Statistical speech modeling for detecting pathologies affecting elder people

Denis Maurel

Internship supervisor :
Carmen Garcia Mateo

Internship advisor :
Mounim El Yacoubi
Sonia Garcia

April 2015 - September 2015

Contents

Acknowledgments

I would like to thank all the people that have contributed to the success of this internship.

First, I want to thank Pr. Mounim EL YACOUBI from Télécom SudParis for his help without which I wouldn't have been able to find this internship, and also for his support all along the preparation phase and the internship in itself. His advises allowed me to determine what I wanted to do as a researcher on the go.

I want to thank my supervisor, Pr. Carmen GARCIA MATEO from the university of Vigo, for her warm welcome, her help all along this internship and also for the opportunities that she gave me during this journey.

I want to thank Dr. Paula LOPEZ and Pr. Yohan PETETIN for their help and their advises on the technical aspects of this project.

I want to thank Pr. Bernadette DORIZZI from Télécom SudParis for her support and her help during the beginning of this travel.

I also want to thank all the people of the team in Vigo for their warm welcome and their help when I needed it.

Finally I want to thank all the people that have supported and advised me during this journey. A special thank for my friend Nicolas GRANGER, with whom the biggest part of the ideas present in this report have been discussed.

Chapter 1

Introduction

Detecting pathologies such as Parkinson's disease, Alzheimer's disease, and Mild cognitive impairment is a challenging task and may require exploiting several behavioural modalities like handwriting, speech, gesture and emotions expressed through the human face. The goal of this project is to investigate speech for this purpose.

Statistical models have been extensively used for speech recognition over the last three decades and have shown impressive performance. In this project, the objective is to develop statistical models to detect pathological states within elder population through voice analysis. Since the goal here is not recognition but rather pathology detection irrespective of the voice content, the model developed should be able to detect and/or exploit relevant speech characteristics such as prosody and other pathology-related acoustic features. The project will be divided into two main parts, signal processing and statistical modelling. The former aims at extracting pertinent acoustic features for the task intended, and doing so, two points of view will be studied. The first one will consider global features which describe the total length of a sample, and the second will focus on the temporal variation of lower level descriptors, the Mel-Frequency Cepstral Coefficients (MFCC). The latter will consist of developing models characterizing pathological and non-pathological speech (associated with healthy people) in a Parkinson Disease (PD) database. Two main machine learning tools have here been considered : for feature extraction, genetic algorithms will be used, then in order to perform the "learning" phase of this project, a special kind of neural network introduced by G. Hinton in [7] will be studied.

This internship has been considered as an entrance in the research world. When I chose to work in a research team, my first objective was to discover another way of thinking. Because I already had experiences in some companies, I wanted to see what it is to work in a totally different team. After that, and because of some first experiences in research through projects at school, I wanted to see what it is spending your work time trying to find a solution to a problem that you aren't even sure if this solution exists or not. I wanted to know what it is to search.

The internship takes place in the AtlanTIC research center for Information and Communications Technologies from the university of Vigo in Spain. The areas covered by the researchers from this centre includes digital communications, wireless networks, satellite communications, remote

sensing, computer vision and image processing, audio and speech processing and many others. The preparation phase of this internship has been done in the Nano-INNOV center at Saclay.

My mission during this internship has been to apply some machine learning methods, especially neural networks methods and genetic algorithm, in order to create a tool which could be able to predict the stage of the PD through voice analysis. As describe previously, this mission has been segmented in two different parts. First I had to discover the database, analyse the features that were used and how I could improve the use of the information they contained. Then after that data would have been usable, the learning algorithm will have to be set in order to identify what could describe the most efficiently the PD.

After a brief state of the art about detection of pathologies through voice analysis, this report will be divided according to those two parts. Each of those two parts will follow the chronological line of the study: what were the problem encountered and what were the tools used to deal with them. In the final part of this report will be exposed my experience all along this internship and its impact on my career choice. Finally as a conclusion will be summarized what have been done, and what could possibly be done in order to improve the obtained results.

Chapter 2

State of the Art

The attempt to detect pathologies through voice analysis has been conducted for many years, and nowadays, many studies have performed a score over 99% of disease recognition for some diseases and under some conditions. In [12] for example, non-linear method, also known as dysphonia measures, have been used in order to detect Parkison's Disease (PD).

About features, many have been used. Voice is much more complex than an image to describe, and so, descriptors, even low level, are hard to define, because each one doesn't represent the same information about the sample, and any computation may lost a part of the information that could be useful for the specific applied method. For example, [12] uses a method based on the aerodynamic flow rate signal at the top of the vocals folds, whereas in [17] and [18], which present state of the art results, searchers use dysphonia measures like shimmer and jitter, which rigorous definitions can be found in [19].

Then, concerning used methods, here again many possibilities have been tested. Not only machine learning algorithm have been used so far, as presented in [12], however, machine learning has demonstrated very good results using algorithm such as random forest and Support Vector Machine (SVR) as in [17]. As the idea of the study is to learn features which will be able to discriminate the presence or not of the disease, an unsupervised method has been chosen here: Deep Beliefs Networks.

Chapter 3

The original database

One thing that must be known about PD is that this disease is not a binary one. Actually, it is described by a specific graduated scale, the Unified Parkinson Disease Rating Scale (UPDRS) and which represents the stage of the disease for the patient. The scale is graduated from 0 to 108, and is a survey which describe the ability of the patient to realize some common actions like thinking, speaking or eating. All in all, the scale is made of 42 items. A more specific description can be found in [14]. The initial goal was to create a tool that could be able to make a regression using voice features in order to predict the UPDRS of the person. We will see that the formulation of this goal is not without consequences.

3.1 First impressions

As in every statistical problem, the first step has been to look at the data distribution. First of all, how is the database built? It contains 1470 samples of different length (the problem of the length will be discussed latter), each one described by 6373 features, and with one UPDRS score associated. The total temporal length of the database is approximately 1.4 hours. The first thing to consider is the number of features, which is more than four times bigger than the number of samples. In a typical machine learning database, the number of features should be far behind the one of samples, if not, the algorithm could encounter problems trying to generalize the intrinsic properties of the base. This aspect of the problem will be commented with the first experiments. After that, let's consider the repartition of the data in relation to the UPDRS label. Using Matlab, we get the following diagram:

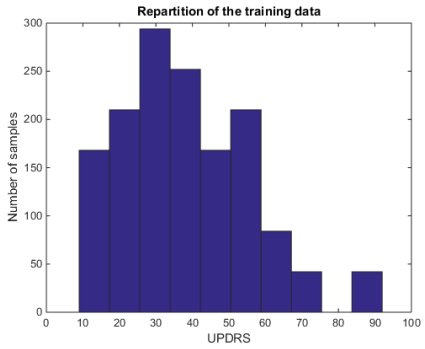

Figure 3.1: Repartition of the training data

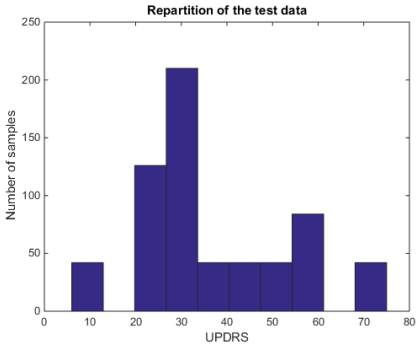

Figure 3.2: Repartition of the test data

First thing to consider: data are not uniformly distributed. Knowing that, a machine learning algorithm that will try to learn this database without modification will focus its attention on the data that represent the majority, it could be a problem if one wants the created tool to be as versatile as possible. Now, just considering that the tool should focus on the earliest stage of the disease, because that is where the health professionals could need help to make a diagnosis, we might get interested in the samples represented with a UPDRS score between 0 and 30 (segmentation get in the French guide [1]). After examination of the labels, it appears that exactly 40% of the database represent an audio recording of a patient with a UPDRS score equal or below 30, and $31,43\%$ with score only below 30. The 60% of the database remaining will be helpful in order to make the algorithm understand what are the main differences between each stage of the disease. To exemplify this, we can consider a child learning what is a transportation. If he only sees cars, he will assume that all transportation are cars. However if he can see cars and motorbike, the child will have a wider idea of what a transportation is, and will have a better understanding of the intrinsic constitution of a transportation.

3.2 A huge problem

First test using Deep Beliefs Network (see Part II for theoretical aspects and new results) weren't good. Trying to generate a regression tool between the 6373 features and the UPDRS score didn't give any good result, with a Root Mean Square Error (RMSE) that was so high that it is not even relevant to mention the numbers here. The fact that there are many more features than samples can be interpreted as a case of noise: the network has too much information to deal with, and it can't efficiently select which features are relevant or not to use. When you want to study a human face but you also consider the color of the sky, or the temperature outside, the network try to interpret this information as useful but it only makes the global error increase.

So the new problem was to proceed to a feature selection. The very first selection has been made using the correlation between each feature and the desired output. However, the results are not expected to be very good, because the relation between audio features and the UPDRS are very

unlikely to be in a linear way. Here are all the correlation coefficients :

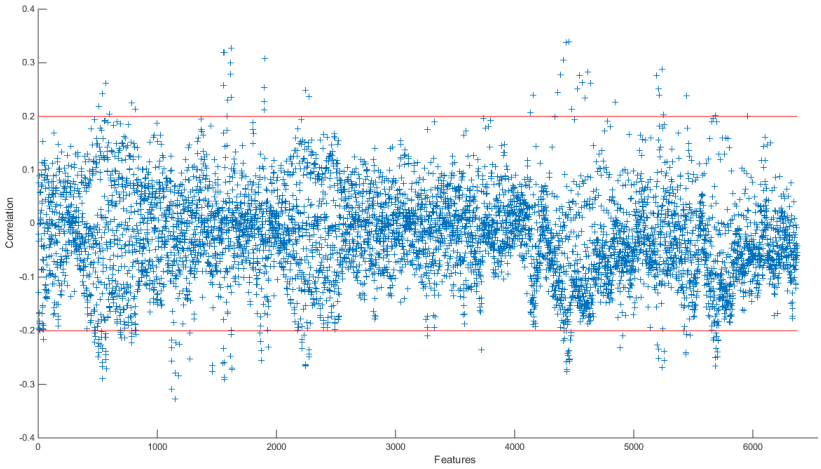

Figure 3.3: Correlation of the 6373 features available in test set

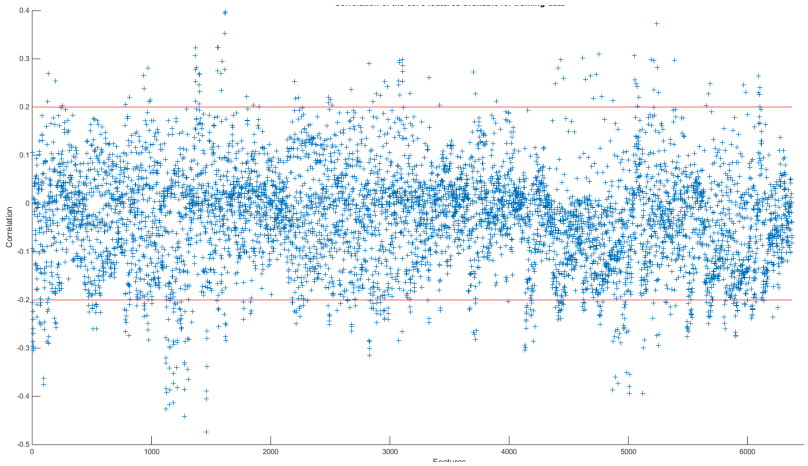

Figure 3.4: Correlation of the 6373 features available in training set

It appears, as expected, that the big majority (99,96%) has an absolute correlation coefficient that is below 0,2, with a maximum of 0,34. After all, some test have still be done only using features with the highest correlation rates, but the results weren't good. The RMSE was lower than with the whole set of features, but it still remained too high to be mentioned here.

An exhaustive search about the possible combinations could have been envisaged, but the number of possibilities was way to high to perform such a thing. Indeed if we look at the numbers,

and if we consider that a feature has only two possibles states (chosen or not chosen), the number of possibilities will be

$$2^{6373} - 1 \approx 10^{1918} \tag{3.1}$$

If each possibility could be evaluated in one nanosecond (10^{-9} second), the exhaustive computation would take

$$10^{1918} \times 10^{-9} s = 10^{1909} s \approx 10^{1904} years \approx 10^{1894} T \tag{3.2}$$

Where T is the estimated age of the universe. Well, six months may be a little short to perform such a method.

Chapter 4

Use of Genetic Algorithms

4.1 On the origin of species by means of natural selection

Another way of selection had to be chosen. The one I was familiar with and which could present interesting results was the use of genetic algorithm. This kind of algorithm can be used in order to find, not necessarily the best, but at least a solution that is really close to the optimal one. This kind of algorithm is inspired by the theory of evolution of Charles Darwin [4]. Indeed, a population will be created initially, then the most efficient individuals will survive while the least adapted to the environment will die. As a living population, the remaining individuals will reproduce, introducing some mutation to the children. This cycle is repeated until an appropriate solution (individual) is found by the process.

Here is the principle of the algorithm. You need to have a problem, and a function (also called fitness) which will determine the "score" of the individual which represents a possible solution. An individual is a possible solution, but it doesn't have to be close to any optimum (as we don't know them yet, or if they even exist). For example, for our problem, an individual can be a 6373-size binary vector that will represent if a feature i is ($individual[i] = 1$) or is not ($individual[i] = 0$) selected for the current solution. We suppose here that the fitness is known (the choice of the fitness will be discussed later). The initialization of the algorithm consists in a production of a population of N different individuals. For each individual, the algorithm compute its score using the fitness function. When it is done, a first selection is done. A fixed percentage of the population is chosen (50% has been used here), and the percentage of the population with the highest score are kept, the others are trashed. Those 50% will be the elite part of the population. After that, in order to create a whole new population, the remaining individuals will reproduce: two will be chosen at random, then their "genes" (which are here represented by the selection of some features) will be crossed in order to get a new individual. In order to simplify the computation cost, a section is randomly chosen, and the first part of the child consists in a copy from the beginning to section of the first parent, whereas the second and last part of the child will consist in a copy from the section to the end of the second parent. The following figure sum up the reproduction process:

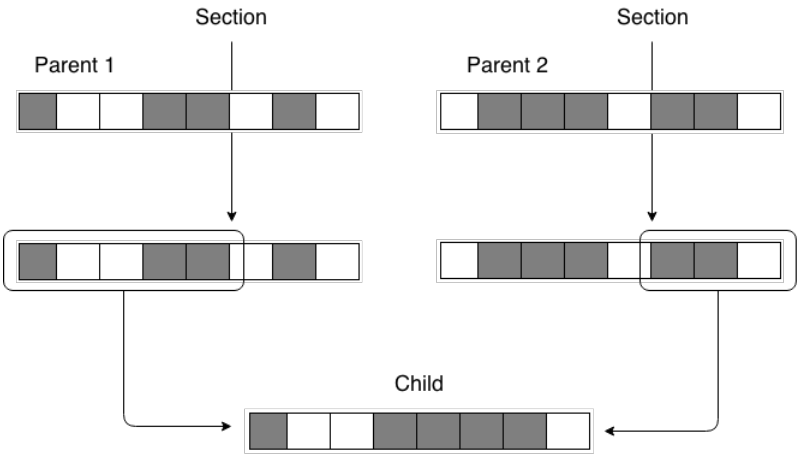

Figure 4.1: Reproduction process for genetic algorithm

Children created this way will represent 25% of the new population. Here again, the percentage is fixed at the beginning of the algorithm. The idea behind this reproduction process is that if two individuals have a relative good score, the score of their possible child is more likely to be good rather than the score of a totally random child. The same process is visible in nature, if two individuals are more adapted to their environment, their childr is more likely to survive later.

The second process that will make the population evolve to a possible optimum, is the process of mutation. As is nature, when a new individual is born, it might appears some modifications in its genotype that will modify its possible actions toward its environment. The same idea is used in genetic algorithm. A random individual is chosen in the elite part of the current population, and some mutation are randomly introduced in its genotype. The choice of a random individual rather than a child has been preferred for the simple reason of the additional computation cost that the creation of a child would have entailed. To create a mutation, a vector of the same size than the individual chosen is created, and each coordinate is randomly set to 0 or 1 using a Bernoulli distribution with a probability of success designed to put a fixed mean percentage of mutation in the whole genotype. Then we perform a "exclusive or" between the initial vector and the mutation vector to finally get the new individual. Here is summed up the mutation process:

XOR	**0**	**1**
0	0	1
1	1	0

Table 4.1: Logic table of XOR

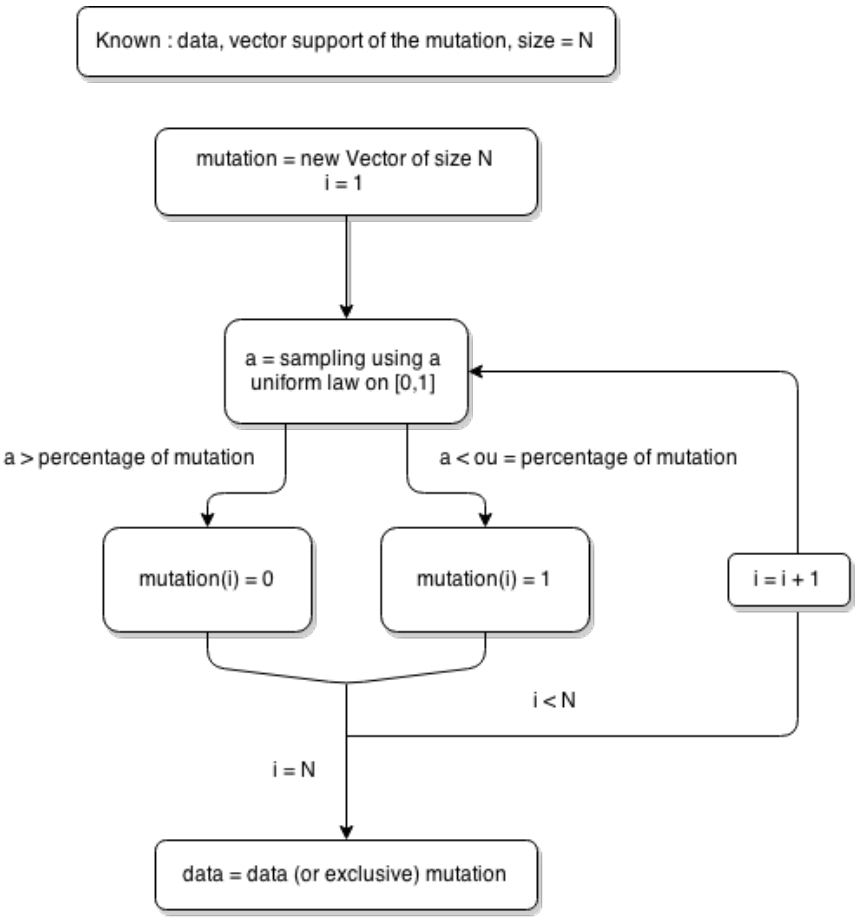

Figure 4.2: Mutation process for genetic algorithm

This process will create 25% of the new population during each generation. Now, at each generation, the genetic algorithm create a whole new population of possible solutions :

- 50% via elitism
- 25% via reproduction
- 25% via mutation

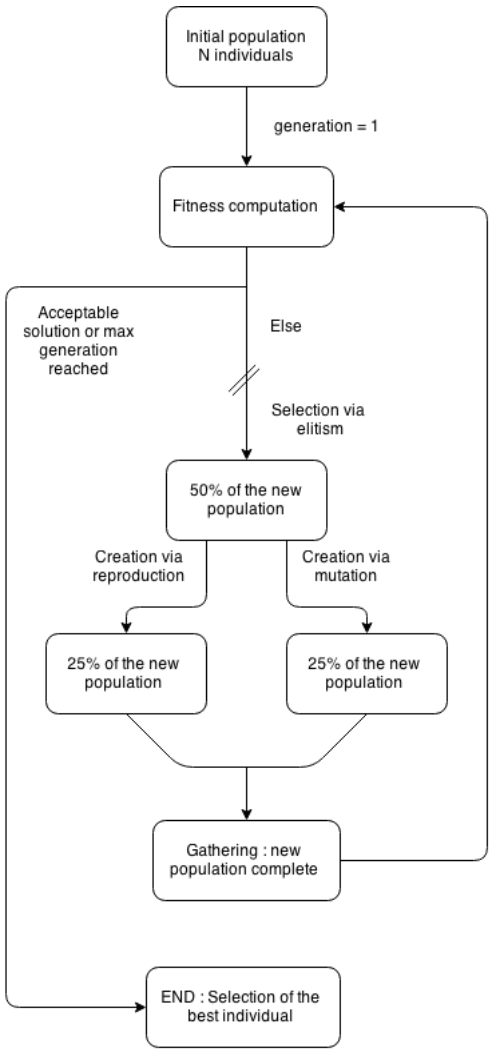

Figure 4.3: Full process of a genetic algorithm

After the new population is complete, their score (fitness) is computed, and if the best result is not good enough, another generation can begin. A summary of the whole process done during a genetic algorithm can be found on Fig 4.3.

4.2 Fitness and results

About the most important part of the algorithm, the fitness function, the first thing that had to be looked at was the kind of solution that was needed. In our case and because we wanted a features

selection, selected features should provide information on the desired output, while avoiding at most as possible any redundancy. A solution has been found in [9] and [3]. The idea of the information provided by a random variable on another is represented by the use of the mutual information, the statistical tool which can be visualized as the proportion of information that the knowledge of a variable allows to get on another. Knowing that, the bigger the mutual information, the closer will be the two studied variables.

Knowing two random variables, the mutual information can be computed this way

$$I(X,Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y) log \frac{p(x,y)}{p(x)p(y)} \tag{4.1}$$

So the first part of the fitness function will be the sum of all the mutual informations between each feature selected by an individual and the output. Assuming that O is the variable representing the output, we get

$$F_1 = \sum_{X_i \in features} I(X_i, O) \tag{4.2}$$

Now, the problem is to reduce the redundancy provided by the use of two similar variables. To achieve this, the mutual information will be used again. Indeed, the redundancy between two variables can be interpreted as the information that each variable can provide to each other. Knowing that, the term that will represent the redundancy in the set of features of an individual will be

$$F_2 = -\sum_{X_i \in features} \sum_{X'_j \; infeatures, X'_j \neq X_i} I(X_i, X'_j) \tag{4.3}$$

Finally, the fitness function consists in the sum of those two terms. To make them comparable, the mean is used instead of the intial value, so for an individual, its fitness function will be, assuming that n is the number of features selected

$$F = \frac{1}{n} F_1 + \frac{1}{n(n-1)} F_2 \tag{4.4}$$

Knowing the whole process, it is possible now to launch a first selection through possible features. Parameters have been set in such a way that approximately 200 features are chosen in the first population, and 10% of the individual can mutate in each generation. The most expensive part in term of computational resources is the defintion of I(X,Y), because the process has to determine the three distribution P(X), P(Y) and P(X,Y). This is also why the number of features selected is so low in comparison of the global number of features. After 68 hours of training on the available servers in the university, a first individual has been selected, and now in order to test its efficiency, a second training is performed using only those about 200 features in order to see if a bigger reduction of the features could improve the results. The idea is the following : in the first training,

the population has reached an area that is globally suitable for the current problem. Now that this is area is found, a more specific exploration can be performed in order to find the local (and potential global, but it can't be verified) optimum of the fitness function. The number of features has been divided by two, so the average length of vectors in the second population will be 100, and the mutation rate is again 10%. After this second training (that has been a lot shorter than the first one), better solutions have been found, and the best one has been saved. A third attempt has been conducted, but the results have not been improved, as the best solution after training was still the solution of the second attempt.

After this training phase, 98 features have been selected.

The numbers of features describing the database is now lower than the number samples. Assuming that the dimensionality reduction performed will describe more precisely the available data, another DBN training has been performed now using the set of 98 features. However, another problem appeared : due to the lack of data, a hard overfiting appeared, and the network only learned by heart all the samples present in the training base, and recognise almost none of the samples present in the test database. The first problem was to remove a possibly too complex representation in order to get a de-noised and simpler one, but now the problem is that there are not enough data. The second part of this internship has been to try to reprocess available data in order to get as much information as possible.

Chapter 5

Use of Deep Neural Networks

5.1 Introduction to Restricted Boltzmann Machines

A restricted Boltzmann Machine is a specific type of Boltzmann machine constitued by one visible and one hidden layer. Units on the visible layers serve as an input, which is why they need to be visible, while units on the hidden layer are computed using the visible layer and so serve as an output. Both layers are entirely connected one to the other. However, there are no connections between units of the same layer. This property allows to develop training algortihms further described in this paper.

It is quite common to extend unit states to use continuous values, especially on the input layer so that it matches the type of dataset values. Binary units on the other hand, can signal the presence of a specific extracted feature on the output. In speech recognition problems, data is typically real-valued, whereas for black and white images, the model would use binary valued neurons.

Nowadays, Restricted Boltzmann Machines (RBM) are widely used as a building block for Deep Beliefs Network. Their role in the final architecture will be to extract features from the data presented in their input layer. As a result, stacking RBM can produce increasingly more abstract features of the initial input.

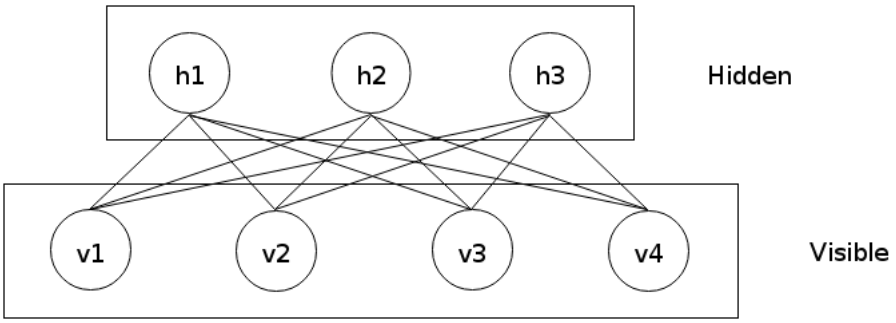

Figure 5.1: Sample Restricted Boltzmann Machine

As the network is evolving in a probabilistic paradigm, what first interests us is the joint distribution $p(\mathbf{v}, \mathbf{h}; \theta)$ which depends on the input v (the visible units), the output h (the hidden units) and the parameters of the model θ (weighs and biases for example). So the joint distribution is defined as the following formula:

$$p(\mathbf{v}, \mathbf{h}; \theta) = \frac{\exp(-E(\mathbf{v}, \mathbf{h}; \theta))}{Z} \tag{5.1}$$

where Z is a normalization factor, also known as the partition function:

$$Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta)) \tag{5.2}$$

and where E si defined as the energy of the joint configuration

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_{i,j} v_i h_j w_{ij} \tag{5.3}$$

Once the joint distribution can be accessed, it is possible to determine the marginal probability of a visible vector **v**:

$$p(\mathbf{v}; \theta) = \frac{\sum_{h \in hidden} \exp(-E(\mathbf{v}, \mathbf{h}; \theta))}{Z} \tag{5.4}$$

The main thing that appear here is that the higher the absolute energy of a specific configuration, the higher the probability of this configuration to appear.

5.2 Unsupervised Training

As with Boltzmann Machines, training consists in maximizing the probability of the network to to be in the states of the training samples. This is otherwise achieved by minimizing the negative log-likelihood of the visible states $-\log p(\mathbf{v}; \theta)$ as a optimization criterion.

Using a gradient descent approach in order to find the optimum, the update rule for the RBM weights is given by [13]:

$$\Delta w_{ij} = E_{data}(v_i h_j) - E_{model}(v_i h_j) \tag{5.5}$$

This update factor is made of two terms: the first one is the data expectation, and is easily computable. The second one however, is the expectation under the distribution defined by the model (when the network is used as a generative tool) and is intractable in practice. In order to approximate this term, the employed method is called *Contrastive Divergence* (CD) [7].

5.3 Contrastive Divergence

To compute the term $E_{model}(v_i h_j)$ from 5.5 analytically would be far too long, as one would have to test every possibility for the couple (visible, hidden) that could possibly appear in the network. Since, the network defines a probability distribution, Monte Carlo techniques can be used, and more precisely Gibbs Sampling in the case of a Markov Chain (see [16] for a thorough introduction on Gibbs Sampling). The Gibbs sampling is used in order to get samples from the underlying distribution of the model. This samples are then used to reckon the expectation $E_{model}(v_i h_j)$. Contrastive divergence then adds on further approximation in order to speed up the process.

Gibbs sampling can generate samples according to the joint distribution, and only requires the conditional probabilities of units activation knowing the other units. Due to the architecture of RBM, this means computing $p(\mathbf{v}; \mathbf{h}, \theta))$ since units from a same layer are independent. This is significantly simpler than computing the joint probability and makes the algorithm more suitable for the training process

Starting with a random input vector v^0, Gibbs sampling steps are as follows:

- Initialize $\mathbf{v}^0$
- loop:
    1. sample $h^i \sim p(\mathbf{h}|v^i)$
    2. sample $v^{i+1} \sim p(\mathbf{v}|h^i)$
    3. $i \leftarrow i + 1$

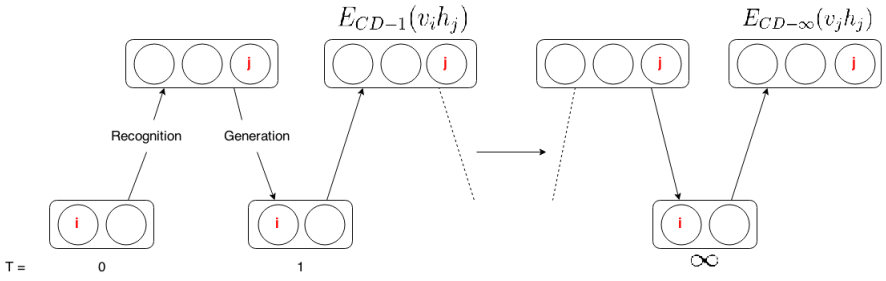

Figure 5.2: Process of Gibbs Sampling

$(\mathbf{v}_i, \mathbf{h}_i)_{i\to\infty}$ would be an accurate sample from $p(\mathbf{v}, \mathbf{h}; \theta)$. However, to have enough samples in order to get the whole distribution law would take far too long time.

Instead of that, G. Hinton [7] has proposed the method of the Contrastive Divergence (CD), method that has been widely studied and used since its emergence [2] and which is based on the Gibbs sampling. The Contrastive Divergence is a method which is used to approximate the distribution defined by the model. It's based on a reformulation of the optimization criterion using the Kullback- Leibler divergence in order to use the idea that the distribution defined by the model and the distribution defined by the network have to be as similar as possible. This method uses two tricks in order to speed up the sampling process:

- In order to approximate the distribution of the data, one can use an example extract from the data as the v^0. By doing so, the distribution from the sample generated will be closer to the distribution the network is trying to reach.
- If k is the number of iterations of the Gibbs sampling, it is possible to stop the process quite soon (k small). Actually, $k = 1$ shows excellent results in practice.

When the Gibbs sampling has been applied to every data sample, it becomes possible to calculate the second term of the gradient $E_{model}(v_i h_j)$, and so it becomes possible to compute the update rule of the parameters of the model. In practice, the expectation is not considered for all the datas at the same time. It's possible to find in the literature [5] a method based on a mini-batch rather than on a full batch. This method is used in practice because it highly decreases the computational time of the update rule.

5.4 Introduction to Deep Beliefs Network

5.4.1 Definition

Deep belief networks are based on the asumption that RBM achieve the reconstruction of missing input units by extracting the meaningful information from the available data on the hidden layer. As a result, the Deep Belief Network architecture proposes a progressive abstraction process using stacked RBM. The output of one RBM is the state of its hidden neurons and serves as the input of the next RBM.

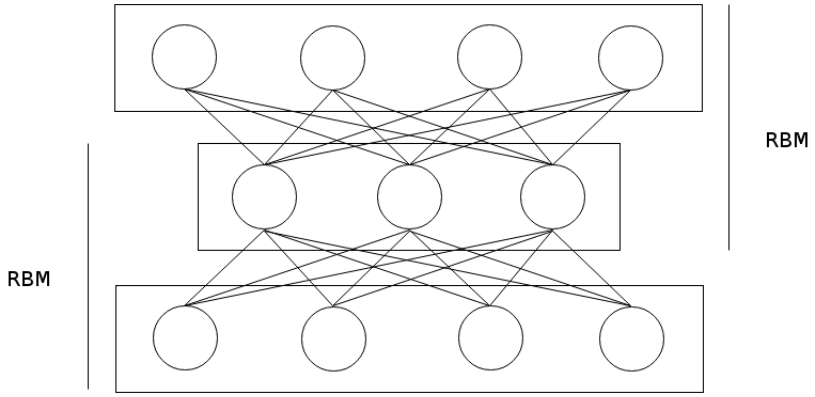

Figure 5.3: Structure of a Deep Beelief Network

5.4.2 Unsupervised Training

Training is first done layer-wise in a greedy fashion. The first RBM is adjusted using the training dataset. Then for each of the successive layers, training data is computed as the value of the hidden layer from the previous RBM for each of its training samples. The idea behind this training is that the network will learn features of increasingly higher abstraction through its layers. The more the feature is abstract, the better it can possibly describe the original dataset. With human faces for example, a DBN will learn how to represent edges in the first layers, then what the differents parts of the faces (nose, mouth...) are and finally how to represent whole faces.

5.4.3 Fine-tuning

After layer-wise pretraining comes the fine-tuning operation which works on the whole system. Several techniques are available for either supervised or unsupervised data samples.

In [8], a relatively simple unsupervised algorithm is proposed. The network is mirrored after pretraning, the initial network performs the usual recognition process while the mirrored part (re)generates the intput. Then, stochastic binary units above the input layer are replaced by their continuous activation probability function, hence transforming the whole network into a continous auto-encoder function. Back-propagation is finally performed to improve regeneration quality. Authors of the article demonstrate that their trained DBN outperforms state-of-the-art techniques on the MNIST classification test.

A much more complex unsupervised algorithm still based on the separation of recognition and generation weights is proposed in [7]. The process is divided into two phases and is known as Up-Down algorithm. During the wake phase, the first half of the network performs recognition and weights of the generation units are adjusted so as to maximise the probability of regenerating the input. The sleep phase that follows adjusts recognition weights with the same objective. The authors present this algorithm as an extension of the wake-sleep algorithm [6] using contrastive divergence instead of Gibbs sampling in gradient computations.

Both of the above-mentioned algorithms are unsupervised and therefore advantageous for databases with few or no expert data.

Semi-supervised learning can be achieved on dataset with partial labelling. The pseudo-labels technique sets missing visible units to their most probable value with respect to the distribution modeled by the network. On the MNIST handwritten character recognition test, [10] shows improved results when enabling pseudo-labels and adding 60000 unlabeled samples to the initial training which contained 600 or 1000 samples. Although the comparison of both experiments can seem a little bit unfair due to the large size discrepancy, many real-world datasets such as voice recordings and text images present similar ratios, and semi-supervised learning can have the leverage to use their potential.

5.4.4 Generation

Generation of examples with an initial database can be quite interesting in some case. For example, when an experiment requires more data than it's possibly for the scientists to get, an DBN can be trained to then generate samples that will be highly correlated to the distribution of samples. DBN can also be used in the case of partial destruction in the data. That's a quite common use of deep networks, as it can be used with Stacked Denoising Autoencoders as well [20]. [7] demonstrate a similar procedure for a DBN which is trained using the MNIST database, and then used as a generative model with a specific label clamped on. The generative aspect of a DBN is especially interesting because it introduces some variability in the output of the model. Indeed, as the model is based on probability distributions and because every bit of data is generated in a non deterministic way, one can get many variations of a same result. An example of use of this aspect is the reproduction of the human handwriting, which needs to introduce some variability in order to be quite similar to the initial handwriting.

5.4.5 Recognition

A DBN can be used either as a generative model, or as a discriminative model. When used as a discriminative model, and as it has been mentionned before, after pre-training the DBN in order to make it extract the most pertinent features of the initial dataset, a second training phase using labeled data is performed. This kind of network appears to be very efficient in classification tasks, as it can be seen in [7]. Another strong point of this kind of network is that it can be paired with any other discriminative model (a Support Vector Machin or a Multi Layer Perceptron for example) in order to achieve better performances. The flexibility of a DBN is produced by the greedy layer-by-layer training, which allow the network to learn features which can be humanly understandable.

5.4.6 Performances

Deep Belief Networks with pre-training and fine-tuning often show state-of-the-art performances, even if specialized architectures and training algorithms keep the lead on computation efficiency or with slighly higher performances [15]. Still, the cost of training only increases llineary with the size of the dataset and the architecture is very versatile. Indeed, it has been sucessfully applied to images (figures and face recognition) as well as texts [8].

On the MNIST test, DBN compete with state of the art algorithms with an initial test error rate of 1.25% achieved in 2006 [7], and further improved to 0.82% [11] using much complex architectures and training.

5.5 How to extend the database

First there were too many features, now there are too few samples. The key point of this part is that in the initial database, samples were described by features which were averaged over the time, so the part of the information which was about variations in time could possibly have been lost. In order to change that, it has been decided that all the samples will be reprocessed using different features. After all the first part about dealing with the initial database, the choice of totally changing the description model was quite a hard choice, but it was also quite mandatory, because everything that has been tried in order to stop the overfiting has failed. So in our new database, each sample will be represented by a T x M matrix, where T is the number of frames in a sample (T is not the same for every sample) and M is the number of MFCCs that are extracted from the sample in our case, M = 16. MFCCs (for Mel-Frequency Cepstral Coefficients) are coefficients often used in voice analysis, and can be seen as a low level descriptor of the sample. In our case, the fact that they are low level is welcome because the network can perfectly work with this kind of features. For example, in most papers that can be found on the Internet, image analysis is almost always done using pixels of the image, the most low level descriptors that can be used for images.

To compute the data this way has significantly changed the repartition of data, this because all samples have different lengths. Studying now frames and not samples, we get the following repartition:

Figure 5.4: Repartition of the training data after framing

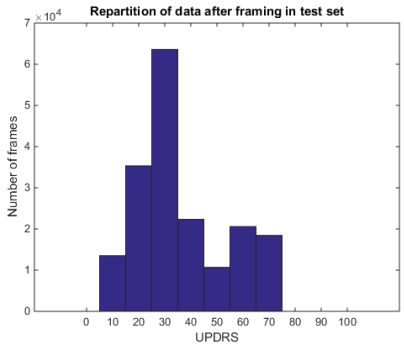

Figure 5.5: Repartition of the test data after framing

The reprocessing of the database seems to increase the quantity of information after a UPDRS of 30, especially for the training dataset where frames with a UPDRS around 40 are now more

numerous that those with a UPDRS around 30. The quantity of information may possibly have been increased, however, the lack of data for UPDRS below 30 is still present.

5.6 Training of a DBN

Now that the samples of the initial database are represented as "cepstral images", another DBN is trained. In order to have a similar size for every input, samples are concatenated in order to get small temporal images of the sample. The hypothesis that is made here is that variations that are very important for the classification of the PD can be found in small temporal context (tremor in the voice for example), and so, knowing the totality of the sample is not mandatory. In order to avoid to cut the sample on possible important variations, the small input images are created using an overlap method, which consist in the superposition of two images that are next each other in time. For example, with a number of frames concatenated of 16 and an overlap of 8, the following images will be created : - image from 1 to 8 - image from 5 to 12 - image from 9 to 16 ...

The overlap process can be summarized this way:

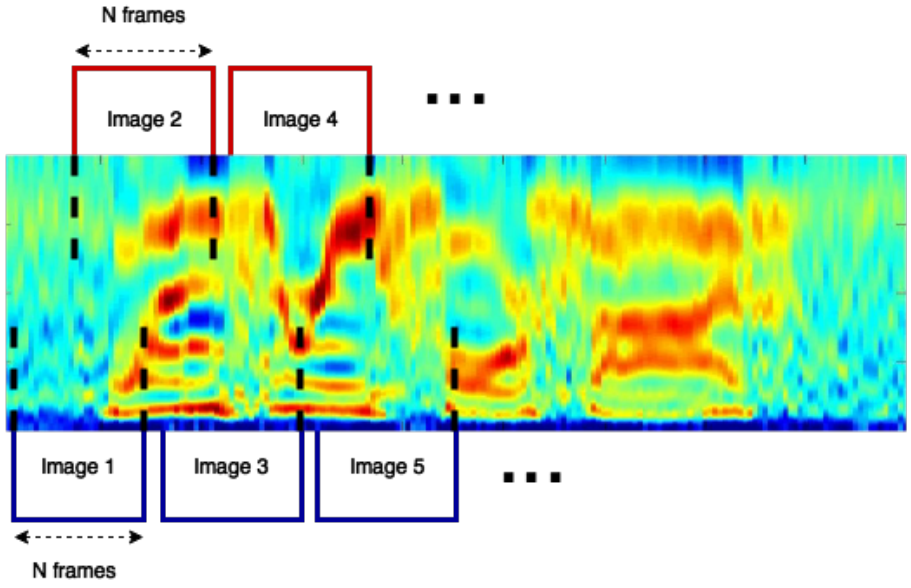

Figure 5.6: Segmentation of the cepstral image and overlap

In 5.6 is described the overlap process used during this internship. Indeed, 30 frames have been concatenated for each frame and the overlap was of 15.

After this segmentation process, a DBN has been trained using the obtained images. It can be useful to remember that the DBN training consist in a pre-training of a future neural network. After this pre-training, the network is considered as a standard Multi-Layer Perceptron (MLP), the

original neural network for which a Deep Beliefs Network can be used as an initializer. During the training of the MLP, the resultant error can be viewed in 5.7 and 5.8.

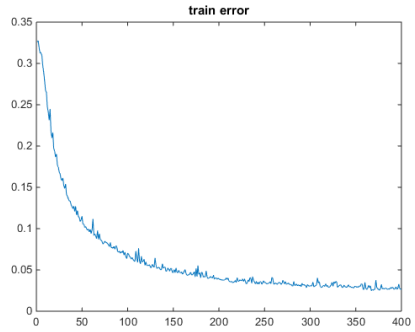

Figure 5.7: Train error

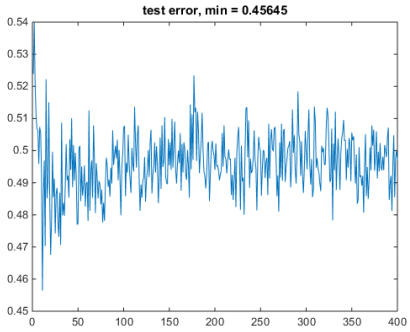

Figure 5.8: Test error

What can be seen on 5.7 and 5.8 a typical case of overfiting: the train error is far lower that the test error, and it even continue to decrease even if the test error is quite stable (when averaged on some iterations) for about 200 iterations. However, one can notice that the lowest test error are at the very beginning of the training, and both errors start sensitively under 1, it means that the pre-training had an effect, the network has learned some representation that are useful for the classification, if not, both starting errors would have been far closer to one.

Now, if one takes a look at data repartition following the initial segmentation of the UPDRS (0-12 / 13 – 30 / 31 - +), one can see that the proportion of each class is not equal for training data (training data is the only one for which the repartition actually matters in this case because that is the base that the network will use to try to understand the problem):

Segment	**0 - 12**	**13 - 30**	**31 - +**
Proportion	2.46%	36.54%	61%

Table 5.1: Proportion of each class in the training database

Actually, the first class (which is the more interesting in this problem with the second one) is almost absent of the database, so it it very unlikely that the network could learn a good representation of it. If one looks at the result table of the class predicted by the network:

Segment	**0 - 12**	**13 - 30**	**31 - +**
Proportion	$1.5\% \pm 1\%$	$31\% \pm 2\%$	$66\% \pm 3\%$

Table 5.2: Proportion of each class at the output of the network

It seems pretty close to the first distribution, but if now is considered not only the proportion, but also the proportion of each class in the misclassification set (each sample for which the network gave a label different from the real one), one gets:

Segment	**0 - 12**	**13 - 30**	**31 - +**
Missclassification proportion	$13\% \pm 1\%$	$67\% \pm 1\%$	$19\% \pm 1\%$

Table 5.3: Proportion of each class in the misclassification set

It appears that the majority of errors concern the second class, which is actually the most important class in the current problem because it is the class of people for whom the PD really starts to develop symptoms, and so it is the class for which health professionals could need help establishing their diagnosis. The network has to be improved.

An idea that appeared soon was to use a majority vote in order to get the most likely class among the analysed concatenated frames. The idea is obviously to use the whole information present in a sample in order to classify it. After some training, the network perform following performances:

Error	**Standard error**	**Post-vote error**
Percentage	$49.5\% \pm 1\%$	$42\% \pm 3\%$

Table 5.4: Error of the network

The error still remain far too high in order to get used in a medical context. The next problem was how to deal with the bad repartition of data. In order to avoid the predominance of a certain class, the training set is changed by picking for each class the number of concatenated frames present in the least important class, here the first one. With this new database, further trainings are performed:

Error	**Standard error**	**Post-vote error**
Percentage	$46\% \pm 1\%$	$37\% \pm 1\%$

Table 5.5: Error of the network with the equalized database

It appears that even if the size of the database is far under the size of the initial database, the results are sensitively better than before. An interesting point here is that the problem doesn't seem to only come from the lack of data, but also from how the database is designed. However if one takes a look at the misclassification repartition for each class:

Segment	**0 - 12**	**13 - 30**	**31 - +**
Proportion	$3\% \pm 1\%$	$65.5\% \pm 1\%$	$31\% \pm 1\%$

Table 5.6: Proportion of each class in the misclassification set with equalized database

The proportions are quite similar to previous study with unequalized database. It's possible to think that the problem doesn't only come from the database itself here, but rather (or also) from the underlying information described by each class. However many other segmentations have been tested, and so far, none of them has allowed to get better results.

The results presented in 5.6 are so far the best results that could have been obtained with this method and this database. It seems difficult to improve this knowing the huge lack of data (especially for the first stage of the disease).

Chapter 6

The experience given by this internship

6.1 Integration in the team and time organisation

The first important aspect of this internship has been the integration in a new team. If I compare to my previous experience in a development team in a previous stage, the context was quite the same. I was suppose to work on a very specific task in an open space with all my colleagues, and everybody knew everybody. It was the same here, each member of the research team had a specific task with the same global theme of voice analysis. The integration was very fine, and everybody has been very friendly since I arrived.

Now, the first, and maybe only problem, I had during this internship has been the language barrier. Actually, in France, I attended some conferences were speakers really insisted on the importance of the English in the research world, I had this really simplistic idea that as I will work in a research team, everybody could easily speak English. However, and even if some people were fluent, I have discovered that this idea was far too simplistic and it wasn't easy for some people to restart to speak English, as it was a language they haven't spoken sometimes for years. However even with those difficulties, I had conversation with members of team almost everyday, and they have made much more efforts with English than I have done with Spanish, and for that I am really thankful to those who tried to speak to me even with the problem of the language.

A point that has been very surprising and very pleasant was the total liberty of how I could organize my agenda. All the discussions I had with my classmates about their third year intership were about an internship in the industry, and so I was really pleased the flexibility that was possible to have here. Research is about idea, so when at the end of the day I still had ideas to explore, I stayed a little more. It wasn't an obligation of whatever, it was just a logical consequence of the work I was able to do at this time. On the opposite, when sometimes ideas were missing, it was possible to manage my agenda in order to take some rest before coming back to continue to work. I was very pleased to the human side that was present in this kind of work.

6.2 A new experience in the research world

Now about the experience with the research world. My only previous experience was during a project at school. This project was a semester long and consisted in the research about a given theme by group of three students. However, we had to manage our agenda between this project and the other part of the cursus, leading to a problem on time investment all along the project. It was also hard to fix meeting with other students, as none of us had the same agenda. So it was quite hard to work in good conditions, and the results at the end of the project weren't really inspiring. At the very end of this project, I really had mixed feelings about by possible orientation as a researcher. However after a research specialisation, I tried again to see what does this world look like. And here comes the internship during which I had the opportunity what research is about. In comparison of an "industrial" internship, the work paradigm was totally different. As said before, the aim is not to work in order to achieve a fixed and quantifiable objective. As a trainee researchers, I had to question everything I was doing: was it relevant? What are the possible results? How could it improve the current state of the project? I also discovered that it's not because a solution seem to work that it gives good results, sometimes the problem is far deeper than expected and one has to completely change its mind in order achieve something totally different. An example of that, and which was the most instructive to me, was the example of the use of genetic algorithm in order to perform a feature selection. The idea, which has been inspired by a specific paper, has been proven to be able to produce good results. The combination of genetic algorithms and mutual information as a criteria of evaluation has already been used in literature. And after that the algorithm has given a sufficient solution, it appeared that the solution wasn't good enough for the problem that was currently studied. The problem wasn't the algorithm used, the code or the computation, it was a more abstract problem which actually was the representation that was currently used to describe data. Even if the genetic algorithm could have found the very best solution, this solution won't have resolved this intrinsic problem of the database.

What this internship taught me about research, is that this field actually actually bears very well it's name. It's not about going in a single direction knowing that it's the right way to follow, which is the case of the work in a company, it's about making many mistakes, in order to explore a wide space of possibilities, knowing that there are not just good are bad solutions, there are many which have to be studied to be compared and then to be used.

But sometimes things are too as bright as expected. During the second month of this internship, I spent hours trying to make my network to learn something, without any idea of what could be wrong. And then a short meeting with others researchers working in the same science field has helped a lot, making me to move back in order to look in a more abstract way what was happening with my work. The part that I have found, and that I will certainly continue to consider as the most difficult in a research work, is that it's impossible to spend, days, weeks, or even months working on the same project without the conviction that there is a way to achieve what you want to do. This kind of fate is really hard to keep when since you are young, the only thing that guide your work is the logic and the pragmatism that are so specific to mathematics and computer science.

Concerning my future orientation. This internship has definitely convinced me to continue in this way. Even if there has been a lot of hard points, the good ones were far more satisfying than those that could have been found in the business world. When my code has worked when I was

in a development team, I was happy, but when my network finally succeed in learning something, it was far better than that. My impression during this trip has been that every little result was a victory, and every fail was an opportunity to learn something, in order to later get a result.

6.3 Conferences about research

At the beginning of my internship in Vigo, I have been convicted to a seminary presented by a searcher who had worked in research for many years. The speaker was here to give us a feedback about his personal experience in research, and the speech could be considered as a presentation of the future life of most of us a researchers. It has been be informative, because the speaker took the time to detail every aspect of the research world, such as the publication, the cooperation in a team, the way we have to use tools, is it interesting to be very specialised… And as an example of determination, the speaker talked about Andrew Wiles, the mathematician who proved the Fermat theorem and who spent more than ten years on a problem for which nobody knew if it was possible to find a demonstration. And he also remind us that, for one researcher like Andrew Wiles, many hundreds spend their entire like searching without finding. It was very interesting, because even if the passion on the speaker was obvious, he also presented us the drawback of such a career, and knowing that, I had the impression to know everything I had to know to take a decision about my future professional life.

After that, I get invited a second time to a seminary in the university, but this time I was the speaker. Indeed, after a presentation given to the whole team who was working on voice analysis, Pr. Garcia Mateo told me that it could be a great opportunity to try to do this presentation in front of unknown people. So I performed a presentation about Deep Beliefs Network in front of unknown people who were interested by the subject. It was the first time that I had to present something just to make people learn something, and it was also the first time I had to do a serious presentation in English. This experience has been really enlightening because it gave me an overview of what I will have to do I give lessons in the future. I realised that it is not easy at all to stay interesting during twenty minutes, so I assume that to make a two hours long lesson may be kind of a challenge. But I really enjoyed this experience.

6.4 A trip to Szeged

From June 19th to 23th , I have been invited to the 6th meeting of the COST action in Szeged in Hungary to present the work that has been done during the beginning of my internship. It has been an opportunity to discover a new country I never went to, but it also was an excellent opportunity to talk with researchers in order to get new ideas to continue my work about Parkinson. During this meeting, 20 countries were represented. First about global impression, I have been really pleased to see how easily people could start a discussion, which could be about work or even just about small talk. The ambiance was far away from the one that could be found in the school world. As the youngest person in the assembly, and just being a trainee among researchers, I first felt quite intimidated, but this feeling finally vanished with the first conversations I have had. People were there to exchange ideas, to create links, or even to initiate cooperation, and I really enjoyed this way of thinking.

That is how some people naturally came to talk to me about my work (present and future). I first talked with a Czech researcher who also was working on the Parkinson detection, and he also was working on a possible regression on the UPDRS. His work was far more advanced than mine, and so it allowed me to ask him about the ideas that he tried to set up. This exchange was really interesting because finally, my interlocutor told me that he could give us access to his database, as long as we don't spread it and that the name of the person responsible for this database is mentioned in our future publication. With just one conversation, the lack of data, one of the biggest problem that has been encountered so far, could possibly have a beginning of solution.

Another important step during this meeting has been the moment when I had to present, in front of about forty internationals researchers, the work I have been doing for approximately three months. The presentation went well, but the hardest part was the question time. During this time, an Israeli researcher clearly told be that I was wasting my time using a deep neural network approach, and that was a mistake to use such kind of methods on this kind of database. He also recommended me to try a first approach using Support Vector Machin rather than DBN to have an approximate baseline. After some discussion and clarification, I went back to my seat, quite frustrated actually. During the coffee break, the Israeli researcher that talked to me during my presentation came to apologize, and this time gave me really interesting point to study, like the "distance learning", which may be able to improve the initial database, and I also ask him for his opinion on some points of my work. Finally, this conversation has been really successful because it gave me some new ideas to test and it was really interesting to have another point of view on my work, even if the introduction has been quite rude.

What I will remember from this meeting is that the exchange of ideas and of point of views is fundamental when one works in research field. A ten minutes conversation can make one has ideas that would have taken him weeks or months to find alone.

Chapter 7

Conclusion

To put it in a nutshell, the research conducted all along this internship have known two main phases. During the first one, the given database has been analyzed and some training have been performed. However, as the underlying structure didn't seem to be efficient enough, a feature selection has been performed using genetic algorithm and mutual information between each variable and the output.

During the second phase of the research, another point of view has been adopted, and the entire database has been reprocessed in order to extract different kind of features, this time using time as a second dimension pour each sample. Results were improved and get improved again when the amount of samples in each class has been equalized in the whole database.

This internship has been successful in that it has allowed me to clearly define how I want to enter in the work world, and all different experiences I had during this trip confirmed that I want to start a career in the research world.

Bibliography

[1] Guide d'évaluation de l'updrs. `http://www.cofemer.fr/UserFiles/File/ECH.2.5.1.UPDRSa.pdf`.

[2] Yoshua Bengio and Olivier Delalleau. Justifying and generalizing contrastive divergence. *Neural computation*, 21(6):1601–1621, 2009.

[3] Vahid Chahkandi, Mehrdad Jalali, Mahsa Mirshahi, and Ali Hosseini. Feature selection with hybrid mutual information and genetic algorithm.

[4] Charles Darwin. On the origins of species by means of natural selection. *London: Murray*, page 247, 1859.

[5] Geoffrey Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 9(1):926, 2010.

[6] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

[7] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[8] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[9] Jinjie Huang, Yunze Cai, and Xiaoming Xu. A hybrid genetic algorithm for feature selection wrapper based on mutual information. *Pattern Recognition Letters*, 28(13):1825–1844, 2007.

[10] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, 2013.

[11] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM, 2009.

[12] Max Little, Patrick McSharry, Irene Moroz, and Stephen Roberts. Nonlinear, biophysically-informed speech pathology detection. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 2, pages II–II. IEEE, 2006.

[13] Keiji Nagatani and Manabu Hagiwara. Restricted boltzmann machine associative memory. In *Neural Networks (IJCNN), 2014 International Joint Conference on*, pages 3745–3750. IEEE, 2014.

[14] Movement Disorder Society Task Force on Rating Scales for Parkinson's Disease et al. The unified parkinson's disease rating scale (updrs): status and recommendations. *Movement disorders: official journal of the Movement Disorder Society*, 18(7):738, 2003.

[15] Marc'Aurelio Ranzato and Martin Szummer. Semi-supervised learning of compact document representations with deep networks. In *Proceedings of the 25th international conference on Machine learning*, pages 792–799. ACM, 2008.

[16] Philip Resnik and Eric Hardisty. Gibbs sampling for the uninitiated. Technical report, DTIC Document, 2010.

[17] Athanasios Tsanas, Max Little, Patrick E McSharry, Jennifer Spielman, Lorraine O Ramig, et al. Novel speech signal processing algorithms for high-accuracy classification of parkinson's disease. *Biomedical Engineering, IEEE Transactions on*, 59(5):1264–1271, 2012.

[18] Athanasios Tsanas, Max A Little, Patrick E McSharry, and L Ramig. Using the cellular mobile telephone network to remotely monitor parkinsons disease symptom severity. *IEEE Transactions on Biomedical Engineering*, 2012.

[19] Athanasios Tsanas, Max A Little, Patrick E McSharry, and Lorraine O Ramig. Nonlinear speech analysis algorithms mapped to a standard metric achieve clinically useful quantification of average parkinson's disease symptom severity. *Journal of the Royal Society Interface*, 8(59):842–855, 2011.

[20] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.