POLITECNICO DI TORINO

MASTER OF SCIENCE COURSE

BIOMEDICAL ENGINEERING

Analyzing movement patterns to facilitate the titration of medications in late stage Parkinson's disease

Supervisors
Prof. Danilo DEMARCHI
Prof. Paolo BONATO

Candidate
Riccardo DI DIO

July, 2019

Academic Year 2018-2019

Abstract

Parkinson's disease is the second most common neurodegenerative disorder after Alzheimer. Nowadays more than 10 million people worldwide are affected and this number is increasing at a rate of 60 thousands new diagnoses per year in US only. It has an impact on the US healthcare industry of 23 billion dollars per year represented by direct and indirect costs.
Periodically quantify the severity of the symptoms is important to arrange the medication doses and schedule the intake times to avoid or keep at minimum the side effects of the medications and keeping always a low level of impairment caused by the symptoms. This is a problem of drug titration, typical of those drugs that have a wearing off effect, like Levodopa, used for Parkinson's Disease.
Today to quantify the severity of motor symptoms, simulated tasks are performed by patients under the observation of clinicians who patiently assigns scores to each task. These procedures take a lot of time to both patients and clinicians and represent an important part of the total cost upon the healthcare industry.
Wearable sensors and modern techniques of data analysis and Artificial Intelligence can help in this task by predicting the severity of motor and non motor symptoms, allowing for continuous monitoring, objective analysis and saving time and money to patients and hospitals. The Motion Analysis Laboratory (Harvard Medical School) collected data from 27 patients with Parkinson's Disease, during performed tasks in the laboratory environment and simulated activity of daily life (SADL) in an apartment-like environment.
This thesis is part of a bigger project where different hospitals and universities in Boston, MA are involved called BlueSky project. Aim of this project is to use all the information from all different kind of sensors (accelerometric and physiological data) to accurately predict changes in the severity of Parkinson for both motor and non-motor symptoms.
The main focus of the thesis is on the analysis of accelerometric data acquired from wrists and feet. However the first months have been spent analyzing other physiological data such as Galvanic Skin Response and Heart Rate Variability for the study of non motor symptoms.

Finally, a machine learning approach has been used. Different regressor models have been tested, the focus has been on the study of temporal patterns through Recurrent Neural Networks, the overall error is reported in terms of Root Mean Squared Error as used in regression problems, however the characteristics of the classifier are derived

from the regressor to a better comparison among different models. The best results obtained are using a LSTM (Long-Short Term Memory) network on the output of the RF (Random Forest) leading to an overall accuracy of 81.4% (4 classes) and RMSE of 0.55. However this approach is better only in the laboratory dataset, while in the apartment dataset the only Random Forest has better performances.

Acknowledgements

First of all I want to thank my family for the support given during these years and above all for having made possible my Erasmus experience in Madrid and my thesis development in Boston.

A super special thank has to be given to my grandparents, who really helped me and sustained me in all my choices. A huge thank also to my uncles Mimmo, Franco and Claudio for their support. This experience wouldn't be possible without the help of my parents, I love you.

My master degree has mainly been developed abroad, 6 months in Madrid and 8 months to Boston. I'm deeply grateful to my family who made these experiences possible. I met so many people and made new friends worldwide and this is way more important of any degree or honor. I really want to thank all the friends who made these experiences ***AWESOME****.*

Particularly I really want to thank: Alex, Davide, Cristina, Michela, Paul, Salvo, Rebecca, Paulina, Jacob, Mauro, Christina, Belén, Lester, Alejandro, Giselle, Natalie and then Christina, Sofia, Sara, Nico, Francesco, Alejandro, Stefano, Marco and all the guys of the Motion Analysis Lab.

A SUPER special thanks to my American family, Jerry, Christina, Melissa and Rosaria and Jerry. A thank is a must for all what you did for me. You'll always have a family in Italy.

Another thanks to all my Friends of a life. Thanks for your support and jokes and good moments, I'm glad for all the beers taken with you guys.

Thanks to Giulia, who has made everything easier and worthier with her good and funny character. I'm glad to having met you in that pub and I'm glad to Peppe and Marco for having me brought (I love you too guys).

Thanks to my personal supervisors: Stefano Sapienza and Federico Parisi, I don't want to thank you for the work done with me if not for the good moments and memories shared. You are amazing people, good luck!

Finally I want to dedicate this work to the future, mine and of the others. Wherever it takes I hope to meet other people as interesting as everyone.

Thank you.

Abbreviations

AD Autosomal Dominant

ADASYN Adaptive Synthetic Sampling Approach for Imbalanced Learning

AI Artificial Intelligence

ANN Artificial Neural Network

ANS Autonomic Nervous System

AR Autosomal Recessive

BiGRU Bidirectional Gate Recurrent Unit

BiLSTM Bidirectional Long Short Term Memory

CM Confusion Matrix

CNN Convolutional Neural Network

CNS Central Nervous System

DBS Deep Brain Stimulation

DSS Decision Support System

FoG Freezing of Gait

GRU Gate Recurrent Unit

GSR Galvanic Skin Response

HAR Human Activity Recognition

HRV Heart Rate Variability

LID Levodopa-Induced Dyskinesia

LSTM Long Short Term Memory

MAE Mean Absolute Error

NLP Natural Language Processing

NN Neural Network

PD Parkinson's Disease

PNS Peripheral Nervous System

RF Random Forest

RMSE Root Mean Squared Error

RNN Recurrent Neural Network

SMOTE Synthetic Minority Oversampling Technique

t-SNE t-Stochastic Neighbor Embedding

UPDRS Unified Parkinson's Disease Rating Scale

Contents

List of Figures

Preface

The aim of this chapter is mainly to list and discuss the works done during the almost 9 months spent in Boston at the Motion Analysis Lab, Department of Physical Medicine and Rehabilitation at Harvard Medical School and affiliated with the Harvard-MIT Division of Health Sciences and Technology, and the Wyss Institute for Biologically Inspired Engineering.

This chapter has strongly been suggested by my supervisor Paolo Bonato, director of the laboratory. This is because many works done during this internship weren't related to the work of my thesis.

The first 1 to 2 months have been spent mainly working on **HRV** signals, investigating the nature of non-linear dynamic signals and applying non-linear analysis on ECG using the Kubios GUI, Matlab® and deploying by myself other Matlab scripts. Particular attention has been given in the study of physiological effects of the Autonomic Nervous System in the HRV and the effect of Parkinson's Disease on it.

To follow, the attention moved to another physiological signal: the Electro-Dermal Activity (EDA) or Galvanic Skin Response (**GSR**). This signal took me more time and energies. The analysis started this time basically from the development of a protocol for data acquisition, then the protocol has been tested on me and other few subjects, the data has been acquired using the Shimmer sensor and elaborated from scratch. On the other hand I've also elaborated GSR signals already acquired by the lab, the problem was that these signals for logistic issues, have been acquired in the shoulder (a wrong body location for the GSR). However the analysis done highlighted even if in few subjects, how the number of fluctuations in the GSR changed between state ON and OFF of the disease. Nevertheless the signals were too corrupted and a new experiment with a new protocol was necessary.

Then I moved to the work explained in this thesis.

Of course during this experience I was surrounded by different engineers and physicians working on their own projects who often needed help even if only for data acquisition. The main experiences done in data acquisition regarded experiments with exoskeletons, different EMG acquisitions and the usage of the Vicon system for gait analysis.

I honestly want to thank professor Bonato and Demarchi for having done this experience possible and having enriched my curiosity on the scientific world.

Chapter 1

Introduction

1.1 Parkinson's disease

Parkinson's disease (PD) is a neurodegenerative disease caused by the progressive loss of dopaminergic pathways in an area of the brain going from the substantia nigra in the midbrain to the corpus striatum in the basal ganglia [17]. PD is the most common illness regarding motor symptoms and the second most common neuro disease after Alzheimer. Parkinson's disease has been firstly descripted by James Parkinson in 1817, nowadays 7 to 10 million people are affected with a prevalence of 41 per 100 000 in the fourth decade of life to over 1900 per 100 000 in people over 80 years old [12]. A recent study in US affirms that the prevalence of Parkinson's disease across North America will increase from 680 000 individuals older than 45 to 930 000 in 2020 and 1 238 000 in 2030. [14]. The incidence of PD usually increase with the age and tends to stabilize over 80 years of age. Around 4% of the diagnosis regards people younger than 50 years old.

1.1.1 Neuroanatomy overview

Since different times medical terms and biological concepts will be used, an overview of the basic is briefly reported in the introduction. The nervous system controls all the body activities, maintaining the homeostasis and responding to all different kind of stimuli, both internal and external. It can be anatomically divided in **Central Nervous System (CNS)** and **Peripheral Nervous System (PNS)**.

Peripheral Nervous System

The peripheral nervous system connects the CNS with the rest of the body and it's located in the spinal cord. Its functions are sensory and motors. The motor functions are the ones that interest us the most since many of the PD symptoms are related to these functions. The motor functions are provided by 2 main systems:

- **Somatic**: skeletal muscle effectors
- **Autonomic**: smooth muscles, involuntary activity of the respiration, cardiac activation and gland effectors

In Parkinon's Disease, both these functions are altered. Many studies report autonomic dysregulation focusing on signals like Heart Rate Variability (HRV) or Galvanic Skin Response (GSR). These type of signals (which have been part of my work during the first months at the Motion Analysis Lab) can be taken using wearable sensors in a non-invasive way.
The autonomic Nervous System (ANS) can still be divided in 2 antagonist systems:

- **Sympathetic Nervous System**: fight or flight response
- **Parasympathetic Nervous System**: rest and digest response

These 2 systems together regulate many physiological functions of our organism. For instance the heart rate and the sweating response are both controlled by the ANS. Particularly in the heart rate, the sympathetic have a positive chronotropic effect while the parasympathetic has a negative chronotropic effect. A positive chronotropic effect means increasing the heart rhythm, while a negative chronotropic effect decrease the rhythm. To show how PD affects also the ANN, many different studies conducted even in the Motion Analysis Lab have focused on the study of HRV. The first part of my work has actually been the study of HRV and GSR in subjects with PD to highlight the changes in state ON and OFF of the disease. It's now possible detect tremor in early stages of PD by looking at the only HRV [20] and predicting the FoG akynesia few seconds in advance by sensing the only GSR [27].

Central Nervous System

The central Nervous System is the orchestra conductor of our body. In here all the main tasks start and the various equilibrium of our body are kept under control.

The encephalon or brain is the organ which conduct everything. Anatomically it can be divided in:

- cerebrum
  - telencephalon
  - diencephalon
- brain stem
  - mesencephalon or midbrain
  - pons
  - medulla
- cerebellum

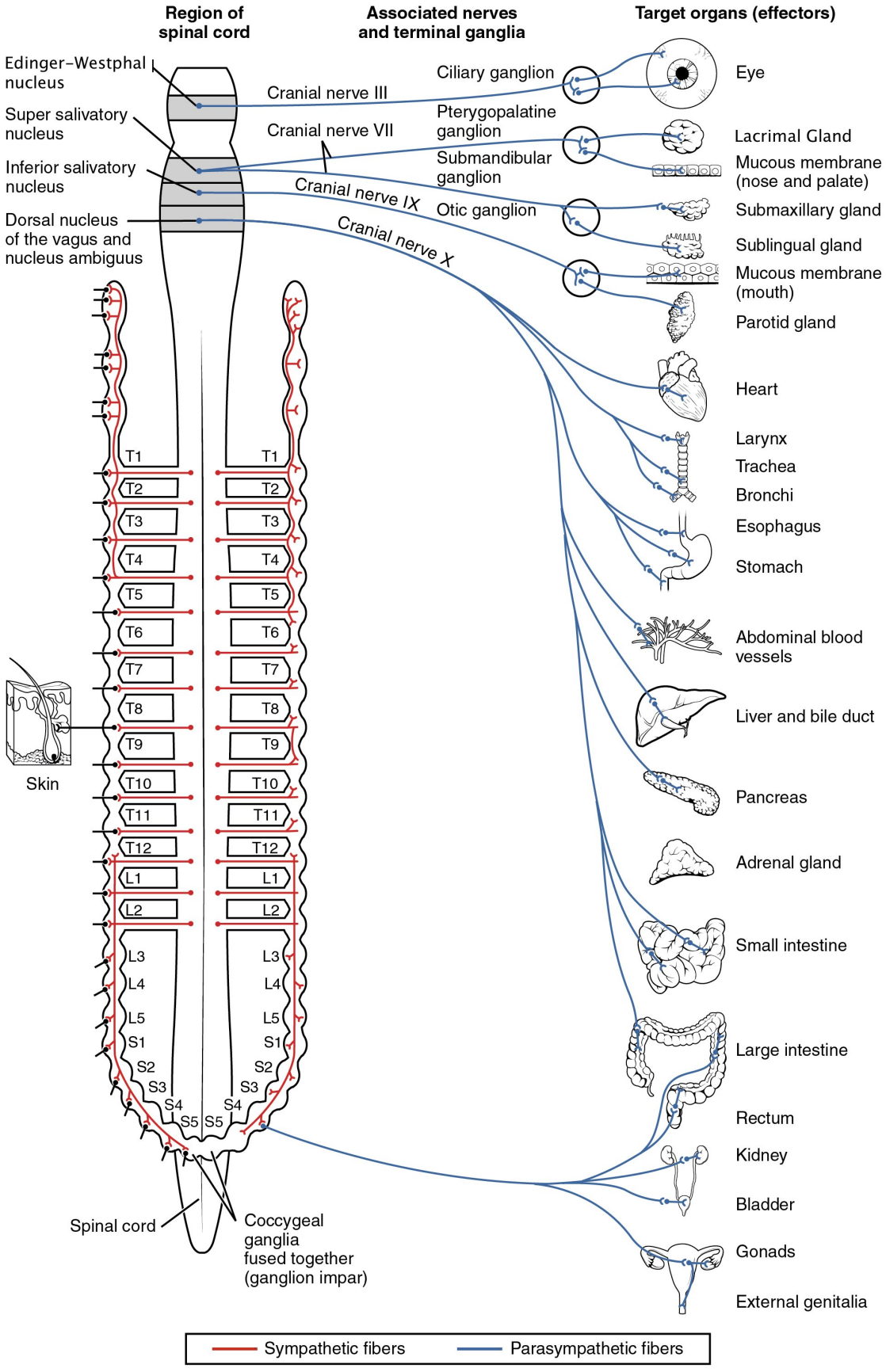


Figure 1.1: Autonomic Nervous System [1]

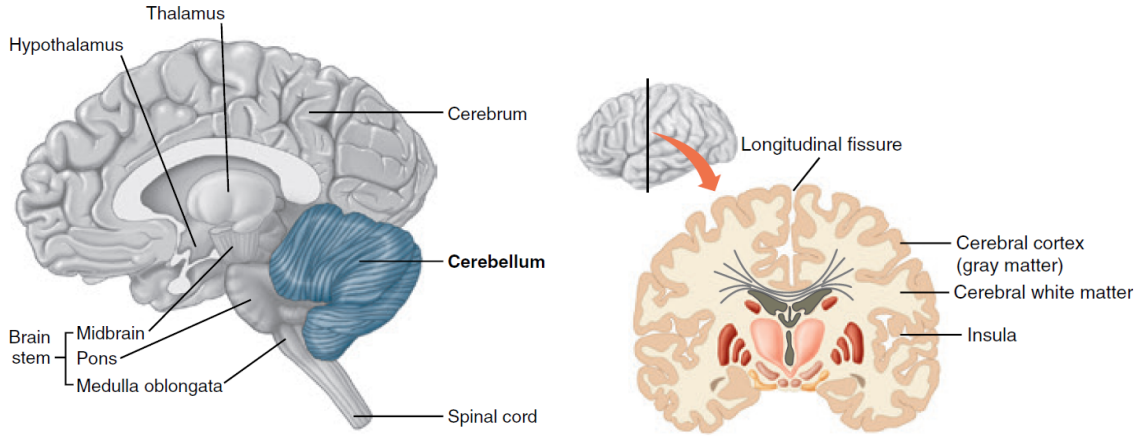


Figure 1.2: Brain structure, sagittal plane view (a) and longitudinal view (b) [19]

Basal Nuclei

The basal nuclei or basal ganglia are nuclei of gray matter that help in controlling the activity of skeletal muscles. When we refer to basal nuclei we are talking about 3 different nuclei which are:

- caudate nucleus
- globus pallidus
- putamen

These nuclei communicate with the **substantia nigra** in the midbrain. In Parkinson's disease the substantia nigra is covered by abnormal aggregations of proteins called Lewy bodies mainly composed of alpha-synuclein. The substantia nigra transmit information through the thalamus to the premotor and prefrontal cortices. This substantia is strongly involved in motor patterns and when PD occurs, the lack of dopamine and the aggregation of Lewy bodies obstruct the normal composition of movements. Figure 1.3 shows how the basal nuclei are connected with other structures.

1.1.2 Biological fundamentals of Parkinson

Looking at the histopathology, there is the loss of dopaminergic neurons in substantia nigra, presence of Lewy bodies (cytoplasmic accumulation of ubiquitylated and misfolded α-sinuclein in neurons) and Lewy neurites (filament inclusions). Problems raise from a lack of dopamine. This neurotransmitter binds specific receptors in different brain areas (nigrostriatal for movement control, memomimic-mesocortical for emotional control or infundibular, which controls hormone release such as Growth

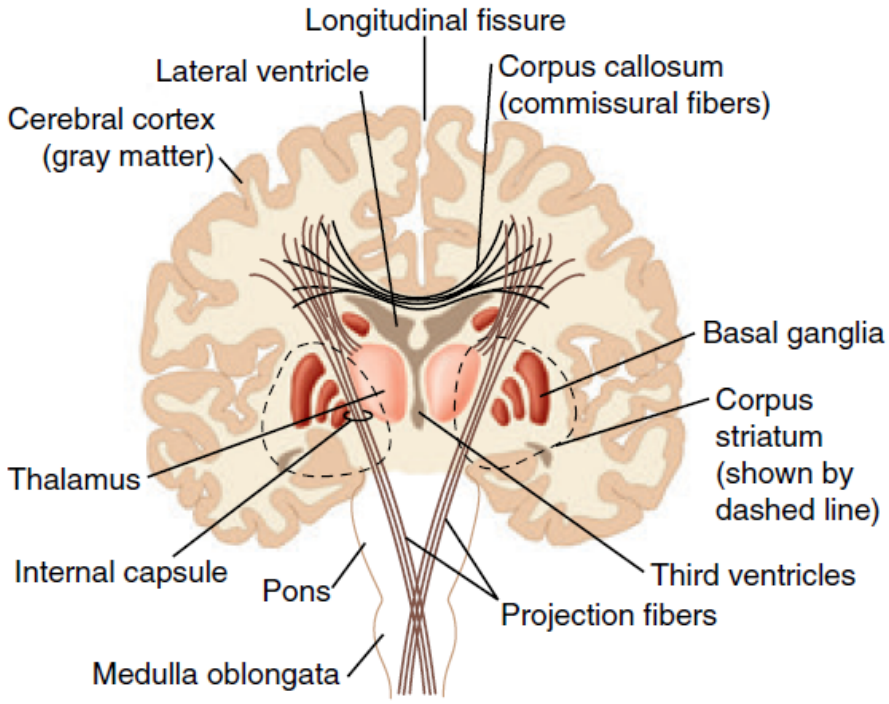


Figure 1.3: Basal nuclei and neighbours

Hormone (GH) and Prolactin Inhibiting Factor). It is produced by tyrosine hydroxylation to L-dopa and then decarboxylated to dopamine. The entire pathway is shown in figure 1.4

It is the precursor of noradrenalin and adrenalin. The 95% of cases of PD are sporadic, while the 5% has a genetic basis (both AD and AR). AD (Autosomal Dominant) form is related to point mutations or increased gene dose of α-sinuclein (accumulations and inclusions). Whereas, AR (Autosomal Recessive) form is characterized by an early onset: there are alterations impinging on genes that are not simply related to the production of α-sinuclein, but other proteins linked to energetic pathways (such as Parkin, PINK1, DJ-1, all related with autophagy).

Thus, even if we do not know precisely α-sinuclein biological function (very likely it is not a crucial one, it is expressed in the dopaminergic neurons of substantia nigra), if the mutation affects Parkin, PINK1 or DJ-1, we will have cells that cannot perform autophagy (in particular mitophagy). For example, PINK1 specifically targets dysfunctional mitochondria (so they are targeted for degradation), if this removal operation doesn't work properly, there will be an accumulation of damaged mitochondria. Anyway, α-sinuclein has not a direct function in reduction of dopamine (which is less available because we lose a neuronal population). This protein has a high turnover: the degradation occurs with the ubiquitination after the glycosylation. Generally, autophagy degrade oligomers, leading to the formation of Lewy bodies and Lewy neuritis, which contain aggregated misfolded proteins, mainly α-sinuclein. Perhaps, these bodies contribute to neuronal death, acting like a "storage" of unwanted material collecting all the aggregates to leave the rest of the free.

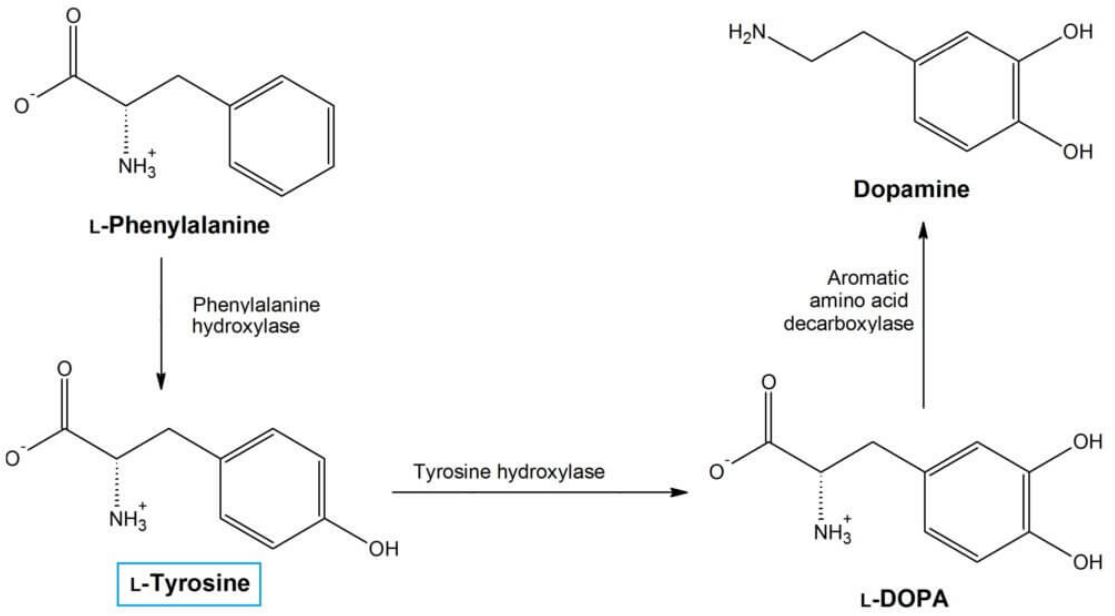


Figure 1.4: L-Dopa pathway [2]

In healthy neurons, α-sinuclein is ubiquitinated by Parkin (Ub-ligase E3) in order to transfer it to the proteasome or with p62 to the autophagy system. In affected neurons, mutated or high amounts or altered (not depending on DNA) α-sinuclein forms profibrils, then fibrils, leading to neurodegeneration. There are 5-6 mutations that are being recognized and all of them are affecting one of the proteins mentioned before (PINK1, Parkin, etc.). Lack of PINK1 is crucial not simply because it is involved in autophagy modulation, but also because it recognizes dysfunctional mitochondria. If there is a mutation of PINK1/Parkin or there are problems with the formation of autophagosome and an engulfment of the lysosome, we have an accumulation of damaged mitochondria, which release factors that are normally segregated in them and cause cell death. Consequence is a condition of oxidative stress leading to inflammation.

Mitochondrial fission and fusion play critical roles in maintaining functional mitochondria when cells experience metabolic or environmental stress. Fusion helps mitigate stress by mixing the contents of partially damaged mitochondria as a form of complementation. Fission is needed to create new mitochondria, but it also contributes to quality control by enabling the removal of damaged mitochondria and can facilitate apoptosis during high levels of cellular stress. Disruptions in these processes affect normal development, and they have been implicated in neurodegenerative diseases, such as Parkinson's. Lack of fission anticipates the accumulation of damaged mitochondria and the pattern could be worse: with fission, we save x% of mitochondria (delaying the loss of energy control), but if we cannot activate the fission, we cannot have this small amount of saving.

To summarize, Parkinson's disease is usually considered a chronic, progressive neurodegenerative movement disorder. However it's well known to have a variety of non-

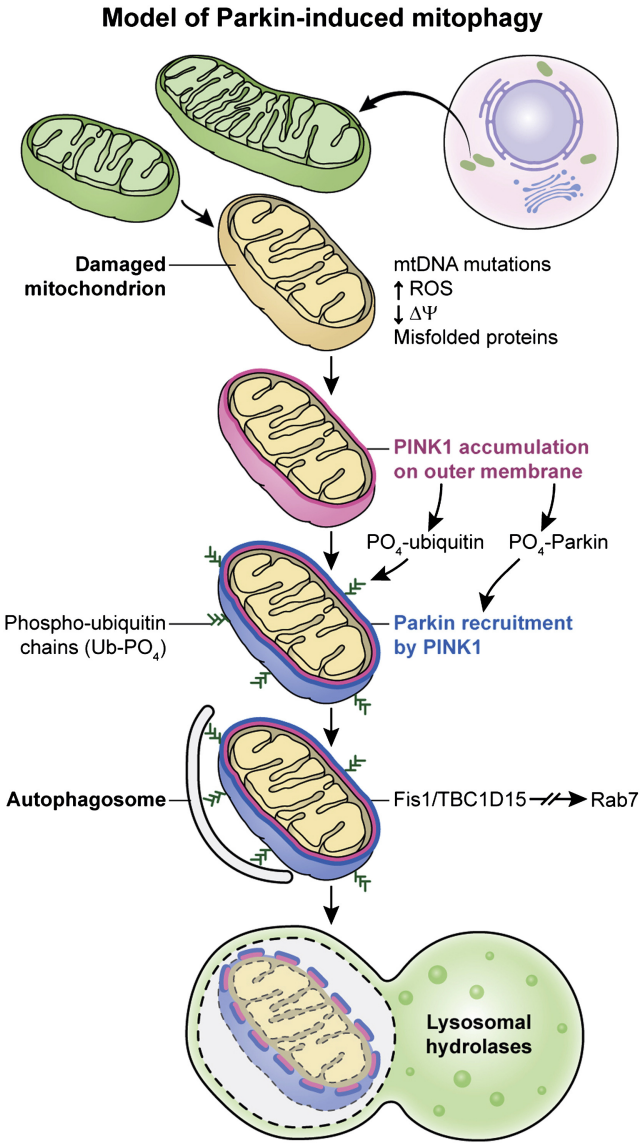


Figure 1.5: Parkin-induced mitophagy [33]

motor symptoms related to the autonomic nervous system. The cause of the disease remains predominantly unknown even if genetic and environmental factors play a key role in the development of the disease. For now there is still no cure to this disease although some medications such as Levodopa are very effective in tackling the symptoms. When Levodopa is no longer enough or the side effects (such as Dyskinesia) are too strong, another alternative is based on an implantable device for deep brain stimulation (DBS).

1.1.3 Symptoms

The symptoms of Parkinson are essentially divided in 2 groups:

1. Motor symptoms
2. Non-motor symptoms

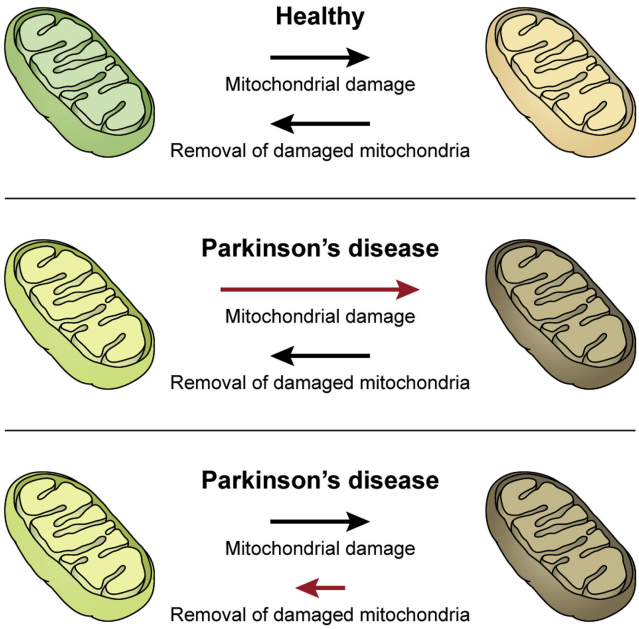
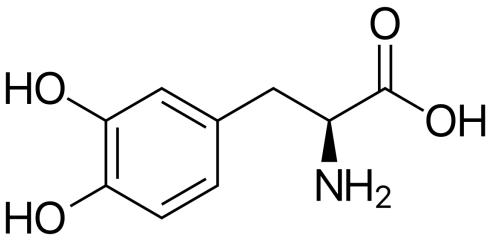
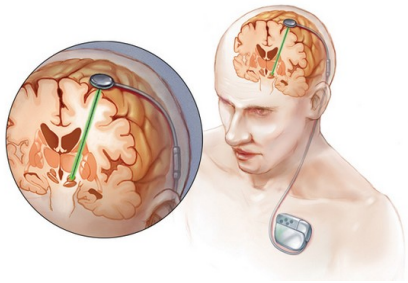


Figure 1.6: Ration between Mitochondrial damage and replacement of mitochondria in PD



(a) Levodopa [3]



(b) Deep Brain Stimulation implantable device [3]

Figure 1.7: Levodopa (a) and DBS (b) are the most common ways to treat Parkinson nowadays

The motor symptoms are usually referred as parkinsonisms since they are the most obvious and spreaded symptoms of this disease, they consist of:

- **Tremor**: only present when muscles are resting and disappear while sleeping as well as with action (one side shakiness)
- **Bradykinesia**: early stages, slowed movements
- **Hypokinesia**: reducted movements
- **Akinesia**: abrogation of movements
- **FoG**: Freezing Of Gait, form of akinesia

While the non-motor symptoms are also called secondary effects and can be summarized in:

- **Cognitive disturbances**: abstract thinking, rule acquisition, planning
- **Visuospatial difficulties**: orientation of drawn lines and facial recognition
- **Impulse control disorders**: gambling, binge eating, compulsive behaviour
- **Dementia** may occur in some subjects
- **Mood difficulties**: depression, apathy or anxiety are the most frequent
- **Sleep and emotional problems**

All these symptoms are so penalizing that the WHO (World Health Organization) rated Parkinson to be on the same level of disability of amputee limbs, congestive hearth failure, deafness, drug dependence and tuberculosis [11].

1.2 Unified Parkinson's Disease Rate Scale (UPDRS)

The severity of Parkinson's disease is scored following the standards of Unified Parkinson's Disease Rating scale. During an UPDRS evaluation, the patient perform scripted tasks under the observation of a clinician who score the subject in 5 different states of severity for each task. It's important to highlight that the UPDRS score is just a way to evaluate the state of the disease in that particular period of time, these scores can change based on medication time and ON/OFF state of the disease, not to mention the natural progress of the disease makes the UPDRS score a tool for temporary evaluation. The tasks are divided in 4 stages, each thought in such a way to take into account different aspects of the disease. Each of these tasks is scored between 0 and 4, as is possible to imagine, this process takes many hours and represents a cost in terms of money and time. Research is focusing on how to automate this process using wearable devices and new data classification techniques.

1.3 BlueSky project

"The term blue skies research implies a freedom to carry out flexible, curiosity-driven research that leads to outcomes not envisaged at the outset." [24] In this outlook, this project aims to investigate the possibility of classifying the motor fluctuation of PD by using wearable sensors and Machine Learning techniques. We don't know what to expect or either the real feasibility of such approach in daily life, however it's possible that in a future more or less closer this work will be used in a real scenario.

The scenario would be the complete characterization of motor symptoms in a remote way and without the need of a clinician to analyze the movements of the subject to understand his level of impairment. This would save money to the healthcare industry and time for both clinicians and patients. To tackle this problem the schema in

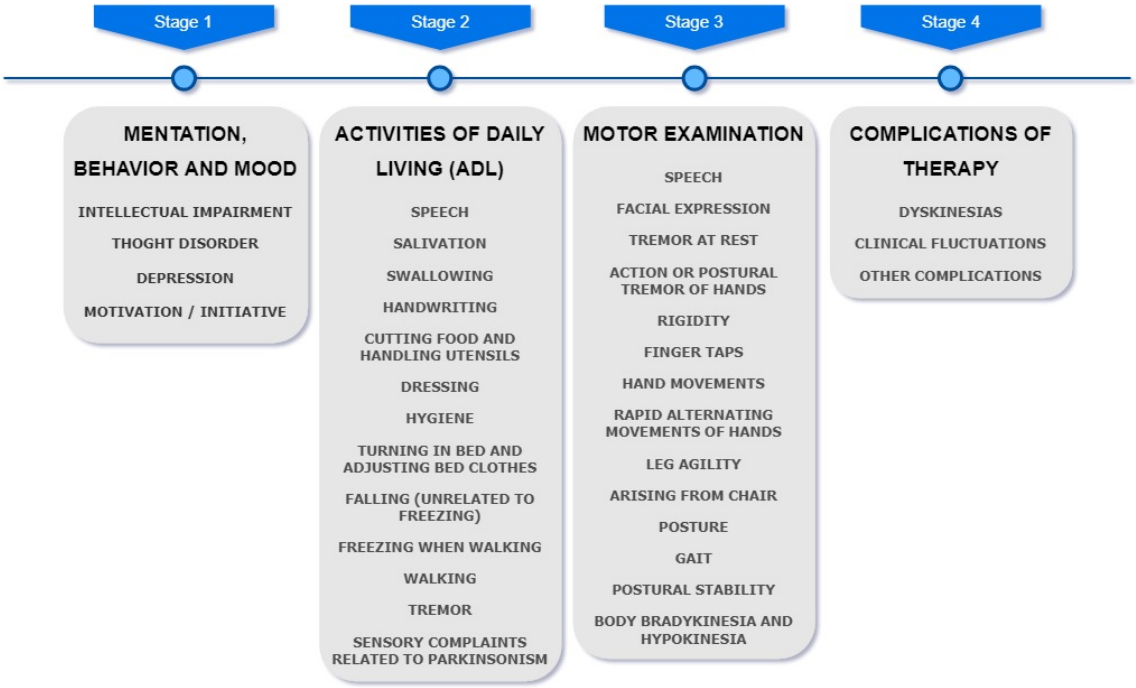


Figure 1.8: UPDRS evaluation

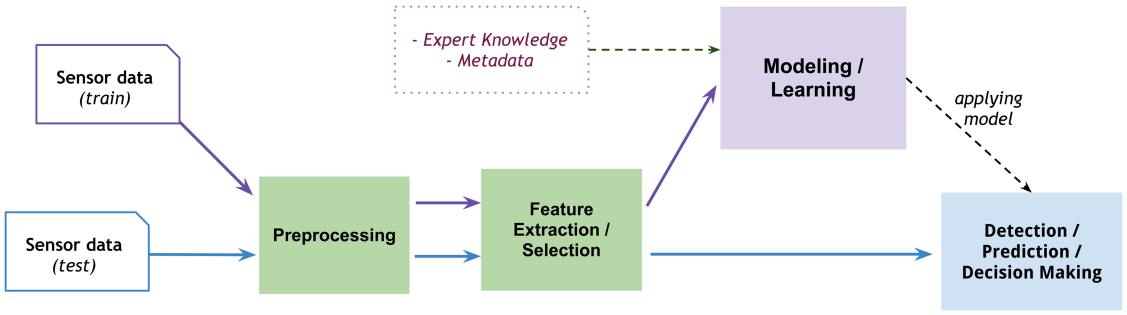


Figure 1.9: General ML schema [13]

figure 1.9 has been adopted. In figure 1.9, the normal scenario of Machine Learning is represented. In our case, the sensor data is the data acquired by the wearable sensors. *Train* is the data acquired in the lab sections while *test* is the data acquired during the apartment sessions. The preprocessing and feature extraction/selection are steps built on the training data and then redone on the test data. Different models, strategies and tuning of parameters have been done to obtain the better results possible on top of the train data. The models that reported the lowest error have been then used on the test data. Aims of this blue sky project is to develop tools for predicting the severity of motor fluctuations in Parkinson. Among the different symptoms, this work will focus on predicting the impairment level of Dyskinesia.

1.4 Dyskinesia

Quoting "parkinson.org" [4] *"Dyskinesias are involuntary, erratic, writhing movements of the face, arms, legs or trunk. They are often fluid and dance-like, but they may also cause rapid jerking or slow and extended muscle spasms. They are not a symptom of Parkinson's itself. Rather, they are a complication from some Parkinson's medications."* Usually the medications refer to Levodopa, a miscalculation of the dose of Levodopa can cause LID, which stands for Levodopa Induced Dyskinesia.

1.4.1 ON/OFF state

With the progression of time in which the subject use Levodopa, the medication starts to work lesser and lesser and its wearing off time starts to decrease, triggering the ON/OFF phenomenon. Ideally if you have a regular Levodopa intake, the symptoms shouldn't vary depending on the medication intake and you should be covered for the entire your day if the dose and times are studied ad hoc. However patients feel better immediately after the medication intake "ON" and worse before taking the next dose "OFF". Eventually the "ON" periods will be shorter and shorter.

In this study, 5 different sessions have been performed and each session had a different transition between ON and OFF state as shown in figure 1.12.

Drug titration

Drug titration refers to an effect of adjustment of medication dose to keep constants the effects against the disease and at the same time keep under control the side effects of the medication itself.

Referred to Levodopa, the side effect is the LID, to avoid the dyskinesia, patients should stay in the state OFF of the disease, however Parkinson patients in state ON can barely walk, often they prefer to overdose in Levodopa and have dyskinetic events instead of staying without the possibility to walk. Figure 1.10 represents the adjustments of doses and how side effects increase with the increasing of doses.

1.5 Study setup

1.5.1 Data Collection

Data collection was structured in such a way to enhance the development of analysis methods of biological data to quantify motor and non-motor signs and motor fluctuations in 2 different environments:

- **Laboratory**: scripted tasks

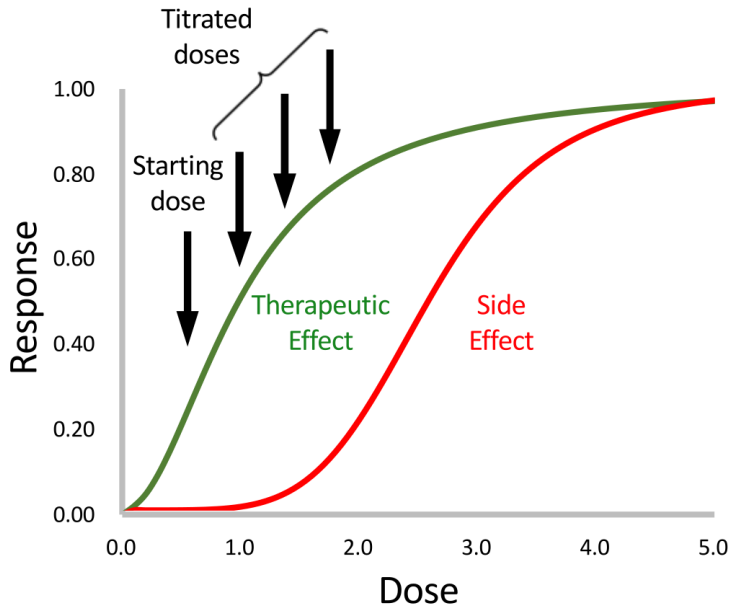


Figure 1.10: Drug titration [5]

- **Home-like**: monitoring of ADL (Activity of Daily Life)

Participants to the study, delayed their medication intake in such a way to arrive in a "OFF" state. In each session the some scripted tasks were performed and scored in the UPDRS scale. Recordings have been performed on 60 healthy volunteers and 27 PD patients in the study conducted at Spaulding Rehabilitation Hospital, Boston,MA.

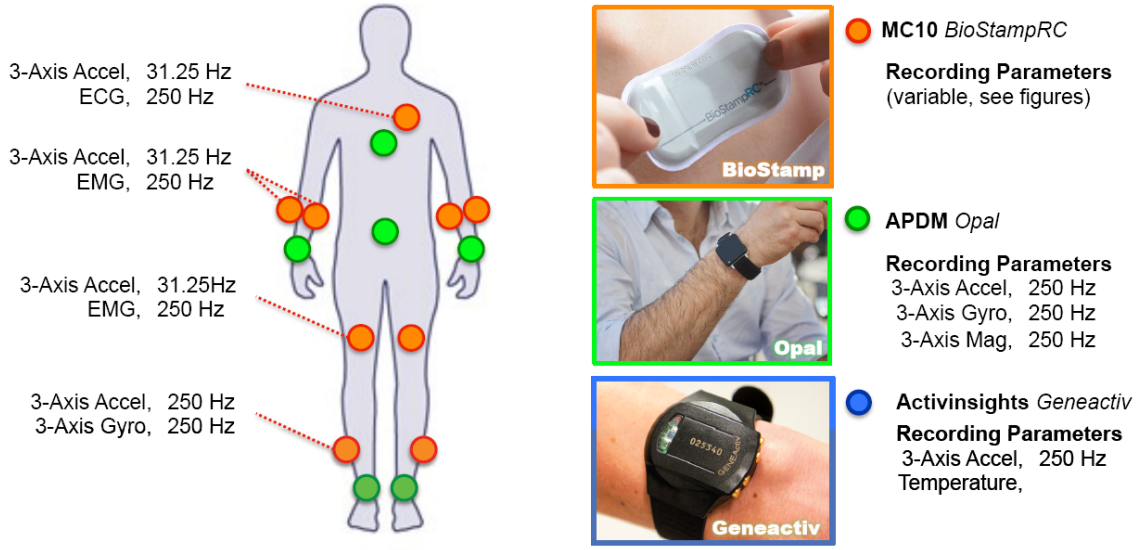


Figure 1.11: Typology of sensors used and body locations

Session 1, 3 and 5 are present for each patient while the presence of session 2 and 4 depend on the medication's effect rapidity. After session 1, participants took their medication and in session 2 the effects of "OFF/ON" transition are studied. Since

some participants have a fast “OFF/ON” transition, these participants won’t have a session 2. Session 3 is performed in a “ON” state, session 4 in “OFF/ON” transition and finally session 5 in “OFF” state again. Session 4 as well as session 2 is optional and depend on the transition rapidity.

Figure 1.12: ON/OFF state among sessions

One of the objectives of the BlueSky project is to create a model for Bradykinesia, Tremor and Dyskinesia able to classify these events in real time in a home environment using data collected by wearable sensors. Future outlooks may be able to implement these classifiers on daily life garments and smart watches and be able to monitor the fluctuation’s trend. As already mentioned this thesis is mainly focused on developing the classifier for Dyskinesia events. All the motor fluctuations are analyzed implementing a windowing technique. Different window lengths have been investigated in literature and a window of 5 seconds length seem to be the one to reach best results. After having pre-processed and windowed the signal, different techniques and strategies have been thought and implemented to reach our goal.

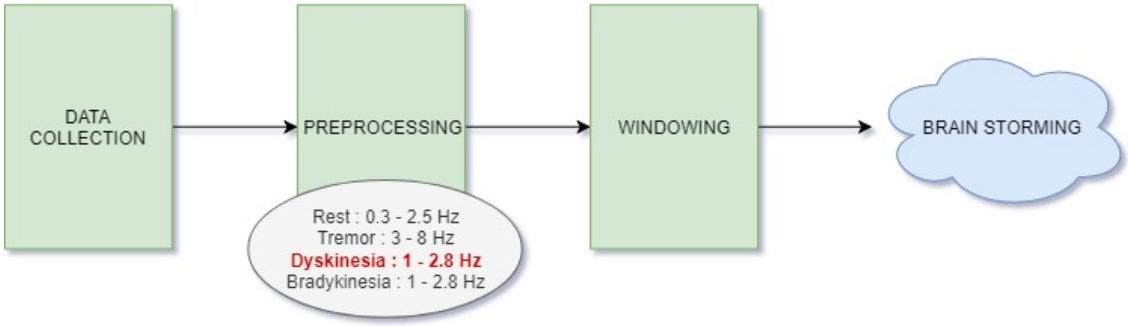


Figure 1.13: From Data acquisition to Brain Storming

It’s interesting notice how the level of dyskinesia is spread among the subjects and between lower limbs and upper limbs, figure 1.14. We can notice that the greatest part of the time the dyskinesia is not present at all and different subjects have different duration of the trial. Moreover, looks like, generally for the population considered, the dyskinesia is more present in the lower limbs than upper limbs.

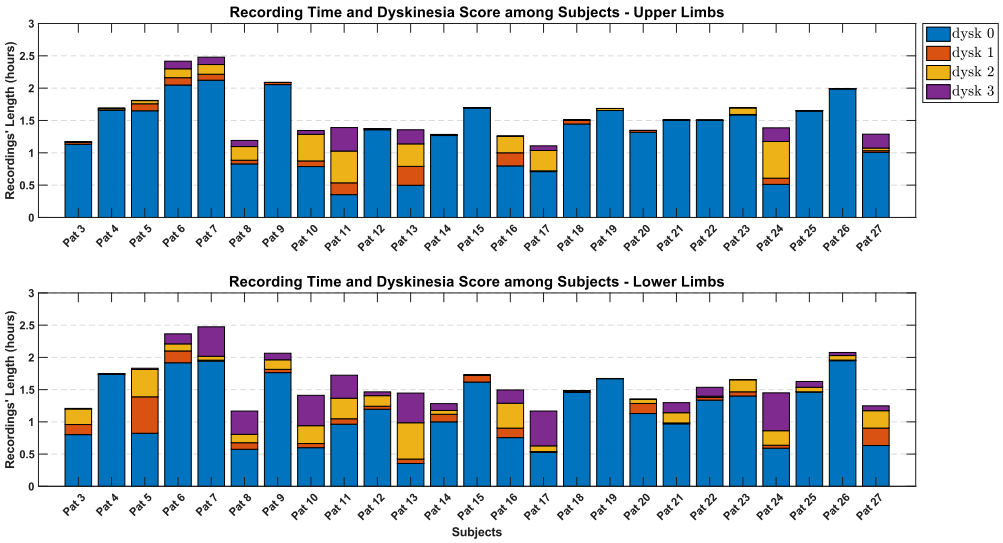


Figure 1.14: How dyskinesia is present among subjects

1.6 Wearable sensors and Artificial Intelligence

Nowadays many different sensors are available to record physiological data in a non invading way. For physiological data we mean all the data retrievable from the body that can allow the understanding of the state of the body. Examples of physiological data are blood pressure, Electrocardiogram (ECG), Electroencephalogram (EEG), Galvanic Skin Response (GSR), Surface Electromyography (sEMG) and many others. All the listed are signals that can be recorded in a non invading way, however there are also physiological signals that require a certain level of invasion (for instance Intramuscular EMG). Moreover wearable sensors are able to sense its location in the space and it's sensible to movements and accelerations, thanks to accelerometer, gyroscope and magnetometer present in the sensor. These signals can't be considered physiological but can be used to understand how movements are performed by the subject and this can ease the process to make diagnoses. These last sensors are particularly used in **Human Activity Recognition (HAR)**. It's in this field where most of the research is focusing at. Many public datasets are available to download and public competitions (above all on kaggle) are spreading worldwide. The kaggle competition HAR with smartphones [6] has been seen by 100k users, 12k have downloaded the dataset and 130 different solutions have been provided. In HAR the goal is to understand which action is the subject doing by analyzing the data from IMU (Inertial Motion Unit). Usually in these problems the most important data derive from tri-axial angular velocity (provided by the gyroscope) and tri-axial accelerometric data (from the accelerometer). Usually this typology of problems require a net to be trained on provided labels. This approach is called Supervised learning.

1.6.1 Supervised learning

Supervised learning is a branch of machine learning where the desired output is known and the model will try to change its weights in order to output something the most similar possible to the desired output. This is an iterative process that requires many labeled data, and a back-propagation algorithm to rearrange the weights. In this subsection some basic training concepts are explained.

Back-propagation

Back-propagation is a way for the net to keep into account the difference between the prediction and the real value. It's important during training since it allows to correct the internal parameters (weights of the net) in a way that minimize the **cost function** which can be seen as a measure of the error. This error can be calculated after every batch or after some random samples and hence this updating of weights can be done during different periods of training. Different algorithms can be applied to minimize the cost function, the most common one is the **stochastic gradient descent**. This algorithm is an optimization algorithm since it's trying to minimize the cost function. The most used cost function is the Mean Squared Error (MSE). Particulars on the algorithms will be investigated further in the Methods chapter. An other very used variant of the stochastic gradient descent is the **adam** optimization.

Chapter 2

Methods

2.1 Introduction

The accelerometric data have been collected from 27 subjects with Parkinson's disease. The patients arrived in an OFF state, meaning they didn't take their medication that morning. There were 5 sessions where a number of tasks has been repeated, session two and four were optional, depending on the rapidity of transition from OFF/ON and ON/OFF of the patient. The wearable sensors used to detect dyskinesia were located on the wrists and the shoes of the subject. Each task had different duration in time and the clinician gave a score in the UPDR scale to each limb where the sensors were attached. Each task has associated one score for sensor. The subjects have been recorded and the dyskinesia severity of each limb has been scored by qualified clinicians who watched the recordings. Figure 2.1 shows how rates are grouped among tasks while table 2.2 is a short summary that link the code of the task with its meaning.

2.2 Machine Learning Approach

The Machine Learning methodology has been adapted. The entire chapter will pay particular attention to each step represented in figure 1.9. The raw signals have been collected using the OPAL wearable sensor from APDM technologies. These data have been then filtered in the domain of interest. 2 bands are interesting for the extrapolation of features, the band in the resting frequencies (between 0.4 and 2.8) and the band in the dyskinesia frequencies (between 1 and 3 Hz).

The filtered signals have been saved and then windowed in windows of 5 seconds length. The choice of 5 seconds derive from further studies and personal experience of the members of the Motion Analysis Lab. In 5 seconds there is information enough to understand if the movement is dyskinetic or not. However there can be 2 different type of dyskinesia, one is slower in the time but quasi-periodic, another is more unpredictable and manifest itself through strong jerks. After the signals have been

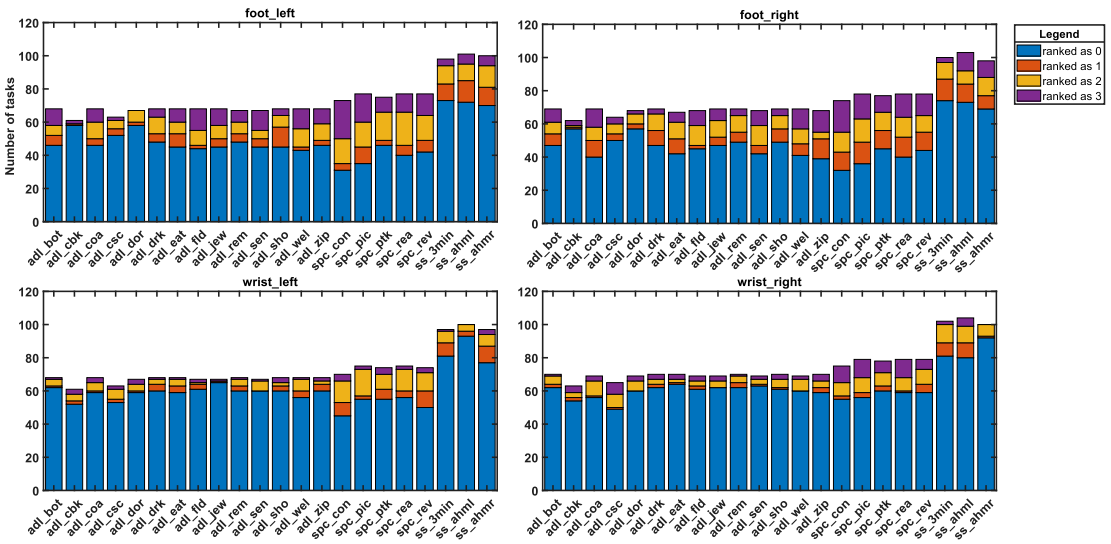


Figure 2.1: Number of tasks for score. From this graph is possible to notice how the scores are distributed among different tasks for each limb

windowed, features are extracted from each window. To solve the problem of misclassified data, clear outliers have been removed, an heuristic threshold on the range of the amplitude of the signal has been superimposed and a first regressor has been used to firstly clear the dataset. This last step is done by selecting only those samples classified by the model with a certain level of confidence. Those samples clearly misclassified are not used to train the following model.

To follow a feature selection is done. Features are selected through **Out Of Bag Feature importance**, implemented in the Random Forest algorithm. This is the most natural choice when the Random Forest is the model chosen. Both binary and multiclass performance are evaluated. The binary model is a classifier, while the multiclass has been implemented as a regressor. This is done because a continuous output is wanted and by doing the regression the model is taking into account the nature of the problem in which an error between 0 and 1 counts less of an error between 0 and 3.

Data visualization is done using the **t - Stochastic Neighbor Embedding (t-SNE)**, a technique that allows the visualization in a 2d or 3d space of multidimensional datasets. From this technique is clear how samples belonging to different classes are overlapped in the feature space.

On the other hand, to keep into account the temporal sequence of the signals, the Long-Short Term Memory recurrent neural network has been studied, tuned and evaluated. Firstly it's been tested on the raw data of each window, then on the extracted features (each feature is extracted on 5 seconds' signal) and finally on the output of the Random Forest. This last procedure can be thought as the brain of a clinician whom internally makes decisions each 5 seconds and eventually he has to find a way to score the entire task based on the sequence of the scores given throughout the task.

This methodology has then been confronted with the median of all the decisions and performance are evaluated in terms of regression (RMSE and MAE are reported) and in terms of classification, confusion matrices are obtained by rounding the prediction of the regressor to the closest integer number.

2.3 Data Collection

The first step of the chain is certainly collect data. To do so, elegibility criteria are needed to be sure of collecting data from the right people and a protocol has to be used for reproducibility purposes. Some statistics of the chosen population are reported. The data are collected in 2 different environments:

- **Laboratory**
- **Apartment**

In the laboratory the patient had to accomplish some standardized tasks, while in the apartment the subject was free to do whatever he wanted. In figure 2.2(a) and 2.2(b) the 2 environments are visible.

(a) Lab setting

(b) Apt setting

Figure 2.2: (a) Laboratory environment (b) Apartment environment

2.3.1 Elegibility Criteria

- People with current diagnosis of PD consistent with UK Parkinson's Disease Society Brain Bank Criteria.
- Responding to Levodopa
- stable configuration of the disease for at least 4 weeks

- able to understand and recognize the *wearing-off* symptoms and recognize their improvement after the dose of Levodopa.

- **NOT** in a current state of neurological disease (PD excluded)

- **NOT** implanted medical devices (DBS included)

Statistics of the subjects are reported in table 2.1 and in figures 1.14 and 2.1

Table 2.1: Participant Characteristics

Characteristic	**Statistics**
Number of subjects	27
Age (years)	66.1 ± 8.2 (42 - 80)
Height (cm)	175.4 ± 10.3 (152 - 196)
Weight (lbs)	182.7 ± 34.9 (116 - 260)
Gender (%)	
Males	19 (73.1)
Females	7 (26.9)
Handedness (%)	
Left	3 (11.5)
Right	23 (88.5)

2.3.2 Protocol

27 subjects have been selected. The table 2.2 resume the tasks that the patients had to do in each section. 5 sections are present, each of them count the same number of tasks, however task number 2 and 4, since were transitional tasks were more likely to not present the entire sequence of tasks. The acronym SADL stands for “Simulated Activity of Daily Life” while CSS stands for “Conversation Speech Session” and were tasks regarding no physical effort but only psychological or emotional. Fig 1.12 shows the state ON/OFF of the patients and how they switch among states because of the L-DOPA assumption.

Task Code	Description
adl_bot	SADL: Shake 5x, open bottle, drink and close
adl_cbk	SADL: Carry a book (out and back 10m) and place it on a table
adl_coa	SADL: Buttoning a lab coat
adl_csc	SADL: Carry a suitcase (out and back 10m) and hold 90s up with forearm at 90°
adl_dor	SADL: Opening and closing a door
adl_drk	SADL: Pour a cup of water and take two drinks
adl_eat	SADL: Eat with a spoon 2x
adl_fld	SADL: On the table, fold a piece of paper in half 4x
adl_jew	SADL: Putting on / removing jewelry
adl_rem	SADL: Remote control use
adl_sen	SADL: Writing a sentence
adl_sho	SADL: Tying a shoe
adl_wel	SADL: Write elelelel (cursive) 10x
adl_zip	SADL: Zipping a zipper
spc_con	CSS: Conversation
spc_pic	CSS: Picture description
spc_ptk	CSS: PATAKA
spc_rea	CSS: Reading
spc_rev	CSS: Reverse counting
ss_3min	sitting_3
ss_ahml	ahm_left_3
ss_ahmr	ahm_right_3

Table 2.2: Brief description of the tasks associated to each code

The patients were recorded and then scored following the UPDR scale explained in section 1.2, page 10 by qualified clinicians. It's to highlight that the clinicians rated the videos, hence they could have viewed more than one time the task performed to give an accurate score.

2.3.3 Hardware

The hardware used during the data collection is the OPAL from APDM wearable technologies. This device offers 3-axis accelerometer, magnetometer and gyroscope. It is one of the most used wearable device in the laboratory for research in motor symptoms.

Table 2.3: OPAL sensors characteristics

	Accelerometer	Gyroscope	Magnetometer
Axes	3 axes	3 axes	3 axes
Range	±166,	$\pm 2000 deg/s$	± 8 Gauss
Noise	120 $\mu g/\sqrt{Hz}$	0.025 $deg/s/\sqrt{Hz}$	2 $mGauss/\sqrt{Hz}$
Sample rate	250 Hz	250 Hz	250 Hz
Resolution	14 bit	16 bit	12 bit

Table 2.4: Hardware characteristics

Hardware Characteristics	
Dimensions	43.7 x 39.7 x 13.7 mm (LxWxH)
Weight	< 25 grams (with battery)
Material	Polycarbonate, Glass
Internal storage	8 Gb, (∼ 450h Storage)
Battery life	Synchronous Logging: 12h, Asynchronous Logging: 16h

2.4 Signal Preprocessing

From the raw data acquired by the sensors to the actual data used to train the different models some operations are done in order to improve the performances of the models.

- Sensors data and events in the lab
  - Load raw sensors data from csv files (OPALs and Biostamps) for each subject and each lab session (1 to 5)
  - Load events data from csv files and create a table with starting and ending time of each task (for all subjects and sessions). Assign to each task the clinical scores. For each motor symptom, the severity classes 3 and 4 are merged (too little samples for class 4)
  - Extract OPALs data, resample at 32 Hz (from original fs = 128 Hz)
  - OPALs data from the lumbar and the sternum signals are not considered in the analysis
  - Merge the sensors data for each session and create a continuous time series from the beginning to the end of the lab visit
  - Filter the signal in the "Dyskinesia band", the filter applied is Chebyshev Type II band-pass filter, with cut-off frequencies: [1 – 2.8 Hz].

  - Segment the filtered data in different signals, each of them will be the signal referred to a task. To each task is assigned a score.
  - Apply windowing to each signal. The signals have been windowed with length of 5 seconds and 50% overlap between 2 consecutive windows.

- Apartment data

  - Load raw data from csv files (OPALs sensors) for each subject
  - Extract OPALs data, resample at 32 Hz (the original is 128 Hz) and fill the gaps with NaNs
  - Apply filtering to the continuous raw sensors data (same filtering described above). The resulting data structure is used as input for the motor symptoms severity estimation algorithms in apartment.

An already existing model has then been used to classify the windows between resting or not resting windows. This model was developed for tremor analysis on the same dataset. However this classifier doesn't allow the separation between voluntary movements and dyskinetic movements. Another model is needed to perform this separation.

2.5 Feature Extraction and Selection

2.5.1 Feature extraction

The extraction of the features is not the aim of this thesis. The features were already extracted at my arrival. However an understanding of the extracted features is necessary. Features can be divided in 5 main different areas. A total of 90 features have been extracted.

Macro areas of features

- Time and frequency domain
- Position, Velocity and Jerk
- Rest band and Dyskinesia band ratio among Rms, range, dominant freq. Amplitude and total power of the PSDs
- Segment Velocity features
- Correlation among sensors

Time and frequency domain are the most traditional family of features. They refer to the set of features that can be extracted in the time domain (such as: mean, std, variance, entropy, kurtosis, skewness, etc.) and the features from the Fourier transform of the signal (maximum amplitude, main component, total power, etc.).

Position, Velocity and Jerk are sets of features that are extracted by processing the accelerometric signals. Paricularly, previous windowing of the signal, the accelerometric signals have been integrated by trapezoid numerical integration to get the the velocity and a further integration is done to get the position. To get the jerk, a derivation is done. All the time and frequency features extracted for the acceleration are extracted for velocity, position and jerk.

Rest band and Dyskinesia band are semi-overlapped bands in the frequency domain, however these set of features could be used to distinguish among voluntary and involuntary (hence, dyskinetic) movements.

Segment velovity features refers to a set of features taken from the work of Keijsers et al. [21].

The last features refers to correlation intra-sensor (among different channels of the same sensor) and correlation inter-sensors (among same channel of different sensors). The firs is useful to understand how the movement of the same limb is done, it's aim is to understand which is the level of correlation among the different directions. The least instead can be more difficult to understand if the nature of dyskinesia isn't clear. It refers to the fact that probably dyskinesia is happening in only one limb, many times dyskinesia can be unilateral meaning that probably there will be a low level of correlation among controlateral and opposite limbs.

2.5.2 Feature selection

Since the principal model chosen for the classification of the dyskinesia is the Random Forest and this model offers the possibility of ordering the features by importance, the feature selection has been performed using the RF model.

Number of features

The features can be mainly ordered by calculating the Variance as a measure of impurity and the Out Of Bag (OOB) error as a metric for the importance of a subset of features. More in detail, to choose the number of features, the following approach can be followed:

Feature Importance

The importance of the features is calculated by keeping into account the impurity in each tree split. For **classification** the measure of the impurity is the *Gini index* while

Algorithm 1 Criterion for choose the number of features

1: Order the features by feature importance
2: **for** ***i*** in number of features **do**
3: Train a RF model with the subset of ***i*** features in the ordered list of features
4: Compute the OOB error
5: Choose the number of features corresponding to the minimum OOB error

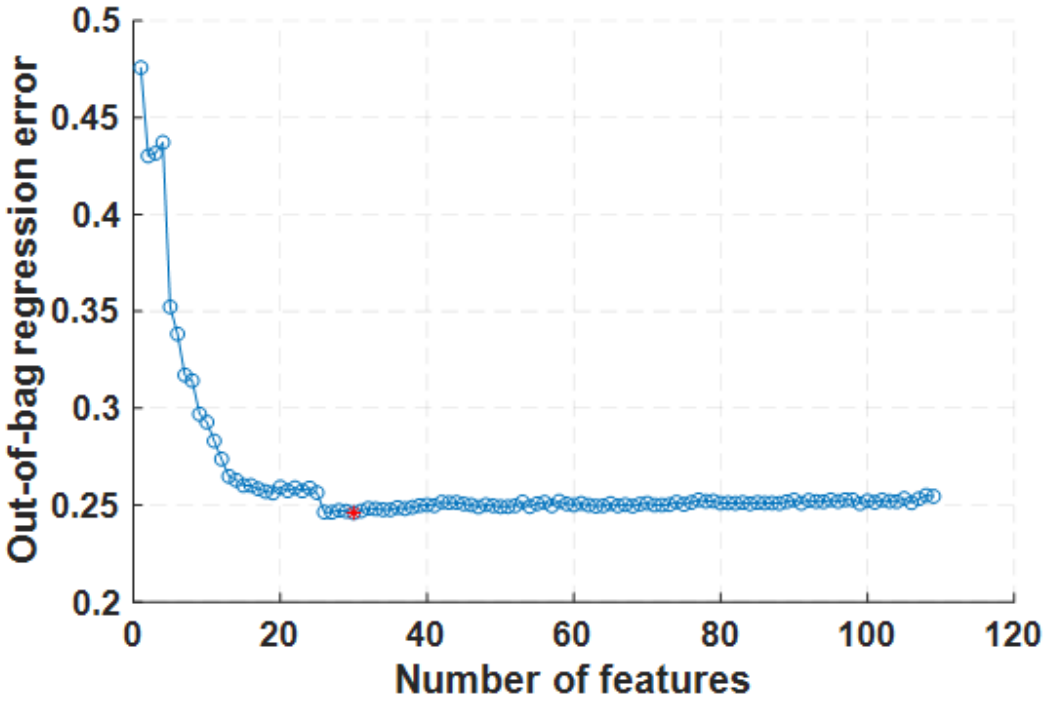


Figure 2.3: Number of features

for **regression** the impurity is measured by the *variance*.

In Random Forest there are typically hundreds to thousands different decision trees, usually these trees are built by looking to a random subset of the features and a random subset of the samples. This is done to guarantee that the trees are not correlated each other and at least, the overfitting is avoided. At each tree splitting, a decision is made. Let's call the split location **node**. In a node a question based on the features present in that node is done. The answer to the question can lead to different branch (or leaves) of the tree. The importance of the feature can be derived by looking at how pure the bucket created by the split is [7].

In a regression approach the impurity is the variance of the bucket. A low variance is expected if the samples ended in a leaf are all similar among each other. This makes sense in a regression problem since the mathematical nature of the samples and their similarity in terms of classes is what matters. A high variance is expected in those buckets created without using a feature that is important. Hence, by measuring the average of the variance among the buckets and keeping into account which features are used and which are not, a ranking of the features can be computed.

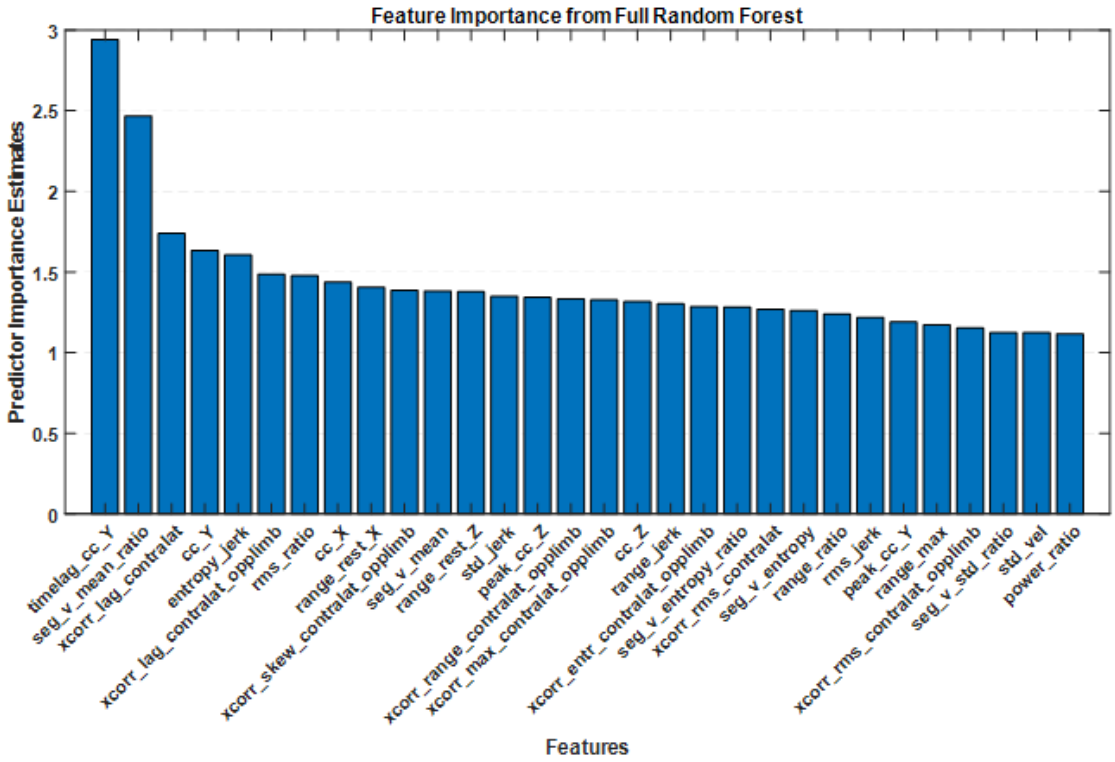


Figure 2.4: Features selected by Random Forest feature selection approach

Figure 2.4 shows the features sorted by feature importance. Only the first 30 features are shown, according to the number of features retrieved from the OOB regression error. It's interesting how the first features are mainly related to cross-correlation features intra and inter sensors. At least in part this could explain why the Random Forest outperforms the Long Short Term Memory nets using raw data. Raw data indeed don't have any information regarding what the other limbs are doing and only rely on their own information. Hence RF models have more information than the LSTM based on raw signals.

Considerations on the Selected features

The correlation has been computed using the xcorr function provided by Matlab®. The first selected feature is the Timelag of the cross-correlation (cc) of the X and Z channel (timelag_cc_Y is a way to call the cc among X ans Z), however also the other cross-correlation in the other channels have been selected. The cross correlation of different channels in the same sensor is important because it's a measure of how the movement is done and which delays are recorded in different directions during a task.

The entropy of the jerk is another feature which clearly makes sense. At the beginning of the work, the idea was to create a binary classifier for voluntary and not voluntary movements based only on entropy features. This is due to the fact that the entropy is a measure of how disordered is a signal, in this case a movement. What expected was a high level of entropy in those signals of clear dyskinesia and low level in pure voluntary movements. Indeed all the models had the problem to not be able to mis-

classify the voluntary movements in dyskinetic movements. This approach will be further investigated in subsection 2.9.1.

2.6 Data balancing

The dataset is strongly unbalanced towards the zero class, hence a data balancing technique is needed. At first an approach based on hierarchical classifiers has been proposed and performed based on the fact that the first classifier would have naturally separated class 0 from the others and the remaining dataset for further classification is already more balanced. This method has been tried, however the poor performance of the binary classifier (in terms of sensitivity) and the fact that a single regressor was explicitly demanded by my supervisor in the lab made another approach the preferred one.

In the chosen approach, a binary classifier is firstly used and then the result of such a classifier is used as a feature with the relative score. This will provide the following RF the ability to understand with how much probability the previous classifier was certain about the fact that the sample represents a dyskinetic movement or not.

2.6.1 SMOTE

While dealing with classification problems, the percentage of the classes in the classes leads an important rule, however there are scenarios where it's not possible to have balanced data. In these conditions the model may try to fit the majority class by creating a bias in the predictions, in the same time the accuracy will stay high but it's not objective since our model is biased in the majority class. It's the same problem for rare diseases or natural disasters. Different strategies see the down-sampling of the majority class in a random way or the oversampling of the minority class.

- **Down-sample** the majority class in a random way.

  In this case there is a huge loss of probably useful data, since we're limitating the number of samples to the class with less data. It's true though that by using this approach, all the data used for training is real data and the classifier won't be biased anymore. However there is a concrete risk that the classifier will mistake more times when tested on elements similar to the ones excluded by down-sampling

- **Oversample** the minority classes by repetition

  The second approach is to decide to not delete any sample, instead of augmenting the minority classes to the number of samples of the majority class. This is done by simple random repetition of the samples in the classes until the same or similar number of samples is contained in each class. However by doing so,

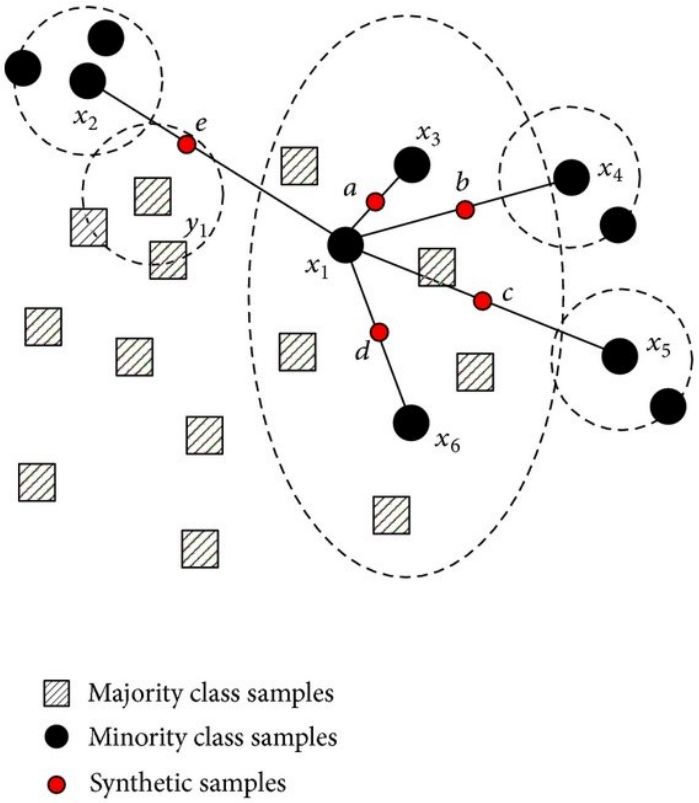


Figure 2.5: **SMOTE** algorithm [18]

a bias is inserted in the minority classes, more in detail, the samples in the minority classes are more important than the samples in the majority class and an overfitting on the minority class is possible.

- **SMOTE** Synthetic Minority Oversampling Technique

  To overcome the problem of overfitting in the minority classes due to simple repetition of the samples, the SMOTE approach is used. This method allow the synthetic repetition of the samples in the minority classes. In this way all the classes will have a similar number of elements and all the elements in each class will be different among each other.

Algorithm 2 SMOTE steps

1: Identify the feature vector and its nearest neighbour
2: Compute the distance between the 2 of them
3: Multiply this distance by a random number between 0 and 1
4: Identify a new point in the line segment by adding the random number to feature vector
5: Repeat the process for identifier feature vector

2.6.2 ADASYN

Another more sophisticated variance of SMOTE is the **ADASYN** algorithm. This algorithm is the algorithm used to balance the datasets. ADASYN stands for Adaptive Synthetic Sampling Approach for Imbalanced Learning.

This method follow the same steps proposed in algorithm 0 but instead of adding the new samples in the segment of the distance among the real samples, ADASYN increase the variability by adding a random number to move out of such segment by 90°. Although this method looks very similar to SMOTE, this simple change is creating new samples that don't present any more the linear dependency with the non synthetic data.

Algorithm 3 ADASYN steps

1: Identify the feature vector and its nearest neighbour
2: Compute the distance between the 2 of them
3: Multiply this distance by a random number between 0 and 1
4: Identify a new point in the line segment by adding the random number to feature vector
5: From this point move of a random factor in perpendicolar direction respect to the distance vector
6: The selected point will be the new synthetic point
7: Repeat the process for identifier feature vector

2.7 Data visualization

Visualizing 90 features in a 3D splace is not an easy task. Different techniques can help, for instance after a dimensionality reduction technique, such as PCA or after the chosen feature selection method, it's possible to decide to plot the samples in the feature space constituted by the 3 main features. However there is a technique which is created appositely to solve this problem, the **t-Stochastic Neighbour Embedding**.

2.7.1 t-SNE

This technique allows data visualization of multiple features in a 2D or 3D space without performing any rotation of the matrices nor dimensional reduction. The points plotted and the referring axes don't have any more a physical significance. The only aim of this technique is to separate as mush as possible elements of different classes by playing with different factors.

The t distribution

The Student's t is one of the biggest breakthrough in statistics as it allows inference to small samples with unknown population variance. This setting has a huge real life application. The t-normal distribution looks very similar to the normal distribution but it has fatter tails. This is important because it allows for a higher dispersion of variables as there is more uncertainty.

The formula to identify the t-distribution is

$$t_{n-1,\alpha} = \frac{\bar{x} - \mu}{s/\sqrt{n}} \tag{2.1}$$

n: sample size
$\bar{x}$: sample mean
μ: population mean
$s/\sqrt{n}$: Standard error of the sample

It's very similar to the normal distribution, indeed it's an approximation of it. However there are degrees of freedom. Usually for a sample of n there are $n-1$ degrees of freedom. After 40 degrees of freedom, the t-statistic is almost identical to the z-statistic (normal distribution). Hence, as the degree of freedom change with the sample size, for a large number of samples the t-statistic and the z-statistic can both be used. However, the importance of t-statistic is in a little number of samples.

Introduction to t - Stochastic Neighbour Embedding

t - Stochastic Neighbour Embedding (t-SNE) is a statistical method winner of the Merck Viz Challenge in 2012 launched by Kaggle. In the kaggle blog [8] Laurens Van der Maaten, creator of this technique and winner of the competition said: "*T-SNE represents each object by a point in a two-dimensional scatter plot, and arranges the points in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points.*"

The following are the reasons to use this technique in data visualization instead of the PCA followed by scatter plots of the 2 or 3 principal components.

- The main focus of this technique is to model small pairwise distances, like local structure, in the space
- This technique implement a way to correct the enormous difference in volume between a multi-feature space and a 2 dimensional or 3 dimensional map

This is a very strong non linear technique that can be divided in 2 main stages:

1. a probability distribution over couples of samples with all the features is built in a way that similar samples have higher chance of being chosen while points not very similar will have a much lower probability of being chosen.

2. a similar probability distribution is built for the samples in the 2D or 3D space, the t-SNE minimizes the ***Kullback–Leibler*** coefficient between the distributions in the low dimensional space and the high dimensional one with respect to the locations of the samples in the space.

The **Kullback-Leibler** divergence for 2 discrete probability distributions P and Q is defined as:

$$D_{\mathrm{KL}}(P \parallel Q) = -\sum_{x \in \mathscr{X}} P(x) \log\left(\frac{Q(x)}{P(x)}\right) \tag{2.2}$$

while for continuous probability distribution the equation 2.2 become:

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) d\,x \tag{2.3}$$

This divergence is usually used as measure of the **information gain** and means how much information is lost if the distribution Q is used to approximate P. The minimization of this divergence is computed using the descent gradient in t-SNE.

Laurens van der Maaten explained that:

"The similarity of datapoint x_j to datapoint x_i is the conditional probability, $p_{j|i}$, that x_i would pick x_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at x_i." [26].

Being N the number of features and x_i the object explained by the i^{th} feature, t-SNE compute the following probabilities:

$$p_{j|i} = \frac{\exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|\mathbf{x}_i - \mathbf{x}_k\|^2 / 2\sigma_i^2\right)} \tag{2.4}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N} \tag{2.5}$$

t-SNE goal is to reflect the probabilities p_{ij} in a 2D or 3D map as better as possible. To do so, a Student t-distribution is used to estimate similarities q_{ij} among 2 or 3 dimensional objects.

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}} \tag{2.6}$$

The Kullback Leibler divergence of these 2 distributions is expressed by equation 2.2 and the result lead to the location in the low dimensional map of the points.

$$KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{2.7}$$

In figure 2.6 is possible to notice how, nevertheless the power of such technique, the samples are strictly overlapped each other. There is however a trend and at least for samples belonging to class 0 and class 3 (except for some samples) there is a separation. The boundaries among the classes however is really not strict and many points of different class have features similar to other classes.

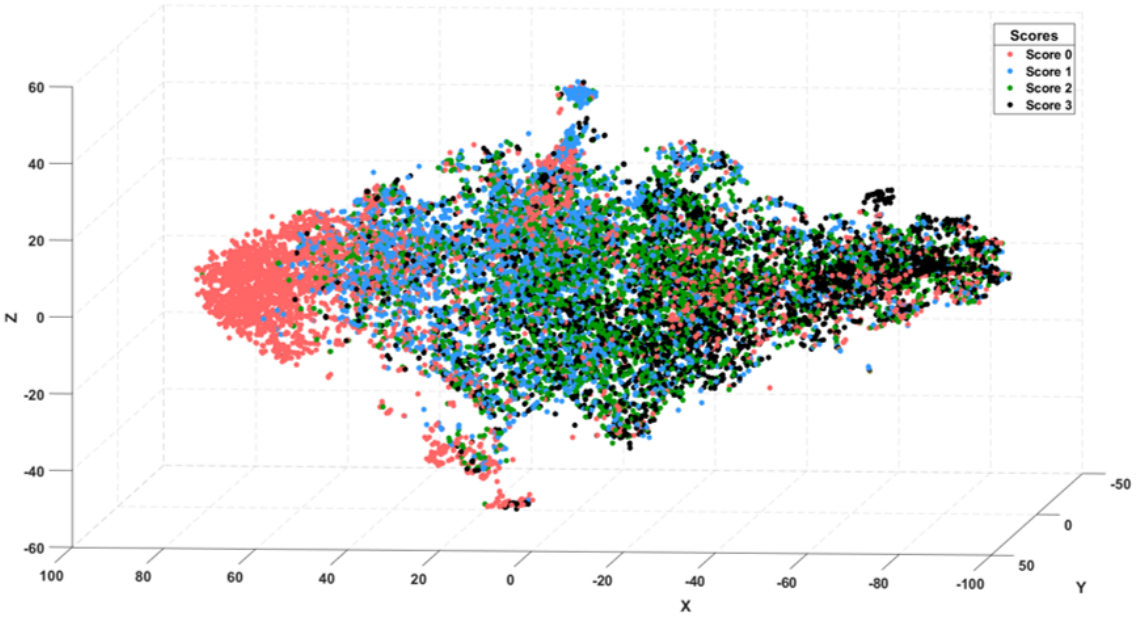


Figure 2.6: t-SNE on down-sampled dataset

This can be explained highlighting the fact that the signals are only accelerometric signals and probably there is not enough information for a proper classification. Moreover the sensors were placed in the only extremity of the limbs, this means that if the dyskinetic event occurred in the proximal site of the limb the sensor might not sense the event at all.

2.8 Models

2.8.1 Random Forest

Details about how Random Forest work and what is it can be found in appendix A. This subsection will focus on the choice of some parameters useful to make the model work properly.

Choice of parameters

The main parameters chosen for make the model work are the number of trees present in the forest. This number usually vary from few dozens to hundreds or thousands of trees. However the more the trees the more the model will be over-fitted. There is a way to estimate the proper number of trees needed.

The **Out Of Bag Error** can be used to chose the correct number of trees for training the model.

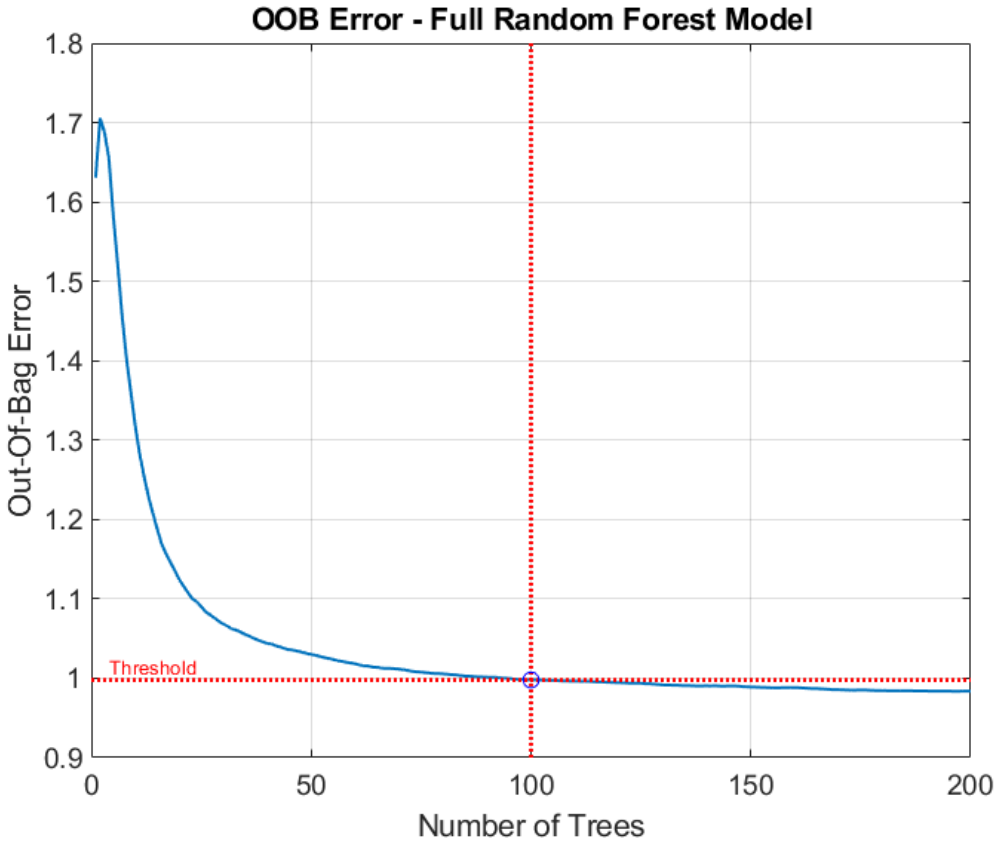


Figure 2.7: OOB to decide the number of trees

In this work 100 trees are sufficient since adding other trees won't decrease that much the OOB error.

2.8.2 LSTM

For the Long Short Term Memory is harder to perfectly set all the parameters to make the model work. Moreover the LSTM require a pre-processing of the data that is not required from the only Random Forest.

Pre-process of data sequences

There are different pre-processing for the data depending on which will be the input to the net. Keeping into account that the 3 main methods have been:

1. **Feed the net with raw data**

   This approach doesn't need a particular pre-process if the data is fed already windowed. Indeed the sequences will have the same length and the LSTM won't have any issue. However a common practice even for this approach would be to standardize the tasks. More on *standardization* will be explained in the next item.

2. **Feed the net with extracted/selected features**

   This approach require **standardization** to be computed. Indeed the LSTM to work at its best require the data with mean centered in 0 and a standard deviation equal to 1.

   $$z(x) = \frac{x - \mu}{\sigma} \tag{2.8}$$

   where μ is the mean and σ the standard deviation. This transformation is generally required when working with RNN.

   After the standardization is computed, the sequences have to be ordered in order to avoid a data loss or a data corruption depending on the method chosen to equalize the length of the sequence in each mini-batch.

   The training session will occur in mini-batch, each of these mini-batch must have all the sequences with the same number of elements (meaning that all the temporal sequences must have the same duration). In our case this is not true, since different task can have different duration. To solve this problem 2 choices can be done:

   - **Zero-padding**: to all the sequences with shorter duration of the longest in the batch will be added a sequence of 0 such as eventually their length will be the same of the longest and all the sequences in the batch will have same number of elements
   - **Shortest**: By choosing this approach, all the sequences longer than the shortest in the batch will be cut to the same length of the shortest.

   Zero-padding is introducing some corruption in the sequence but is not wasting any data. The Shortest method instead is not introducing any noise but is keeping out some information that could be valuable.

   Normally the sequences are mixed before being fed to the net. This approach however increase the corruption of data in case zero-padding is chosen and increase the data loss if the length of the shortest one is the one of the entire mini-batch. So it's important to **order** the sequences previously. Then a proper mini-batch size will be chosen.

3. **Feed the net with the output of the Random Forest** In this case the ordering of the sequences is necessary. However no data standardization is required.

Figures 2.8 and 2.9 represents respectively the effect of zero-padding and truncation on unordered and ordered sequences.

Choice of parameters

The parameters used to train the nets are reported in the Matlab snippet code. This architecture is not the only one tried. Many different architectures have been tried, eventually this has been chosed because 3 layers is the general suggestion for RNN

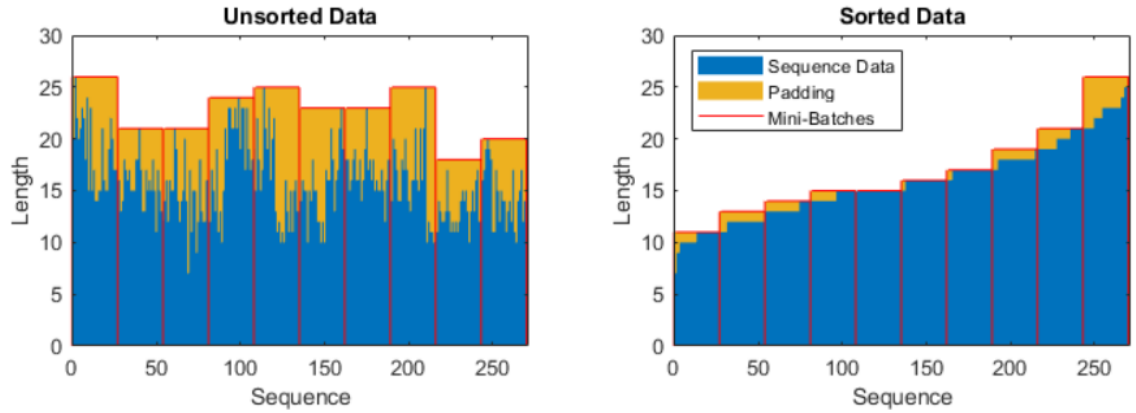


Figure 2.8: Zero-padding to the longest sequence [9]

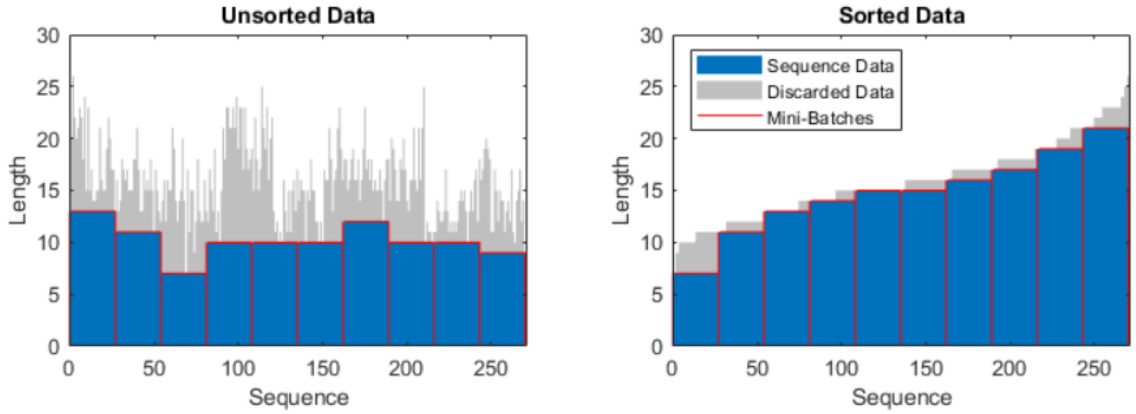


Figure 2.9: Truncation on the shortest sequence [9]

in Human Activity Recognition (HAR) classification problems. However the performance don't change significantly with any change of the architecture (except obviously structures without any sense).

What has been noticed is that by increasing the number of layers the computational time notably increase while the performance don't change that much. Hence the architecture used eventually will use only one layer, however if computational times do not constitute a problem, 3 layers is the best option.

```
inputSize = NumberFeatures;
numHiddenUnits1 = 80;
numHiddenUnits2 = 100;
numHiddenUnits3 = 60;

layers = [ ...
    sequenceInputLayer(inputSize)
    bilstmLayer(numHiddenUnits1, 'OutputMode', 'sequence')
    bilstmLayer(numHiddenUnits2, 'OutputMode', 'sequence')
    bilstmLayer(numHiddenUnits3, 'OutputMode', 'last')
    dropoutLayer(0.3)
    fullyConnectedLayer(1)
    regressionLayer]
```

```
14
15 %% Train the network with architecture and options specified
16 net = trainNetwork(X, Y, layers, options);
```

- **Learning Rate**: 0.005
- **Learning Rate Drop Factor**: 0.5
- **Learning Rate Drop Period**: 10 epochs
- **Number of epochs**: 200
- **Validation**: Leave one subject out
- **L2 regularization factor**: 0.0005
- **Batch size**: 32
- **Optimization algorithm**: adam
- **Batch uniformation**: truncation to shortest

LSTM architecture

The proposed architecture uses 3 layers of BiLSTM, the first 2 are used with "Output" set to "Sequence", the last is set to "Last", meaning that we are interested on classification. However the final layer will be the *RegressionLayer*, this layer translate the output to the continuous and will make the net conscious of the fact that the loss to minimize is the Mean Squared Error. In detail the proposed network is resumed by the following layers:

2.9 Difficulties

The main problem to tackle for the recognition of the windows is the fact that the dyskinetic event can occur as a burst and hence can be present only in one or few windows, while all the others of the task will not present dyskinetic behavior, however since all the windows of the task have the same associated score, some of them have an intrinsic error.

Lee et al.[23] proposed a method to address the potential mismatch between the characteristics of accelerometric data segments of 5 seconds length and the UPDRS score given from the rater to the entire task.

In his work he used the WEKA implementation of Expectation Maximization (EM) to cluster the dataset and remove all the samples that overlap in other clusters, then a RF regressor has been used to predict the membership of new windows. This approach relies on the robustness of the RF to overfit for facing the problem of unbalanced data.

However that approach has been mainly used for tremor recognition and the conditions are slightly different. Indeed in tremor recognition the assumption is to work only in "resting" intervals, meaning that all the movements recognized as voluntary are not evaluated. Moreover tremor usually has a certain level of continuity and, if the task has a severity different from zero there is a good possibility that the tremor has been present throughout the task.

The clustering approach formed as first step 2 main clusters that (by looking the dimensions of the clusters) were consistent with the "resting" and "not resting" classes. However this behavior doesn't allow the correct identification of the 2 clusters "Dyskinesia" and "Not dyskinesia". Moreover it was a bit critical for the data, indeed by applying this procedure most of the data was unused and the training data was too little to properly train a model.

2.9.1 Voluntary movement prediction

Dyskinesia can be very similar to voluntary movements from an accelerometric point of view. The filter applied to the raw data in the band from 1 Hz to 2.8 Hz should remove the components of the movements related to gross changes in the body motion, however it removes only part of the voluntary movements. A model based on the previous classifier is proposed. Figure 2.10 is the schema for assessing the label to the unlabeled data. Indeed since no evaluation has been computed to classify movements between voluntary and not voluntary, an algorithm has to be implemented. The following assumptions are done:

- If the maximum among the range of the 3 accelerometric channels is below a certain heuristic chosen threshold (0.1 m/s^2) then we can consider that no movement at all is happening, hence the window is no further considered.

- A further and better investigation to understand if a movement is present, is then done by the model for resting prediction. If the limb is resting, no matter if there is dyskinesia or not, there isn't a voluntary movement.

- When the movement is detected, it can be both dyskinetic or purely voluntary. To distinguish between them, the dyskinetic score given by the clinician is observed.

After having given the labels to the dataset, a Random Forest classifier has been trained with Leave one subject out technique. This is done to leave the prediction as more general as possible. The new prediction will be the real label assigned to each window. This is done to keep into account also those voluntary movements done in presence of dyskinesia.

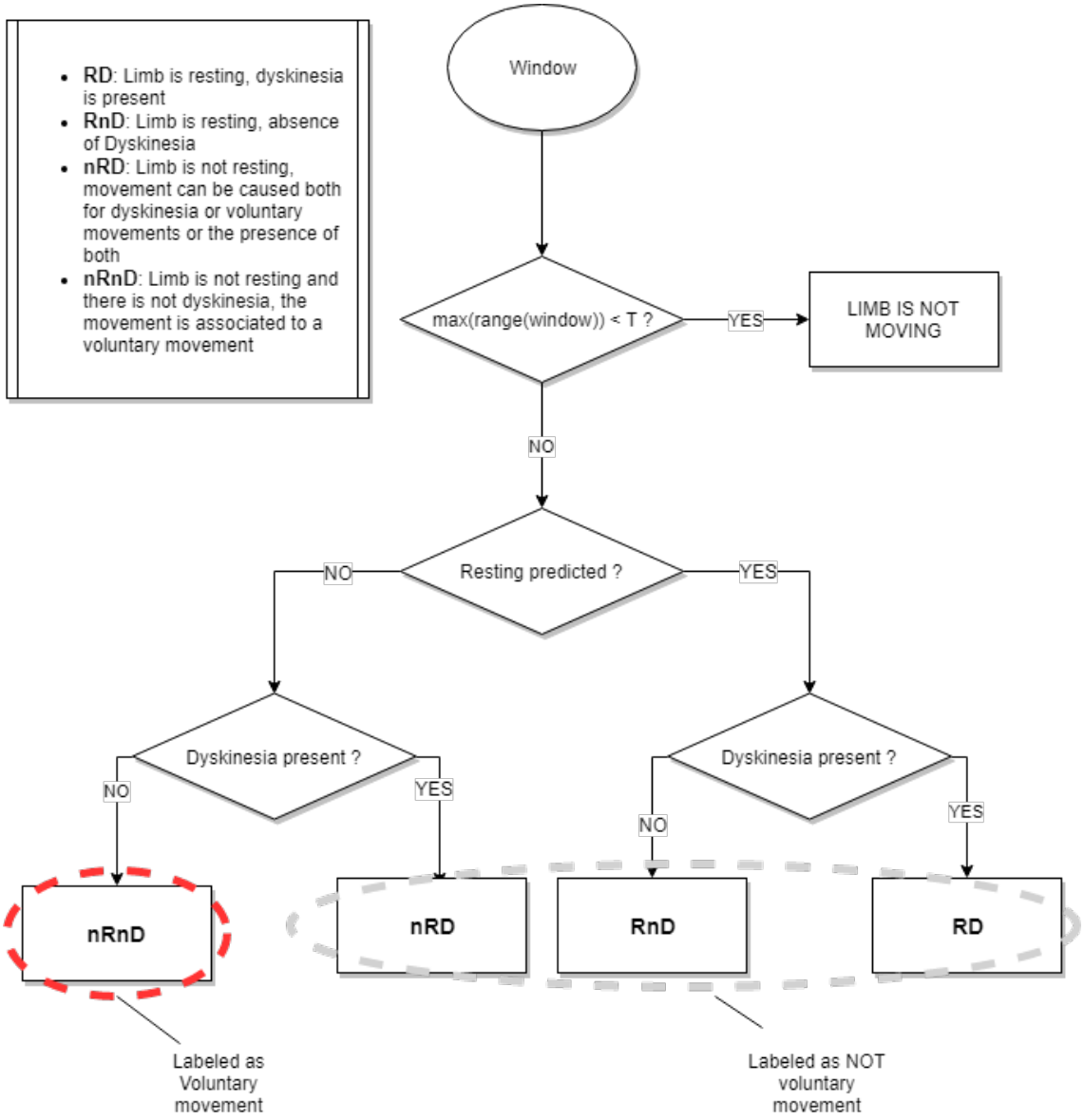


Figure 2.10: Voluntary movement schema for label assessment

2.10 Regression or Classification

One of the main problem was to understand if it is better to tackle the problem by doing a regressor model or a classifier one.

2.10.1 Regression

Regression is generally used when continuous predictions are required. As an example, if you want to predict the height of a person based on some geographical and personal characteristics other than height, you probably need a regressor model. However if you want to categorize people in classes such as “short”, “normal height” or “high”, you need a classifier. In this example is possible passing from regression to classification by setting a threshold in the predicted height.

In our problem, the specifications of the project were to obtain a continuous output, meaning we need a regression approach. For regression, the most used indicators are MAE and RMSE:

Mean Absolute Error (MAE)

Let’s call our ground truth θ and our prediction $\hat{\theta}$, we can define the error of the prediction as the difference between them. This error can be positive, if the prediction is greater than the truth or negative viceversa. However we are mostly interested in understand the absolute value of this error for each predicted value compared with the respective ground truth. Hence we sum all the absolute value of the errors e_i and then we average the results.

$$\text{MAE} = \frac{\sum_{i=1}^{n} \left|\theta_i - \hat{\theta}_i\right|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}. \tag{2.9}$$

This is a very used indicator since its meaning is easily interpretative.

Root Mean Squared Error (RMSE)

In this indicator there is a squared operation instead of the absolute value for the error. This means that large errors will influence much more than smaller ones.The Mean Squared Error (MSE) is the expected value of the squared error.

$$\text{RMSE}(\hat{\theta}) = \sqrt{\text{MSE}(\hat{\theta})} = \sqrt{\text{E}((\hat{\theta} - \theta)^2)}. \tag{2.10}$$

This indicator is penalizing much more all the predictions that are are further from the real value. In our problem this is a good indicator since it’s not the same mistaking between severity 1 and 2 or between 1 and 3. RMSE always for the same reason is very influenced from the outliers.

2.10.2 Classification

Classification instead is used in problems where the final wanted output is a label. It's full of different indicators for classification problems. In this work classification is mainly used for binary classification such as predictions regarding voluntary movements or the model for distinguish among dyskinetic and not dyskinetic movements. To explain the variety of indicators used in classification problems, let's firstly introduce the following terminology valid for a binary classification:

- **True Positive (TP)**: is the amount of elements truly predicted as positive
- **False Positive (FP)**: is the amount of elements predicted as positive but that were negative
- **True Negative (TN)**: is the amount of elements truly predicted as negative
- **False Negative (FN)**: is the amount of elements predicted as negative but that were positive

All these amounts have a place in the so called Confusion Matrix reported in figure 2.11: Starting from these 4 quantities reported in the Confusion Matrix, a really big

		Predicted Class	
		Normal	Attack
Actual Class	Normal	True Negative (TN)	False Positive (FP)
	Attack	False Negative (FN)	True Positive (TP)

Figure 2.11: Binary Confusion Matrix

number of indicators can be calculated, the principals are reported in table 2.5.

Table 2.5: Binary classification indicators

Indicator name	**Mathematical formula**
Accuracy	$\mathrm{ACC} = \frac{\mathrm{TP}+\mathrm{TN}}{P+N} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$
False Negative Rate (FNR)	$\mathrm{FNR} = \frac{\mathrm{FN}}{P} = \frac{\mathrm{FN}}{\mathrm{FN}+\mathrm{TP}} = 1-\mathrm{TPR}$
False Positive Rate (FPR)	$\mathrm{FPR} = \frac{\mathrm{FP}}{N} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}} = 1-\mathrm{TNR}$
Sensitivity (TPR)	$\mathrm{TPR} = \frac{\mathrm{TP}}{P} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} = 1-\mathrm{FNR}$
Specificity (TNR)	$\mathrm{TNR} = \frac{\mathrm{TN}}{N} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}} = 1-\mathrm{FPR}$
Positive Predictive Value (PPV)	$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} = 1-\mathrm{FDR}$
Negative Predictive Value (NPV)	$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FN}} = 1-\mathrm{FOR}$
False Discovery Rate (FDR)	$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TP}} = 1-\mathrm{PPV}$
False Omission Rate (FOR)	$\mathrm{FOR} = \frac{\mathrm{FN}}{\mathrm{FN}+\mathrm{TN}} = 1-\mathrm{NPV}$
F1 Score	$F_1 = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV}+\mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}$

Chapter 3

Results

Different methods and models have been tried to get the lowest error possible during predictions. Following the results in terms of Confusion Matrixes (CM) and main indicators of the principal approaches are reported.

3.1 Random Forest model

3.1.1 Binary classification

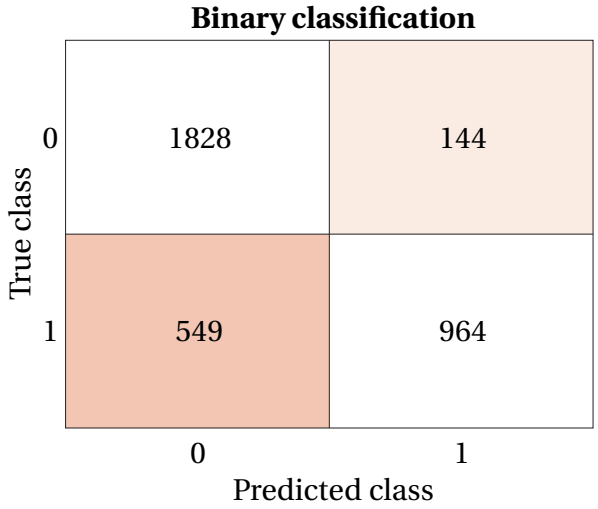


Figure 3.1: RF performances in binary classification

Accuracy: 89.3%
Sensitivity: 63.7%
Specificity: 97.1%

3.1.2 Multi-class regression

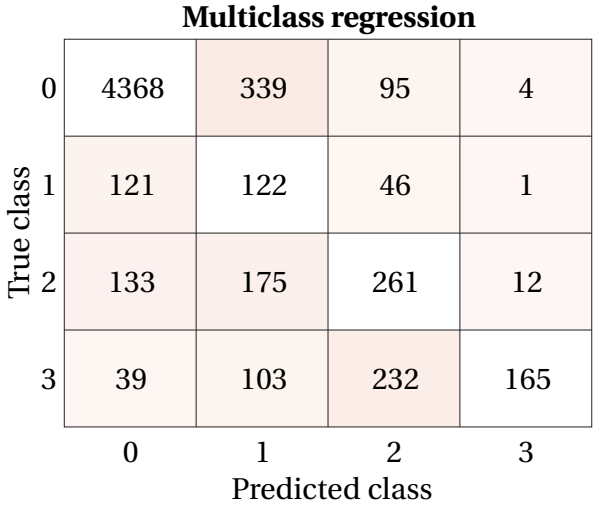


Figure 3.2: RF performance in multi-class regression

For multi-class classification, specificity and sensitivity have been calculated for each class to show the differences. Then to summarize the overall performance, the results have been averaged.

Table 3.1: RF performances

	Sensitivity	**Specificity**	**Precision**	**F1 Score**
Class 0	0.91	0.79	0.94	0.92
Class 1	0.42	0.90	0.17	0.24
Class 2	0.45	0.93	0.41	0.43
Class 3	0.31	0.99	0.91	0.46

Accuracy: 79.1%
Sensitivity: 52.1%
Specificity: 90.5%
Precision: 60.5%
F1 Score: 51.2%

In terms of regression the RMSE and MAE are reported:

RMSE: 0.64
MAE: 0.29

3.2 LSTM model

3.2.1 Raw data performance

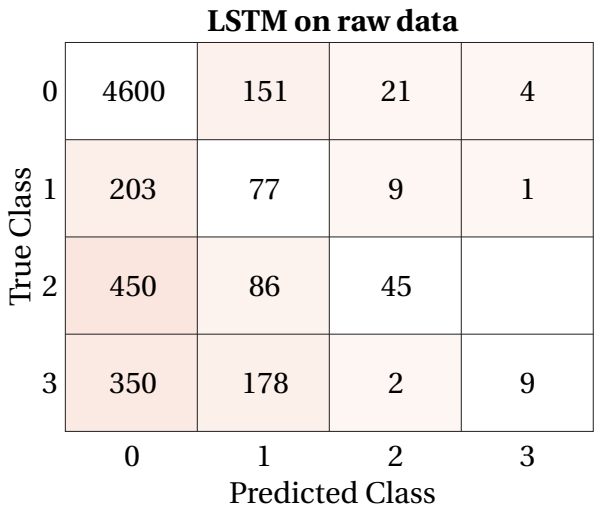


Figure 3.3: LSTM performances on raw signals

Table 3.2: Performance LSTM on raw signals

	Sensitivity	**Specificity**	**Precision**	**F1 Score**
Class 0	0.96	0.29	0.82	0.88
Class 1	0.27	0.93	0.16	0.20
Class 2	0.07	0.99	0.58	0.14
Class 3	0.02	0.99	0.64	0.03

Accuracy: 76.5%
Sensitivity: 33.1%
Specificity: 80.3%
Precision: 55.1%
F1 Score: 31.3%

In terms of regression the RMSE and MAE are reported:

RMSE: 1.12
MAE: 0.89

3.2.2 Features based performance

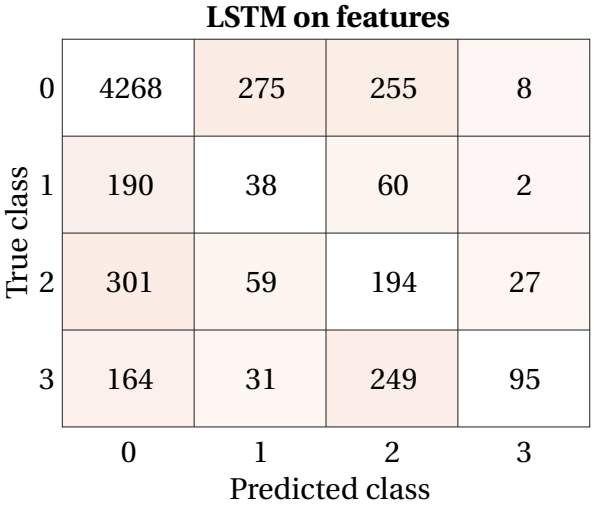


Figure 3.4: LSTM performance on features

Table 3.3: Performance LSTM on features of the signal

	Sensitivity	**Specificity**	**Precision**	**F1 Score**
Class 0	0.87	0.58	0.89	0.88
Class 1	0.09	0.96	0.13	0.11
Class 2	0.26	0.93	0.33	0.29
Class 3	0.72	0.93	0.18	0.28

Accuracy: 73.9%
Sensitivity: 48.4%
Specificity: 84.9%
Precision: 38.2%
F1 Score: 39.0%

In terms of regression the RMSE and MAE are reported:

RMSE: 0.85
MAE: 0.41

3.2.3 LSTM on RF output

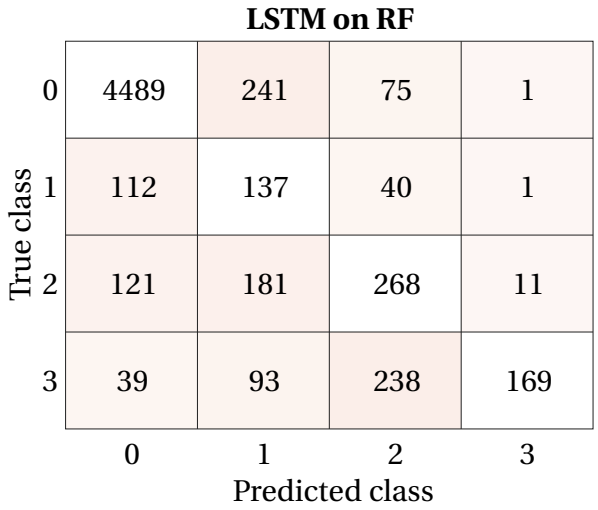


Figure 3.5: LSTM performance on RF output

Table 3.4: Performance LSTM on RF

	Sensitivity	**Specificity**	**Precision**	**F1 Score**
Class 0	0.94	0.78	0.93	0.94
Class 1	0.21	0.97	0.47	0.29
Class 2	0.43	0.94	0.46	0.44
Class 3	0.93	0.94	0.31	0.47

Accuracy: 81.4%
Sensitivity: 62.8%
Specificity: 91.0%
Precision: 54.5%
F1 Score: 53.6%

In terms of regression the RMSE and MAE are reported:

RMSE: 0.55
MAE: 0.26

3.3 Apartment data

In the data apartment, only the best models are tried. Hence, the only Random Forest and the LSTM after the RF are confronted.

3.3.1 Random Forest multiregression

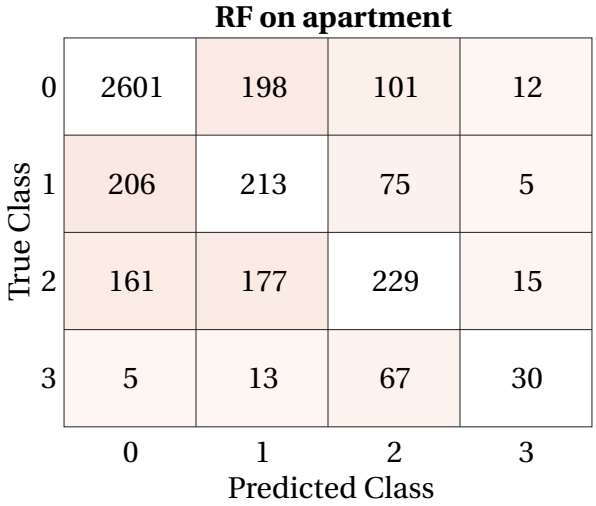


Figure 3.6: RF on apartment data

Table 3.5: Performance RF on apartment data

	Sensitivity	Specificity	Precision	F1 Score
Class 0	0.87	0.75	0.89	0.88
Class 1	0.34	0.92	0.43	0.38
Class 2	0.43	0.90	0.39	0.41
Class 3	0.61	0.96	0.23	0.34

Accuracy: 73.5%
Sensitivity: 56.5%
Specificity: 88.3%
Precision: 48.6%
F1 Score: 50.3%

In terms of regression the RMSE and MAE are reported:

RMSE: 0.69
MAE: 0.38

3.3.2 LSTM on RF - Apartment

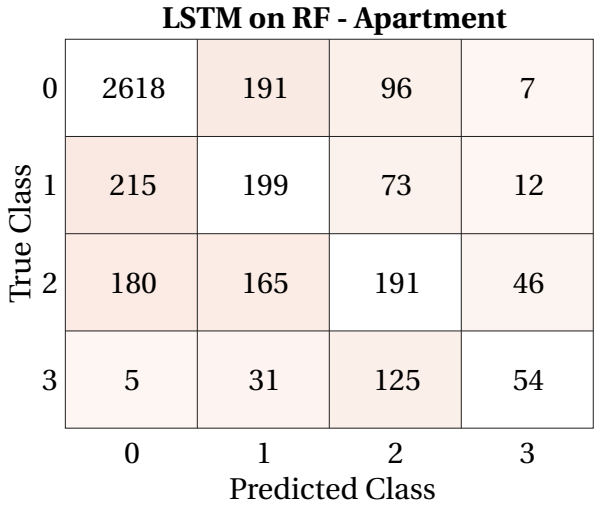


Figure 3.7: LSTM on RF for apartment data

Table 3.6: Performance LSTM on RF for apartment data

	Sensitivity	**Specificity**	**Precision**	**F1 Score**
Class 0	0.87	0.75	0.90	0.89
Class 1	0.34	0.92	0.40	0.37
Class 2	0.39	0.89	0.33	0.36
Class 3	0.45	0.96	0.25	0.32

Accuracy: 72.7%
Sensitivity: 51.3%
Specificity: 88.1%
Precision: 46.9%
F1 Score: 48.3%

In terms of regression the RMSE and MAE are reported:

RMSE: 0.90
MAE: 0.46

3.3.3 Error per subject

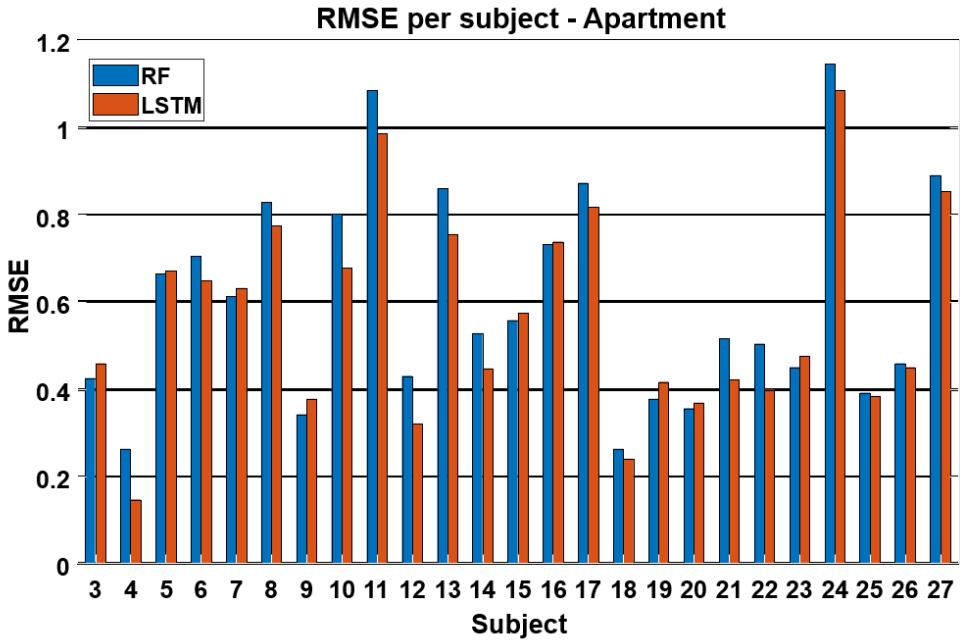


Figure 3.8: RMSE per subject

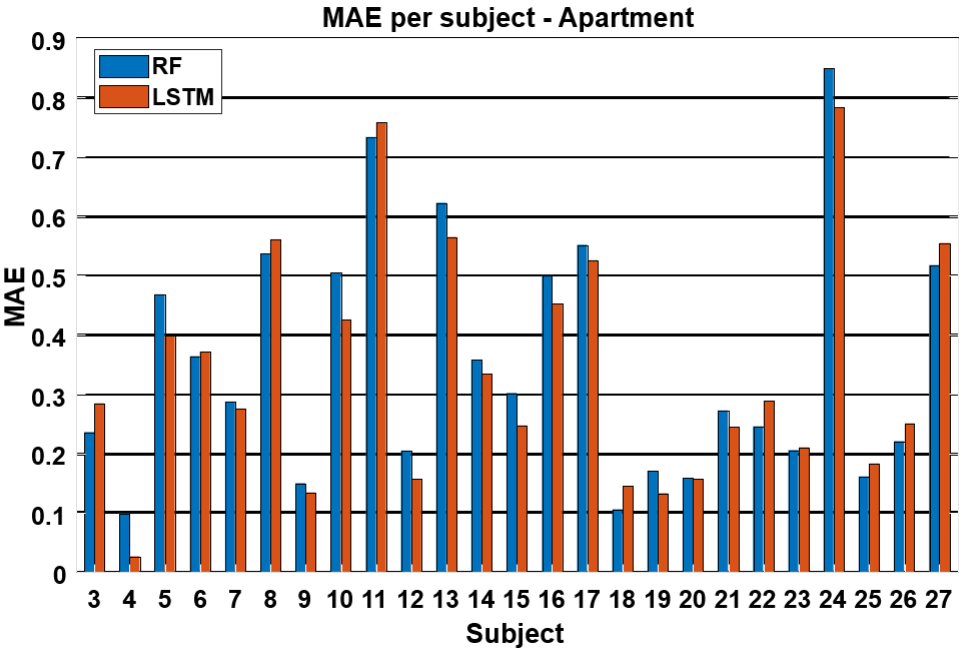


Figure 3.9: MAE per subject

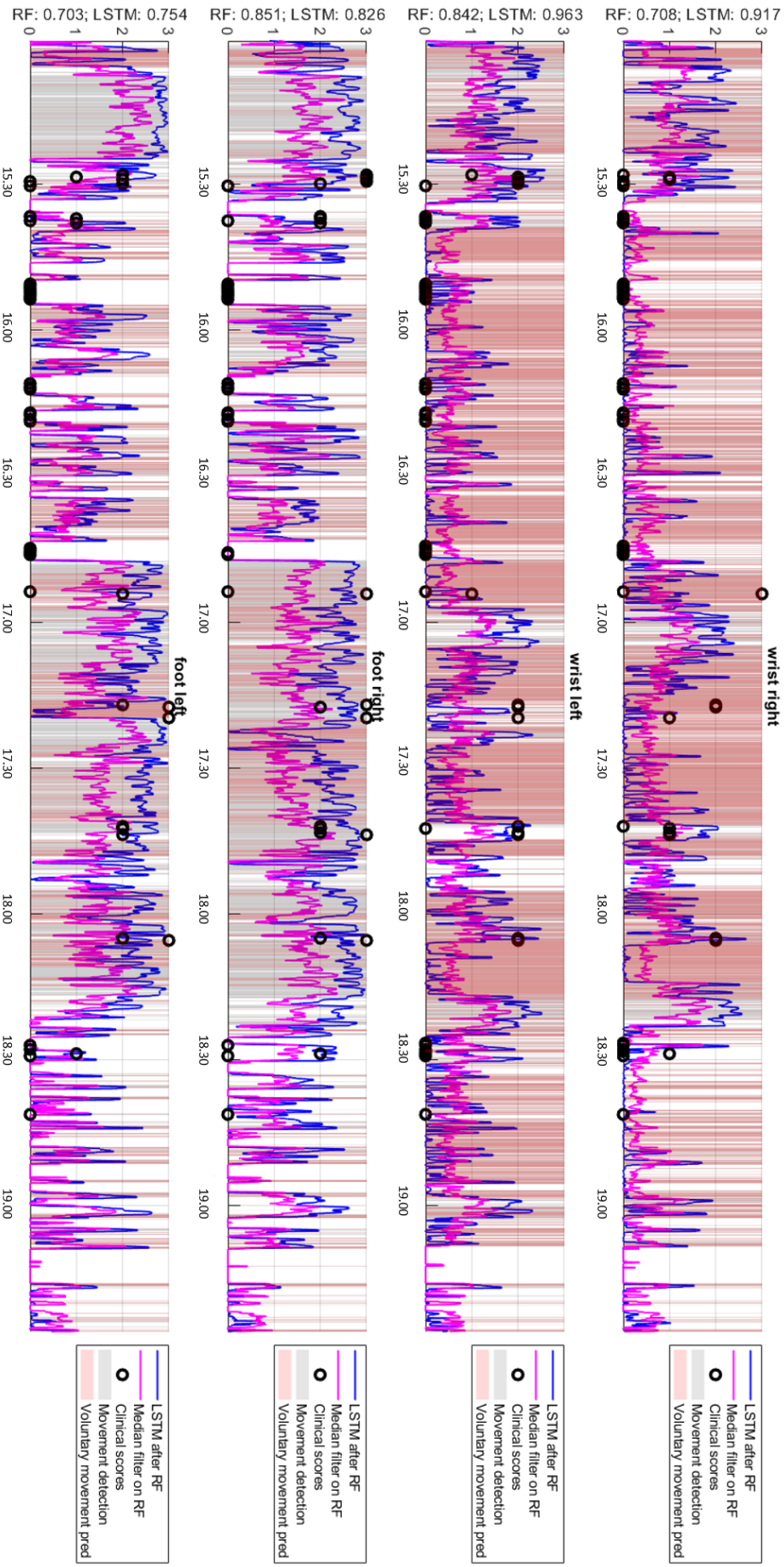


Figure 3.10: Total data series

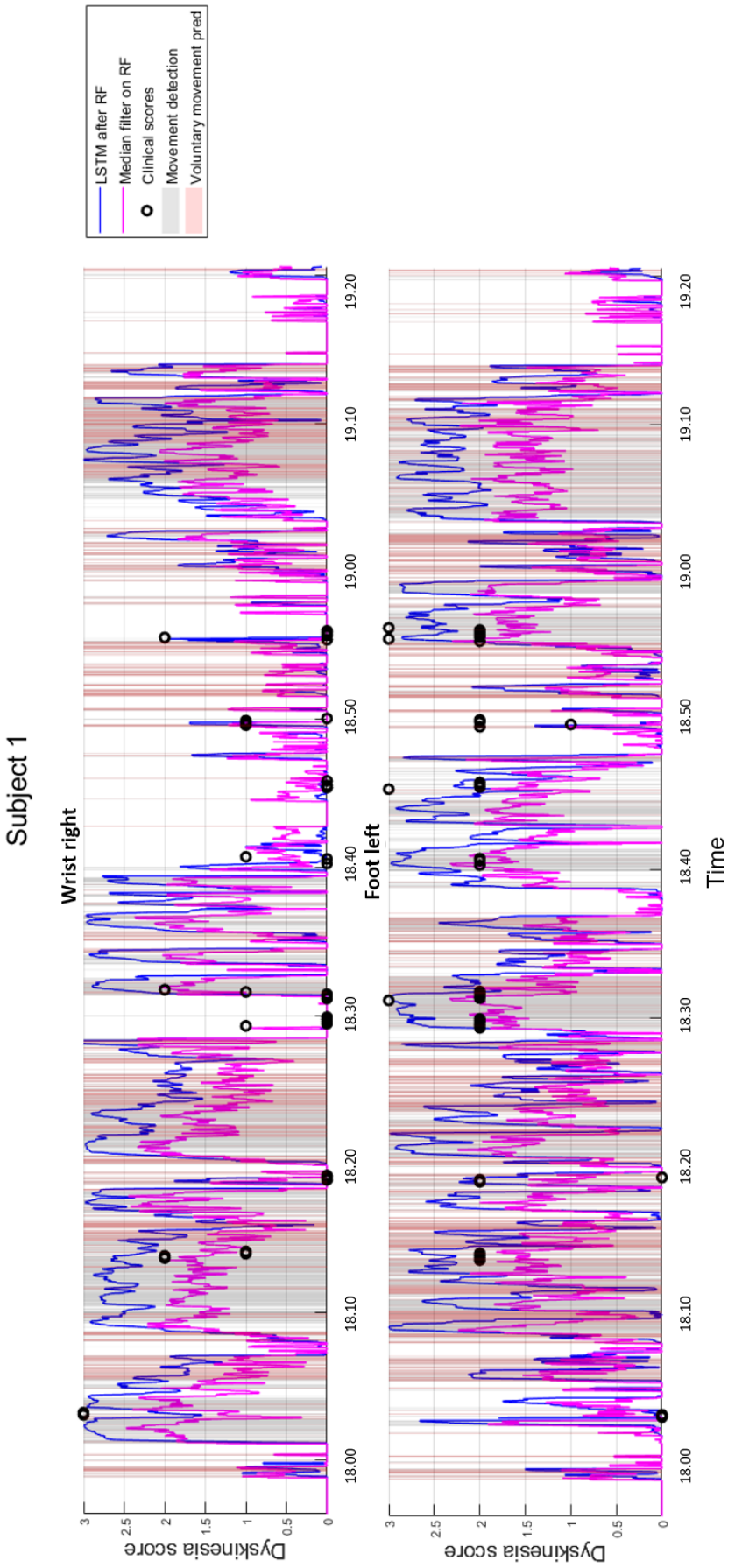


Figure 3.11: Data series

Chapter 4

Conclusions

To conclude, the performance of the proposed models are still too poor to be used in real-life scenario. The lack of sensitivity makes the classifier too unreliable. However this research has been conducted on the basis of only accelerometric sensors. This was a specification to be respected and it was just an attempt to see if, with modern deep learning techniques, the information inside the accelerometric signals was enough to perform with good performance the classification. Usually the motor symptoms are studied using not only the accelerometer but also magnetometer, gyroscope and many times also Electromyography and/or Vertical Ground Force sensors (these last sensors are actually mainly used for Freezing of Gait or other forms of akinesya).

Other interesting fact is the evaluation of the LSTM net on the output of the RF. This idea has been elaborated to give a sense of temporal continuity of the predictions of the Random Forest. It's interesting notice how the LSTM improve the results of the RF in the laboratory dataset while its effect is not relevant in the apartment dataset. This can be explained by the fact that in the laboratory the tasks were repeated different times and since they were standardized a certain similar pattern among the subjects is present in the same task. On the other hand, the subjects were totally free in the apartment environment, hence the resulting temporal sequence was different from the laboratory. Since the LSTM was trained on the predictions of the RF but in the lab, it makes sense that it performs better than the median of the predictions in the lab but not in the apartment where the conditions changed.

Nevertheless this work showed how wearable sensors and modern techniques of Artificial Intelligence can be very useful in the medical field not only for medical images and diagnosis but also in the field of medical signals and their classification even in a very complex scenario like the prediction of severity of LID. It's important to highlight the fact that in recognition of these symptoms there is also an intrinsic error due to the fact that different clinicians can disagree in the severity of dyskinesia of a movement. Indeed the experience of the clinician play a key role in its scores. The doctor will tend to assign the worst possible score to the subject who present the highest level of impairment he has ever seen, however another doctor, with usually more experience

won't tend to assign the highest severity very easily.

Finally the results obtained in the Apartment environment are the first ever. To best in my knowledge, no other studies ever tried to predict Levodopa Induced Dyskinesia in a totally free environment. All the precedent studies reported results in terms of scripted tasks in the laboratory environment.

Clinical observation for drug titration is for now not intended to be overtook by algorithms and machines. It's a too delicate issue and a clinical decision is still necessary. However in a future, thanks to the development of always more sophisticated sensors and understanding of physiological data and motor patterns would be possible to deploy an AI-powered DSS (Decision Support System) for clinicians for all the different motor symptoms of Parkinson's disease.

Chapter 5

Future Outlooks

Since it's actually the first time an experiment like this (in the apartment) has ever been conducted, it's plenty of future works and improvements. For sure it's very possible to improve the Recurrent Neural Network by a better understanding of the architecture and of the parameters to set the net. However the real future improvement would be to consider not only the accelerometric data but also data from other sensors, indeed the gyroscope can be very useful to understand the real orientation of the limbs and this can be a key feature for predict dyskinesia.

In my opinion the idea of training a model in data obtained from scripted tasks is a bit limitative if then the model has to be tested in real-life scenarios. What I would suggest as future work is to have a *training set* obtained in similar conditions of the *test set*. This means that the data acquisition would both be taken in a simulated apartment full of cameras and then the dyskinetic movements will be rated by experts. However the data should be enough to train a model.

Another major issue was the lack of true positives. There were too many true negatives and few cases of dyskinesia. Moreover all the cases of dyskinesia were overlapped in the space of features. Probably a bigger dataset with more clear true positive would be the best for a proper development of the model.

Finally, many other paths can be tried in the Machine Learning field, it's plenty of different Deep Learning techniques that might be more efficient than the Random Forest. The investigated one (LSTM) didn't lead to great results, however other paths could be tried. To itemize some of them:

- CNN 1D for signal analysis
- GRU instead of LSTM
- Autoencoders for feature extraction
- Boltzman Machines for feature extraction

Another limitation could have been the sensors location. It's possible that since they were in the body extremity, they weren't that sensible to dyskinetic movements oc-

curring in the proximal part of the limb. For instance if a sever dyskinetic movement occurred on the thigh, the clinician rated the symptom as 3, however it's possible that in the foot the movement was near to be null. In similar way, if the dyskinetic movement appear on the thigh and the real score is 3, the sensor will now sense a different signal respect to the dyskinesia present in the thigh. This will confound a lot the model since it will see signal with different features have the same score and signals with similar features have the same score.
To try to fix this issue, more sensors may be needed.

Otherwise a total different and new approach could be used to tackle this problem, instead of focusing on the mere signals acquired by different body locations or even physiological signals, since clinicians rates the these symptoms based on videos an AI could do the same.

Indeed we can discuss infinitely about the sufficiency or not of the information contained in accelerometric signals. The fact is that since the clinician is rating a video, the video has for sure the information to make that decision. Hence a total different model would be needed. A real future outlook could be the development of DSS (Decision Support Systems) based on Artificial Intelligence that could be connected on cameras in the very apartment of the subject. This would avoid the encumbrance of some wearable devices but could creat some bureaucratic and privacy issues.

The alternative could be standardize the apartment and the camera location and ask to the subject to arrive in the location and set him free to do whatever he wants while observed by cameras connected to the AI.

Appendix A

Random Forest

Random Forest is an ensemble method created from decision trees. An ensemble is a way to use many different weak learners and combine their contribute in a way to create a better model. In the **classification** setup of the ensemble, the different learners give their contribute by *voting* their result, the final result will be the one which received more votes. In the **regression** setup the different results are *averaged*.

A.0.1 Decision Tree

A decision tree is a simple model where many different decisions are taken in a binary mode. Each decision brings to a branch with a different path of further decisions, the input and the output are connected by this path of different decisions usually based on simple thresholds. Traditionally decision trees can be create manually, their power is in the easy explication and comprehension of the connection between input and output. A decision tree can be interpreted as a flowchart-like structure where each node test a feature of the dataset, each branch is the result of the test and each leaf is the final class (*classification tree*) or value (*regression tree*).
All classification tree algorithms have to address two common problems:

"How to split a node t and when to stop splitting it?"
The term **Classification And Regression Tree (CART)** is a hypernym used to refer to both regression and classification decision trees. These trees are based on the same general idea but they slightly differ on the procedure used to determine where and how the branch is split [16].
C4.5 and CART follow the exhaustive search approach where the basic steps are exposed in algorithm 4. Being X the set of predictor variables and S a subset of the values taken by X, in such a way that $S \in X$. [25]

The stop criterion in algorithm 4 is related to the level of impurity of the child nodes. The metric for impurity depends on the algorithm used, nowadays the scikit-learn tool in Python and the Matlab functions to create DTs, both use the CART algorithm which use the Gini impurity. There are literally tons of different algorithms for choos-

Algorithm 4 Pseudocode for tree construction - **Exhaustive search**

1: Start at the root node
2: **for** each X **do**
3: Find the set S that minimizes the sum of the node impurities in the two child nodes and choose the split $S^* \in X^*$ that gives the minimum overall X and S
4: **if** Stopping criterion is reached **then**
5: Exit
6: **else**
7: Apply step 2 to each child node in turn

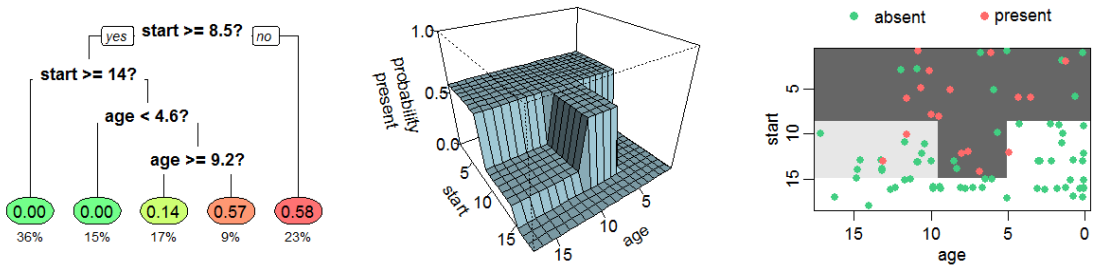


Figure A.1: This is an example of a decision tree structure. **Left**: the leaves show the probability of a certain event based on the decisions upon the input variables. **Middle** 3-dimensional plot of the decision tree based on the path represented on the figure in the left, **Right**: Aerial view of the middle plot. The chances of the event studied is higher in the darker areas. [28]

ing how and where split a branch. During this work, the Random Forest has been developed on the basis of Decision Trees created with the CART algorithm. **Gini impurity function**: It's an indicator of how often an item chosen randomly from the subset is mislabeled if it was labeled in a random way according to the distribution of labels in the subset. Supposing $i \in \{1,2,...,J\}$ being J the total number of classes and p_i the part of elements labeled as i in the subset. The Gini impurity represents the sum of probability $\sum_{i=1}^{J} p_i$ of an element labeled as i times the probability $\sum_{k \neq i} p_k = 1 - p_i$ of mistaking during classification of the element.

$$I_G(p) = \sum_{i=1}^{J} p_i \sum_{k \neq i} p_k = \sum_{i=1}^{J} p_i(1-p_i) = \sum_{i=1}^{J} (p_i - {p_i}^2) = \sum_{i=1}^{J} p_i - \sum_{i=1}^{J} {p_i}^2 = 1 - \sum_{i=1}^{J} {p_i}^2$$

Decision trees are very easy to implement and are useful when the problem is easy to model and usually few features are involved. However as a drawback they easily tends to overfit on the training set, hence their generalization power is little. Here different procedures can be used to enhance this aspect.

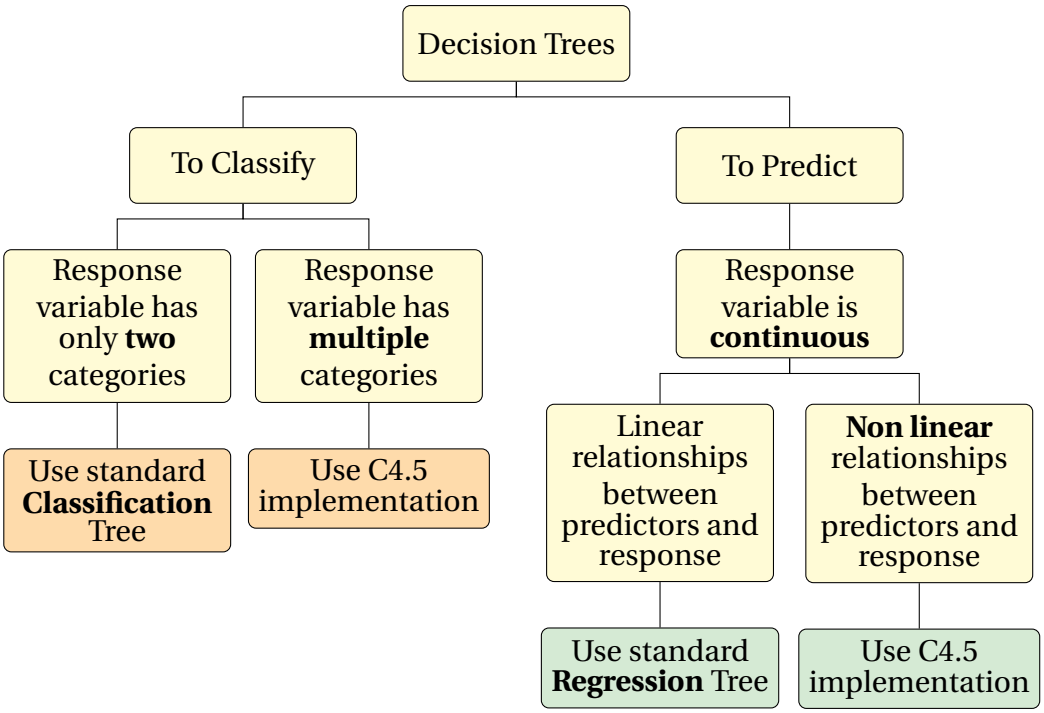


Figure A.2: How to choose the algorithm for Decision Tree

A.0.2 Bagging

The word *bagging* comes from **b**ootstrap **agg**regation. The general idea consists in splitting the data in random different ways, feed these subdata to decision trees, each of them will provide different results, to reduce the variance the results will be averaged if a regressor is desired or a voting system will be used if a classifier is preferred instead.

IDEA: build a number of decision trees on bootstrapped training samples and, each time a split in a tree is considered, randomly chose m predictors (as split candidates from the full set of M predictors.

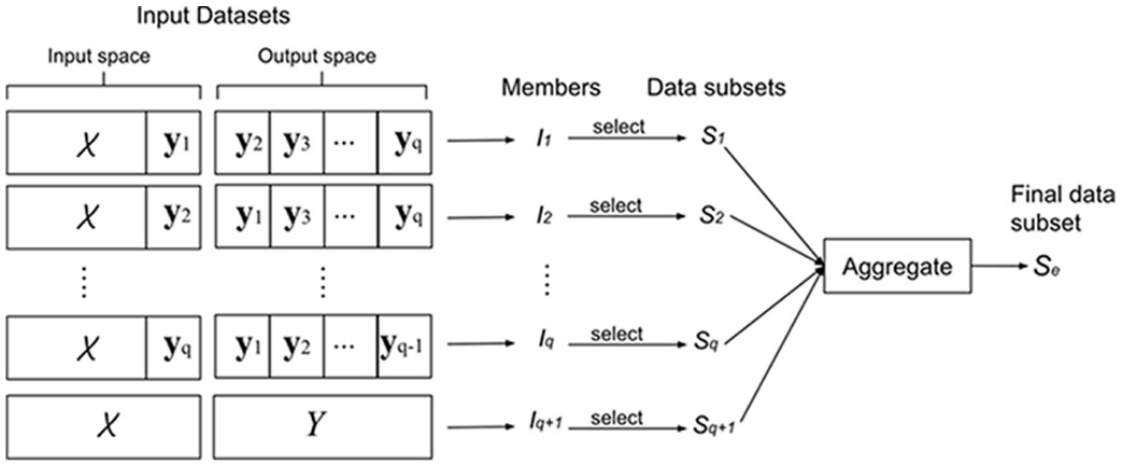


Figure A.3: General Ensemble schema [31]

Appendix B

Neural Networks for Time Series

One clear deficiency of Neural Network models compared to symbolic models is the difficulty in dealing with temporal series and be able to keep in consideration items in an ordered way. Prediction problems constitute a special subclass of function approximation problem in which the following values need to be determined from past values. HAR (Human Activity Recognition) is a classical field where there is a clear need to keep into account previous items in order to predict the followings or classify a determined movement. Using the sequence of the signal in movement will help to model complex activity details and enhance the performance in recognizing patterns useful for classification and regression models. Recently many studies have explored different types of RNN for HAR.

B.0.1 Convolutional Neural Networks

Bidimensional CNNs can extract translational invariant local features in a great way, however they become ineffective when modelling temporal patterns in data. But one dimensional CNNs are actually one of the possible approaches. With RNN, you would use a cell that takes as input previous hidden state and current input value, to return output and another hidden state, so the information flows via the hidden states.

With one dimensional CNNs, you would use time-invariant filter functions in parallel to sliding windows along the input sequence, and stack such windows on top of each other, in such a way that higher-level windows would look for patterns within the lower-level patterns. Using such sliding windows can be useful for finding repeating patterns within the temporal data. However, nowadays, Matlab doesn't have any procedure to implement CNN-1D on temporal data. For this reason Python has been used to try this approach.

B.0.2 Recurrent Neural Networks

If you think to your brain which contains billions of neurons, they're all connected together somehow and there are many loops without any single direction. When we make a decision *now* we are not making that decision based only on what we see/hear *now* but based also on past events. In figure B.1 the hidden layer is just connected

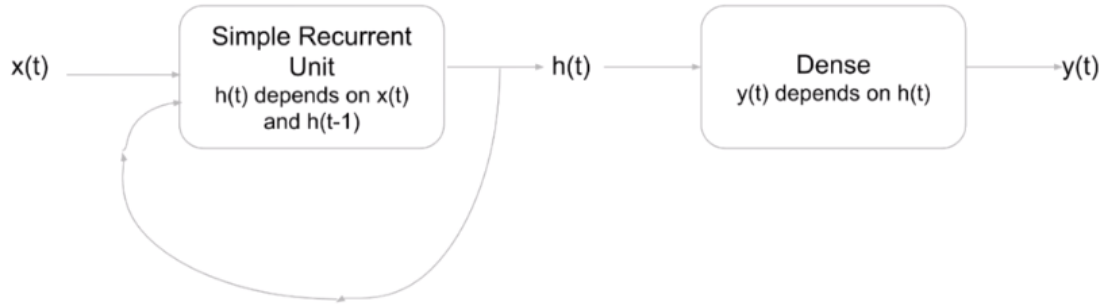


Figure B.1: Most basic Recurrent Neural Network

back to itself. Since the independent variable here is time our 3 major data variables need to be indexed by time so we call our input $x(t)$ instead of just x, we call our output $h(t)$ instead of y and we call the hidden layer value $h(t)$ instead of just h.

$$x(t) = given \tag{B.1}$$

$$y(t) = softmax(W_0^T h(t) + b_0) \tag{B.2}$$

$$h(t) = f(W_i^T x(t) + W_h^T h(t-1) + b_h) \tag{B.3}$$

$y(t)$ only depends on $h(t)$, calculating $y(t)$ is as simple as usually is, it's just a regular dense layer. We multiply our transposed weight matrix and add our bias vector and apply an activation function which can be a sigmoid, an hyperbolic tangent, a softmax or whatever else. The layer of interest in a RNN is the hidden layer, to calculate $h(t)$ we have both a term that depends on $x(t)$ and a term that depends on $h(t-1)$. Since $x(t)$ is a vector of size D and $h(t)$ is a vector of size M, we need to set the weight matrix to be consistent with these vector sizes. Assuming we're doing multi classing classification where K is the number of classes:

$$shape(x(t)) = D \qquad shape(W_i) = D \times M \tag{B.4}$$

$$shape(h(t)) = M \qquad shape(W_h) = M \times M \tag{B.5}$$

$$shape(y(t)) = K \qquad shape(W_o) = D \times K \tag{B.6}$$

One way to think about $h(t)$ is that is just a feature vector as it was in a feed forward neural network but it also contains a *memory of the past* via $h(t-1)$. You can think at W_h as a weight matrix that tells $h(t)$ which part of $h(t-1)$ are important to remember and which are not. One way to think to a RNN is just to unroll it along time, so imagine time going horizontally and the Neural Network going upward then we can unroll the NN, a schematic representation is shown in figure B.2. This brings us to a very important idea which is about not thinking to a sequence as a sequence in time but rather as a static set of inputs. In such a case we have simply a Neural Network with shared weights, so we see that W_i, W_h and W_o all show up t times. So it's a specially designed NN with the same weights repeated over and over again. In this scenario it's like all

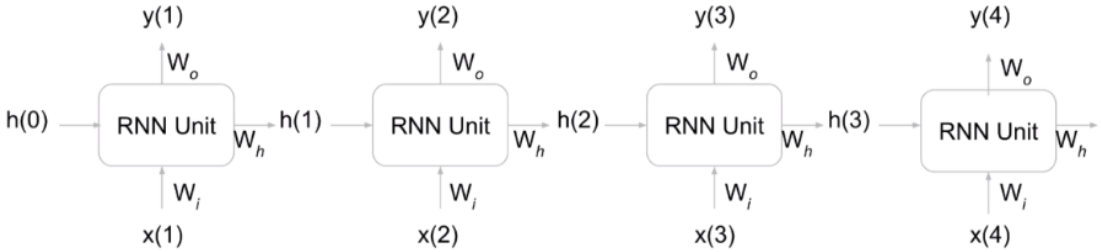


Figure B.2: View of a RNN unrolled as a Feed Forward ANN

happening at the same time with this not common architecture for a ANN. Theoretically these weights could also be different weights but it's more common while using the RNNs to use the same weights. A question that may raise is *"why is not possible to feed a Feed-forward Neural Network with my sequence? Why do we need this special architecture?"*. Suppose to have a sequence $x(t) = [x(1), x(2), x(3), x(4)]$ but instead of having T (in this case $T = 4$) different input vectors of size D, we have a big input vector of size $T \times D$. In an analog way, there will be T different h and T different y in order to have all the same variables of before except that now they're all concatenated together.

Let's calculate the total number of parameters for a Feed-forward net with one input layer, one hidden layer and one output layer in which each layer has the same number of neurons (note that this is required in our hypothesis of comparing RNN with Feed-forward ANN):

- Input parameters:
  - $T = 161$ (sequence length)
  - $D = 3$ (input dimensionality)
  - $M = 25$ (hidden layer size)
  - $K = 4$ (number of output classes)
- $W_{i\to h} = T \times D \times T \times M = 1\,944\,075$ parameters
- $W_{h\to o} = T \times M \times T \times K = 2\,592\,100$ parameters

- Total: 4 536 175 parameters without counting the bias terms.

$W_{i\to h}$ is the input to hidden weight matrix
$W_{h\to o}$ is the hidden to output weight matrix

This is a huge number of parameters and is exhaustive in terms of computation time. However it's been mathematically demonstrated that the Feed-forward architecture is able to approximate every kind of function, so it should work.

Let's have a look of the number of parameters involved in a Recurrent architecture.

- Input parameters:
  - $T = 161$ (sequence length)
  - $D = 3$ (input dimensionality)
  - $M = 25$ (hidden layer size)
  - $K = 4$ (number of output classes)
- $W_{i\to h} = D \times M = 75$ parameters
- $W_{h\to h} = M \times M = 625$ parameters
- $W_{h\to o} = M \times K = 100$ parameters
- Total: 800 parameters without counting the bias terms.

In RNN you have a structure which allows you to have a fraction of the weights and allows you to use those weights in a more intelligent way.

The numbers used are the numbers in my thesis where each windows was long Other than the fact of huge number of parameters, Feed-forward Neural Netwroks suffer of the fact that the input must be equal in size. However normally sequences don't satisfy this requirement. This problem is also present for RNN however there are ways to tackle and partially solve the issue. Recurrent Neural Network (RNN) contains connections from output nodes to previous layers and they allow interconnections among nodes of the same layer. RNNs are, nowadays, the gold standard for temporal series analysis and they find huge utilization in Natural Language Processing (NLP) and in general, in every field in which prediction is the goal.

Recently many studies have explored different kind of RNN models for HAR. Inoue et al. (2016) looked at the best combination of architecture and optimal parameter values to get the best performance possible. They noticed that by increasing the number of layers, the computational time and the memory usage increases and they concluded with an optimal architecture of 3 layers.

Many different models of RNN have been proposed along time, in this work mainly 2 different types of RNN have been used:

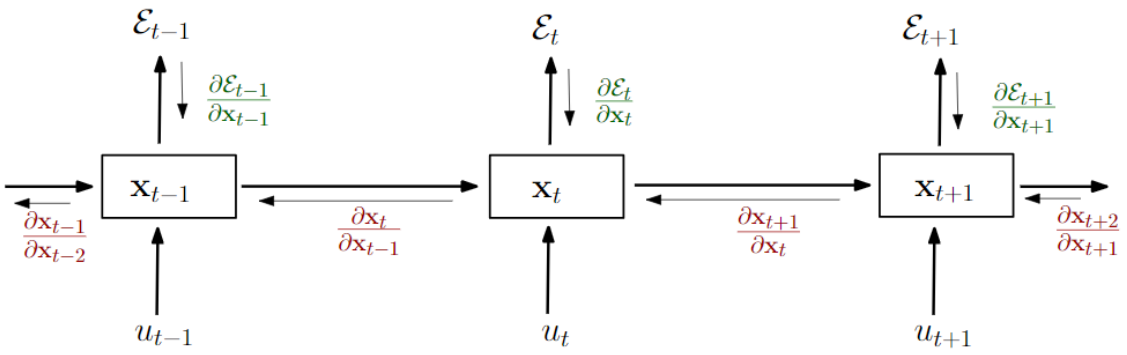


Figure B.3: RNN unrolled in time. Back-propagation highlighted [29]

- **BiLSTM** (Bidirectional Long Short Term Memory)
- **BiGRU** (Bidirectional Gate Recurrent Unit)

The bidirectional approach means that each sequence will pass twice through the different cells, once reversed respect to the other. This reverse-time approach is possible only in offline predictions but it's been shown to considerably increase the performance of both LSTM and GRU.

The long term dependencies problem

The *exploding gradient* is a problem due to the fact that the norm of the gradient will increase until *exploding* during training a Recurrent Neural Network. It's been introduced at first in Bengio *et al.* (1994)[22]. In the gradient descent algorithm we're trying to find the global minimum of the cost function. A generic RNN, with input $\mathbf{u}_t$ and state $\mathbf{x}_t$ where t is the time step:

$$\mathbf{x}_t = F(\mathbf{x}_{t-1}, \mathbf{u}_t, \theta) \tag{B.7}$$

By defining

- $\mathbf{u}_t$ is the input at time t
- $\mathbf{x}_t$ is the hidden state of the network at time t
- $\mathbf{W}_{rec}$: the parameters of the model given as recurrent weight matrix
- $\mathbf{b}$: is the bias
- $\mathbf{W}_{in}$ the input weight matrix, collected in θ for the general case.
- $\mathbf{x}_0$ is given by the user, learned or imposed at 0
- σ is an element-wise function.
- $\mathscr{E} = \sum_{1 \leq t \leq T} \mathscr{E}_t$ represents a measure of the performance of the network

- $\mathscr{E}_t = \mathscr{L}(\mathbf{x}_t)$ is the error obtained from the output at time t

By writing the equations of gradients in a sum of products form, Pascanu *et al.* [30] derived the following formulas:

$$\frac{\partial \mathscr{E}}{\partial \theta} = \sum_{1 \leq t \leq T} \frac{\partial \mathscr{E}_t}{\partial \theta} \tag{B.8}$$

$$\frac{\partial \mathscr{E}_t}{\partial \theta} = \sum_{1 \leq k \leq t} \left(\frac{\partial \mathscr{E}_t}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} \frac{\partial^+ \mathbf{x}_k}{\partial \theta} \right) \tag{B.9}$$

$$\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_{i-1}} = \prod_{t \geq i > k} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1})) \tag{B.10}$$

$\frac{\partial^+ \mathbf{x}_k}{\partial \theta}$ is the partial second derivative of the state $\mathbf{x}_k$ with respect to θ.
"*To understand this phenomenon we need to look at the form of each temporal component, and in particular at the matrix factors* $\frac{\partial \boldsymbol{x}_t}{\partial \boldsymbol{x}_k}$ *that take the form of a product of t−k Jacobian matrices. In the same way a product of t−k real numbers can shrink to zero or explode to infinity, so does this product of matrices*" [30]. This represents an explosion or a vanishing of the long term components making impossible to the model to learn patterns in events temporally far among each other.

B.0.3 Long Short Term Memory LSTM

LSTM are special RNN designed to avoid the problem of Long-Term dependencies. It's actually specially designed to find patterns in long periods of time. They were firstly introduced by Hochrereiter & Schmidhuber in "Long Short-Term Memory" (1997) [32]. Generally we can imagine the RNN to combine in some way the input in time t with the output of the previous cell state. In figure B.4 the RNN uses a simple $tanh$ activation function for each cell, however in this way, there's no protection for $\mathbf{W}_{rec}$ when is back-propagated through the chain. In figure B.5 is represented the structure of the LSTM. The cell state is like a conveyor belt whereas the important information is passed from cell to cell with only few controlled changes. These changes are done by gate layer units.

The first step B.7 shows the behavior of the **forget gate layer**. The sigmoid neural network concatenate the output of the precedent cell h_t and the current input x_t and outputs a value f_t that is informative of what to forget about the previous cell state C_{t-1}.

In step 2, figure B.8 the **input gate layer** and the **input modulation gate layer** decide how to update the old cell state in a way that will keep into account the new information arrived combined with the past cell output.

Step 3, figure B.9 is where the update of the cell state really take place.

Finally, step 4 represented in figure B.10 decides the output of the current cell. This output h_t will be a combination of the current cell state, the previous output and the

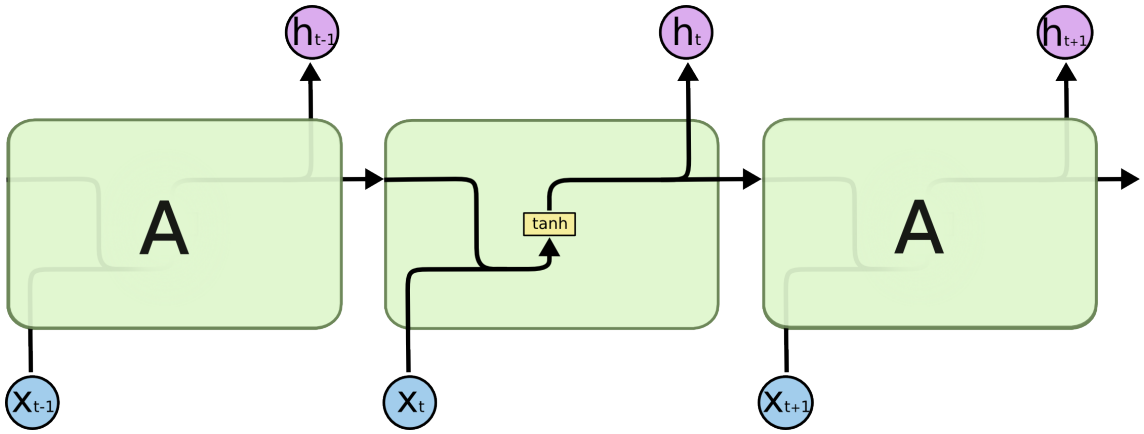


Figure B.4: General architecture of a standard RNN, the repeating module contains a single layer [34].

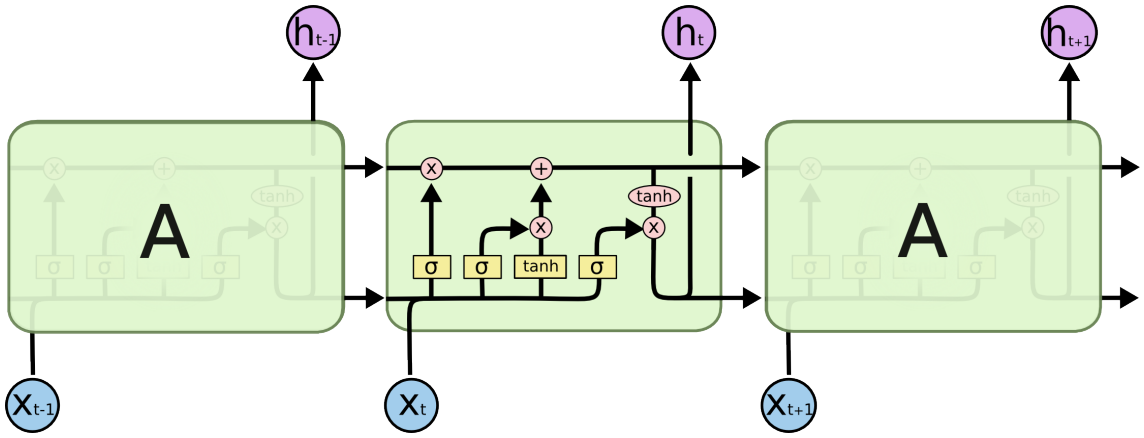


Figure B.5: LSTM schema, the repeating module contains 4 interacting layers [34].

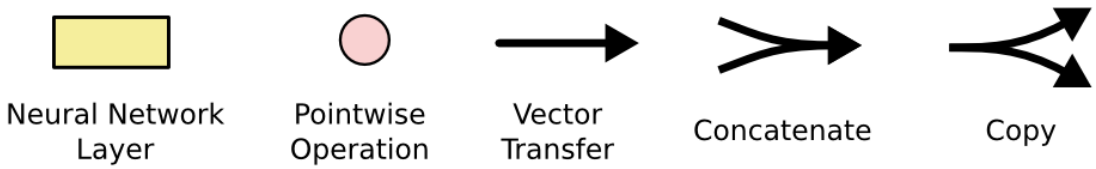


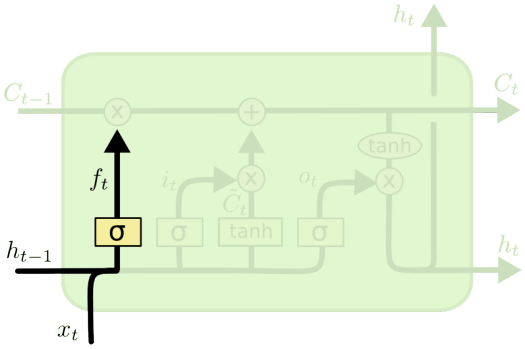
Figure B.6: Legend of the symbols used in the LSTM schemas [34].

current input. Since h_t is finally obtained by a point-wise multiplication between the output of a *sigmoid* layer and a *tanh* function, its range will stay in (-1, 1).

B.0.4 Gate Recurrent Unit GRU

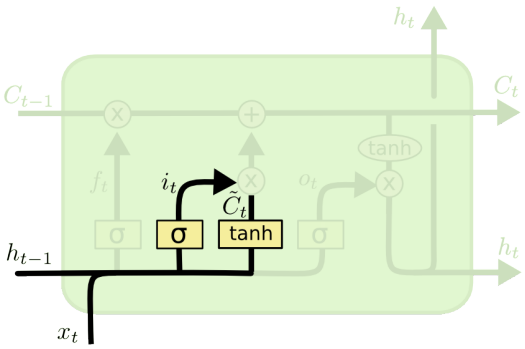
This is a pretty dramatic change of the LSTM, it's been firstly introduced by Cho et al. in 2015 [15]. It's mainly composed by 2 gates: the update and the **reset gate** and the **update gate**. This last gate the forget and input gate of the LSTM in a single one.

The main advantage of the GRU over the LSTM is a minor number of trainable parameters resulting in more robustness to overfitting and a decrease in terms of computational time. Figure B.11 shows the math behind each gate, while figure B.12 shows



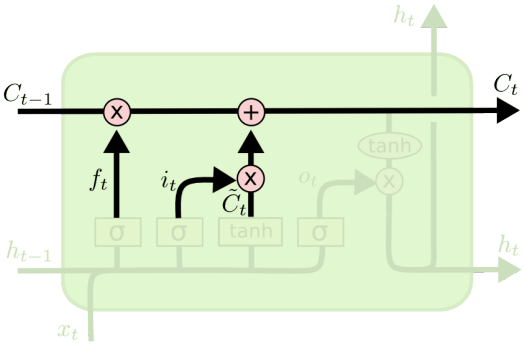
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure B.7: LSTM Step 1 [34].



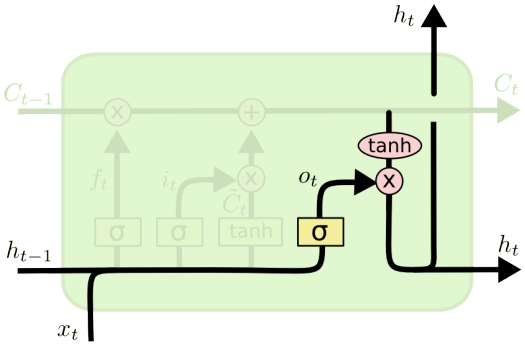
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure B.8: LSTM Step 2 [34].



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

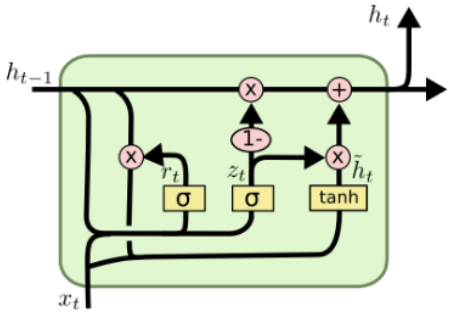
Figure B.9: LSTM Step 3 [34].



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure B.10: LSTM Step 4 [34].

the differences among the gates in LSTM and GRU.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure B.11: Gate Recurrent Unit - GRU [34].

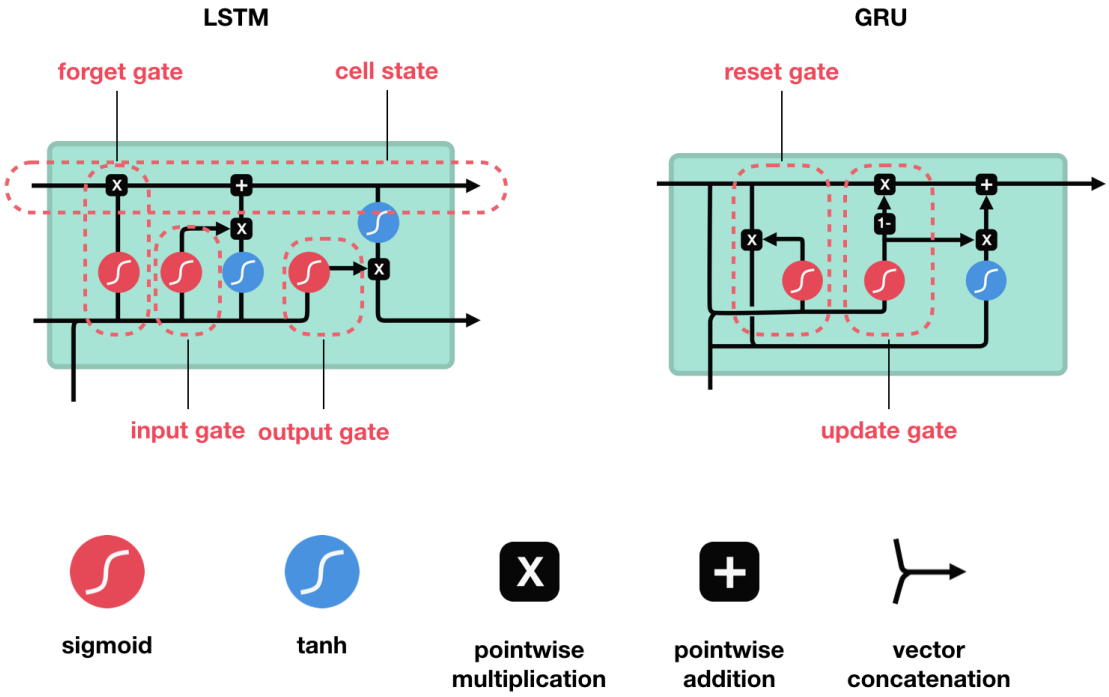


Figure B.12: LSTM and GRU, comparison among gate layers [10]

Bibliography

[1] URL: https://en.wikipedia.org/wiki/Autonomic_nervous_system.

[2] URL: https://www.dietspotlight.com/l-tyrosine/.

[3] URL: https://epilepsyu.com/deep-brain-stimulation-dbs-for-medically-refractory-epilepsy-in-the-u-s/.

[4] URL: https://parkinson.org/Understanding-Parkinsons/Symptoms/Movement-Symptoms/Dyskinesia.

[5] URL: https://en.wikipedia.org/wiki/Drug_titration#/media/File:Titrated_doses.svg.

[6] URL: https://www.kaggle.com/uciml/human-activity-recognition-with-smartphones.

[7] URL: https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f.

[8] URL: http://blog.kaggle.com/2012/11/02/t-distributed-stochastic-neighbor-embedding-wins-merck-viz-challenge/?fbclid=IwAR2tjLK2_ptJ4RnppBsjmXE3pl1PM58VNEF6p60mt0NpY81cknp6zImyKaA.

[9] URL: https://www.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html.

[10] URL: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21.

[11] Claas Ahlrichs and Michael Lawo. "Parkinson's Disease Motor Symptoms in Machine Learning: A Review". In: *Health Informatics: An International Journal (HIIJ)* 2 (Nov. 2013). DOI: 10.5121/hiij.2013.2401.

[12] Sabrina Angelini. "Parkinson's Disease: From Pathogenesis to Pharmacogenomics". In: (2017).

[13] Hadi Banaee, Mobyen Uddin Ahmed, and Amy Loutfi. "Data Mining for Wearable Sensors in Health Monitoring Systems: A Review of Recent Trends and Challenges". In: *Sensors* 13.12 (2013), pp. 17472–17500. ISSN: 1424-8220. DOI: 10.3390/s131217472. URL: https://www.mdpi.com/1424-8220/13/12/17472.

[14] Marras C.Beck et al. “Prevalence of Parkinson’s disease across North America”. In: (2018). DOI: 10.1038/s41531-018-0058-0. URL: https://doi.org/10.1038/s41531-018-0058-0.

[15] Kyunghyun Cho et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. In: (2014).

[16] *Classification and regression trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

[17] Mark Hallett and Werner Poewe. *Therapeutics of Parkinson’s disease and other movement disorders*. 2018.

[18] Feng Hu and Hang Li. “A Novel Boundary Oversampling Algorithm Based on-Neighborhood Rough Set Model: NRSBoundary-SMOTE”. In: (2013).

[19] Moini Jahangir. *Anatomy and Physiology for Health Professionals*. 2019.

[20] Yoon JH et al. “Heart rate variability to differentiate essential tremor from early-stage tremor-dominant Parkinson’s disease.” In: (2016).

[21] Noel L.W. Keijsers, Martin W.I.M. Horstink, and Stan C.A.M. Gielen. “Automatic Assessment of Levodopa-Induced Dyskinesias in Daily Life by Neural Networks”. In: (2003).

[22] “Learning Long-Term Dependencies with Gradient Descent is Difficult”. In: (1994).

[23] S. I. Lee et al. “A novel method for assessing the severity of levodopa-induced dyskinesia using wearable sensors”. In: *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. Aug. 2015, pp. 8087–8090. DOI: 10.1109/EMBC.2015.7320270.

[24] Belinda Linden. “Basic Blue Skies Research in the UK: Are we losing out?” In: *Journal of Biomedical Discovery and Collaboration* 3.1 (Feb. 2008), p. 3. ISSN: 1747-5333. DOI: 10.1186/1747-5333-3-3. URL: https://doi.org/10.1186/1747-5333-3-3.

[25] Wei-Yin Loh. “Classification and regression trees”. In: (2011). DOI: 10.1002/widm.8.

[26] Laurens van der Maaten and Geoffrey Hinton. “Visualizing Data using t-SNE”. In: (2008).

[27] Sinziana Mazilu et al. “Prediction of Freezing of Gait in Parkinson’s from Physiological Wearables, An Exploratory Study”. In: (2015).

[28] Stephen Milborrow. URL: https://commons.wikimedia.org/w/index.php?curid=68192267.

[29] “On the difficulty of training Recurrent Neural Networks”. In: (2012).

[30] Sebastian Raschka. URL: https://sebastianraschka.com/images/blog/2015/singlelayer_neural_networks_files/perceptron_gradient_descent_1.png.

[31] Oscar Reyes, Habib Fardoun, and Sebastian Ventura. "An ensemble-based method for the selection of instances in the multi-target regression problem". In: *Integrated Computer Aided Engineering* 25 (July 2018). DOI: 10.3233/ICA-180581.

[32] Schmidhuber Sepp Hochreiter. "Long Short-Term Memory". In: (1997).

[33] "The Roles of PINK1, Parkin, and Mitochondrial Fidelity in Parkinson's Disease". In: (2015).

[34] *Understanding LSTM Networks*. 2015. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.