali vantaggi apportati da questa architettura sono la capacitá di rilevare features spazio-indipendenti nei dati non processati e il numero ridotto di parametri da addestrare. Le CNN hanno mostrato buone prestazioni nella classificazione dei dati EEG, permettendo di ridurre le fasi di processing e di estrazione delle features, solitamente critici in quanto potenziali bias possono essere introdotti.
La presenza di epoche contenti il potenziale evocato nei dataset é solitamente limitata: lo sbilanciamento tra epoche contenenti l'evento e il non-evento ne rende difficile l'individuazione. A tal proposito, un metodo di bilanciamento dei dati é necessario per migliorare la detezione degli eventi.
Le CNN sono caratterizzate da un alto numero di iperparametri, i quali sono parametri non appresi durante la sessione di training ma definiti a priori. Una ricerca per l'identificazione della migliore configurazione di iperparametri puó migliorare le prestazioni della rete.

**Il Progetto**

Lo scopo di questo progetto é stato quello di ottimizzare le prestazioni di una CNN giá presente in letteratura, EEGNet, per la classificazione del potenziale di errore.
Il lavoro si é concentrato principalmente sulla ricerca del metodo migliore per bilanciare i dati, dal momento che lo sbilanciamento dei dati causa una scarsa

detezione delle epoche contententi l'ErrP. In particolare, un nuovo metodo di data augmentation basato su modelli ARX viene proposto: un modello ARX viene identificato per ogni instanza del dataset e in seguito utilizzato per la generazione di nuovi segnali. I nuovi segnali, diversi da quelli originali ma con le stesse caratteristiche che definiscono il potenziale di errore, vengono aggiunti al dataset in modo da avere uno stesso numero di epoche contenenti l'ErrP e non. Sono state individuate diverse tecniche per generare i nuovi segnali e, di conseguenza, sono state implementate diverse varianti del metodo di data augmentation. Questi metodi sono stati validati confrontando le prestazioni con metodi di bilanciamento tradizionali presenti in letteratura: oversampling, undersampling e class weights.
Un'altra area di interesse del progetto riguarda l'ottimizzazione degli iperparametri. É stata implementata una strategia di ricerca per trovare la migliore configurazione di iperparametri e le prestazioni della rete con la nuova configurazione sono state valutate e confrontate con quella originale.

**Metodi**

Il dataset utilizzato é costituito da registrazioni EEG di sei soggetti nel corso di un esperimento specifico per la generazione di ErrP.
La fase di preprocessing dei dati consiste in un filtraggio spaziale CAR e in un filtraggio passa banda tra 1 e 10 Hz. Le epoche sono poi ottenute estraendo il segnale nell'intervallo [150, 650] *ms* dopo la presentazione del feedback.
I dati sono stati randomizzati e la rete é stata addestrata con un approccio stratified 5-folds cross-validation, bilanciando ad ogni iterazione il set di training con la tecnica selezionata.
I metodi di bilanciamento basati su ARX sono stati implementati seguendo la seguente procedura: un modello ARX viene generato per ogni epoca contenente ErrP e ogni modello viene poi utilizzato per generare nuovi dati at-

traverso la distorsione dell'input esogeno dato in ingresso al modello ARX. Le distorsioni che sono state valutate sono la modifica dell'ampiezza, l'aggiunta di rumore bianco e deformazione, mediante dilatazione o contrazione, del segnale. Le tecniche sono state testate sia separatamente, creando un dataset con dati ottenuti mediante una singola tecnica, sia combinate, includendo dati ottenuti con diversi tipi di distorsione.
Le prestazioni della rete sono state testate con un'analisi inter-soggetto, includendo tutti i dati, e con un'analisi intra-soggetto, utilizzando separatamente i dataset dei singoli soggetti. I risultati della classificazione sono stati riportati in termini di F1-score, accuratezza bilanciata e Utilility gain function. Quest'ultima metrica quantifica il guadagno che puó essere apportato in un dispositivo BCI mediante l'introduzione di un classificatore di ErrP.
La ricerca degli iperparametri é stata eseguita con un approccio grid search identificando quattro cicli di ottimizzazione: gli iperparametri con caratteristiche simili sono stati raggruppati in cicli e testati insieme. Ad ogni iterazione é stata scelta la configurazione che risultava nelle piú alte prestazioni in termini di F1-score per il set di validazione. I criteri scelti per terminare la procedura sono il raggiungimento della convergenza o di un elevato numero di ripetizione dei cicli.

**Risultati**

Nell'analisi inter-soggetto le migliori prestazioni sono state ottenute dai metodi ARX. I risultati ottenuti per il set di test sono, in termini di F1-score, uguali a $\sim$ 78 % per le tecniche ARX amplitude e ARX amplitude+ noise e, in termini di accuratezza bilanciata, a $\sim$ 87 % per l'ARX amplitude+ warping. Per i metodi ARX si é riscontrata una minore differenza nelle prestazioni tra training e test e una riduzione del numero di Falsi Positivi rispetto ai metodi tradizionali.

Nell'analisi intra-soggetto si nota una distinzione nei risultati ottenuti tra due gruppi di soggetti: i soggetti 1, 2 e 5 hanno ottenuto buoni risultati a differenza dei soggetti 3, 4 e 6. Le migliori prestazioni sono state ottenute con i dati del soggetto 2 risultando in termini di F1-score uguali all' 87.9 % per il metodo ARX mix e in un'accuratezza bilanciata del 92.72 % per la tecnica ARX amplitude+warping. Per il secondo gruppo di soggetti le prestazioni sono risultate inferiori e i metodi tradizionali, specialmente la tecnica class weights, ha portato ai risultati migliori.
La ricerca degli iperparametri ha apportato un miglioramento nelle prestazioni sul set di test del 2,1 % in termini di F1-score e del 3,7 % in termini di accuratezza bilanciata. La configurazione finale degli iperparametri differisce da quella iniziale principalmente per due caratteristiche: si é verificata una differenziazione nei layer di pooling, dropout e attivazione tra i due blocchi principali dell'architettura EEGNet e il numero di filtri dei layer convoluzionali ha subito un incremento.

**Conclusioni**

I metodi ARX hanno mostrato una migliore capacitá di generalizzazione, generando un numero inferiore di Falsi Positivi e ottenendo migliori prestazioni rispetto ai metodi tradizionali. Tuttavia, é da notare che ció non si é verificato per quei dati che sono risultati anomali rispetto agli altri: per i dati dei soggetti 3, 4 e 6 i metodi tradizionali hanno performato meglio.
Un potenziale miglioramento nelle prestazioni di un dispositivo BCI con l'introduzione di un classificatore ErrP bilanciato con i metodi ARX é stato evidenziato dalle Utility gain functions. Inoltre, dal confronto fatto con alcuni studi presenti in letteratura, é risultato che l'uso di EEGNet combinato al metodo di data augmentation basato su ARX ha piú alte prestazioni rispetto a tutti gli studi con approcchi simili considerati.

Tra i metodi proposti, la tecnica ARX warping risulta essere un'eccezione: ha ottenuto, infatti, prestazioni inferiori rispetto agli altri.
Infine, la ricerca degli iperparametri ha confermato che un miglioramento delle prestazioni puó essere raggiunto mediante l'ottimizzazione di essi. Tuttavia, necessita di miglioramenti: la ricerca, cosí come é stata implementata, richiede un elevato tempo computazionale e una ricerca specifica per i metodi basati su ARX deve essere implementata.

# Contents

# List of Figures

# List of Tables

# Acronyms

BCI Brain Computer Interface

ErrP Error Potential

EEG electroencephalogram

CNS Central Nervous System

ACC Anterior Cingulate Cortex

fMRI functional Magnetic Resonance Imaging

EPSP excitatory post-synaptic potential

IPSP inhibitory post-synaptic potential

ERP event-related potential

DL Deep Learning

CNN Convolutional Neural Network

ANN Artificial Neural Network

GAN Genrative Adversarial Network

ROC Receiver Operating Characteristic

AUC Area under the ROC Curve

SSVEP steady state visually evoked potentials

MI Motor Imagery

BNCI Brain/Neural Computer Interaction

| | |
|---|---|
| CAR | Common Average Reference |
| ReLU | rectified linear unit |
| ICA | Independent Component Analysis |
| MRCP | ovement-related cortical potential |
| SMR | sensory motor rhythms |
| DCNN | Deep Convolutional Neural Network |
| ARX | AutoRegressive eXogenous |
| EOG | electrooculography |
| FIR | Finite Impulse Response |
| ReLU | Rectified Linear unit |
| ELU | Exponential Linear Unit |
| SGD | Stochastic Gradient Descent |
| RMSprop | Root Mean Square propagation |
| FC | Fully Connected Layer |
| TP | True Positives |
| TN | True Negatives |
| FP | False Positives |
| FN | False Negatives |
| AR | Autoregressive |
| RMS | Root Mean Squared |
| AIC | Akaike Information Criterion |

# Chapter 1

# Introduction

Chronic conditions such as amyothrophic lateral sclerosis, spinal cord injuries or stroke lead to a total miscommunication between the nervous system and the voluntary muscles, resulting in severe motor disabilities [1].
Historically, for patients suffering from those neurological pathologies, the prognosis for recovery has been poor [2]: the consequences are the loss of autonomy and difficulty in accomplishing their daily activities.
Research in the last decades had evaluated several ways to both improve their quality of life and reduce the cost of intensive care: one of them is the design of Brain Computer Interfaces (BCIs) [3].
BCI are devices that create a non-muscular channel between the brain and an external device, allowing to bypass the lesion and to restore the functionality [2]. Its main objective is to interpret the user's intention through the elaboration of the acquired brain signal and to generate a correct output to control tools like computers, wheelchairs, speech synthesizers, assistive appliances and neural prostheses, which in turn will carry out the desired action [3].
Even though those interfaces are very promising and have a variety of appli-

cations, their use is nowadays limited: BCIs are error-prone and the decision-making system requires a lot of repetitions to decode the subject's will. Therefore, a high level of fatigue and attention are required for the user and if the BCI's output is incorrect the device is controlled in a wrong way being potentially dangerous.

By adding an additional block that validates or rejects the BCI's output it would be possible to improve the performance and the reliability of the overall system. This block should not require additional fatigue for the patient and should be easily implementable.

The Error Potential (ErrP) is a specific brain activity, spontaneously generated when humans make or perceive an error. It can be observed in the electroencephalogram (EEG) with a certain latency after the awareness of the error as a signal with a stereotypical shape.

An Error Potential classification system can potentially overcome current BCIs limitations: by identifying the presence (or not) of an ErrP in the acquired signal it is possible to discern if the decision taken by the BCI system agrees with the user's expectations, therefore with his intention.

## 1.1 The ErrP: neurophysiology and applications

### 1.1.1 Neurophysiology

The Central Nervous system (CNS) is constituted by the brain and the spinal cord. The brain is the main processing unit: its function is to elaborate, integrate and respond to the sensory information.

The outer part of the brain is the cerebral cortex. Anatomically, it can be divided in two hemispheres which can be subdivided in four lobes: frontal, parietal, occipital and temporal lobe (figure 1.1).

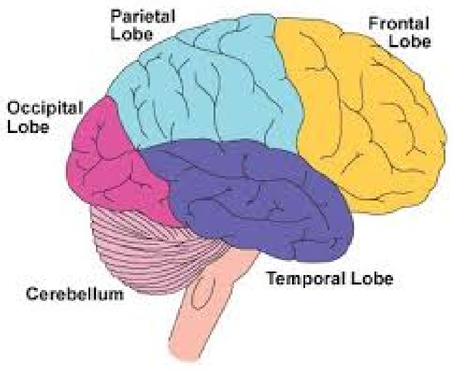


*Figure 1.1: Cerebral cortex's lobes*

For each lobe, it is possible to identify specific functions.
The frontal lobe is involved in the control of skeletal muscular activity, the parietal lobe is involved in the tactile, pressure, vibrations, pain and temperature perception, the occipital lobe in the visual stimuli processing and the temporal one mainly in the auditory and olfactory stimuli processing [4].
Neurons, the unit cells of the nervous system, are arranged in the cerebral cortex in structures called columns and layers. A very complex network of communication between neurons is obtained through specialized structures called synapses.
By studying neurons architecture in the cerebral cortex, in 1909 Korbinian Brodmann identified 47 different brain areas, each one involved in a different function. Nowadays 52 Brodmann areas have been identified [5]. By studying the connectivity between brain regions and between neurons, it is possible to identify specific networks devoted to specific cognitive processes.
The Error Potential is generated by a high-level, generic, error processing system [6]. *Error-processing* concerns the fact that the system is involved in

detecting an error; with *high-level* it is intended that the system is associated with the executive processes mediated by the frontal areas of the brain. Eventually, *generic* refers to the high flexibility of the system, capable of identifying errors in a wide variety of contexts.
As it will be further discussed, the Error Potential is generated in a variety of different tasks: it can be elicited by a response to a negative feedback, by the awareness of a self-committed error or just by looking another device making an error. Furthermore, it can be elicited by a negative feedback presented in different sensory areas such auditory, visual or somatosensory modalities [7]. This suggests that the system generating the response is independent from the stimulus origin and it is flexible to very different tasks.
Several studies [6], [8] suggested that the region in which the ErrP is generated is the frontomedian wall, in particular the anterior cingulate cortex (ACC) (figure 1.2). By analyzing functional magnetic resonance images (fMRI), the ACC was identified as the activation brain area during erroneous trials [9].

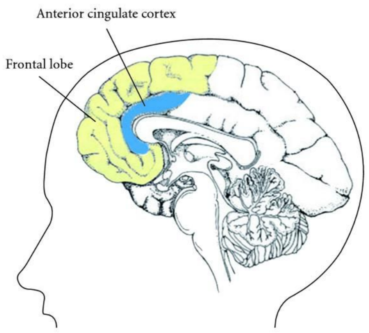


*Figure 1.2: Anterior cingulate cortex*

Frontal areas of the brain, including the prefrontal cortex, the anterior cingulate system and the basal ganglia contribute to the executive control. This system is involved in planning actions, decision making and plays a role when a subject faces a new or difficult task [10]. Executive control also concerns response monitoring, therefore ensuring that the consequences of an action are consistent with the intent. In [6] it has been suggested that this mechanism is consistent with the error-processing system, confirming the role of the anterior cingulate cortex in the generation of the Error Potential.

### 1.1.2 The Electroencephalography

The Electroencephalography is a technique used to acquire the electrical activity of the brain. The main source of the signal is the synchronized activity of a population of neurons, whose electrical activity sums up and therefore can be recorded from the scalp as a significant and localized signal.
Neurons are excitable cells, they transmit information through ionic currents that generate a measurable electric field. It is possible to distinguish two types of neuronal activation: the *action potential*, generated when the membrane potential reaches a certain threshold, characterized by a rapid depolarization (1 or 2 ms) and a return to the rest condition (repolarization) and the *post-synaptic potential*. The latter one is caused by a synaptic activation, which is mediated by neurotransmitters, and consists in a ionic current that in case of an excitatory potential (EPSP) is carried by positive ions and in case of inhibitory potential (IPSP) is carried by negative ions. This activation is slower (10 ms), allowing the temporal summation of post-synaptic currents from many neurons.
For this reason, it constitutes the main contribution to the EEG signal (figure 1.3).

Neurons' ionic current can be modelled as a current dipole oriented along the dendrite: to measure a net current it is required that neurons are aligned with each other to sum up the single contributions. The pyramidal neurons have such organization along the cortex and are characterized by an open-field potential: they are the main source for the EEG [11].

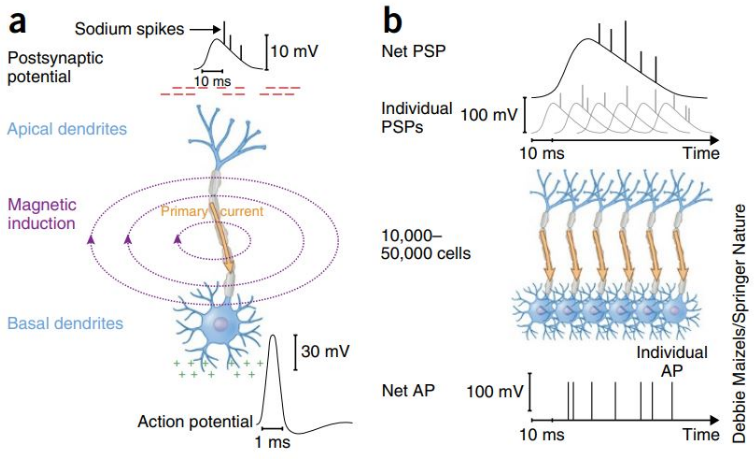


*Figure 1.3: a) Post-synaptic potential and action potential of a single pyramidal neuron. b) Neurons with synchronized activity. Temporal summation of the single post-synaptic potentials leads to a measurable electric signal. [12].*

EEG signal is measured at the level of the scalp: it consists in a set of measurements of the voltage difference between pairs of electrodes. The electrodes can be placed directly on the skin (*dry electrodes*) or with a conductive gel (*gel-based*). They can be placed singularly on the scalp or fitted to an elastic cap, such that their positions are fixed. Standard montages for the electrodes location have been defined: the 10-20 International system [13] establishes that the electrodes are placed at distances in percentage of the 10-20 range between Nasion and Inion. Depending on the position, electrodes are marked as: *Fp* for Frontal Pole, *C* for Central, *P* for parietal, *O* for Occipital and $T$ For temporal. Electrodes placed along the central line are marked with $z$, while electrodes on the left with an odd number and on the right with an even number (figure 1.4 ).

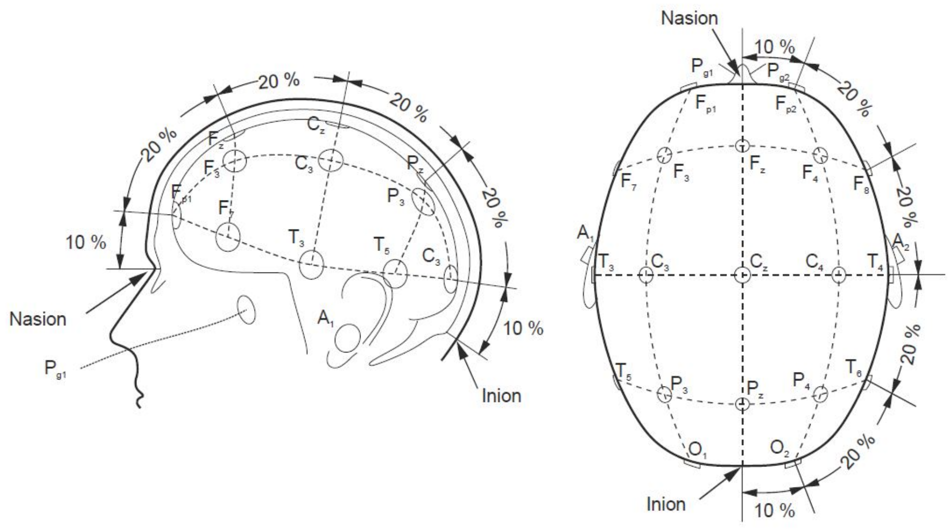


*Figure 1.4: The 10-20 International system for EEG electrodes placement [3].*

Standard numbers of electrodes employed are 32, 64, 128, 256: by increasing the number of electrodes, the spatial resolution increases.
The main problem with the EEG signal is its poor spatial resolution: the signal recorded on the scalp results from the propagation of the neurons activity trough conductive tissues and it is strongly attenuated by the skull due to its high resistivity. For this reason it is difficult both to localize the source of the activity, given the EEG measurements at the scalp (*inverse problem*), both to determine the potential at each electrode given the brain source (*forward problem*) [14].
Moreover, EEG signal is quite noisy, also because of disturbances introduced by the measurement system.

However, it is characterized by a very good temporal resolution, making it optimal to follow brain's activity during time and capture its dynamic.
As a matter of fact, EEG is employed in a variety of different tasks and experimental paradigms, being a very flexible technique, non-invasive, recorded with non-bulky, silent and non-expensive instrumentation.

### 1.1.3 Event-related potential

EEG signal resembles the activity of a conglomeration of different neural sources, making it difficult to isolate individual neuro-cognitive processes. From a statistical point of view, it is a stochastic non-stationary signal.
It is possible to elicit a certain brain response by providing a specific external stimulus: the voltage change, specifically related to the stimulus and time-locked to it, is called event-related potential (ERP) [15].
The segment of the EEG in which the realization is detected is called epoch. This signal can be considered transient and deterministic since it has a finite duration in time and it has a stereotypical shape.
The ERP is localized in the brain's regions where the stimulus is processed, so that its realization is mainly localized in specific electrodes. Usually it has a small magnitude respect to the background EEG, so that it is necessary to employ signal processing to extract it.
As mentioned before, the EEG is a stochastic signal while the ERP is deterministic, so by averaging many epochs in which the ERP is expected it is possible to separate the ERP from the noise, i.e. the background EEG. The waveform obtained is called *grand average* ERP.
Visual, auditory, somatosensory stimuli with specific patterns and paradigms are provided: the ERP, depending on the type of stimulus, will present specific peaks at specific latencies and will be localized in different brain areas,

therefore detected in different electrodes.
A typical paradigm which is used to extract the ERP is the *oddball paradigm* [16]. The subject performs a task in which some visual stimuli are presented: a sequence of 80% Xs and 20 % Os with a blank inter-stimulus interval. After the session the ERPs elicited by the Xs and Os are extracted, by isolating the epochs after the presentation of the stimuli and they are averaged. The ERP waveforms are obtained for the X and for the O for each electrode: they are characterized by different and specific features. (figure 1.5).

### 1.1.4 The Error Potential

The Error Potential (ErrP) is an event-related potential occurring in the EEG when a subject recognizes that an error has been committed, either by himself or in an action that he is observing [17].
It was first studied by two independent groups [18], [19] in 1990s and described as a negative peak (Ne) and later a positive (Pe) deflection in the event-related potential of incorrect choice reactions.
Nowadays the concept of ErrP has been extended to a variety of different tasks when the resulting action, executed by the subject himself or by an external system, is in disagree with the subject's expected output.
As mentioned before, the Error Potential is an event-related potential, therefore it is a deterministic part of the EEG signal that occurs with a certain latency after the presentation of the stimulus, therefore the error.
By averaging together several epochs of EEG signals associated with an erroneous response, it is possible to identify the characteristic shape of the ErrP.
It has been shown that, depending on the task, four types of ErrP can be distinguished.

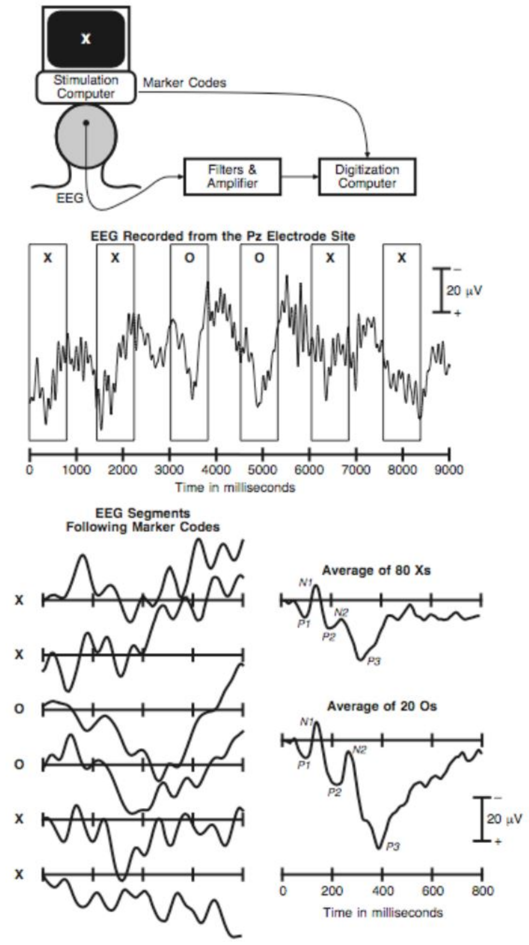


*Figure 1.5: Example of an ERP experiment. The user looks at a screen in which frequent Xs and infrequent Os are presented. The EEG is recorded, the EEG segments are isolated and marked with 'X' or 'O' depending on the stimulus. Then the ERP is obtained by averaging the single epochs. It's possible to notice the specific peaks for the two averages [16].*

The *response ErrP* is elicited when the subject who is performing the task recognizes his own error. The main component of the signal is a negative potential showing up 80 $ms$ after the incorrect response followed by a larger positive peak showing up between 200 and 500 $ms$ [20].

When the error is made aware by a feedback presented to the subject, the generated response is named *feedback ErrP*. The main component here is a negative deflection occurring 250 $ms$ after the presentation of the feedback indicating an incorrect performance [6].
The *observation ErrP* is elicited while the subject observes an erroneous action of another person or device. In this case the negative deflection shows up 250 $ms$ after the incorrect response of the operator [21].
Finally, the *interaction ErrP* is evoked when an error occurs in the interaction between an user and a machine. It is characterized by a negative peak 250 $ms$ after the error, a positive peak at 320 $ms$ and a negative peak 450 $ms$ after [21].
The former two types of ErrP are elicited by the incorrect action of the subject himself performing the task and recognizing his own error. The latter ones are instead elicited by an error made by another device, being the subject just looking at the result of the task. Those two types are the one of our interest, since they could be elicited during a BCI experiment when the subject's intention is misclassified by the BCI system, so that the resulted action is not the one desired and the ErrP is evoked.
A typical realization of the *interaction ErrP* can be seen in figure 1.6.

### 1.1.5 Application in BCI

"A BCI is a computer-based system that acquires brain signals, analyzes them, and translates them into commands that are relayed to an output device to carry out a desired action." [23]
BCI devices allow patients with severe motor disabilities to interact with their surroundings through the control of an external device.
A typical BCI device is constituted by: an acquisition system, a preprocessing

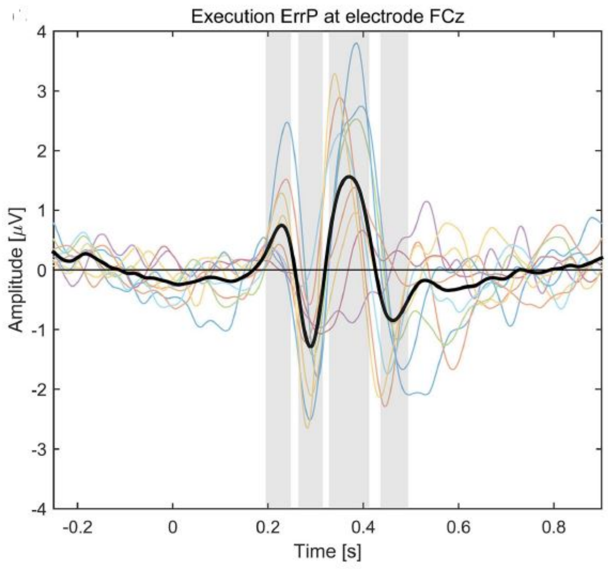


*Figure 1.6: Average of error-correct EEG trials (thick line) for the channel FCz [22].*

system, a feature extraction system, a classification algorithm and an external device. The brain signal is recorded, it is preprocessed to amplify, remove artefacts and digitized it. The preprocessed signal is further analysed to extract the features: structured data needed for the classification algorithm to decode the user's intent. Once the classification algorithm provides its output, the corresponding control signal is sent to the external device to execute the desired action.

Nowadays, the use of BCI in real life is still limited because of some problems: BCIs are frequently prone to errors in the recognition of the subject's intent [24] and the training period necessary to build the classification algorithm is usually long and tedious for the patient [25].

Moreover, in applications such as the P300 speller, the identification of a

character requires many repetitions. The subject has to look at a matrix of characters and directs his gaze to a target character while each row and column, in turn, is randomly flashed. Detecting the P300, i.e. the evoked potential elicited by the flash, in a single trial is very difficult so that the intensification sequence has to be repeated several times to identify a single character [26]. This requires a lot of attention for the subject and the system performs very slow.
A way to increase the speed and the reliability of the BCI classifier is to use the Error Potential as a corrective signal.
As mentioned before, the Error Potential is generated not only when the subject himself commits an error, but also when an erroneous feedback from an external device is presented or when there is an error during the interaction with it. Once the BCI classifier provides its output, the selected command can be sent to the user as a feedback: if no Error Potential is detected the action is executed, otherwise the decision is discarded and the trial repeated (figure 1.7).

This would not require any additional fatigue for the subject since the ErrP is generated implicitly just for error awareness without need of training or asking him to actively generate it [27]. Moreover, it would increase the speed of the decision system because a lower number of repetitions could be carried out.
Another highlighted problem is that the BCI's classifier does not require just an initial training, but the procedure must be repeated since the EEG signal naturally changes over time, both between different sessions and within a single session [28].
Therefore, another use of the Error-potential is in error-driven learning: the parameters of the task's classifier can be updated whenever an Error Po-

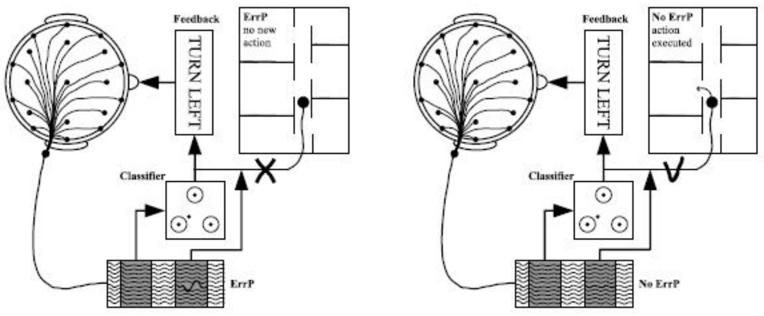


*Figure 1.7: Application of the ErrP detection in a brain-controlled robot. The subject receives a feedback indicating the classifier's decision. If the feedback generates an ErrP (left) the command is discarded, is no ErrP is detected (right) the action is executed [24].*

tential is detected to learn from misclassification [29] or it can be used for reinforcement learning [30]. In this case the user monitors the performance of an an autonomous agent which has learning capabilities and takes low-level decisions. Each time an Error Potential is detected, a negative reward value is given to it in order to prevent in the future the erroneous action.
With those approaches, any intermediate training session could be avoided since the classifier is able to learn and adapt to changes.

## 1.2 Deep learning

Deep learning (DL) is a sub-field of Machine learning, which includes a set of methods that can learn from raw data.
While most Machine learning techniques require the design of a feature extractor to transform the raw data into a suitable feature vector for the classifier, Deep learning classifiers can be fed with raw data and are automatically

able to discover the most appropriate representation needed for detection [31]. This has many positive implications: a domain expertise is not required to structure data and no bias can be introduced by handling data in a non-optimal way. Moreover, the pipeline of the model is simplified letting the "machine" to learn how to learn from data.
A Deep learning architecture takes as inputs the raw data and through multiple layers which perform non-linear transformations results in a higher and more abstract level of representation of the data. During the process irrelevant features for the classification task are discarded while aspects that are important for discrimination are amplified [31].
This approach has been widely used in many fields such as image and speech recognition, natural language understanding showing successful results.
Focusing on the use of Deep Learning methods for EEG, a review of current applications is presented [32]. Within the studies reported, 86% focused on classification of EEG data for sleep staging, seizure detection and prediction and BCI, 9% on the improvement of processing tools such as learning features from EEG or handling artefacts and 5% on generating new data for data augmentation purposes.

### 1.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of deep neural networks characterized by the employment of the mathematical operation called convolution [31]: the layers of the networks are constituted by filters, whose parameters are learned during the training process, that are applied to the input data through convolution.
They are one of most prominent Deep Learning architectures employed typ-

ically for image detection and classification purposes. The most beneficial aspects of CNNs are the reduction of the number of parameters that the network has to learn and the spatial-independent features [33].
For instance, in a face detection application, it is not important where the face is located in the image, it will detect it regardless its position.
Moreover, the features going into deeper layers become more and more abstract: in a image classification at first edges are detected, then contrasts between regions, shapes and then higher level features. This allows to capture in data some aspects that are spatial-independent, because the network is not designed to look at spatial properties but instead to look at common features in the image.
In [32] the Authors reported that CNN is the most often used DL architecture for EEG data classification showing that it performs well also with time series data.
The main building blocks of a CNN are:

- convolution layer
- pooling layer
- fully connected layer

In a *convolution layer* each input data, presented as a multi-dimensional vector, is convoluted with a filter.
Using a convolution kernel respect to a classic node of an artificial neural network (ANN) implicates much less parameters [33]. Suppose to have an input data of dimension 12 x 12 x 3: to connect this input with a neuron of a standard ANN we would need 12 x 12 x 3 weighted connections, therefore 12 x 12 x 3 parameters to train.
With a CNN, instead of having a full connection, a filter which covers a

sub-region of the matrix is convoluted with it and shifted along the matrix. Suppose to choice a filter of dimension 3x3: the number of parameters to be trained is 3x3x3.

The main idea is that the parameters are shared between different regions of the matrix, since the same filter shifts all along the matrix and there is sparsity of connections: in each layer each output value depends only on a small number of inputs.

A *pooling layer* operates a downsampling of the input matrix: it divides the data matrix in sub-matrices, substituting them with a single value that summarizes the magnitude of the local region. This is done to reduce the dimension of the data compacting it. An important aspect of *pooling layers* is that they do not include any parameter to be trained.

Finally, a *fully connected layer* is a layer in which each node is connected with all the inputs, as in a regular ANN.

The main advantage to have less trainable parameters is that the training session requires less time and deep learning approaches are usually very time consuming.

Further description of the components of a CNN will be presented in Chapter 2.

### 1.2.2 Data imbalance

An imbalance dataset is a dataset in which there's no equal occurrence of examples belonging to different classes. The most prevalent class is called the *majority class*, while the rarest class is called the *minority class* [34].

Usually, especially in ERP datasets, the rare events are the ones of interest but the scarcity of examples makes very complex and challenging their identification. In fact, because of the imbalance, a bias in favor of the majority

class is usually introduced [35].
The main strategies to deal with an imbalance dataset are [34]:

- Resampling techniques
- Cost-sensitive learning

With *resampling techniques* it is intended a modification of the input data which is done before training the model. The main idea is to rebalance the data either by creating new instances of the minority class in case of *oversampling* or by removing instances of the majority class in case of *undersampling.*
In case of oversampling it is possible to randomly replicate instances of the minority class or to apply *data augmentation* techniques to generate new data.
The main problem with the oversampling approach is that, since the training set is constituted by repeated instances, it may lose in generalization and cause overfitting. Instead, with an undersampling approach, depending on the degree of deleted samples, the dataset could lose valuable information [35].
In *cost-sensitive learning* some costs are assigned to the instances in a way that a higher cost for the misclassification of minority class with respect to the majority class samples is provided. This method is less popular [34] since cost matrices have to be identified and usually the misclassification cost is unknown from the data. Moreover, if it is not done by experts, some biases can be introduced in the learning model.
In [36] it is reported that both undersampling and oversampling approaches have been applied to classical EEG classification tasks, such as epileptic seizures or transitional sleep stages identification.
In literature many methods of data augmentation have been proposed: some

distortions may be operated on the data for instance by adding noise, usually Gaussian noise [37] or new data can be generated by generative algorithms, such as the use of Generative Adversarial Network (GAN) [38].
Class imbalance affects also performance evaluation: the metric to quantify the classifier performances has to be chosen wisely. Accuracy results to be affected by the imbalance, being biased toward the majority class: while the majority class instances are correctly classified, a lack of identification of the minority class samples is verified [34]. Other metrics, such as *balanced accuracy, F1-score* and *ROC AUC*, are robust to the imbalance present in the dataset [36].

### 1.2.3 Hyperparameters optimization

One of the main features of a CNN is that it is characterized by a high number of hyperparameters [39]. Hyperparameters are a-priori parameters of the learning model which are not learned during the training session, in contrast to the others parameters of a network. They include both parameters that define the geometry and size of the layers in a network architecture and variables that define the learning process.
Hyperparameters values affect the performance of the model, so they have to be adapted to the specific learning problem: a hyperparameter optimization process can be defined.
The two classical approaches for optimization [40] are:

- Grid search
- Random search

Hyperparameters to be optimized have to be chosen and for all of them a range of possible values have to be identified.

With the grid search technique all the combinations between values of the hyperparameters are evaluated while with the random search a random sample of possible configurations is chosen and tested. The learning process is then executed with the different configuration sets and the one resulting with the highest performance is chosen.
Grid search requires more time but it guarantees to reach the optimal set, while with random search this is not ensured. The main advantage to use random search is the reduction of optimization time, in particular when important hyperparameters are tested against less influential ones and testing all the possible values of the latter does not affect too much the performances (figure 1.8).

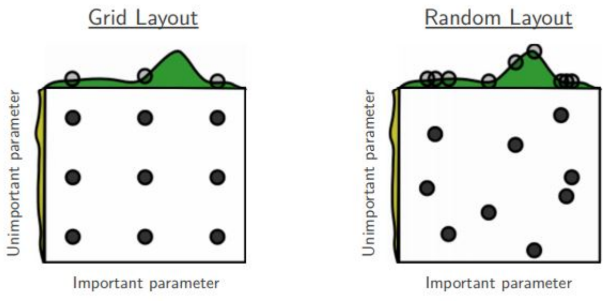


*Figure 1.8: Grid layout (left) requires the definition of a set of equally distributed values for the hyperparameters tested. With a random layout (right) random combinations of hyperparameters are chosen. It is better to choose the latter approach when important parameters are tested against less important ones to reach an optimal time cost [40].*

It has to be noticed that nowadays new methods for hyperparameters optimization have been designed, such as Bayesian optimization [41].

## 1.3 State of the art

### 1.3.1 Current Error Potential applications

In 2014 a review of the applications of Error potential in BCI was presented [27]. The Authors identified two main uses of the ErrP in BCI: ErrP can be used as a *corrective signal* to reject those task-classification decisions who elicit an error awareness by the user or in a *error-driven learning*.
In the latter case the ErrP signal was used in adaptive BCIs to update the parameters of the task classifier in order to track the non-stationarities of the EEG signal without requiring any intermediate training sessions [29] or it was applied for reinforcement learning purposes to provide a feedback to an autonomous agent [30].
Regarding the last years of publications, the use of ErrP as a *corrective signal* was employed in a variety of applications.
In [42] the ErrP is used to verify the incorrect classification of steady state visually evoked potentials (SSVEP): a flashing light generates an electrical activity in the occipital region at the same frequency of the light stimulus. The main objective to use ErrP in this case is to improve the performance of such interfaces in real world applications, in which the lighting conditions are not optimal and many external disturbances would decrease the user's attention.
ErrP in the motor imagery field was considered in [43] and [44]. "Motor imagery is a method of identifying a user's intention through the EEG characteristic that appears when the user imagines a certain movement" [44]. The classification accuracy of such signals is low so that they are usually combined with other signals like SSVEP or ERP: the Error Potential could be used as a corrective signal to prevent the execution of wrong commands.

In [45] the application is the P300 speller: a flashed matrix of letters is shown and the target letter the user wants to pronounce is detected by identifying the P300 wave. The contribution of this study is the introduction of a double ErrP detector: if an ErrP is detected after the feedback, the symbol shown is deleted and replaced by another one, which is the second most possible. If also the second feedback elicits an ErrP, the first selected symbol is re-selected.
The application of ErrP in *reinforcement learning* is present in publications as [46]. A human-robot interface is developed, in which the robot is controlled by human actions. The human executes different gestures and the robot chooses the actions depending on the gesture type. The robot learns the meaning of the gesture by receiving a human feedback: the feedback is obtained through the ErrP detection from the recorded EEG. If an ErrP is recognized, a negative feedback is transferred to the robot's learning algorithm to "discourage" that action next time, while if no ErrP is detected a positive feedback is sent. The main reason to use the ErrP as feedback is that generating an explicit feedback for each action for the user is very demanding and tiresome while the ErrP is generated implicitly.
Regarding the classification methods used in those studies, traditional approaches were chosen. Random forest was found as the most performing for [42], in [44] Linear discriminant analysis is used while in [46] a Support vector machine is applied for ErrP classification.
None of them used a Deep Learning approach. In the next section some articles notably using Deep learning techniques, in particular Convolutional Neural Network architectures, are presented.

### 1.3.2 Related works

CNN architectures applied for the detection of the Error Potential are presented in this section in order to provide a practical framework of the current state of the art for this application.

**a) deep ConvNets** [47]

This CNN architecture is fed by EEG data which are preprocessed by spatial filtering using common average reference (CAR) and by isolating the band of the signal between 1 Hz and 10 Hz. The activity related with ErrP is verified for each electrode to select the most activated ones: 2 electrodes (FCz and Cz) out of 64 are held.

In order to overcome the data imbalance due to the fact that ErrP epochs are much less than non-ErrP ones, the Authors opt for an oversampling technique: data belonging to the minority class are randomly chosen and replicated.

The CNN architecture implemented is composed by five layers each one containing a convolution layer, a pooling layer and a ReLU activation function. To deal with a possible overfitting problem that may arise from the oversampling, batch normalization and dropout layers are added.

Performances of the network are tested both with a single-subject approach and with a cross-subjects one. The results in terms of average accuracy on the test set are for the single-subject analysis equal to 80.15 % ± 4.14 % while equal to 79.79 % ± 2.96 % in the cross-subjects one.

It is reported that the network benefits from the addition of batch normalization and dropout layers with an accuracy improvement of 4 % .

**b) ConvNet-based pipeline** [48]

In this case the CNN is not fed by raw data or by data minimally preprocessed. Instead, specific preprocessing steps are defined to extract ErrP

features, including: artefact removal, whitening and cropping. In the artefact removal step the activity of bad channels is substituted by inferring a new one through interpolation between close electrodes. Eye blinking, movement artefacts and trial-specific artefacts are identified through ICA inspection and removed. The feature set is normalized through a whitening process and then it is randomly cropped for each training iteration. It is reported that this latter step prevents the identification of false local minima.
No methods to deal with data imbalance are reported.
The CNN designed is constituted by two layers, each one including a convolution and a pooling layer.
The network is tested with a cross-subjects approach comparing the performance by using just FCz and Cz channels data and by using all channels.
The CNN performs better with all the channels, yielding to an average accuracy on the test set of 77.45 % ± 2.35 % for error trials and of 84.10 % ± 0.30 % for correct trials.
The designed network is compared with other literature approaches: a Gaussian classifier, a linear Support Vector Machine, a Dynamic Bayesian Network and a majority-vote strategy. The proposed CNN outperforms all those methods.
**c) EEGNet** [49]
The CNN presented in this section is the one which will be analysed in this thesis.
The Authors' objective is to build a CNN able to classify EEG signals from different BCI paradigms: they test EEGNet on P300 visual-evoked potential, ErrP, movement-related cortical potential (MRCP) and sensory motor rhythms (SMR). The network is constituted by four main blocks including convolution, pooling, batch normalization and dropout layers.

Some particular convolution layers are used: Separable and Depthwise convolution layers. They are usually used for image classification algorithms and have less parameters to be trained.
Data are preprocessed by downsampling at 128 Hz and band-pass filtering between 1 and 40 Hz. All the channels data are used.
To overcome the problem of data imbalance a class weights approach is chosen: weights are assigned to data depending on their belonging class, in a way that minority class epochs weight more during the training session.
The performance of EEGNet is compared both with other DL architectures present in literature and with other traditional approaches in a within and cross-subjects analysis.
In the within-subject classification EEGNet outperforms the other approaches for MRCP and ErrP while is not significantly better for P300 and for SMR. In the cross-subject analysis there is not a net improvement respect to the literature.
The performance for the ErrP dataset is $\sim$ 0.81 in terms of AUC (Area under the ROC curve) in the within-subject analysis, obtained as the average between all subjects and all folds of the 4-folds cross-validation, while in the cross-subjects analysis the AUC is $\sim$ 0.74, averaged across 30 folds.

**d) DCNN with GAN** [38]

A modified version of EEGNet is proposed, where some additional convolution layers are added to make the network deeper.
The Authors overcome the data imbalance problem by designing a Generative Adversarial Network (GAN): a Deep Learning data augmentation approach that generates a new set of data from the existing ones.
Data are downsampled to 128 Hz and band-pass filtered between 1 Hz and 10 Hz. Bad epochs are rejected and muscular and eye artefact are removed

through ICA. Eventually, data are re-referenced by computing the average reference.

The performance of the network is evaluated both in single and cross-subjects sessions by comparing different degrees of addition of the new generated data. Overall the best result is achieved with a double stage of augmentation (2 part of the dataset out of 3 are new signals) resulting in an average accuracy of 87.94 % on the test set.

Also in this case the network is compared with other literature methods, both DL and traditional approaches, outperforming all of them.

## 1.4 Aim of this work

BCIs have a large room for improvement: the use of the ErrP as a feedback respect to the decision taken by the classifier is very promising. It would speed up the classifier decision process and improve the device performances in terms of user's intent detection.

For this purpose it is necessary to develop an ErrP classifier that has high performances in order to achieve a gain by its addition to a BCI device [50].

Nowadays, Deep learning techniques have provided a high-level of performances in image, audio and speech recognition tasks. By being fed with a large amount of raw data, a DL network is able to identify the most useful features to classify samples.

Among different architectures, Convolutional Neural Networks are the most prominent ones, being adaptable in a variety of field. Such technique has already been tested on time-series data and in particular on EEG classification tasks, achieving good results.

The aim of this work is to model a Convolutional Neural Network for the classification of the Error Potential. In particular, starting from an architecture already present in literature [49], EEGNet, this thesis is focused on its optimization.

When an ERP detection is considered, it is worth remembering that the number of instances including the event, the ErrP in this case, is much less than the remaining ones. The first focusing area is to study different ways to address with data imbalance which, as already mentioned, significantly affects the network performances. Different methods to balance data have been compared: oversampling, undersampling and class weights.

In this thesis, a novel method for data augmentation is hypothesized and studied: new data have been generated through an ARX model. Taking the instances with ErrP, an ARX model is fitted on each of them and then used to generate new data through the modification of the exogenous input. This new approach is compared with the classical ones to validate its performances.

The second focusing area is the hyperparameters optimization: the building blocks of EEGNet, the number and size of the filters, the parameters defining the learning process are evaluated. Through an optimization strategy the optimal set of hyperparameters specific for the application and the dataset used is identified.

The work is organized as follows: in Chapter 2 the dataset and the software used are presented. EEGNet is described and the data classification strategy is reported, including the adopted cross-validation approach and the metrics chosen. The methods used for balancing data are described and the newly-introduced ARX data augmentation method is presented. Eventually, the hyperparameter search implemented is described.

In Chapter 3 the ARX generated data are inspected to verify the similarities with the original data. The results of the ErrP classification are reported both in terms of intra and inter-subject analysis, comparing the performances of the different balancing techniques. The hyperparameters search results are then presented: performances before and after the optimization are reported. Finally, in Chapter 4 a discussion regarding the obtained results is provided and possible future developments are identified.

# Chapter 2

# Material and Methods

## 2.1 Dataset

The dataset used in this thesis for the evaluation and optimization of the proposed EEGNet is published by [30] and it is available as an open-access BCI dataset of *BNCI Horizon 2020: Monitoring error-related potentials* [51]. It is constituted by EEG recordings during an ErrP-specific experiment performed by 6 subjects (mean age 27.83 ± 2.23) in two recording sessions.

The experiment paradigm consists of reaching a target location, i.e. a coloured square, through a moving cursor. The working area is constituted by 20 possible horizontal positions where the cursor and the target square may be located. At each time step the cursor moves by a step toward the location of the target. Once the target is reached, the cursor remains in place and the target appears on the screen in a new location. A graphical representation of the experiment is presented in figure 2.1.

Subjects are asked to monitor the movement of the cursor, knowing that its objective is to reach the target, but without having any control over the cursor itself, i.e. they are observing the machine working. In order to elicit the

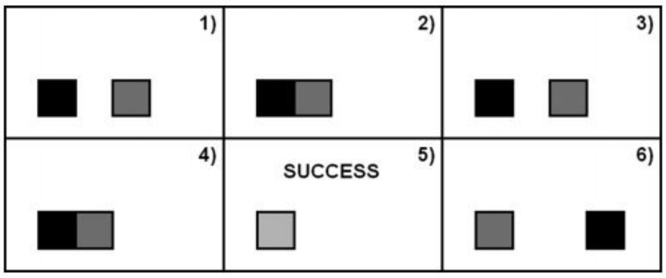


*Figure 2.1: Illustration of the experiment protocol: 1) target is on the left of the cursor, 2) cursor moves towards the target 3) cursor moves in the opposite direction 5) cursor reaches the target and 6) the target is moved in another location. Figure taken from [21].*

ErrP, at each time step it may happen that the cursor moves in the wrong direction with 20% of probability. The cursor is moved with a rate of 0.5 Hz (every 2s) and each movement represents one trial of the experiment. A session is constituted by 10 blocks, each one of approximately 50 trials. Subject performed two sessions with a gap of several weeks.

EEG signal is recorded with 64 electrodes at a sampling frequency of 512 $Hz$ using a *BioSemi ActiveTwo* system. Electrodes are placed according to the 10-20 International System.

Globally, the dataset is constituted by 6437 epochs of which 1322 include the ErrP. A summary of the epochs for each of the 6 subjects is presented in table 2.1.

| **Subject** | **ErrP epochs** | **non-ErrP epochs** | **ErrP occurrence (%)** |
|---|---|---|---|
| Subject 1 | 235 | 809 | 22.51 % |
| Subject 2 | 242 | 838 | 22.41 % |
| Subject 3 | 188 | 848 | 18.15 & |
| Subject 4 | 211 | 841 | 20.06 % |
| Subject 5 | 241 | 882 | 21.46 % |
| Subject 6 | 205 | 897 | 18.60 % |
| **Total** | 1322 | 5115 | 20.54 % |

*Table 2.1: Number of epochs distinguished in ErrP and non-ErrP is presented for each subject of the dataset. The percentage of trials including the ErrP is reported in the last column.*

## 2.2 Software

Code is implemented with Python 3.8 in Jupyter Notebook environment.
Preprocessing is done on *MNE* [52], an open-source Python package for visualization and analysis of neurophysiological data.
The CNN implementation is performed with *Tensorflow* 2.3 [53] using *Keras* API [54]. Tensorflow is a machine learning open-source library that supports large-scale training: it is used in a wide variety of applications, being mainly focused on training deep neural networks for large datasets [53].
In order to optimize time requirements, the model is trained on a NVIDIA GeForce GT 730 GPU.
*Scikit-learn* [55] package is used to implement the hyperparameters search and the cross-validation strategy.

## 2.3 Data preprocessing

A preprocessing pipeline has been defined to extract from the EEG the epochs, i.e. the segments of the signal localized just after the movement of the cursor in which the ErrP may be present.

Preprocessing allows to "clean" data, by removing noise and isolate the components of the signal inherent with the brain activity under study, and to downsize, by extracting from each signal a smaller but representative vector. A smaller size of the data allows to reduce the time needed to train the model, but conversely, a procedure that requires many preprocessing steps is time consuming. As discussed before, Deep Learning architectures are able to learn from raw data and they extract themselves the features: for this reason, it was chosen to simplify the preprocessing, not including any procedures like artefact, eye blinking or eyes movements potential removal.

A pipeline of the preprocessing is presented in figure 2.2.

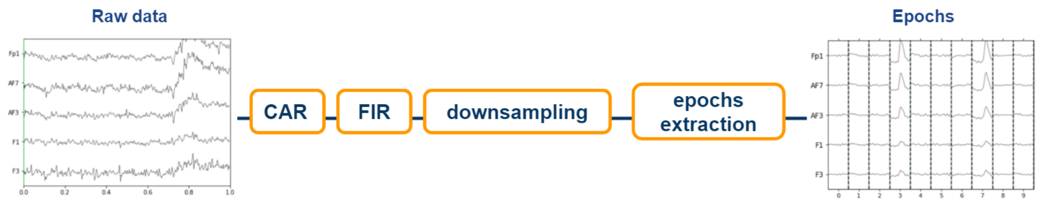


*Figure 2.2: Data preprocessing pipeline. From raw data common average is subtracted (CAR), then a FIR filter is applied to extract the band between 1 and 10 Hz, data are downsampled to 64 Hz and finally epochs are extracted.*

Raw EEG data are spatially filtered with a Common Average Reference (CAR) approach: the average potential over all the 64 electrodes is computed and subtracted from each electrode signal. By spatial filtering the noise present globally over the electrodes can be reduced, making the detec-

tion of low amplitude signals in the EEG easier. Data are then band-pass filtered between 1 and 10 Hz with a FIR filter: the Authors in [21] suggested to consider this band since ErrP is a relatively slow cortical potential. Data are downsampled to 64 Hz and the epochs are extracted between 150 ms and 650 ms after the presentation of the feedback. This range is chosen according to the expected latency of the ErrP signal [24].
Filtering allows to extract the specific range of frequencies expected in EEG when cerebral activity related to error-processing is present, while downsampling is executed mainly for size reduction.
The parameters of the preprocessing steps have been defined according to literature [21], [24], [30], [28].
Most of the works in literature [28], [21], [30] use data of FCz and Cz channels for the classification: in this study we opted to analyze all the EEG channels in order to exploit all the information contained in the recorded data.
Each epoch results in a matrix of dimension (64, 37) which is then given as input to the network.

## 2.4 EEGNet

EEGNet is a Convolutional Neural Network architecture presented in [49].
The code is available on GitHub at [56].
It has been chosen for this study because it is a very flexible and generalizable tool, as demonstrated by the good performances on different types of BCI paradigms, including ErrP classification [49].
Moreover, the Authors reported that it can be trained with very limited data. This is crucial in EEG applications since recording a big dataset requires many sessions, so high cost, time and fatigue for the users.

### 2.4.1 Architecture

A graphical representation of EEGNet architecture and a summary of the filters' size, number of parameters and outputs are presented in figure 2.3 and table 2.2, respectively.

Each building block is now presented.

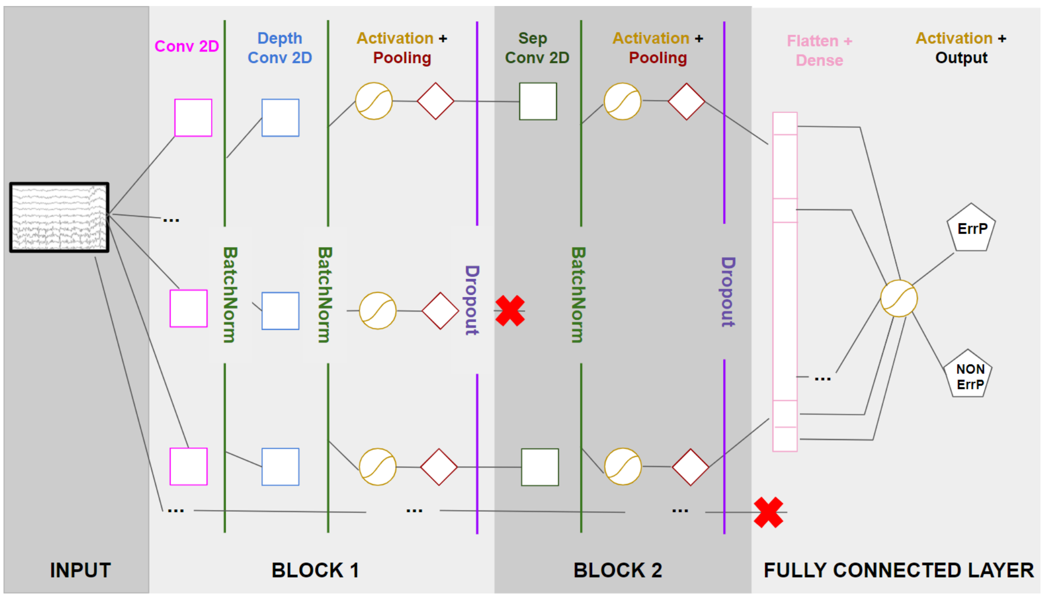


*Figure 2.3: Overall architecture of EEGNet. The network is constituted mainly by three blocks: Block 1 includes a Convolution Layer and a Depthwise Convolution layer, Block 2 a Depthwise Separable Convolution Layer and the Fully Connected layer includes a flatten, a dense layers and the classification activation function.*

| **Block** | **Layer** | **# filters** | **Size** | **#params** | **Output** | **Options** |
|---|---|---|---|---|---|---|
| **1** | Input | | | | (C, T, 1) | |
| | Conv 2D | F1 | (1, 32) | 32* F1 | (C, T, F1) | padding= "same", stride= 1 |
| | BatchNorm | | | 2* F1 | (C, T, F1) | |
| | Depht Conv 2D | D* F1 | (C, 1) | C* F1* D | (1, T, F1* D) | padding= "valid", stride= 1 |
| | BatchNorm | | | 2* F1* D | (1, T, F1* D) | |
| | Activation | | | | (1, T, F1* D) | ELU |
| | AvgPool | | (1, 2) | | (1, T//2, F1* D) | stride= 1 |
| | Dropout | | | | (1, T//2, F1* D) | p= 0.5 |
| **2** | Sep Conv2D | F2 | (1, 8) | F2*(D* F1)+ 8* D* F1 | (1, T//2, F2) | padding ="same", stride= 1 |
| | BatchNorm | | | 2*F2 | (1, T//2, F2) | |
| | Activation | | | | (1, T//2, F2) | ELU |
| | AvgPool | | (1,4) | | (1, T//8, F2) | stride=1 |
| | Dropout | | | | (1, T//8, F2) | p=0.5 |
| **FC** | Flatten | | | | (1* T//8 * F2) | |
| | Dense | T//8* F2 | | | 1 | |
| | Activation | | | | 1 | Sigmoid |

*Table 2.2: EEGNet architecture is presented. It is a modified version of [49]. The three blocks of EEGNet are described (FC stands for fully connected layer) in terms of number of filters, size, parameters and data output size. Values reported are C= number of electrodes, T= time points, F1= 8, D=2, F2= 16.*

## INPUT

Data are presented as a 4D matrix of dimension (*samples, channels, time points, kernels*).

*Samples* is the number of data in input: size depends on the set given in input.

*Channels* is the number of electrodes: 64

*Time Points* is the number of data points for each electrode: it is equal to 37.

*Kernel* dimension for the input is equal to 1. It is the dimension on which the CNN works by concatenating at each layer the results of the operations computed by the filters.

### BLOCK 1

#### Conv 2D

Conv 2D is a convolution layer, characterized by filters that operate through convolution.

A convolution filter slides over the input matrix and for each position the sum of the element-wise multiplication between the filter and the input sub-matrix is computed: this results in a single element of the output matrix (figure 2.4).

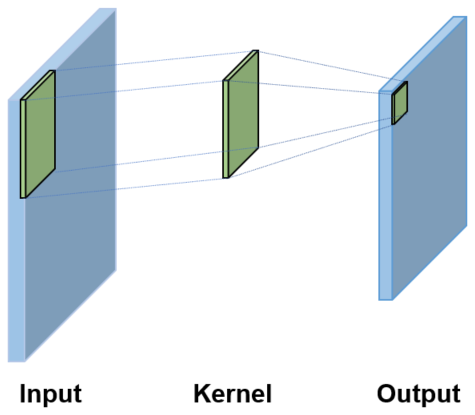


*Figure 2.4: Convolution layer: the input matrix is convoluted with the filter. Each new value of the output matrix is obtained by a convolution between the filter and an equal-size part of the input matrix*

The convolution layer may have some additional features:

- *stride*: number of steps by which the filter is moved each time it slides
- *padding*: number of values added at the borders of the input matrix. This is done to address the fact that edges of the matrix are multiplied fewer times respect to the central values. Two options are available: "same" and "valid". "Same" padding means that the addition to the

borders leads to an output matrix with the same size of the input while "valid" means that no padding is computed.

This layer is constituted by F1 convolution filters of size (1, 32, 1). F1 is the number of filters and it is set at 8, "same" padding and stride equal to 1 are applied according to [49].
The first dimension of the filter is set at 1 in order to not perform any convolution respect to the *channels* dimension, while the second dimension is set at half the sampling rate (64 Hz) in order to extract frequency information from 2 Hz above [49]. The third dimension must always agree with the input's one to compute the convolution: for this, reason it is equal to 1 .
This layer, by processing just the *time points* dimension, operates a temporal filtering of data for each electrode outputting F1 feature maps containing the EEG signal at different band-pass frequencies [49].

**BatchNorm**

Batch normalization is a layer that helps stabilizing the training session by setting the distribution of the inputs at zero mean and unit variance. This allows to remedy for the so-called *internal covariant shift* [57], resulting in an acceleration of the training session.
It is called batch normalization because normalization is made not over the entire input dataset, but on the single batches, i.e. the subsets of the dataset that one at time feed the network.

**Depth Conv 2D**

This layer is constituted by depthwise convolution filters.
This kind of convolution differs from the standard one by computing the convolution separately at each *kernel* level. As mentioned before, in standard convolution the filter's *kernel* dimension must agree with the input's one because the operation is computed over the volume. A convolution between

an input of dimension (4, 4, 3) with a kernel of dimension (2, 2, 3) results in an output of dimension (3, 3). Conversely, with a depthwise convolution layer the filter is applied singularly to each volume layer so that is possible to decouple the convolution between different matrix's *kernels* [58]. The result of a depthwise convolution between an input of size (4, 4, 3) and a filter of size (2, 2, 3) is a (3, 3, 3) matrix (figure 2.5).

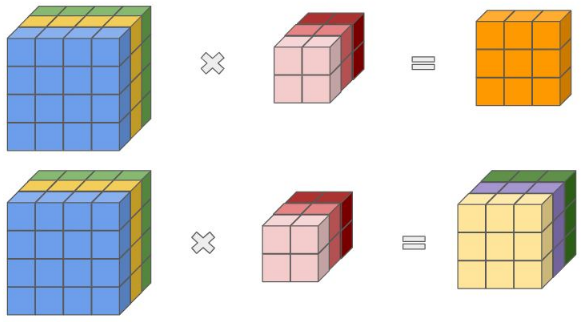


*Figure 2.5: Standard convolution (top) and Depthwise convolution (bottom). In the standard one the convolution is made over the volume resulting in a 2D matrix, while in the depthwise each volume is treated separately and the resulting kernels are concatenated in a 3D matrix.*

By having the same filter's size, therefore the same number of parameters to train, it can be noticed that the output matrix obtained with the depthwise convolution is 3D while the one obtained with a standard convolution filter is 2D. This can be considered as a reduction of parameters: to obtain the same output a standard convolution layer would have required three more filters.

This layer is constituted by D*F1 filters with size (64, 1). D is set at 2, stride is equal to 1 and no padding is applied. The filter size is (64, 1): the

convolution is computed just on *channels* dimension. This results in a spatial filtering computed between the electrodes for each time point and separately for each bands obtained by the Conv 2D layer.
For this reason, each spatial filter can be optimally fitted on the specific band.

**Activation**

Activation functions are either linear or non-linear functions that compute the weighted sum of the node's inputs establishing if each node of the network is activated or not [59].
The most used ones are presented in figure 2.6.

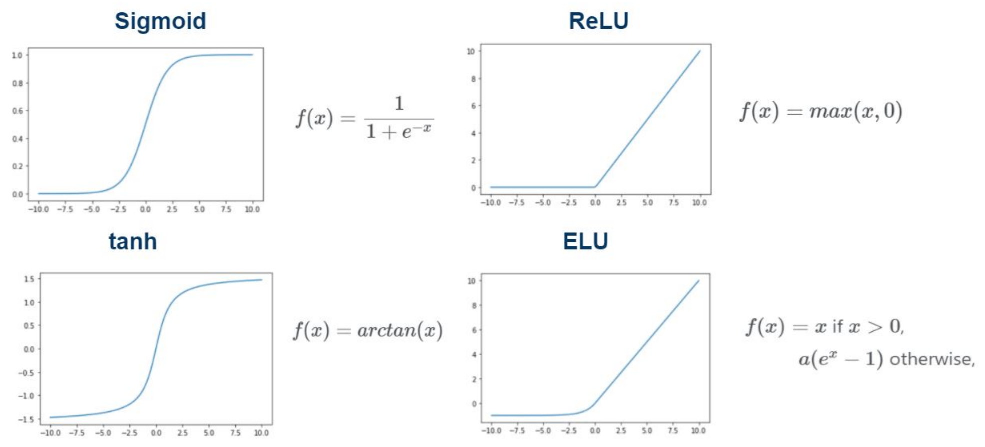


*Figure 2.6: Typical activation functions used in Deep Learning*

*Sigmoid* provides an output between 0 and 1: for this reason it is usually applied in the output layer of the network, where samples have to be classified.
*Tanh* is better than sigmoid in the hidden layers since values lie between -1 and +1: as mentioned before, optimization algorithm performs better if samples distribution is centered around zero.

The *Rectified Linear Unit* (ReLU) is a nearly linear function that forces negative values to zero. It guarantees faster computation since no exponential or divisions operations are computed and results in a higher level of performance and generalization than sigmoid and tanh [59].
Eventually, the *Exponential Linear Unit* (ELU) is usually applied to speed up the training. Conversely to ReLU, it includes negative values so that the output mean is closer to zero.
The activation function used in this layer is the ELU.

**Pooling**

Pooling layers have the main feature to reduce the size of the input by "summarizing" the information with no need of parameters to be trained.
The two main types of pooling layers are *Max pooling* and *Average pooling.* The output matrix is obtained by dividing the input into sub-matrices, according to the size of the pooling filter, and taking the maximum of it in case of Max pooling or the average in case of Average pooling (figure 2.7).

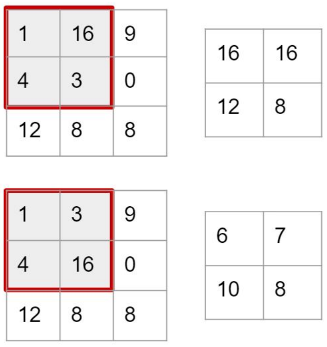


*Figure 2.7: Pooling layer with filter's size equal to (2, 2). Max pooling (top) and Average pooling (bottom) are applied to the same input. The resulting matrices are shown on the right side.*

In this layer, an Average pooling of dimension (1, 2) is applied to downsample data to 32 Hz. Also for pooling layers *stride* value can be chosen: it is set at 1.

**Dropout**

A dropout layer performs the exclusion of random nodes at each training iteration. It is characterized by a hyperparameter called dropout rate: it defines the probability assigned to each node to be deleted.

Dropout layers, by randomly excluding nodes, simplify and add causality to the network. This is useful both for the regularization of the network and for reducing overfitting. As a matter of fact, if random nodes are eliminated, the weights do not rely on single inputs but there's a spread out of the relevance of the nodes: this prevents outputs from becoming strongly correlated between each other, which in turn leads to overfitting [60].

In a standard *dropout* layer, for each input, random elements can be deleted or not, while with *spatial dropout* each input is either all maintained or all deleted [60]. The latter one promotes higher independence between different inputs.

A dropout layer with probability equal to 0.5 is included at the end of Block 1.

**BLOCK 2**

**Sep Conv 2D**

This layer is constituted by depthwise separable convolution filters. This type of filter is a depthwise convolution filter followed by a pointwise one.

The pointwise convolution has dimension (1, 1, *kernel*) so that at first each *kernel* is obtained separately through the depthwise filters and then they are merged optimally through the the 1 x 1 convolution (figure 2.8).

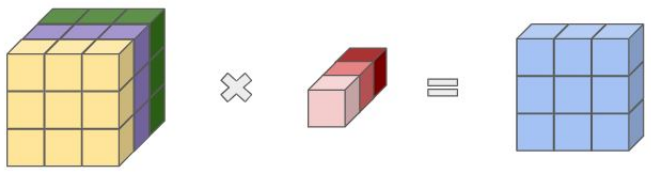


*Figure 2.8: A pointwise convolution filter is a filter of dimension (1,1, volume size) so that it combines the kernels in a 2D matrix*

F2 depthwise separable convolution filters of dimension (1, 8) are included. F2 is set at 16, stride is equal to 1 and "same" padding is applied.

**Activation**

Also in this block ELU function is applied.

**Pooling**

An Average pooling of dimension (1, 4) is applied for dimension reduction. The *stride* is set at 1.

**Dropout**

A dropout layer with dropout rate of 0.5 is placed at the end of Block 2.

**FULLY CONNECTED LAYER**

**Flatten + Dense**

A flatten layer flattens the data in order to create a 1D vector and then the dense layer connects every input with a weight.

This is a traditional ANN node in which each input is connected with each output.

**Activation**

This is the last layer of the network: it classifies data assigning a class, i.e ErrP or non-ErrP.

In this layer, sigmoid activation function is used.

### 2.4.2 Training procedure

Training session consists of finding the parameters of the network that better classify the data.
To do that a loss function and an iterative method must be defined: the latter one will optimize the network's parameters to minimize the loss.
The loss function used is the *binary cross-entropy*: the cross-entropy between true and predicted labels is computed. The smaller is its value, the closer are the probability distributions of the predicted classes to the real ones.
The most common iterative methods are: SGD, RMSprp and Adam.
SGD and RMSprop are characterized by one hyperparameter called *momentum* while Adam has two hyperparameters, two *momenta*.
All of them need an additional hyperparameter: the *learning rate* which determines the decay rate of the algorithm while reaching the minimum [61].
In EEGNet, Adam is used with learning rate equal to 0.001, 1st momentum to 0.9 and 2nd momentum to 0.999. The hyperparameters' values are defined according to [61] in which Adam algorithm is introduced.
The network is trained in 300 *epochs* and the *batch size* is set at 16. The former hyperparameter defines the number of times on which the algorithm goes over the data during the training session, while the latter represents the number of training samples that are used during training in order to make one update of the network parameters [57].

## 2.5 Data classification

Performances of EEGNet are evaluated in both an *intra-subject* and an *inter-subjects* analysis.
In the intra-subject analysis EEGNet is trained separately with each sub-

ject's data while in the inter-subjects analysis the entire dataset is used to train the network.

A pipeline of the strategy applied to classify and evaluate the performances of EEGNet is presented in figure 2.9.

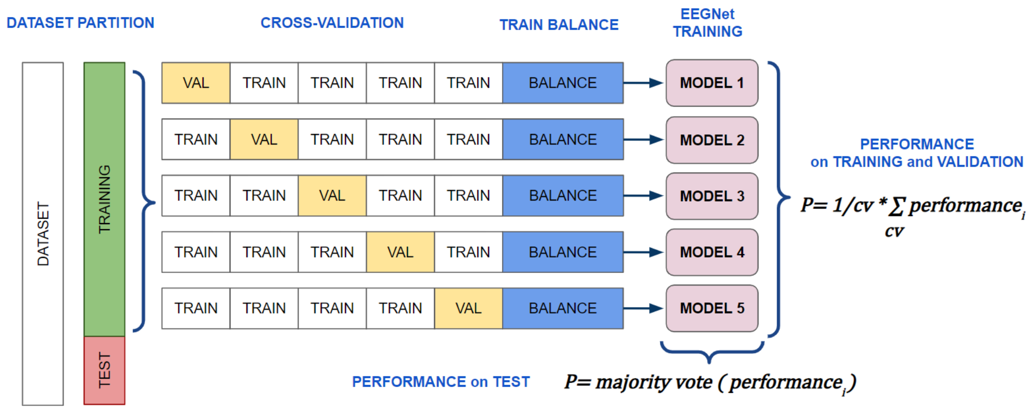

DATASET PARTITION
CROSS-VALIDATION
TRAIN BALANCE
EEGNet TRAINING
DATASET
TRAINING
TEST
VAL TRAIN TRAIN TRAIN TRAIN BALANCE MODEL 1
TRAIN VAL TRAIN TRAIN TRAIN BALANCE MODEL 2
TRAIN TRAIN VAL TRAIN TRAIN BALANCE MODEL 3
TRAIN TRAIN TRAIN VAL TRAIN BALANCE MODEL 4
TRAIN TRAIN TRAIN TRAIN VAL BALANCE MODEL 5
PERFORMANCE on TRAINING and VALIDATION
P= 1/cv * ∑ performance_i cv
PERFORMANCE on TEST
P= majority vote ( performance_i )

*Figure 2.9: Data classification pipeline. Partition in training and test sets is done. 5-folds cross-validation is performed: for each iteration 1 out 5 parts is used as the validation set and the training set is balanced with the chosen data balancing method.*

Each step of the procedure is now described.

### 2.5.1 Dataset partition

*Training set* is the part of the dataset that is used to train the model: the network is fed with those data and learns from them the parameters that better classify data.

*Validation set* is the part of the dataset that is unseen by the network during the training session. This means that the parameters are not fitted on it, therefore it is useful to test the performance of the model. The main objective of the validation set is to do model selection, i.e. estimating the performances

of different models on it to choose the best one [62]. For instance, in the hyperparameters optimization process hyperparameters are chosen according to the ones that result in a higher performance on the validation set.
*Test set*, similarly to the validation set, is the part of the dataset not used for training, but just to evaluate performances. No decisions are taken on it but it is considered at the end to test the ability of the network to predict classes on independent data.
Data are divided in 80 % for training and validation and 20 % for test. The partition between training and validation is done during cross-validation.
As it will be discussed in Chapter 3, there is a high variability between subjects' data: in order to obtain a heterogeneous distribution of data in the three sets, it has been decided to shuffle data before partitioning them into training and test sets.

### 2.5.2 Cross-validation

Cross-validation is a method for estimating the prediction error.
Due to scarcity of data, usually it is not possible to have a large set for validation: by using a small part of the dataset to evaluate the performances, the results may not represent the overall behaviour of the model and the obtained results lose in generalization [62].
With *K-fold cross-validation* the dataset is divided in $K$ parts: $K$ iterations are executed and in each one *K-1* folds are used for training and 1 for validation. Typical values for $K$ are 5 or 10.
It has been decided to use a *Stratified 5-folds cross-validation*: folds are created in a way that the percentage of samples for each class are constant for each fold. Since data are imbalanced, by stratifying it is ensured that the minority class samples are included in all the folds.

Therefore, in each iteration, 80 % of the training set is used for training while 20 % is used for validation.

### 2.5.3 Train balance

The dataset is highly imbalanced: ErrP epochs are much less than non-ErrP ones so that the classification may be biased toward the majority class [35]. For this reason, a balancing technique is applied before feeding the network. The different approaches will be discussed in *Balancing methods* section.
For each iteration of the cross-validation process the imbalance on the extracted training set is evaluated, by counting the number ErrP and non-ErrP samples, and addressed according to the balancing strategy chosen. In particular, it may require either the addition or removal of samples from the training set or the assignment of weights to the instances.
Data are shuffled and ready to feed the network.
No balancing is performed on validation and test sets.

### 2.5.4 Performance evaluation

#### 2.5.4.1 Metrics

The confusion matrix for the classification results is reported in table 2.3.

| | **ErrP** | **non-ErrP** |
|---|---|---|
| **1** | TP | FP |
| **0** | FN | TN |

*Table 2.3: Confusion matrix for the classification. The first row reports the effective presence of the Error Potential (ErrP or non-ErrP) while the first column represents the network's output (1 for positive classification, 0 for negative). Depending on the agreement or not between real and predicted values the following indexes can be defined: TP = True Positives, TN = True Negatives, FP= False Positives, FN= False Negatives,*

The metrics chosen to evaluate performances are *accuracy*, *F1-score* and *balanced accuracy*. It has to be noticed that the latter two are robust to data imbalance [36] while the former is biased: it is reported in this study just to compare it with literature results.
Accuracy is a measure of the samples correctly classified over the total number of samples:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

F1-score is calculated as follows:

$$F1 = 2 \cdot \frac{precision \cdot sensitivity}{precision + sensitivity} \tag{2.2}$$

where $sensitivity = \frac{TP}{TP+FN}$ and $precision = \frac{TP}{TP+FP}$.

Balanced accuracy is the average between the *sensitivity* and *specificity* of the classifier:

$$Balanced\ accuracy = \frac{sensitivity + specificity}{2} \tag{2.3}$$

where *specificity* is defined as: $specificity = \frac{TN}{TN+FP}$.

#### 2.5.4.2 Utility metric

The above metrics quantify the performance of a classifier, but it would be interesting also to evaluate if the improvement of ErrP detection would generate a real benefit applied to a BCI system. To this purpose the Utility metric is considered.

Utility is a metric introduced in [50] for the quantification of the performance of a BCI system. In particular, its definition for a BCI device integrating an error-correction system is as it follows:

$$U = \frac{log_2(N-1)(pr_C + (1-p)r_E + p - 1)}{c} \tag{2.4}$$

where $N$, $p$ and $c$ are parameters of the BCI classifier itself, being the number of possible classifier's outputs, the accuracy of the classifier and the duration of a single trial, respectively. $r_C$ and $r_E$ are, instead, referred to the error-correction system, being the recall for correct trials and the recall for errors:

$$r_C = \frac{TN}{TN+FP} \qquad\qquad r_E = \frac{TP}{TP+FN} \tag{2.5}$$

In particular, what it is evaluated in this thesis is the Utility gain, thus the the gain that can be introduced in the performances of a BCI system by adding an Error Potential-based correction system. It is defined as the ratio of the Utilities, as it follows:

$$g = \frac{pr_C + (1-p)r_E + p - 1}{2p - 1} \tag{2.6}$$

A gain $> 1$ means that an improvement in the performances can be obtained by adding the error-correction system, while a gain $< 1$ highlights that its addition is counterproductive.
Utility gain, as a function of the BCI system's accuracy $p$, is compared between the different balancing method classifiers proposed, to identify which one can provide a higher gain, thus a higher improvement in a BCI system performances.

#### 2.5.4.3 Overall score

With 5-folds cross-validation five different models are obtained by training with different subsets of the training set. It results in a score for each iteration for both the three sets.
Final scores on training and validation sets are calculated by averaging the five single scores, while a *majority vote* approach is used for the test set.
Majority vote strategy is applied to combine classifiers outputs to assign a class to labels. If the classifiers make independent errors, the majority vote outperforms the best classifier [63]. Five predictions are obtained on the test set: majority vote counts the votes for each class for each sample and selects for each one the class with higher occurrence. Metrics are calculated on this final resulting vector.

## 2.6 Balancing methods

### 2.6.1 Data augmentation with ARX

A novel method for data augmentation is introduced in this section.
ARX has already shown good performances in modelling ERPs such as event-related auditory potentials [64] and somatosensory evoked potentials [65].
Here, ARX models are used for data augmentation purposes: new instances including the ErrP are generated by fitting an ARX model on each ErrP epoch of the original dataset and then using the model with modified input to simulate new data.
During the classification process the new data are included in the training set to overcome the imbalance between ErrP and non-ErrP epochs. In the intra-subject analysis, only ErrP pattern (and models) obtained from the specific subject dataset are included, while in the inter-subjects analysis new epochs are randomly picked.

#### 2.6.1.1 ARX modelling

ARX is a a linear parametric approach to model time series. In particular, it is defined as the sum of an autoregressive (AR) process driven by white noise and of the contribution of an exogenous input, a signal with a specific shape. A mathematical notation for the ARX model structure is as follows:

$$y_i(t) = \sum_{j=1}^{p} a_j y_i(t-j) + \sum_{k=d}^{q+d-1} b_k u(t-k) + e_i(t) \tag{2.7}$$

The acquired signal $y_i(t)$ can be modelled as a linear combination between an AR model of order $p$, taking in account the previous samples of the signal multiplied by the coefficients $a_j$ and the contribution of white noise $e_i(t)$, and an exogenous part of order $q$ which includes previous samples delayed by $d$

of the exogenous input $u(t)$ multiplied by coefficients $b_k$.

The autoregressive part of the formula models the background EEG, therefore the brain activity not related with error-processing. It is a non-stationary stochastic signal, already modelled in literature [65] through an AR system. The exogenous input, instead, models the deterministic part of the signal, i.e. the ErrP itself.

Non-preprocessed signals are considered: ErrP epochs extraction is performed by selecting the portion between $[-1.5,\ 2]\ s$, in which 0 s is the instant of presentation of the incorrect feedback (the incorrect movement of the cursor).

An ARX model is identified for each single epoch. The three main steps for generating the models are: *exogenous input computation*, *ARX model identification* and *ARX model validation* (figure 2.10). .

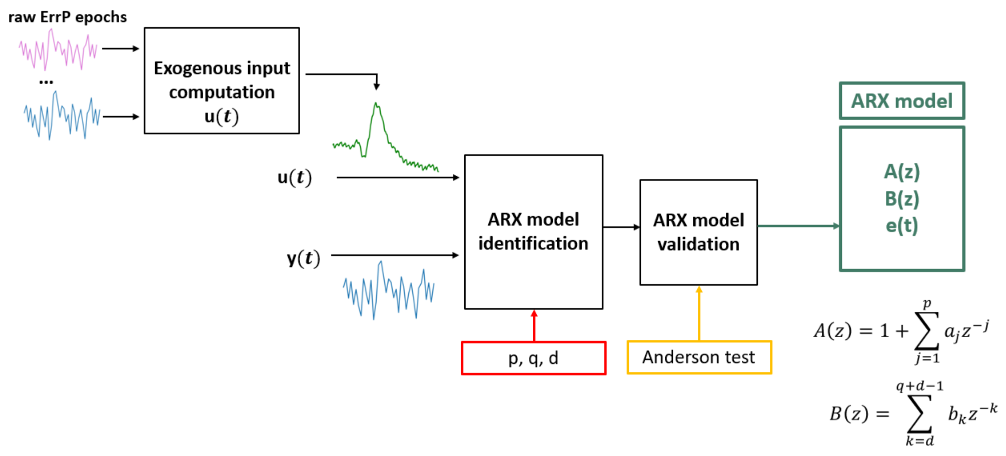


*Figure 2.10: ARX modelling scheme: Exogenous input u(t) is computed from the raw ErrP epochs. For each epoch y(t) an ARX model is identified and validated. The results of the procedure are the coefficients of ARX model and the white noise e(t) which drives the model.*

**Exogenous input computation**

The exogenous input is generated by averaging the raw ErrP epochs.

With averaging it is possible to decrease the effects of the stochastic part of the signal and enhancing the deterministic features. By doing so, the waveform obtained represents the realization of the Error Potential.

Epochs are excluded from averaging if one of the following criteria is fullfilled [64]:

- the root mean squared (RMS) difference from the average of all epochs is greater than two times the standard deviation of the RMS difference of all epochs
- the maximum slope of the epoch is greater than two times the standard deviation of the maximum slopes of all epochs

The computed average, excluding those epochs, is used as the exogenous input $u(t)$ for the generation of all the models.

**ARX model identification**

This step includes the identification of the coefficients $a_j$ and $b_k$, the orders $p$, $q$ and the delay $d$ of the model.

An ARX model is fitted on each ErrP epoch of the dataset, generating a model for each time series of each electrode.

The identification of the coefficients is performed through a *least square* approach, minimizing the following figure of merit [64]:

$$y_i(t) = \frac{1}{N}\sum_{j=1}^{N}(y_i(t) - \hat{y_i}(t))^2 \tag{2.8}$$

where N is the number of time samples, $y_i(t)$ is the signal itself and $\hat{y_i}(t)$ is the model's prediction.

A search for the optimal orders of the model is performed. In particular, the orders which result in the minimum value for the Akaike Information Criterion (AIC) are chosen [64].

Tested values for $p$, $q$ and $d$ are presented in table 2.4.

| **Parameter** | **Values** |
|---|---|
| $p$ | [1,7] |
| $q$ | [1,7] |
| $d$ | [0,4] |

*Table 2.4: Range of values tested for the orders p, q and the delay d of the ARX model are reported.*

**ARX model validation**

The models generated are tested to verify their suitability to describe the epochs. *Anderson test* [66] with confidence interval of 95% is performed to check for residuals whiteness. If the model is able to adequately fit signals, no information is contained in the residuals, therefore they follow a normal distribution, otherwise it is rejected.

#### 2.6.1.2 New signals generation

A graphical representation summarizing how new signals are generated is presented in figure 2.11.

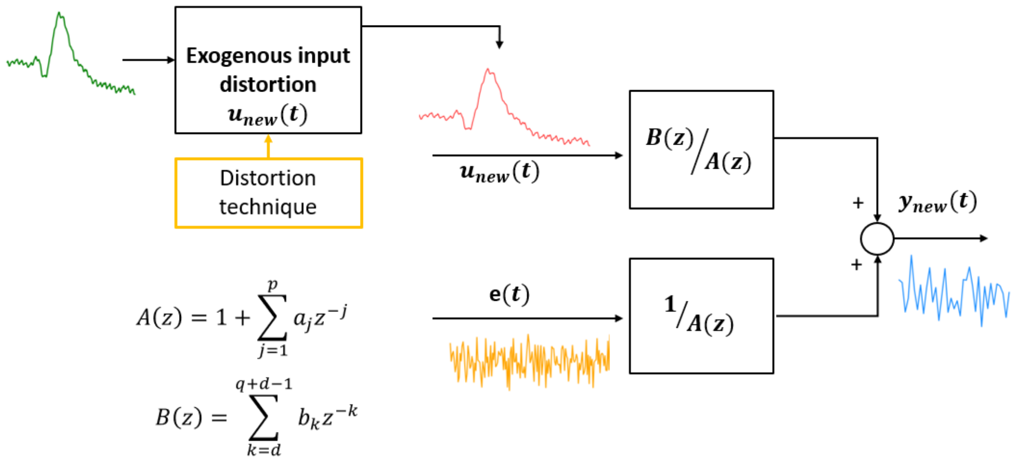

Exogenous input distortion
$u_{new}(t)$
Distortion technique
$u_{new}(t)$
$B(z)/A(z)$
+
$y_{new}(t)$
+
$e(t)$
$1/A(z)$
$A(z) = 1 + \sum_{j=1}^{p} a_j z^{-j}$
$B(z) = \sum_{k=d}^{q+d-1} b_k z^{-k}$

*Figure 2.11: New signals generation scheme: the exogenous input is distorted with the selected transformation. The ARX models obtained are used to simulate the new signals through the distorted exogenous inputs.*

Once the ARX models have been identified, by applying 2.7, it is possible to simulate the epochs.
New signals can be generated by using the same formula but operating some modifications. In particular, some distortions of the exogenous signal $u(t)$ have been evaluated:

- *amplitude*: the amplitude of the waveform is modified
- *noise*: white noise is added to the exogenous input
- *warping*: the exogenous signal's waveform is shrank

Due to the high imbalance a large new dataset is needed: different new versions of the exogenous signal are generated in order to generate multiple new signals by using the same ARX models driven with different inputs. When the newly generated data are added to the training set to balance, the different types of distortions are either treated separately, by considering a

dataset of new signals obtained with the same transformation with different values, or they are combined, by considering a dataset including new signals obtained with different types of transformations.
A summary of all the combinations and the related ranges of values tested for each transformation is presented in table 2.5.

| **Type of distortion** | **Range of values** | **Transformation** |
|---|---|---|
| *ARX amplitude* | $a \in [0.5, 1.5]$ | $u_{new}(t) = a \cdot u(t)$ |
| *ARX noise* | $\mu = 0,\ \sigma \in [0.1, 0.9]$ | $u_{new}(t) = u(t) + WN(\mu,\ \sigma)$ |
| *ARX warp* | $shrink \in [75, 125]\%$ | $u_{new}(t) = shrink(u(t))$ |
| *ARX amplitude + noise* | $a \in [0.5, 1.5]$<br>$\mu = 0,\ \sigma \in [0.1, 0.9]$ | $u_{new}(t) = a \cdot u(t)$<br>$u_{new}(t) = u(t) + WN(\mu,\ \sigma)$ |
| *ARX amplitude + warp* | $a \in [0.5, 1.5]$<br>$shrink \in [75, 125]\%$ | $u_{new}(t) = a \cdot u(t)$<br>$u_{new}(t) = shrink(u(t))$ |
| *ARX noise + warp* | $\mu = 0,\ \sigma \in [0.1, 0.9]$<br>$shrink \in [75, 125]\%$ | $u_{new}(t) = u(t) + WN(\mu,\ \sigma)$<br>$u_{new}(t) = shrink(u(t))$ |
| *ARX amplitude + noise + warp* | $a \in [0.5, 1.5]$<br>$shrink \in [75, 125]\%$<br>$\mu = 0,\ \sigma \in [0.1, 0.9]$ | $u_{new}(t) = a \cdot u(t)$<br>$u_{new}(t) = u(t) + WN(\mu,\ \sigma)$<br>$u_{new}(t) = shrink(u(t))$ |

*Table 2.5: All the combinations and the related range of values for the distortions applied at the exogenous input are reported. In the last column the formula by which the distortion is calculated are specified.*

Finally, the newly generated data are preprocessed with the same steps presented in the *Data preprocessing* section.
A graphical inspection is done to verify that the signals generated are not too different from the original ones.

### 2.6.2 Other balancing methods

Resampling methods includes those methods that operate by adding or removing samples from the training set.
The two implemented in this study are:

- *oversampling*
- *undersampling*

Oversampling takes the training set and randomly duplicates samples of the minority class until balance is reached.
Undersampling takes the training set and randomly removes samples of the majority class until balance is reached.
Besides these two methods, also the *class weights* approach used in [49], where EEGNet is introduced, is considered. It does not modify the training set but assigns a weight to each sample depending on the class in a way that minority class samples "weights" more during training.
In particular, class weights are calculated according to the following formula:

$$weight_i = \begin{cases} 1, & \text{if } i \text{ in } majority\ class \\ \frac{\#majority}{\#minority}, & \text{if } i \text{ in } minority\ class \end{cases} \tag{2.9}$$

where $\#majority$ is the size of the majority class samples and $\#minority$ is the size of the minority class ones.

## 2.7 Hyperparameters search

Hyperparameters of EEGNet are optimized in order to improve classification performances.

Both hyperparameters related with EEGNet architecture and with the training algorithm are considered.
Search has been divided in four cycles so that for each one different groups of hyperparameters are tested. The hyperparameters that are more related to each other are grouped together: for instance, the ones regarding the iterative method or the ones regarding the number of filters of the layers are included in the same cycle to be tested together. The cycles are ordered in a way that, at first, the most important hyperparameters, thus the ones that affect most the performances, are optimized and then the search continues with the cycles including the less relevant ones.
In cycles in which important hyperparameters are tested a grid search approach is applied, while in those cycles in which less relevant hyperparameters are tested or in which the possible ranges of values are unknown a-priori are tested with random search. Once random search is performed, a smaller range of values for the hyperparameters is identified and for those grid search is performed.
For each cycle the configuration that results in the highest metric score on the validation set is identified: the metric optimized during the search cycles is the *F1-score*. The hyperparameter configuration is modified according to the results and the next cycle is started. The procedure is repeated until convergence is reached, i.e. the hyperparameters set does not change, or when a given high number of repetitions is reached.
The cycles now presented are the ultimate ones: they are all performed with grid search. A random search was initially performed to verify which hyperparameters affect the most the results and what ranges consider.
This is the best configuration for each cycle that has been obtained.

### 2.7.1 1st cycle

In the first cycle the hyperparameters related with the optimization algorithm are tested: *learning rate, batch size* and *iterative method.* They define how fast and with which algorithm the minimum of the loss function is reached.
In literature it is reported that those are the ones that most affect the network's performances, especially the learning rate [40].
Learning rate is typically defined in range $[0, 1]$ and greater than $10^{-6}$ while batch size is defined as a power of two value between 1 and a few hundreds [67].
The tested values and the EEGNet default values are presented in table 2.6.

| **Hyperparameter** | **Tested values** | **Default value** |
|---|---|---|
| *learning rate* | 0.0001, 0.001, 0.01, 0.1, 1 | 0.001 |
| *iterative method* | Adam, RMSprop, SGD | Adam |
| *batch size* | 16, 32, 64, 128, 256 | 16 |

*Table 2.6: 1st cycle of hyperparameter search. The hyperparameters and the related tested and default values are specified.*

### 2.7.2 2nd cycle

In the second cycle different types of layers and activation functions are tested. The layers considered are the *pooling, dropout* and *activation* ones.
In EEGNet in both Block 1 and 2 an activation, a pooling and a dropout layers are present and defined in the same way.
Here a differentiation between Block 1 and Block 2 is opted.
The activation function in the last layer is not modified, since, to make the classification, the sigmoid is the most appropriate function.
The default and search values are reported in table 2.7.

| **Hyperparameter** | **Layer** | **Tested values** | **Default value** |
|---|---|---|---|
| *pooling layer* | poolingType1 | Average Pooling, Max Pooling | Average Pooling |
| | poolingType2 | Average Pooling, Max Pooling | Average Pooling |
| *dropout layer* | dropoutType1 | Dropout, Spatial Dropout | Dropout |
| | dropoutType2 | Dropout, Spatial Dropout | Dropout |
| *activation layer* | activationType1 | ELU, ReLU | ELU |
| | activationType2 | ELU, ReLU | ELU |

*Table 2.7: 2nd cycle of hyperparameter search. The hyperparameters and the related tested and default values are specified.*

### 2.7.3 3rd cycle

In the third cycle the number of filters of the three convolution layers are evaluated: *F1, D* and *F2*.
Starting from the values defined in [49], a range $[value - 2,\ value + 2]$ is considered for both F1 and D.
F2 in [49] is defined as the product of F1 and D, here it is tried also to decouple its definition from the two other values.
A summary of the search is reported in table 2.8.

| **Hyperparameter** | **Tested values** | **Default value** |
|---|---|---|
| *F1* | 6, 8, 10 | 8 |
| *D* | 1, 2, 4 | 2 |
| *F2* | 6, 8, 10, 12 14, 16, 18, 20, 22, 24, 26, 32, 36, 40, 44 | 16 |

*Table 2.8: 3rd cycle of hyperparameter search. The hyperparameters and the related tested and default values are specified.*

### 2.7.4 4th cycle

In this cycle the *dropout rate* of the dropout layers, the *momenta* of the iterative method, and the *learning decay* are tested.If the iterative method is *Adam* two momenta have to be tested, otherwise just one.The learning decay is a technique in which the learning rate is not constant but it is initialized with a large value and then it decays during the iterations. It should prevent to end in local minima and helps the optimization [68]. Learning decay is not applied in EEGNet originally, here its addition is evaluated.
A summary of the tested values is presented in table 2.9.

| **Hyperparameter** | **Tested values** | **Default value** |
|---|---|---|
| *dropout rate* | 0.3, 0.5, 0.7 | 0.5 |
| *momentum* | *beta 1*= 0.9, 0,99, 0.999 | 0.9 |
| | *beta 2*= 0.99, 0.995, 0.999 | 0.999 |
| *learning decay* | True, False | False |

*Table 2.9: 4th cycle of hyperparameter search. The hyperparameters and the related tested and default values are specified. If the iterative method is SGD or RMSprop the momentum considered is just beta 1, otherwise both of them have to be optimized.*

# Chapter 3

# Results

The Chapter presents the main results of the thesis and it is organized as following. At first a graphical validation of the data, both the original data and the generated ones, is performed. Grand average ErrP is plotted to identify the characteristic shape of the event-related potential. For the original dataset the differences between the subjects curves are inspected, while for the ARX generated data an evaluation on the goodness of the data augmentation method is done.
Then EEGNet results are presented in terms of inter-subjects and intra-subject analysis. The performances obtained with the ARX-based data augmentation methods are compared with the other approaches, i.e. class weights, oversampling and undersampling. Results are reported in terms of accuracy, F1-score and balanced accuracy. Finally, a discussion on the impact of these methods in terms of improvement of the Utility gain metric is also provided.
Eventually, the hyperparameters search process is presented, reporting the final configuration of the network and its performance after the optimization.

## 3.1 Preprocessed data: graphical inspection

At first all the subjects' data are considered: the grand average ErrP, computed over all the epochs of all the subjects, is presented in figure 3.1 for channel FCz and Cz. It is obtained by computing the difference between the averaged ErrP epochs and the non-ErrP ones. As discussed before, the error-processing activity is detected mainly in the FCz and Cz electrodes: for this reason, the waveforms are reported for these channels.

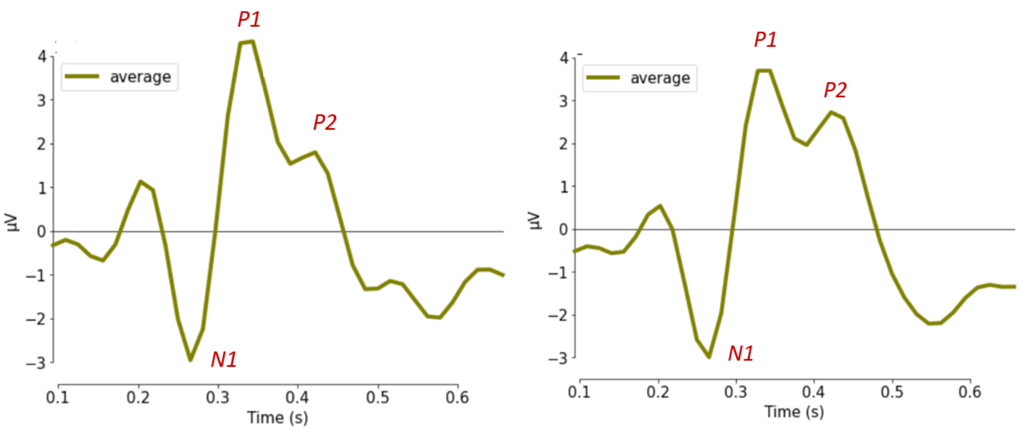


*Figure 3.1: Grand average ErrP over all subjects' data: averaged ErrP epochs minus averaged non-ErrP epochs for channel FCz (left) and Cz (right). The curves are characterized by three main peaks indicated in the figures: N1, P1 and P2.*

The curves obtained are characterized by an amplitude belonging in the $[-3,\ 4]\ \mu V$ range and by three main peaks: a negative peak occurring at $0.266s$, a first positive peak at $0.344s$ and a secondary positive peak at $0.422s$. After $0.55s$ the signals lie under the baseline activity.
By considering these time instants the topographical maps are plotted in order to verify which are the electrodes, and consequently the brain areas, showing more intense electrical activity at the peaks occurrences. The maps

are reported in figure 3.2.

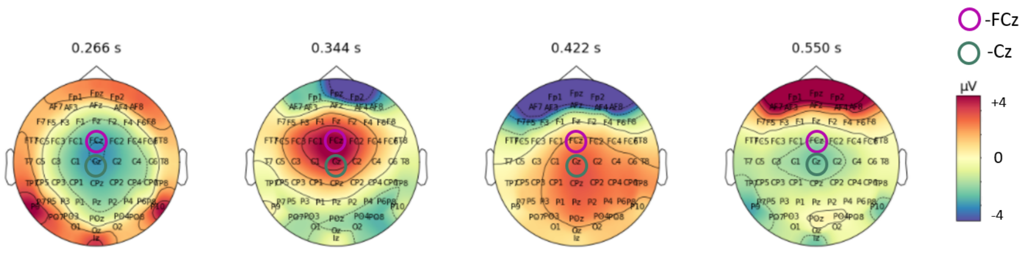


*Figure 3.2: Topographical maps showing brain areas activations. In particular, the maps refer to the time instants of the ErrP waveforms' peaks: 0.266 s, 0.344 s, 0.422 s, 0.550 s. The pink circle highlights the FCz electrode localization, while the green one the Cz electrode. The colors of the maps identify the electrical activity range between* $-4\mu V and + 4\mu V$.

An effective decrease of the activation at the negative peak and an increase of the electrical activity at the positive peaks latencies can be noticed in the medio-frontal areas of the brain.

Following, a distinction between subjects' data is made to evaluate the intra-patient differences in activations: the grand average ErrP curves for each subject are plotted and compared to the one constituted by all subjects' data (figure 3.3). In particular, it can be noticed that the latencies of the peaks are almost the same for all the subjects while there are differences in the amplitude of the peaks: in FCz channel Subject 1 and Subject 3 present higher peaks, while Subject 2 and 5 have comparable amplitudes in respect to the average; for the Cz electrode, Subject 1 and Subject 2 show higher amplitude, while Subject 3 and Subject 5 present lower peaks. It can be noticed also that the second positive peak is less pronounced in some subjects.

Subject 4 and Subject 6 have a different behaviour respect to the others (figure 3.3): in particular, by evaluating the signals from the Cz electrode, it is possible to observe that the two positive peaks are not distinguishable, but a wider unique positive peak occurs. For Subject 6, neither the negative peak is noticeable.

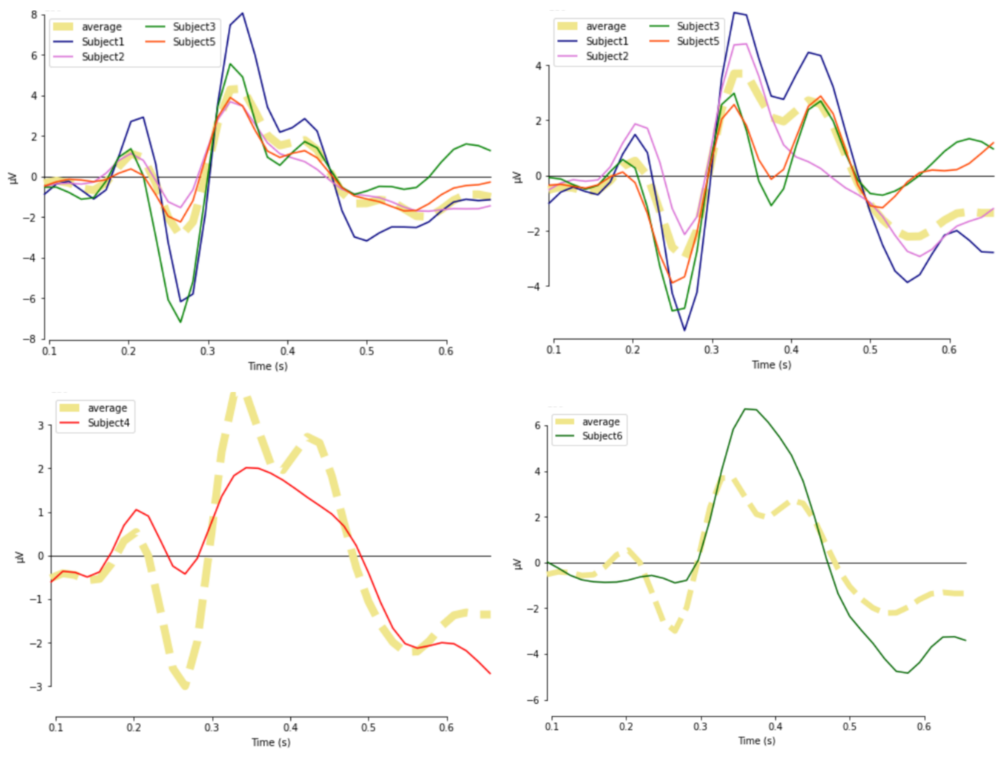


*Figure 3.3: On top: single subjects' grand average ErrP curves compared to the average one (dashed line) for FCz (left) and Cz (right) electrodes. On bottom: ErrP waveforms for Cz electrode respect to the average one (dashed line) for Subject 4 (left) and Subject 6 (right).*

## 3.2 ARX-based data augmentation

The results regarding the ARX-based data augmentation approach are now presented. At first results about the models generation process and then about the new signals are reported.

To generate the models, the exogenous input is computed, following the criteria presented in Chapter 2: the generated waveform is shown in figure 3.4. In particular, the signal plotted refers to the FCz channel, used as a representative example.

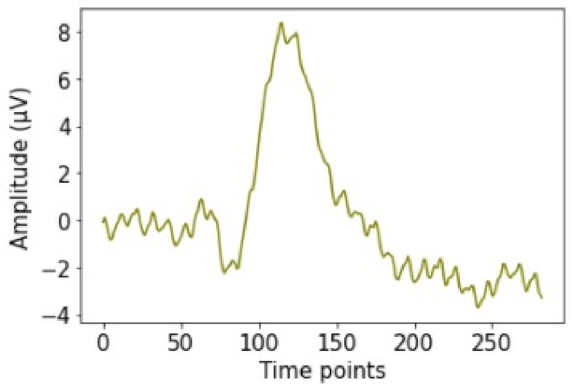


*Figure 3.4: Computed exogenous input for ARX's models generation*

It is characterized by an amplitude ranging from $[-4,\ 8]\ \mu V$.

To generate the models, a model identification is performed on each ErrP epoch of the dataset: overall the models which are validated by the Anderson test result to be 367.

The new data are generated by using those models and computing a distortion on the exogenous input. As mentioned before, the distortions operated concern the change of amplitude, addition of white noise and shrinking of the curve.

The dataset consists of 6437 epochs, of which 1322 are ErrP epochs. After the partition in training, validation and test sets the imbalance on the training set is at its maximum of 4119 epochs. To generate enough new signals with 367 models, a series of distortions for the exogenous signal are necessary: the range of values within the distortion has to be computed is defined and then a set of equally distributed values within the range is chosen to generate the new exogenous signals. The ranges applied for each transformation are the ones reported in Chapter 2.

In figure 3.5 the resulted exogenous inputs obtained through the changes of amplitude, noise addition and warping techniques are reported. Also in this case the FCz channel signal is plotted.

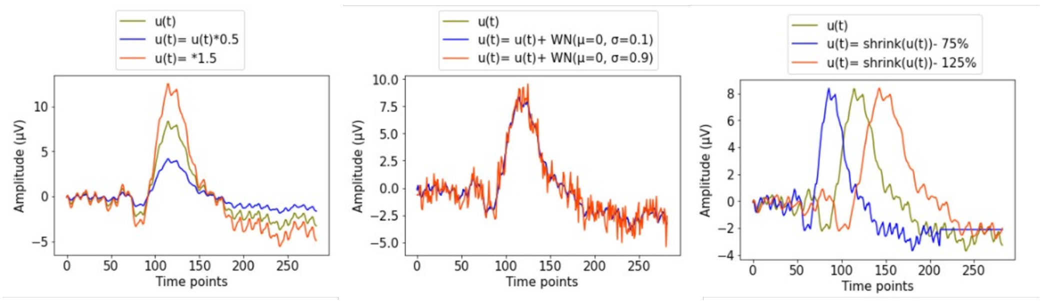


*Figure 3.5: Exogenous signal distortions: change in amplitude (left), white noise addition (center) and warping (right)*

The new generated datasets, obtained both by considering new signals obtained with a single distortion technique and by grouping signals obtained with different techniques, are then inspected to verify if they have the same characteristics of the original dataset: the grand average ErrP curves are compared to the overall original one.

Since the ARX new data are just ErrP epochs, the non-ErrP epochs, needed to generate the waveforms, are extracted from the original dataset. The results for all the ARX methods are reported in figure 3.6 and 3.7.

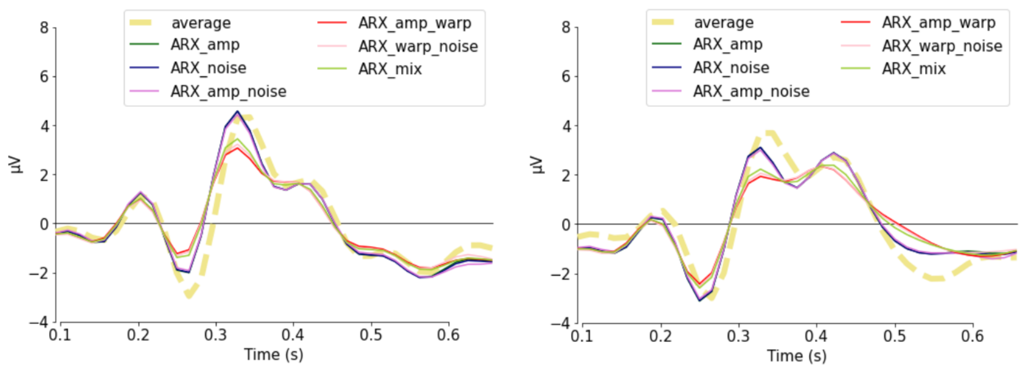


*Figure 3.6: Grand averaged ErrP for the ARX's new data compared to the original one (dashed line). The waveforms are reported for channel FCz (left) and Cz (right).*

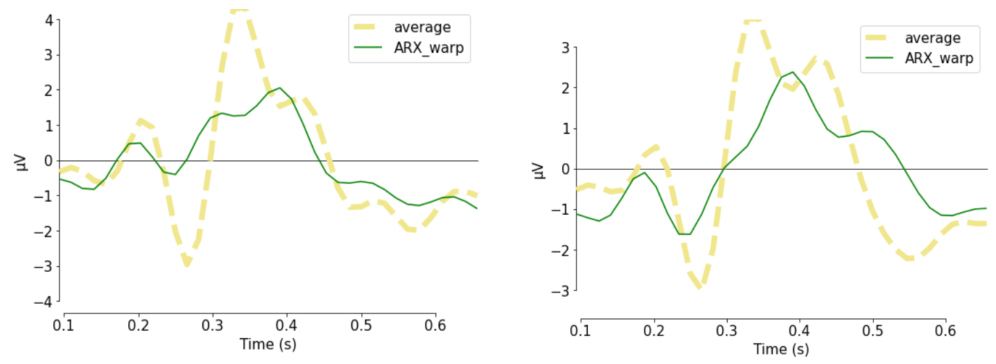


*Figure 3.7: Grand averaged ErrP obtained by considering the ARX warping data respect to the original one (dashed line). The waveforms are reported for channel FCz (left) and Cz (right).*

As expected, all the ARX techniques, except for the ARX warping, generate new data that are very similar to the original ones. The waveforms present the same peaks latencies and amplitudes (figure 3.6). Conversely, the ARX warping technique introduces some noticeable changes to the shape of the curve: the amplitude range is smaller and the peaks latencies are not respected. In particular, it can be observed that the positive peaks for the Cz channel occur with a significant delay respect to the original grand average ErrP.

## 3.3 Inter-subjects analysis

In the inter-subjects analysis EEGNet performances are evaluated on the whole dataset, without making any distinction on the single subjects' data. The results obtained are presented for the traditional balancing methods (class weights, oversampling and undersampling) and for the introduced ARX-based data augmentation approaches. In the following figures the bar-graphs, showing the results obtained on the test set through majority vote approach, are reported for accuracy (figure 3.8), F1-score (figure 3.9) and balanced accuracy (figure 3.9).

Regarding the traditional methods, it can be noticed that class weights and oversampling have comparable results while undersampling results show the lowest performances for all the three metrics.

The performances by applying the ARX-based methods outperform the traditional ones in all the cases. A particular case is the ARX warping technique which results in a low value for the F1-score but a high value for balanced accuracy and accuracy. The scores obtained by the methods which establish the addition of data obtained with a single distortion of the exogenous input (ARX amplitude, ARX noise and ARX warping) are not so different from the ones in which data obtained with different distortions are grouped.

Overall, the methods which result in the highest scores are the ARX amplitude, ARX amplitude + noise, ARX amplitude + warp and the ARX mix.

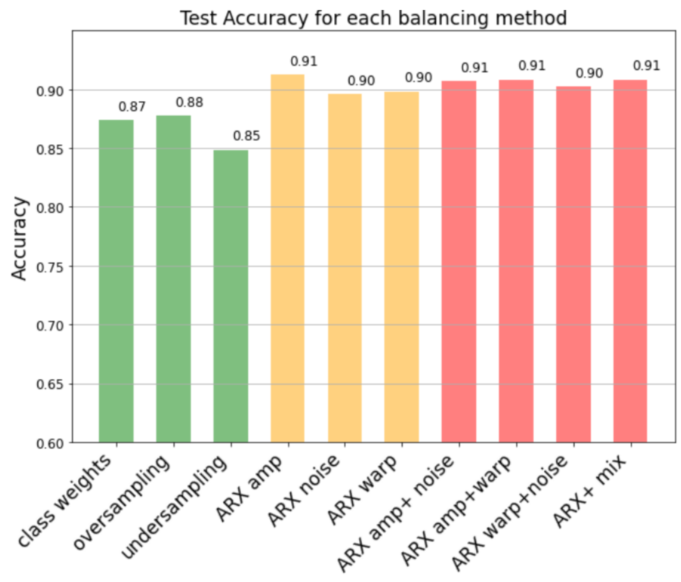


*Figure 3.8: Inter-subjects accuracy for the test set for each balancing method. The green bars indicate EEGNet results with the traditional balancing methods, the orange bars with the ARX-methods by using just one distortion technique on the exogenous input, while the red ones with combined distortions techniques.*

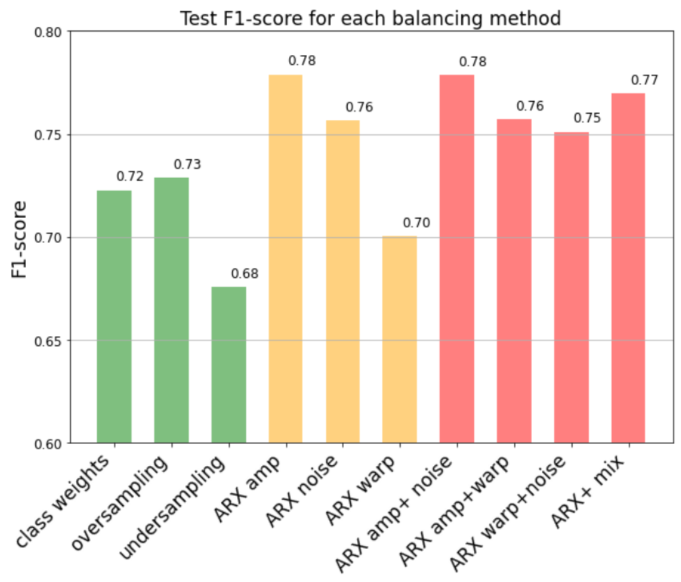


*Figure 3.9: Inter-subjects F1-score for the test set for each balancing method.*

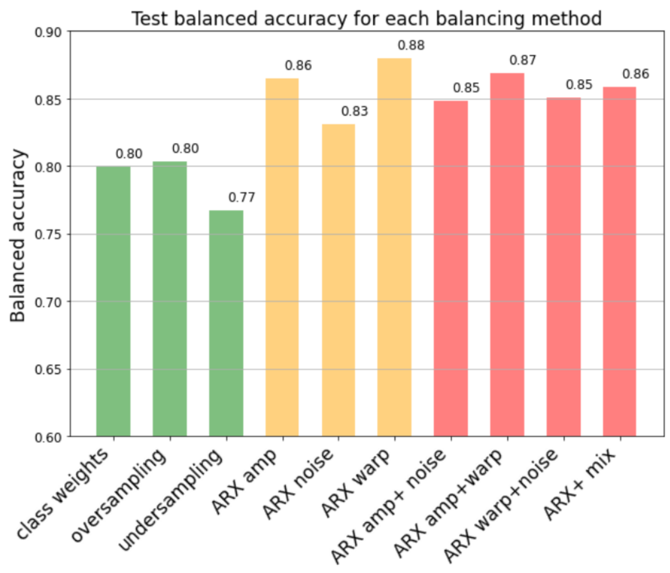
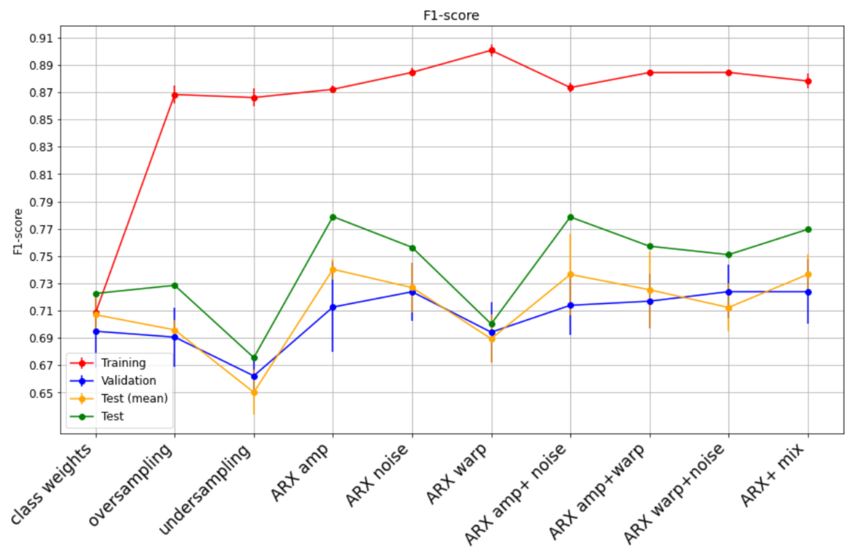


*Figure 3.9: Inter-subjects balanced accuracy for the test set for each balancing method.*

For the sake of comparison, the performances on training and validation sets are reported in figure 3.10 together with the already presented results on the test set.

In this thesis the majority vote approach has been chosen to evaluate the performances on the test set: in the following graphs the results on the test set obtained by averaging the results of each iteration of the cross-validation process, are also shown just to report the different behaviours of the two approaches. Further mentions about test set results, if not explicitly reported, concern the results obtained with the majority vote strategy. By comparing them, it can be observed that averaged test results are more similar to the validation set results. The majority vote scores are always higher except for the traditional techniques when balanced accuracy is considered.

Overall, there is a difference between training and test results: in terms of F1-score the difference is high while for the balanced accuracy the gap is reduced. This is verified for all the balancing methods excluding the class



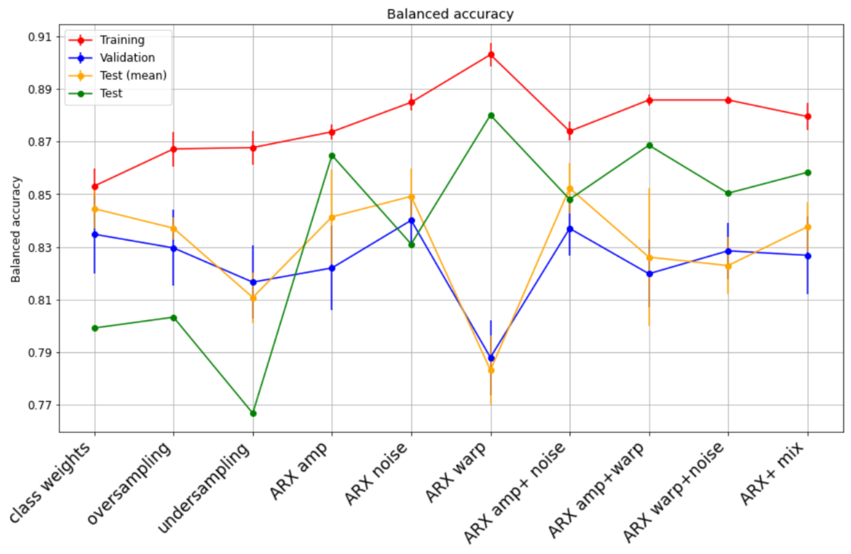


*Figure 3.10: Training, validation and test sets performances in terms of F1-score (top) and balanced accuracy (bottom). The test results are reported both in terms of majority vote results (green line) and averaged on the iterations of the cross-validation (orange line).*

weights approach in which the performances are comparable for the three sets.

For both the metrics, the highest difference is obtained with the undersampling approach being equal to $\sim$ 19% for the F1-score and $\sim$ 9% for the balanced accuracy. A reduction of the gap between training and test sets can be observed for the ARX amplitude, ARX amplitude + noise and the ARX mix approaches.

The results on the training set for the ARX warping technique confirm the discrepancy observed in the test set results between F1-score and balanced accuracy: the gap occurring between training and test performances is high if considering the F1-score, while it is smaller if considering the balanced accuracy. However, it can be noticed that the gap is high also for the balanced accuracy if the averaged test and the validation set scores are considered.

To have an additional point of view on the results some confusion matrices are presented in table 2.3 : the ones obtained with the traditional methods (top row of table 2.3) and the ARX amplitude, ARX warping and ARX mix (bottom row of tabel 2.3) are shown.

| *Class weights* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 211 | 115 |
| **0** | 47 | 915 |

| *Oversampling* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 212 | 112 |
| **0** | 46 | 918 |

| *Undersampling* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 203 | 140 |
| **0** | 55 | 890 |

| *ARX amplitude* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 199 | 54 |
| **0** | 59 | 976 |

| *ARX warping* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 153 | 26 |
| **0** | 105 | 1004 |

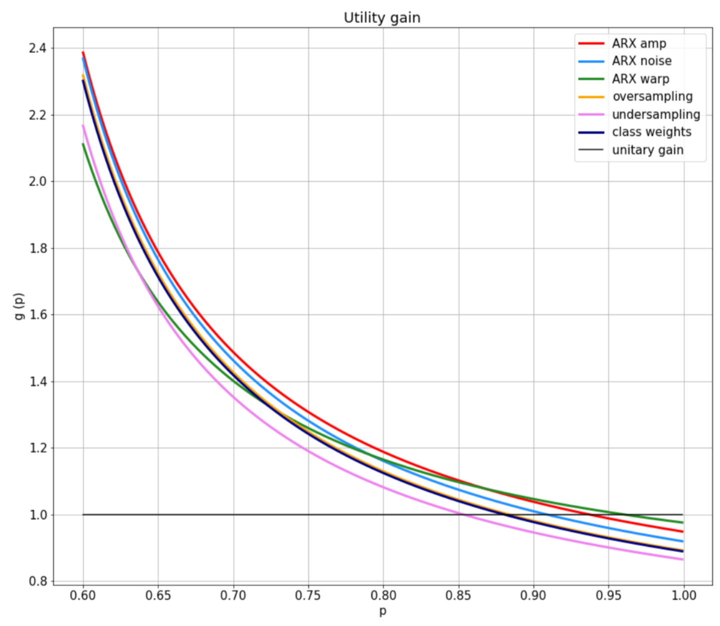
| *ARX mix* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 197 | 57 |
| **0** | 61 | 973 |

*Table 3.1: Confusion matrices of the classification for traditional methods (on top) and ARX-based data augmentation approaches (bottom).*

Class weights and oversampling have similar confusion matrices: it was expected by the similar test performances. Conversely, with the undersampling method both the number of False Positives and False Negatives increase. The confusion matrices of the ARX methods show that the number of False Positives is significantly decreased by those approaches. The number of False Negatives, instead, is a little higher for the ARX amplitude and the ARX mix while it is doubled for the ARX warping, which again is the method with the poorest performances.

Finally, Utility gains $g(p)$ are reported in figure 3.11 for each balancing method. The curves represent the potential gain that can be introduced in a BCI system by adding an ErrP-based correction system: higher is the gain, higher is the benefit. The curves are in function of the BCI classifier accuracy p, thus the gains for the different balancing methods can be compared and the best one can be identified by fixing a value for p and verifying which curve shows the biggest value on the y-axis. As discussed in Chapter 2, if gain $<1$ the addition of an error-correction system is counterproductive: the unitary gain is reported in the figures as a black line: when the functions go under that line any improvement is gained by adding the ErrP classifier. Traditional methods provide an improvement until the BCI classifier has accuracy $p = \sim 88$ % while the ARX-based methods range from $\sim 91$ % to $\sim 96$ %.

Overall, it can be noticed that the ARX-based methods provide a higher gain: the combined techniques and the ARX amplitude are the ones with higher values.



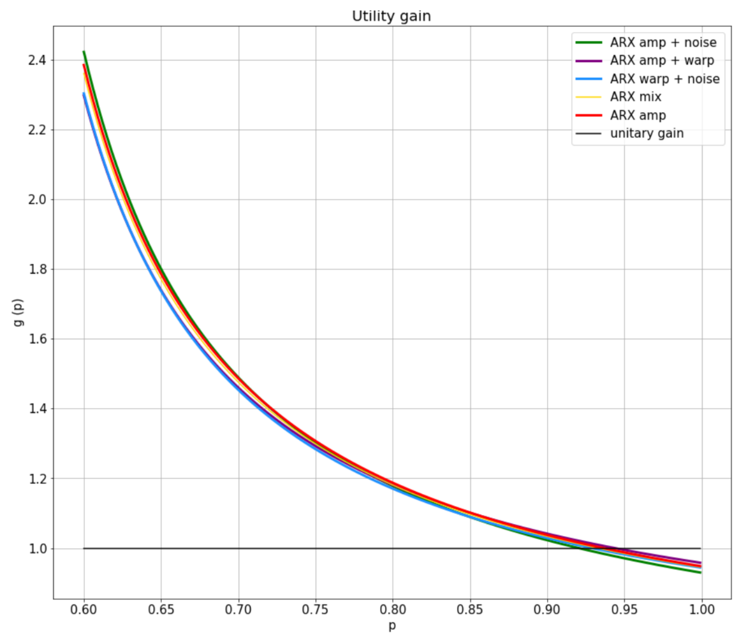
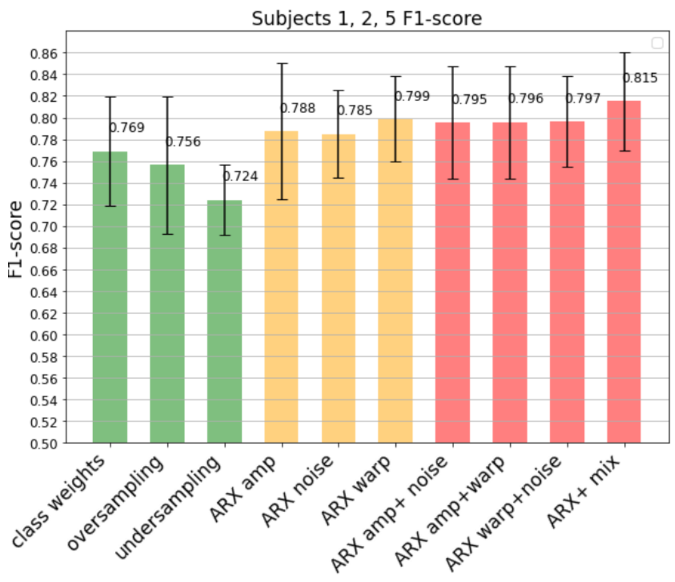


*Figure 3.11: Inter-subjects Utility gain functions: on top, traditional methods and ARX-based with a single techniques are reported, on bottom the ARX combined techniques compared to the ARX amplitude. The curve of the gain is in function of p, i.e the accuracy of the BCI classifier.*

## 3.4 Intra-subject analysis

EEGNet performances on the single subjects' data are now reported.
In general, the results present high variability between subjects: Subject 1, 2 and 5 data show higher performances respect to Subjects 3, 4, and 6. The averaged results for the two groups in terms of F1-score and balanced accuracy are presented in figure 3.12 and figure 3.14.
By considering the first group, i.e. Subject 1, 2 and 5, it can be noticed that the results obtained with the ARX-methods are higher respect to the traditional methods. It can be observed from the black segments, representing the standard deviations of the distributions, that there is a high variability of the scores between the subjects, especially for the F1-score.



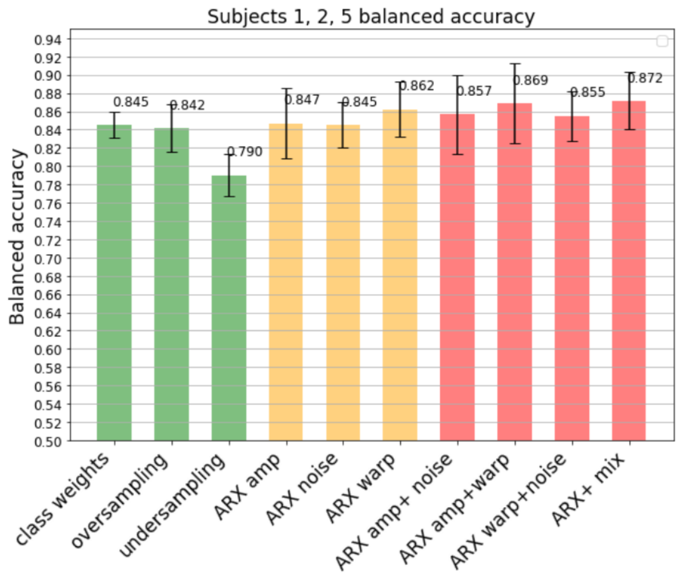
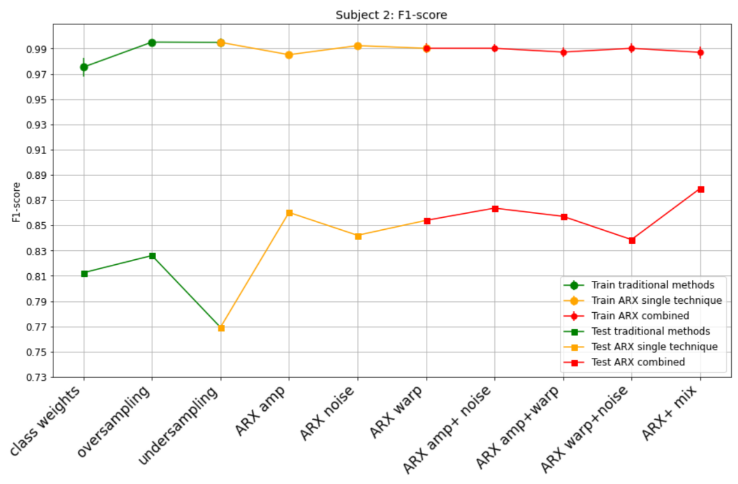


*Figure 3.12: Subjects 1, 2 and 5 averaged results are reported in terms of F1-score (top) and balanced accuracy (bottom) for each balancing method. Colors identify the different class of methods: green represents the traditional methods, orange the ARX with a single technique and red ARX with combination of techniques. The black segments define the standard deviation of the distribution over the subjects.*

Subject 2 data are the one performing better, resulting in a balanced accuracy 92.72 % for the ARX amplitude + warping technique and 87.9 % in terms of F1-score for the ARX mix. The results for both training and test sets are reported in figure 3.13. It can be noticed that the ARX methods lead to an improvement of the performances and a reduction of the gap that is present between the training and test sets performances. The highest performances are obtained with the ARX amplitude, ARX amplitude + warping and ARX mix.



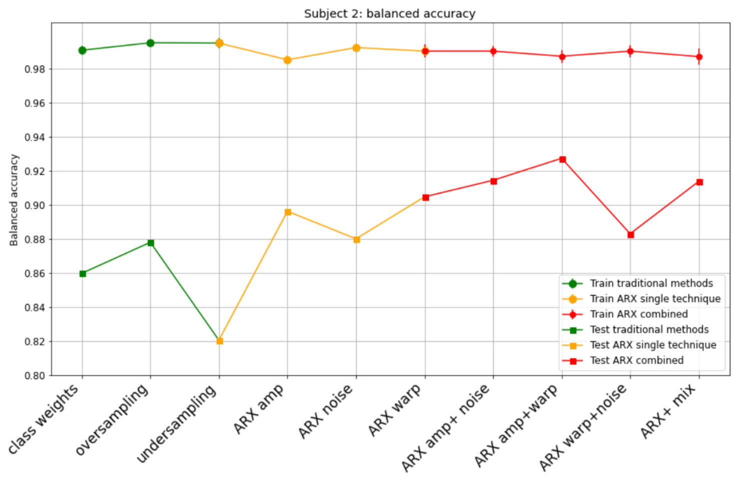
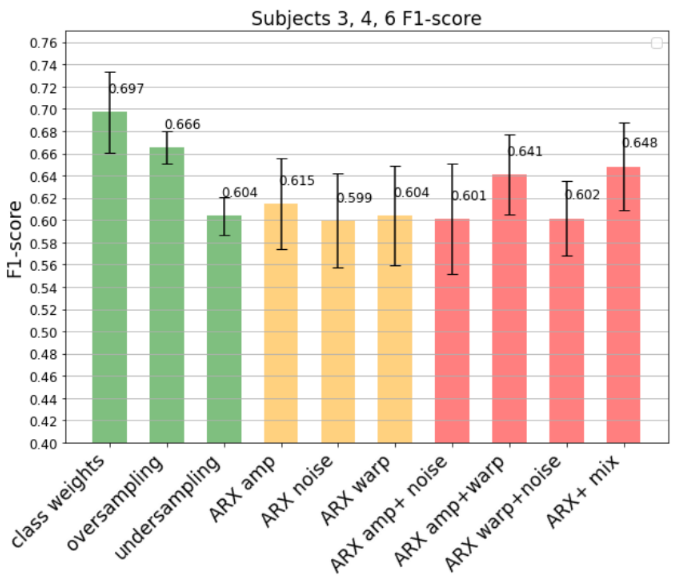


*Figure 3.13: Subject 2: results in terms of F1-score (top) and balanced accuracy (bottom) are reported for each balancing method.*

The results for the second group, i.e. Subjects 3, 4 and 6, are reported in figure 3.14. These subjects' data have lower performances respect to the first group, especially in terms of F1-score. The method that performs better is class weights, resulting in a score of 72.72 % for the F1-score and 81.74 % for the balanced accuracy for the Subject 6. It can be observed that, ARX mix and ARX amplitude + warping show comparable scores if the balanced accuracy is considered.



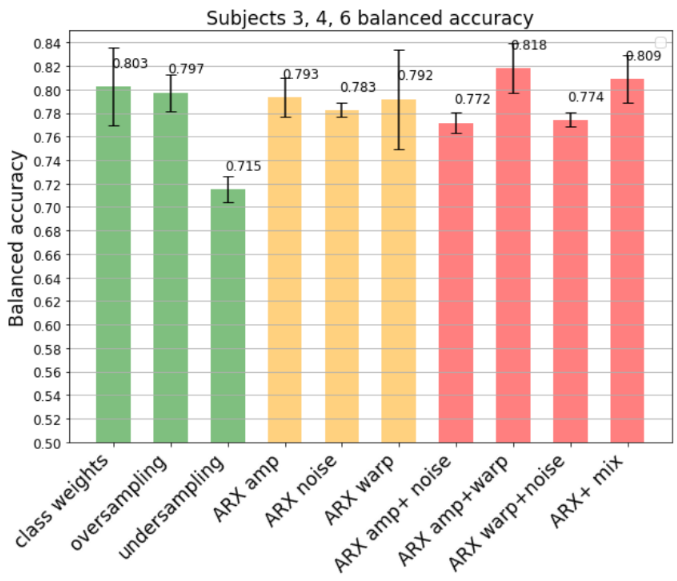
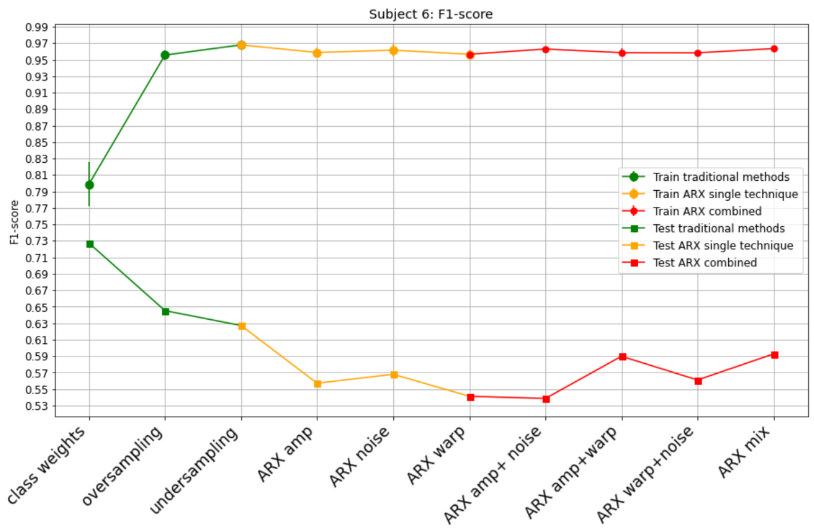


*Figure 3.14: The results of Subjects 3, 4 and 6 averaged are reported in terms of F1-score (top) and balanced accuracy (bottom) for each balancing method.*

Scores for training and test sets of Subject 6, which results in the lowest performances, are reported in figure 3.15.



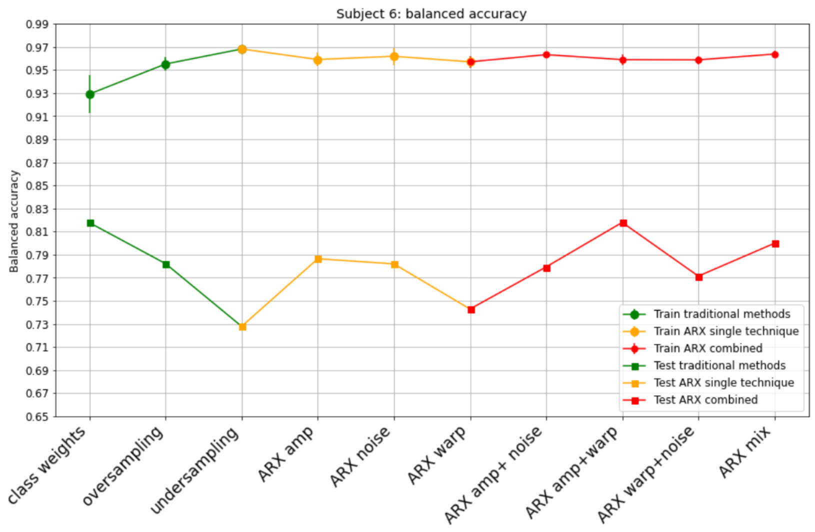


*Figure 3.15: Subject 6: results in terms of F1-score (top) and balanced accuracy (bottom) are reported for each balancing method.*

It can be observed that, in terms of F1-score, class weights and oversampling have significant higher results, having also the class weight score on the test set very close to the training one. Regarding the balanced accuracy, ARX

amplitude + warping shows comparable results to the class weights one. For both the metrics the gap between training and test performances is high, reaching a maximum of $\sim 40$ % in terms of F1-score for the ARX amplitude method.
The confusion matrices for Subject 2 and 6 are reported in table 3.2: the one referring to class weights, ARX amplitude+ warping and ARX mix are selected.

| *Class weights* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 39 | 13 |
| **0** | 5 | 159 |

| *ARX amp+ warp* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 36 | 4 |
| **0** | 8 | 36 |

| *ARX mix* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 40 | 7 |
| **0** | 4 | 165 |

| *Class weights* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 36 | 15 |
| **0** | 12 | 158 |

| *ARX amp+ warp* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 23 | 7 |
| **0** | 25 | 166 |

| *ARX mix* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 24 | 9 |
| **0** | 24 | 164 |

*Table 3.2: Confusion matrices of the classification for Subject 2 (top) and Subject 6 (bottom) for class weights, ARX amplitude + warping and ARX mix methods.*

In all the cases the ARX methods result in a lower number of False Positives than the class weights approach. The number of False Negatives is comparable between class weights and the ARX methods for Subject 2 while it is doubled in case of Subject 6.
Eventually, the Utility gain functions are reported for each subject: for each one just the functions related to the balancing methods that provide the higher gains are shown (figure 3.16).

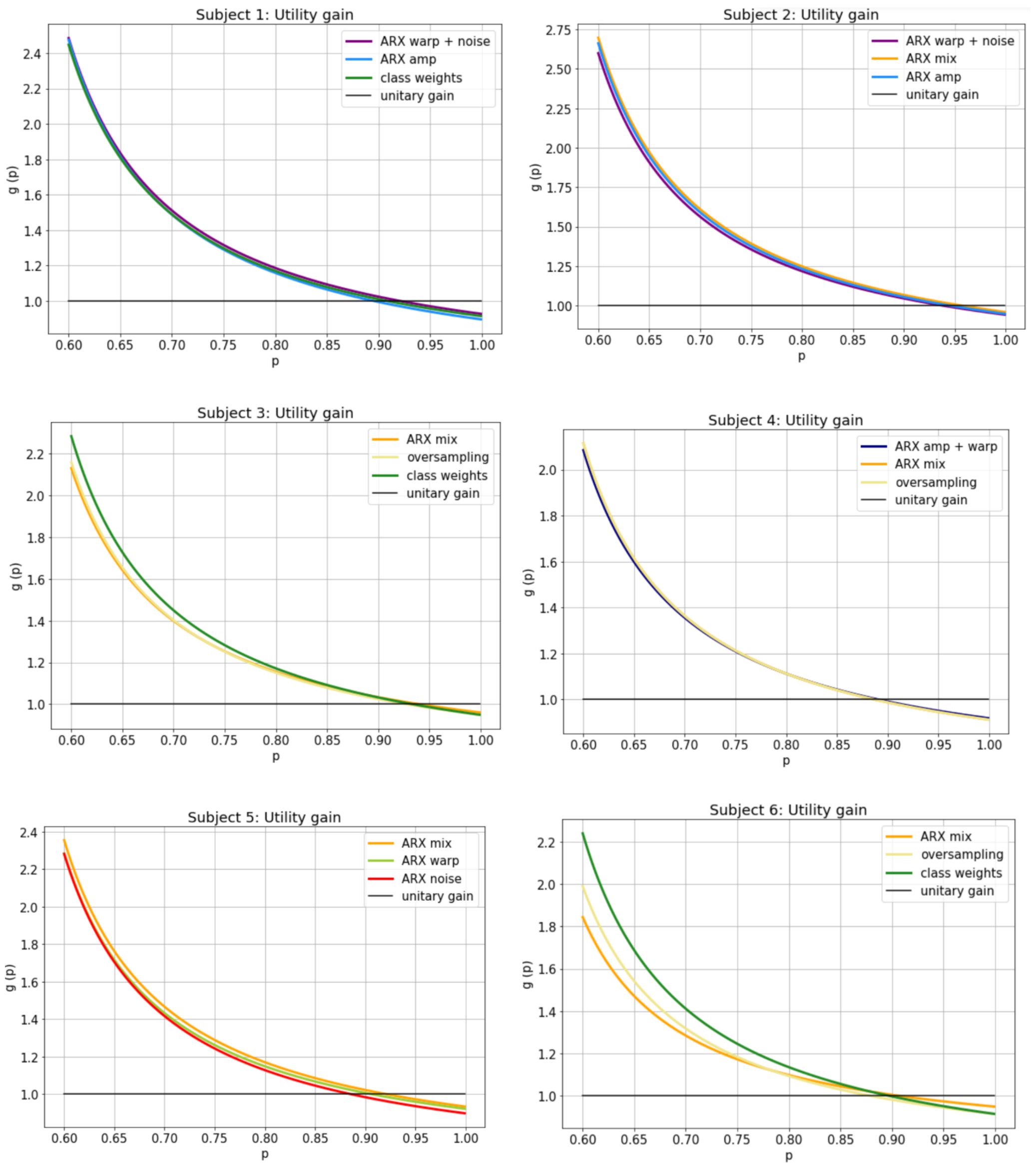


*Figure 3.16: Intra-subject Utility gain functions for each subject.*

They confirm the results previously presented: higher gain is obtained for Subject 1,2 and 5 for the ARX-methods approach while class weights and oversampling provide better results for Subjects 3,4 and 6. Within the ARX methods the combined techniques and in particular, the ARX mix, provide higher scores.

## 3.5 Hyperparameters optimization

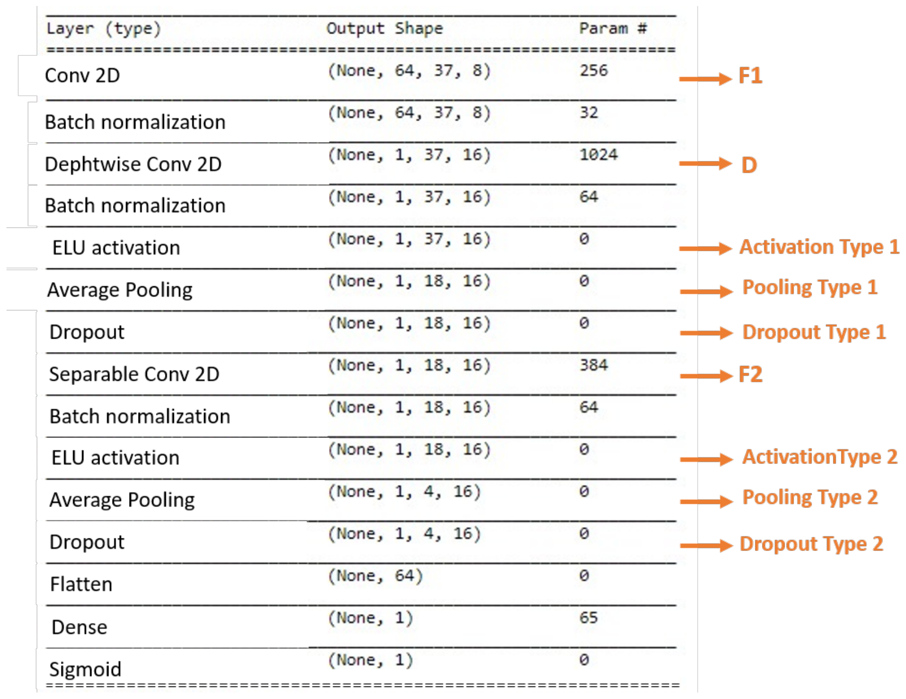
### 3.5.1 Default hyperparameters

The default EEGNet hyperparameters configuration is reported in table 3.3.

| | **Hyperparameter** | **Value** |
|---|---|---|
| **Optimization algorithm** | batch size | 16 |
| | learning rate | 0.001 |
| | iterative method | Adam |
| | beta 1 | 0.9 |
| | beta 2 | 0.999 |
| | learning decay | False |
| **EEGNet Block 1** | Pooling Type 1 | Average Pooling |
| | Dropout Type 1 | Dropout |
| | Activation Type 1 | ELU |
| | F1 | 8 |
| | D | 2 |
| | dropout rate | 0.5 |
| **EEGNet Block 2** | Pooling Type 2 | Average Pooling |
| | Dropout Type 2 | Dropout |
| | Activation Type 2 | ELU |
| | F2 | 16 |
| | dropout rate | 0.5 |

*Table 3.3: EEGNet's default hyperparameters configuration*

As discussed before, the hyperparameters evaluated are related both to the optimization algorithm and to the EEGNet's architecture itself. The EEG-Net default architecture is presented in detail in figure 3.17, specifying the number of trainable parameters and the types of hyperparameters for each layer.



| Layer (type) | Output Shape | Param # | |
|---|---|---|---|
| Conv 2D | (None, 64, 37, 8) | 256 | F1 |
| Batch normalization | (None, 64, 37, 8) | 32 | |
| Dephtwise Conv 2D | (None, 1, 37, 16) | 1024 | D |
| Batch normalization | (None, 1, 37, 16) | 64 | |
| ELU activation | (None, 1, 37, 16) | 0 | Activation Type 1 |
| Average Pooling | (None, 1, 18, 16) | 0 | Pooling Type 1 |
| Dropout | (None, 1, 18, 16) | 0 | Dropout Type 1 |
| Separable Conv 2D | (None, 1, 18, 16) | 384 | F2 |
| Batch normalization | (None, 1, 18, 16) | 64 | |
| ELU activation | (None, 1, 18, 16) | 0 | ActivationType 2 |
| Average Pooling | (None, 1, 4, 16) | 0 | Pooling Type 2 |
| Dropout | (None, 1, 4, 16) | 0 | Dropout Type 2 |
| Flatten | (None, 64) | 0 | |
| Dense | (None, 1) | 65 | |
| Sigmoid | (None, 1) | 0 | |

*Figure 3.17: EEGNet default architecture: for each layer the data output shape is specified, the number of parameters and the relative hyperparameters (specified in orange).*

The total number of trainable parameters of the network is 1889, of which 80 are hyperparameters.
A hyperparameter related with the optimization algorithm but not included in the search is the number of epochs. It defines the number of times that the algorithm goes through the entire dataset during training. It has been decided to not search the value for it: as shown by figure 3.18, which refer to the trend of the metrics over the epochs, the performances are stable around 300 epochs. The training curve, in fact, present an initial steep increment for the first epochs and then it stabilizes around the 200th epoch for all the

three metrics. For this reason, the epoch hyperparameter is maintained at its default value of 300.

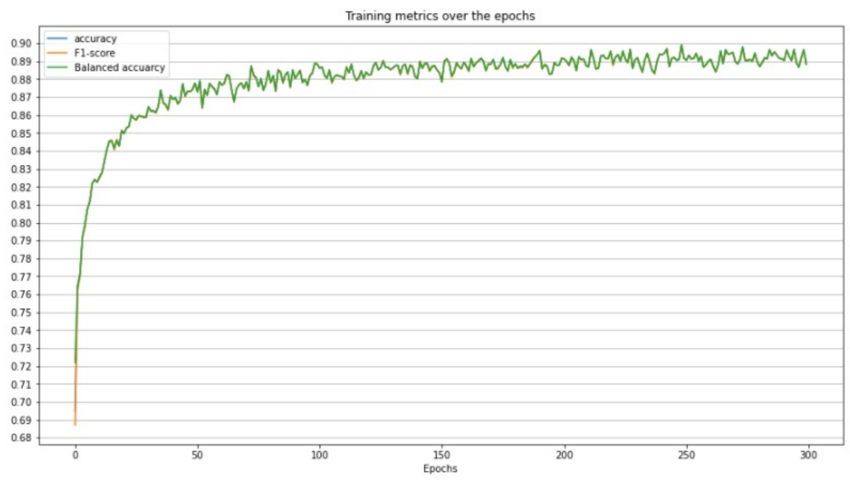


*Figure 3.18: Accuracy, F1-score and balanced accuracy trend over the epochs during training session.*

### 3.5.2 Hyperparameters optimization

A search for the optimal configuration of the hyperparameters of the network is executed. It is performed with a grid search approach through the four cycles defined in Chapter 2. The validation set performances are evaluated at each cycle and the F1-score metric is optimized.

Regarding the balancing method applied on the data, since the search was performed before the development of the ARX-based data augmentation methods, the oversampling approach has been used.

As discussed before, the criteria to stop the search are the convergence or, in alternative, a significant number of repetitions. Convergence was not reached so that the procedure was repeated three times, resulting in three steps each one constituted by the same four cycles.

The final configuration of the hyperparameters obtained is shown in table 3.4.

| | **Hyperparameter** | **Value** |
|---|---|---|
| **Optimization algorithm** | batch size | 64 |
| | learning rate | 0.001 |
| | iterative method | Adam |
| | beta 1 | 0.999 |
| | beta 2 | 0.99 |
| | learning decay | False |
| **EEGNet Block 1** | Pooling Type 1 | Max Pooling |
| | Dropout Type 1 | Dropout |
| | Activation Type 1 | ELU |
| | F1 | 12 |
| | D | 4 |
| | dropout rate | 0.3 |
| **EEGNet Block 2** | Pooling Type 2 | Average Pooling |
| | Dropout Type 2 | Spatial dropout |
| | Activation Type 2 | ReLU |
| | F2 | 26 |
| | dropout rate | 0.3 |

*Table 3.4: EEGNet's default hyperparameters configuration*

Regarding the optimization algorithm hyperparameters, batch size is increased and the two momenta of Adam's algorithm are modified. Regarding EEGNet architecture, pooling, dropout and activation layers are differentiated between Block 1 and Block 2. The number of filters F1, D, F2 of the convolution layers are increased respect to the default ones. Dropout rate is reduced. With the new configuration, the total number of parameters is equal to 5537, of which 172 are hyperparameters.

The F1-score trend over each cycle of the search for the validation set is presented in figure 3.19.

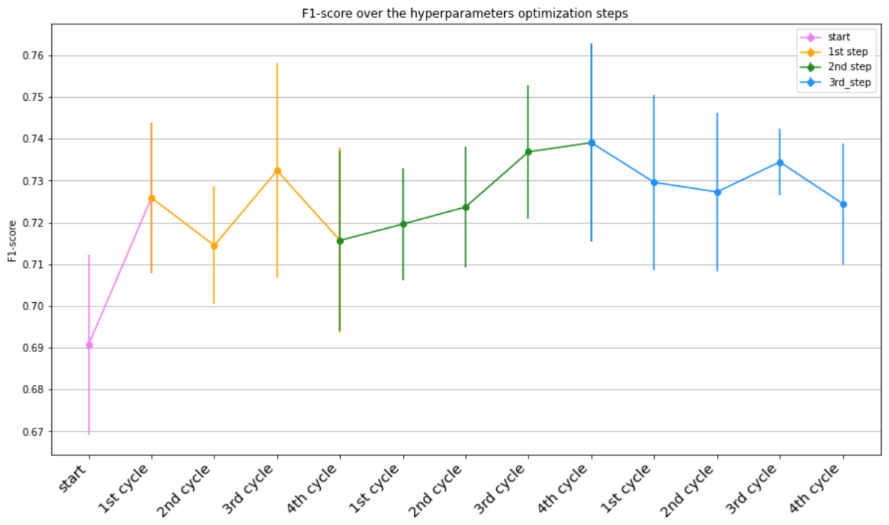


*Figure 3.19: Validation set's F1-score trend over each cycle of each step of the hyper-parameters search.*

The initial F1-score value is of 69 %, at the end of the optimization steps it reaches a value of 72.45%. It can be noticed that until the second step the metric value increases while during the third step it starts decreasing: the maximum is reached at the 4th cycle of the second step when the F1-score is equal to 73.9 %.

### 3.5.3 Performance evaluation

The accuracy, F1-score and balanced accuracy for training, validation and test sets, before and after the search, are presented in figure 3.20.

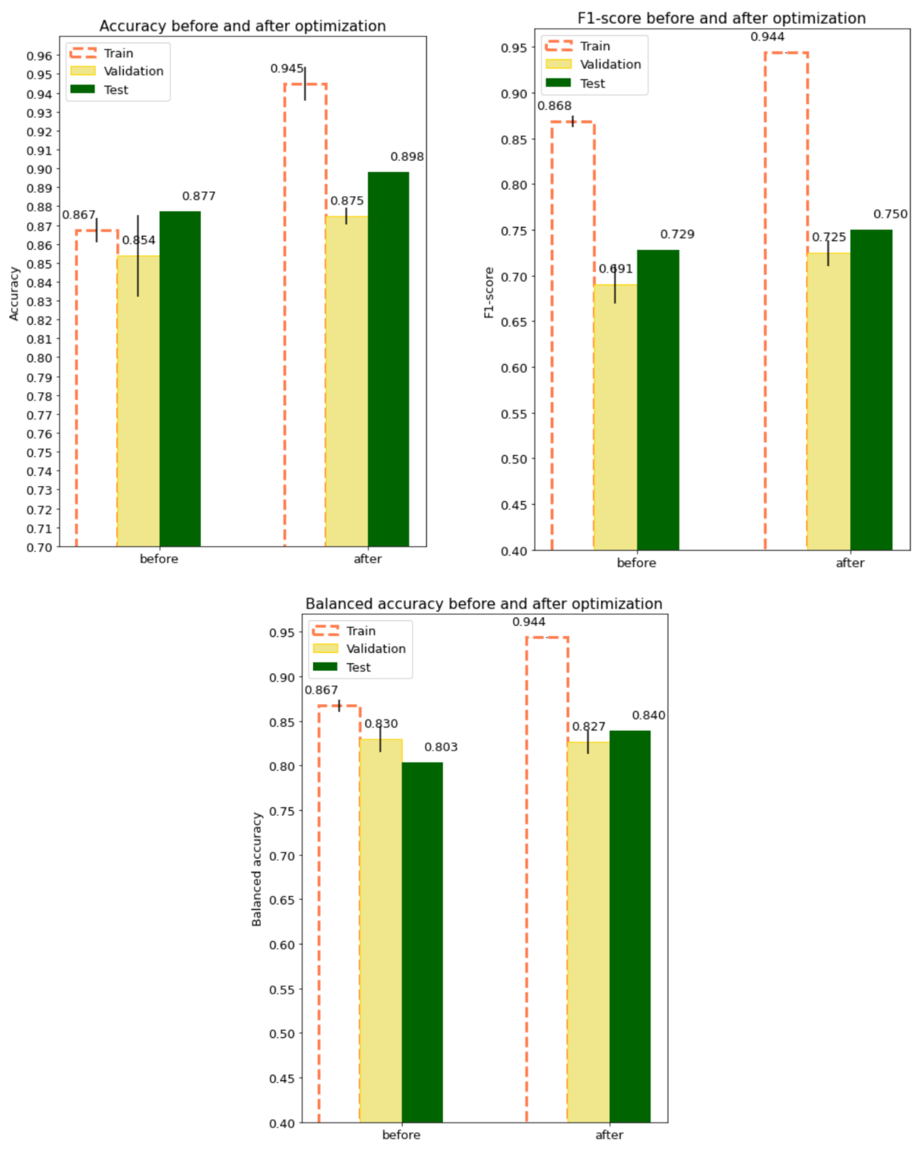


*Figure 3.20: EEGNet performances before and after the hyperparameters optimization for training, validation and test sets. All the three metrics are reported: accuracy (top left), F1-score (top right) and balanced accuracy (bottom).*

The performances are improved for the training and test sets for all the three metrics. Validation set results are improved if considering accuracy and F1-score but not in terms of balanced accuracy, in which a reduction is observed.

By looking at the confusion matrices (table 3.5), it is possible to observe that the number of False Positives is reduced after the optimization, while the number of False Negatives increases.

| *Before optimization* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 212 | 112 |
| **0** | 46 | 918 |

| *After optimization* | | |
|---|---|---|
| | **ErrP** | **non-ErrP** |
| **1** | 196 | 69 |
| **0** | 62 | 961 |

Table 3.5: Confusion matrices of the classification before and after the optimization

Utility gain function before and after the hyperparameters search is presented in figure 3.21.

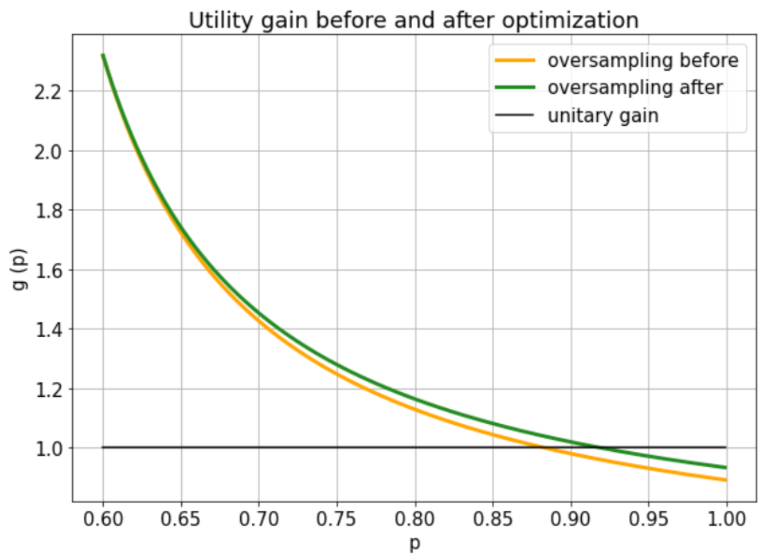


Figure 3.21: Utility gain function before and after the hyperparameters optimization.

A higher gain can be noticed in the function after the optimization, in particular when the BCI classifier accuracy $p$ is greater than 70 %. The network before the optimization provides an improvement in the performances of the system if $p$ is less than $\sim$ 87 %, while after the optimization p can have $\sim$ 91 % of accuracy.

# Chapter 4

# Discussion

## 4.1 Inter-subjects analysis

The results in the inter-subjects analysis reveal that the main problem to address by balancing the training set is the loss of generalization of the model. As discussed before, ERP classification is challenging mostly because of the scarcity of representation of the events: in the dataset used the imbalance between ErrP and non-ErrP epochs is of 3793 samples, being the ErrP epochs just 1322.

With undersampling a loss of information is verified: the exclusion of such a high number of instances leads to a significant reduction of the size of the dataset. As a matter of fact, the performances with this approach are the lowest considering all the metrics.

With oversampling the results are better but the difference in performances between training and test set is still high: a gap of $\sim 14$ % is present by considering the F1-score. It can be an evidence of the arise of overfitting: the ErrP epochs are added multiple times to address the high imbalance so that the performances on training are high but the model is not versatile in the classification of independent data.

Class weights performances are similar to the oversampling ones. However, it can be noticed that the gap between training and test performances is reduced: it confirms the fact that with oversampling loss of generalization arises. As a matter of fact, the score for oversampling in training, especially considering the F1-score, is much higher than class weights one, even if performances on test set are comparable. Class weights approach perform well on the test set but it presents the lowest performances in the training set: it highlights the fact that there is a room of improvement for this method and that the assignment of different weights can influence the results. As discussed before, the choice of class weights must be executed by experts in order to not incur in biased performances.

On the other hand, the introduced ARX-based methods result in the highest performances in test. An improvement of $\sim 5$ % for the F1-score can be observed for the ARX amplitude and ARX amplitude+ noise and of $\sim 7$ % in terms of balanced accuracy for the ARX amplitude+ warping method respect to oversampling.

An exception can be observed in the ARX warping technique: it results in a very low value in terms of F1-score while the highest value for the balanced accuracy. It can be explained by the fact that with this technique the highest number of True Negatives is reached, thus improving the specificity included in the balanced accuracy definition, but the lowest number of True Positives is also obtained, thus affecting the F1-score value.

The improvements in terms of True Negatives can be noticed for all the ARX techniques: the False Positives number is halved respect to the traditional methods. As highlighted in [69], it is crucial for an error-correction system to have the lowest possible value for the False Positives: the rejection of correct

outputs generate frustration for the user, lowering motivation and leading to worse performances in the successive trials.
Another improvement introduced by the ARX-based methods is the reduction of the gap occurring between training and test sets performances: the difference for the ARX amplitude method is of 9.32 % in terms of F1-score and of $\sim$ 0.89 % for the balanced accuracy. It is possible to conclude that these approaches have a higher capability of generalization.
Overall, the ARX-based methods which result to be more successful are the ARX amplitude method and the combined ARX techniques. The scores for the ARX amplitude are equal to 91.23 % for accuracy, 77.88 % for F1-score and 86.48 % for the balanced accuracy.

## 4.2 Intra-subject analysis

With the intra-subject analysis the differences between subjects data have been highlighted.
The first group (Subjects 1, 2 and 5) results confirm what it has just been discussed: the ARX methods result in the highest performances, decreasing the gap between training and test sets scores and reducing the number of False Positives. The scores, however, show high intra-subjects variability: the standard deviations for the averaged results reach $\sim$ 13 % for the ARX amplitude in the F1-score.
The results for the second group (Subjects 3, 4 and 6) are low. F1-score reaches a minimum of 53.84 % for the Subject 6 for the ARX amplitude+noise technique. These results can be related to the fact that Subject 4 and Subject 6 data are quite anomalous respect to the others: as reported in the

graphical inspection of the data in Chapter 3, the grand average ErrP curves obtained for these two subjects are very different from the others showing just a positive peak with a different latency. It has to be noticed that in this case traditional approaches perform better: it is made also evident by the fact that in the confusion matrices for the ARX-based methods the number of False Negatives is the half of the total number of ErrP epochs, thus half of the instances are misclassified. This is not observed for the class weights approach. A possible explanation is reported in figure 4.1, where the averaged ErrP epochs and the averaged non-ErrP ones are shown for Subject 1 and Subject 6.

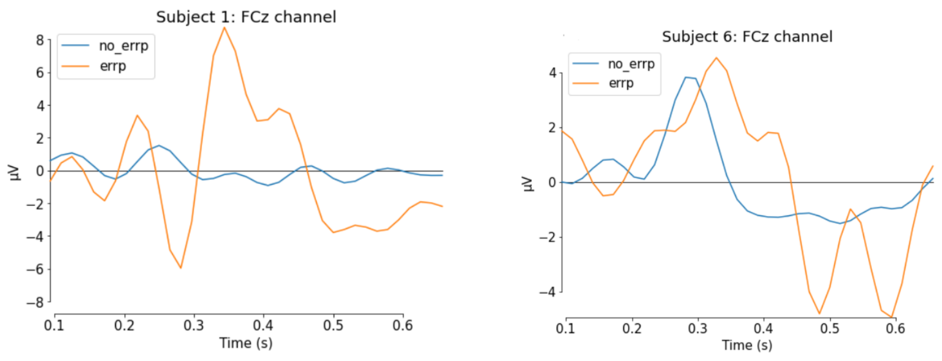


*Figure 4.1: Averaged ErrP and non-ErrP epochs for Subject 1 (left) and Subject 6 (right) for FCz channel.*

It can be noticed that there's a high difference in amplitude between ErrP epochs and non-ErrP epochs for Subject 1. This difference is not highlighted for Subject 6. In particular, before $\sim 0.35s$, the two epochs are on the same amplitude range, with very close peaks. It may be possible that through ARX modelling and then signal generation process, in which the distortions cause changes in amplitude, the difference between events and non-events has become less clear, leading to a higher misclassification rate.

However, in the intra-subject analysis, at least for the first group of subjects, comparable and higher results respect to the inter-subjects analysis are obtained: for Subject 2, the one resulting in the highest performances, the classification scores are equal to $94,91$ % for accuracy, 87.91 % for F1-score and 91.10 % for balanced accuracy for the ARX mix technique. It is possible to infer that EEGNet is able to perform well even with a small dataset.

## 4.3 Data classification strategy

From the reported results some considerations may arise concerning the classification strategy applied.
In the inter-subjects analysis shuffling data before the partition in training and test set is necessary to not affect just a set out of three with anomalous data: a model trained only with Subjects 1, 2 and 5 data would have not classified well Subjects 3,4 and 6 data.
The majority vote approach, as made clear in the plots in the inter-subjects analysis, outperforms the averaged score strategy for the test set. The scores for the test set obtained with the majority vote approach are higher than the ones obtained by averaging the results of each iteration of the cross-validation process and are also higher than the validation set ones: the network benefits from this technique in the evaluation of the test performances.
The accuracy, as it was already expected, it is not a good metric to deal with imbalanced datasets: the scores obtained in the inter-subjects analysis are all similar making the difference between the performances of the balancing methods not clear.

## 4.4 Hyperparameters search

The hyperparameters search highlights that the hyperparameters of a CNN affect the performances of the network. The improvement obtained on the test set by optimizing the hyperparameters configuration is of 2.1 % for accuracy and F1-score and of 3.7 % for the balanced accuracy.
The F1-score on the validation set does not monotonically increase during the cycles: it is due to the fact that, even with the same configuration, the performances of the network could vary by repeating the training.
The resulting configuration highlights that a differentiation between Block 1 and Block 2, in terms of pooling, dropout and activation layers, can improve the performances of the network. The increment obtained in the D, F1, F2 values, i.e. the number of filters of the convolution layers, suggests that a bigger network can perform better. The Utility gain function is improved by the optimization: a higher gain can be reached by optimally fitting the hyperparameters to the dataset used.
Some considerations concerning the hyperparameters search have to be done: the search was not performed by using the ARX-based methods to balance the training set and, as discussed so far, the balancing method chosen affects importantly the network performances.
The search, as it was implemented, is very time-consuming: grid search tests all the possible combinations between the hyperparameters increasing exponentially the time required to execute a cycle of search if a high number of hyperparameters is tested together.
The resulting network has 5365 trainable parameters compared to the initial 1809 ones: it has to be clarified if this increment lowers the performances if a small dataset is used.

## 4.5 Comparison with literature results

The performances of the proposed EEGNet with the ARX-based methods for data augmentation are now compared with the literature. The studies presented in the *State of the art*, except for EEGNet, refer to the same dataset used in this thesis: hence, the results are directly comparable. The metric used in all the papers is the accuracy: for this reason, in this thesis, the scores have also been presented in these terms.
In table 4.1 the performances on the test set are reported both in terms of intra-subject and inter-subjects analysis. It has been chosen to compare the literature results with the ARX-based approaches both with a single and combined techniques: ARX amplitude and ARX mix are selected.

| | **deep ConvNets[47]** | **DCNN with GAN [38]** | **EEGNet+ARX amp** | **EEGNet+ARX mix** |
|---|---|---|---|---|
| **s1** | 86.49% ± 1.13% | 86.90 % | 89.47% | 89.00 % |
| **s2** | 81.45 % ± 1.32 % | 92.11 % | 93.98 % | 94.91 % |
| **s3** | 79.71% ± 0.30% | 91.41 % | 87.50 % | 89.90 % |
| **s4** | 80.87% ± 2.41% | 82.14 % | 85.78 % | 86.73 % |
| **s5** | 74.76 % ± 3.84 % | 86.90 % | 84.49 % | 89.33 % |
| **s6** | 78.61 % ± 1.15 % | 89.41 % | 80.87 % | 85.07 % |
| **inter subj** | 79.94 % ± 2.08 % | 87.94 % | 91.23 % | 90.84 % |

*Table 4.1: Accuracy for each subject and for inter-subjects analysis compared with literature studies [47] and [38]. The ARX amplitude and ARX mix results are reported in the light blue columns.*

The results of deep ConvNets are reported in terms of mean and standard deviation, since for each analysis two scores are presented. In the intra-subject analysis the performance of the network are evaluated at first by using session 1 data for test and then session 2 data. In the inter-subjects analysis the performances are first tested by using Subjects 1, 2, 3 data for test and then Subjects 4, 5, 6 data. The results highlight that both ARX amplitude and ARX mix outperform deep ConvNets in all the cases. In the inter-subjects analysis the improvement is of $\sim 10$ %.

In [38] a modification of EEGNet is presented and a GAN data augmentation method is applied: the performances are more comparable with the ones reported in this thesis. The ARX-based methods have higher scores both in the intra-subject and in the inter-subjects analysis: the ARX amplitude technique leads to an improvement of 3.29 % and the ARX mix of 2.90 % in terms of inter-subjects performances. In the intra-subject analysis just the results of Subjects 3 and 6 have higher results with the GAN approach: this is coherent with the discussion presented earlier about the second group's performances.

Eventually, the results are not directly comparable with the paper in which EEGNet is defined [49], since a different dataset was used. It has been chosen to not use that dataset in this thesis since a first analysis on it showed that the grand average ErrP is quite different from the typical characteristics described in Error Potential studies. The curve obtained for the EEGNet dataset for the FCz channel is presented in figure 4.2.

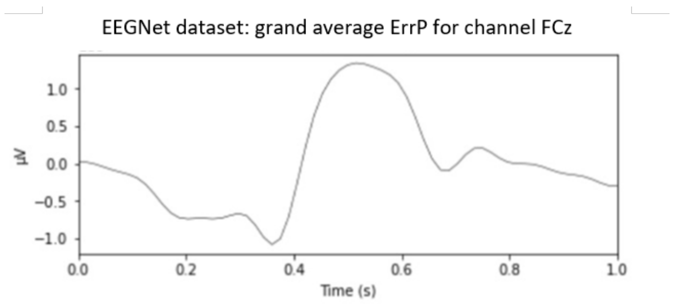


*Figure 4.2: Grand average ErrP curve for channel FCz obtained for the dataset used in EEGNet*

A much smaller amplitude range can be noticed: the signal is in the $[1, -1]$ $\mu V$ range. A much wider unique positive peak is present at $\sim 0.55s$ and also the negative peak presents a different latency.

# Chapter 5

# Conclusions

In this study a CNN for Error Potential classification has been validated: starting from EEGNet, an architecture present in literature, the work has been focused on its optimization. Two directions are followed: a novel data-augmentation technique and a hyperparameter search.
Modelling EEG signals which include event-related potentials trough ARX models was already demonstrated to be successful in literature [64], [65] and in this thesis the results have shown that using them to generate new data is a valid strategy. The signals obtained don't lose the information contained in the ErrP: the grand average curves obtained with the ARX new signals present the same stereotypical shape and latencies of the original data.
The performances obtained by the ARX-based data augmentation methods proposed are always higher than the traditional methods used for balancing (oversampling, undersampling and class weights): a higher capability of generalization is highlighted, reducing the difference in the performances between training and test sets. Moreover, a reduction of the False Positives is observed. This is crucial for the application of an error-correction system in a BCI device: it could improve the user's motivation, allowing to perform

the desired action with less incorrect rejections.
The Utility gain functions reported show that the ARX methods would be able to contribute to the overall performances of a BCI system with a higher gain respect to the other balancing methods. The gain in the performances would be relevant for BCI devices that have a classification accuracy at its maximum of $\sim 96$ %.
Within the presented ARX methods there is not one that outperforms the others. In the inter-subjects analysis the ARX amplitude and the ARX combined techniques show better results and a lower difference between training and test scores while in the intra-subject analysis, by considering just the group of subjects with good results, also ARX noise shows comparable results.
The exception within these methods is the ARX warping: the grand average curve does not present the same characteristics of the original one and its performances, especially by looking at the F1-score, are lower than the other methods. Even if it provides the lowest number of False Positives, the False Negatives value show a high increase if compared with the others methods.
Worse performances can be noticed in the classification of those data which are anomalous respect to the others: for Subject 3, 4 and 6 the traditional balancing methods, especially class weights, result in higher scores.
By the comparison made with the literature studies taken in account in this thesis, it has been observed that the presented EEGNet with the ARX-based data augmentation methods outperform the other strategies.
Eventually, an evaluation on the hyperparameters relevance has been carried out by implementing a search strategy: the results show that the performances of the network can be improved by identifying the optimal hyperparameters configuration. This search has been done in parallel with the

implementation of the ARX methods, so that it was not computed specifically for these approaches. The main changes highlighted by the results obtained are the differentiation of pooling, dropout and activation layers between the two blocks of the architecture and the increase of the number of filters for the convolution layers.

## 5.1 Future works

There is room for improvement for the work so far presented: the main future developments are now presented.
Regarding the CNN architecture:

- **Evaluate EEGNet blocks**: an analysis on the single EEGNet blocks may be carried out to identify which are the layers mainly contributing in the performances. In particular, it could be verified if, by further differentiating the characteristics of Block 1 and Block 2, the performances can be improved, as it was suggested by the hyperparameters search results.
- **Implement a novel CNN**: starting from EEGNet, a novel CNN could be designed by adding new types of layers or, in case, making it deeper by increasing the number of blocks. In general, a personalized architecture could be thought, based on the evaluation about each layer features and its degree of influence in the performances.
- **Improve hyperparameters search**: grid search was performed in this thesis and it has been proven to be very time consuming. Random search does not ensure to find the optimal configuration so it is not suggested in the optimization of important hyperparameters. Other strate-

gies for hyperparameters search can be tried: for instance, Bayesian optimization could improve time requirements ensuring to find the optimal configuration. Eventually, hyperparameters search can be carried out on the network specifically balanced with an ARX-based method.

- **Improve time cost**: training a network is time consuming and a long training session for the user can be stressful. Methods to speed up the training can be hypothesized: for instance, it can be evaluated if the use of just FCz and Cz channels (the electrodes in which the ErrP activity in mainly localized) results anyway in high performances. The dataset size would be reduced, resulting in a faster training session.

The ARX-based data augmentation method is novel so a lot of research for its optimization can be done:

- **Evaluate new type of distortions**: change of amplitude, noise addition and warping have been evaluated. Warping did not show comparable results respect the others but presented the lowest number of False Positives: it could be tried to evaluate if a smaller range of shrinking can improve the results. New types of distortions can be thought: it has to be noticed that, in the first stages, also a shift in time of the exogenous input was tried and it did not perform well.
- **Compare ARX-methods with other data augmentation techniques**: in this thesis ARX was compared with resampling techniques and class weights assignment. Its performances could be compared with other data augmentation approaches to further validate it. The implementation of a GAN could be thought.

Eventually, the proposed EEGNet with the ARX-based data augmentation method can be tested experimentally: an ErrP-specific experiment could be set up and the performances of the network can be evaluated directly in an online application.

# Bibliography

[1] S. Machado, F. Araiijo, F. Paes, B. Velasques , M. Cunha, H. Budde, L. Basile, R. Anghinah, O. Arias-Carrion, M. Cagy, R. Piedade, T. A. de Graaf, A. T. Sack, and P. Ribeiro, "EEG-based Brain-Computer Interfaces: An Overview of Basic Concepts and Clinical Applications in Neurorehabilitation," *Reviews in the neurosciences*, vol. 21, no. 6, pp. 451–468, 2010.

[2] M. O. Krucoff, a. W. S. Shervin Rahimpour, V. R. Edgerton, and D. A. Turner, "Enhancing nervous system recovery through neurobiologics, neural interface training, and neurorehabilitation," *frontiers in Neuroscience*, vol. 10, 2016.

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

[4] F. Martini, M. Timmons, and R. Tallitsch, *Anatomia umana*, vi ed. EdiSES, 2016.

[5] M. Strotzer, "One Century of Brain Mapping Using Brodmann Areas," *Clinical Neuroradiology*, vol. 19, pp. 179–186, 2009.

[6] C. Holroyd and M. Coles, “The neural basis of human error processing: Reinforcement learning, dopamine and the error-related negativity,” *Psychological Review*, vol. 109, no. 4, pp. 679– 709, 2002.

[7] W. Miltner, C. Braun, and M. Coles, “Event-related brain potentials following incorrect feedback in a time-estimation task: Evidence for a 'generic' neural system for error detection.” *Journal of Cognitive Neuroscience*, vol. 9, pp. 788– 798, 1997.

[8] S. Taylor, E. Stern, and W. Gehring, “Neural systems for error monitoring: recent findings and theoretical perspectives,” *Neuroscientist*, vol. 13, pp. 160–172, 2007.

[9] C. Carter, T. Braver, D. Brach, M. Botvinick, D. Noll, and J. Cohen, “Anterior cingulate cortex, error detection, and the online monitoring of performance.” *Science*, vol. 280, pp. 747–749, 1998.

[10] A. Baddeley, *Working memory.* New York : Oxford University Press, 1986.

[11] F. L. da Silva, *EEG-fMRI: physiological basis, technique, and applications.* Berlin: Springer, 2010.

[12] S. Baillet, “Magnetoencephalography for brain electrophysiology and imaging.” *Nature neuroscience*, vol. 20, no. 3, pp. 327–339, 2017.

[13] H. Jasper, “The ten-twenty electrode system of the international federation,” *Electroencephalography and Clinical Neurophysiology*, pp. 371–375, 1958.

[14] C. M. Michel and D. Brunet, “EEG Source Imaging: A Practical Review of the Analysis Steps,” *Frontiers in Neurology*, vol. 10, p. 325, 2019.

[15] M. Coles and M. Rugg, *Electrophysiology of mind: Event-related brain potentials and cognition.* Oxford University Press, 1995.

[16] S. J. Luck, *An Introduction to the Event-Related Potential Technique*. The MIT Press., 2005.

[17] C. Wirth, P. Dockree, S. Harty, E. Lacey, and M. Arvaneh, "Towards error categorisation in bci: single-trial eeg classification between different errors." *Journal of neural engineering*, vol. 17, no. 3, 2019.

[18] M. Falkenstein, J. Hohnsbein, and J. Hoormann, "Effects of errors in choice reaction tasks on the ERP under focused and divided attention," *Psychophysiological Brain Research*, pp. 192–195, 1990.

[19] W. Gehring, M. Coles, D. Meyer, and E. Donchin, "The error-related negativity: an event-related brain potential accompanying errors," *Psychophysiology*, vol. 27, 1990.

[20] S. Kim, E. Kirchner, A. Stefes, and F. Kirchner, "Intrinsic interactive reinforcement learning - using error-related potentials for real world human-robot interaction." *Scientific Reports*, vol. 7, no. 1, 2017.

[21] P. Ferrez and J. del R. Millan, "Error-related eeg potentials generated during simulated brain-computer interaction. ieee transactions on bio-medical engineering," *IEEE transactions on bio-medical engineering*, vol. 55, no. 3, pp. 923–929, 2008.

[22] M. Spuler and C. Niethammer, "Error-related potentials during continuous feedback: using eeg to detect errors of different type and severity," *Human Neuroscience*, vol. 9, no. 155, 2015.

[23] J. J. Shih, D. J. Krusienski, and J. R. Wolpaw, “Brain-computer interfaces in medicine.” *Mayo Clinic Proceedings*, vol. 87, no. 3, pp. 268– 279, 2012.

[24] P. W. Ferrez and J. del R. Millan, “You are wrong!: automatic detection of interaction errors from brain waves,” *IJCAI’05: Proceedings of the 19th international joint conference on Artificial intelligence*, pp. 1413–1418, 2005.

[25] X. Artusi, I. K. Niazi, M.-F. Lucas, and D. Farina, “Performance of a simulated adaptive bci based on experimental classification of movement-related and error potentials,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 4, pp. 480–488, 2012.

[26] N. Schmidt, B. Blankertz, and M.S.Treder, “Online detection of error-related potentials boosts the performance of mental typewriters.” *BMC Neuroscience*, vol. 13, no. 19, 2012.

[27] R. Chavarriaga, A. Sobolewski, and J. del R. Millan, “Errare machinale est: the use of error-related potentials in brain-machine interfaces,” *Frontiers in Neuroscience*, vol. 8, 2014.

[28] A. Buttfield, P. W. Ferrez, and J. del R. Millan, “Towards a robust bci: Error potentials and online learning,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, 2006.

[29] A. Llera, M. van Gerven, V. Gomez, O. Jensen, and H. Kappen, “On the use of interaction error potentials for adaptive brain computer interfaces,” *Neural Networks*, vol. 24, pp. 1120–1127, 2011.

[30] R. Chavarriaga and J. del R. Millan, “Learning from eeg error-related potentials in non invasive brain-computer interfaces,” *IEEE Transac-*

*tions on Neural Systems and Rehabilitation Engineering*, vol. 18, pp. 381–388, 2010.

[31] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 436, 2015.

[32] Y. Roy, H. Banville, I. Albuquerque, A. Gramfort, T. H. Falk, and J. Faubert, “Deep learning-basedel ectroencephalography analysis: a systematic review,” *Journal of Neural Engineering*, vol. 16, 2019.

[33] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” *2017 International Conference on Engineering and Technology (ICET)*, 2017.

[34] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, “Learning from class-imbalanced data: Review of methods and applications,” *Expert Systems with Applications*, vol. 73, pp. 220–239, 2017.

[35] J. Leevy, T. Khoshgoftaar, and R. Bauder, “A survey on addressing high-class imbalance in big data.” *Journal of Big Data*, vol. 5, no. 42, 2018.

[36] Y. Roy, H. Banville, I. Albuquerque, A. Gramfort, T. Falk, and J. F. J., “Deep learning-based electroencephalography analysis: a systematic review.” *Journal of Neural Engineering*, vol. 16, no. 5, 2019.

[37] F. Wang, S. Zhong, J. Peng, J. Jiang, and Y. Liu, “Data augmentation for eeg-based emotion recognition with deep convolutional neural networks.” *Lecture Notes in Computer Science*, vol. 10705, 2018.

[38] C. Gao, Z. Li, H. Ora, and Y. Miyake, “Improving error related potential classification by using generative adversarial networks and deep

convolutional neural networks," *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2020.

[39] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball, "Deep learning with convolutional neural networks for EEG decoding and visualization," *Human Brain Mapping*, vol. 38, no. 11, p. 5391âĂŞ5420, 2017.

[40] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, no. null, p. 281âĂŞ305, 2012.

[41] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, pp. 2951–2959, 2012.

[42] F. Kalaganis, E. Chatzilari, K. Georgiadis, S. Nikolopoulos, N. Laskaris, and Y. Kompatsiaris, "An error aware ssvep-based bci," *2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 775–780, 2017.

[43] Y. Zhang, W. Chen, J. Zhang, and J. Wang, "Extracting error-related potentials from motion imagination eeg in noninvasive brain-computer interface," *2017 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pp. 768–773, 2017.

[44] S. Kim, D. Kim, and L. Kim, "Optimization method of error-related potentials to improve MI-BCI performance," *2019 7th International Winter Conference on Brain-Computer Interface (BCI)*, pp. 1–5, 2019.

[45] A. Cruz, G. Pires, and U.J.Nunes, "Double ErrP detection for automatic error correction in an ERP-based BCI speller." *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society*, vol. 26, no. 1, pp. 26–36, 2018.

[46] S. Kim, E. Kirchner, and A.Stefes, "Intrinsic interactive reinforcement learning- using error-related potentials for real world human-robot interaction." *Scientific reports*, vol. 7, 2017.

[47] S. A. Swamy Bellary and J. M. Conrad, "Classification of error related potentials using convolutional neural networks," *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pp. 245–249, 2019.

[48] J. M. M. Torres, T. Clarkson, E. A. Stepanov, C. C. Luhmann, M. D. Lerner, and G. Riccardi, "Enhanced error decoding from error-related potentials using convolutional neural networks," *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 360–363, 2018.

[49] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "EEGNet: a compact convolutional neural network for EEG-based brain-computer interfaces," *Journal of Neural Engineering*, vol. 15, no. 5, 2018.

[50] B. Dal Seno, M. Matteucci, and L. T. Mainardi, "The Utility Metric: A Novel Method to Assess the Overall Performance of Discrete Brain-Computer Interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 1, pp. 20–28, 2010.

[51] [Online]. Available: http://bnci-horizon-2020.eu/database/data-sets

[52] A. Gramfort, M. Luessi, E. Larson, D. Engemann, D. Strohmeier, C. Brodbeck, R. Goj, M. Jas, T. Brooks, L. Parkkonen, and M. Hämäläinen, "MEG and EEG data analysis with MNE-Python," *Frontiers in Neuroscience*, vol. 7, 2013.

[53] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'16. USA: USENIX Association, 2016, pp. 265–283.

[54] F. Chollet. (2015) Keras. [Online]. Available: https://github.com/fchollet/keras

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[56] Eegnet's code. [Online]. Available: https://github.com/vlawhern/arl-eegmodels

[57] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" 2019.

[58] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2017.

[59] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation Functions: Comparison of trends in Practice and Research for Deep Learning," *ArXiv*, vol. abs/1811.03378, 2018.

[60] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 648–656.

[61] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[62] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning.* Springer, New York, NY, 2009.

[63] C. Orrite, M. Rodríguez, Francisco, Martínez, and M. Michael Fairhurst, "Classifier ensemble generation for the majority vote rule," *Progress in Pattern Recognition, Image Analysis and Applications*, pp. 340–347, 2008.

[64] L. T. Mainardi, J. Kupila, K. Nieminen, I. Korhonen, A. M. Bianchi, L. Pattini, J. Takala, J. Karhu, and S. Cerutti, "Single sweep analysis of event related auditory potentials for the monitoring of sedation in cardiac surgery patients," *Computer Methods and Programs in Biomedicine*, vol. 63, no. 3, pp. 219–227, 2000.

[65] L. Rossi, A. Bianchi, A. Merzagora, A. Gaggiani, S. Cerutti, and F. Bracchi, "Single trial somatosensory evoked potential extraction with arx fil-

tering for a combined spinal cord intraoperative neuromonitoring technique." *Biomedical engineering online*, vol. 6, no. 2, 2007.

[66] M. A. Stephens, "Edf statistics for goodness of fit and some comparisons," *Journal of the American Statistical Association*, vol. 69, no. 347, pp. 730–737, 1974.

[67] Y. Bengio, *Practical Recommendations for Gradient-Based Training of Deep Architectures.* Springer Berlin Heidelberg, 2012, pp. 437–478.

[68] K. You, M. Long, J. Wang, and M. I. Jordan, "How Does Learning Rate Decay Help Modern Neural Networks?" 2019.

[69] M. Spuler, M. Bensch, S. Kleih, W. Rosenstiel, M. Bogdan, and A. Kubler, "Online use of error-related potentials in healthy users and people with severe motor impairment increases performance of a P300-BCI." *Clinical Neurophysiology*, vol. 123, no. 7, 2012.