POLITECNICO DI TORINO

Master Degree in Biomedical Engineering

Master Degree Thesis

Convolutional Autoencoder and Deep Infomax for epileptic EEG signals clustering

Supervisors:

Prof. Gabriella Olmo
Prof. Monica Visintin
Dr. Giulio Franzese

Student:

Alessia Toscano

October 2019

To Paoletto

Abstract

Nowadays, the diagnosis of epilepsy is based only on the experience of neurologists. He/she has to observe hours of EEG signal and try to find a form of waves periodicity.

In principle, five neurologists are needed to diagnose epilepsy, so the aim of this thesis is to cluster epileptic EEG signals, applying different techniques.

The original idea was to use unsupervised learning with a training set of healthy subjects, test the network with both healthy and epileptic signals and find the differences.

To achieve this scope, two different networks have been used: convolution autoencoder and deep infomax. Both of them are analysed below.

As regards results, they are represented with t-sne technique, in order to reduce the EEG information in a 2D space. The idea is to obtain a space in which the two clusters (healthy epochs and epileptic ones) are well separated.

In a second moment, convolutional autoencoder has been used with a training set made of both healthy and epileptic subjects. This approach has the aim to extrapolate a parameter for the identification of the pathology.

Contents

List of figures

List of tables

Introduction

1.1 Physiological bases

The human brain consists of about 60-90 billion of neurons communicating with each other through the propagation of an electromagnetic field. Through the acquisition of the electroencephalographic (EEG) signal, only electrical component is investigated. Therefore, electroencephalographic signal is an expression of the electrical activity of the cerebral cortex.

The main responsible for electromagnetic propagation are the pyramidal neurons, which consist of mainly three parts (as shown in the figure 1): soma (or cellular body) axon and dendrites.

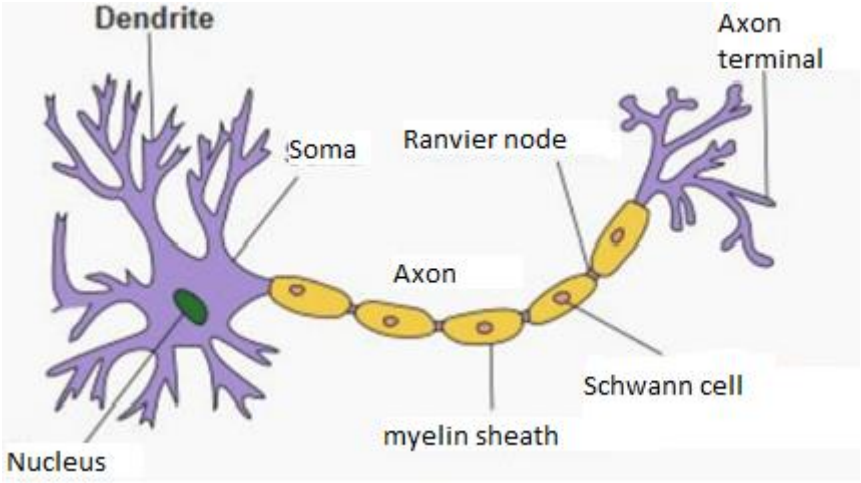


Figure 1: Structure of a neuron

The communication between these two parts, and thus the propagation of the action potential, is ensured by a highly specialized structure known as *synapses*. The signal is propagated in the form of electrical impulses along the axon, the conduction speed is increased by about 1000 times thanks to the presence of the myelin sheath, that is an axon insulation with the extracellular environment. The propagation takes place thanks to the nodes of Ranvier, interruptions of the myelin sheath, where the membrane is exposed to the extracellular environment and the signal propagating for depolarization or saltatory conduction.

So, the action potentials detected by the acquisition of the EEG signal are rapid variations in the membrane potential, that passes from the normal negative value to a positive value and ends with a restoration of the negative potential.

Potentials generated within the brain propagate perpendicularly to the cerebral cortex, so their detection is possible by electroencephalography, even if the signal is heavily attenuated.

1.2 EEG signal characteristics

Electroencephalographic biopotentials can be measured through electrodes placed on the scalp of the subject. The first example of brain bioelectric activity recordings dates back to 1875, when Richard Caton published the results of his experiments on animals. Later, in 1924, Hans Berger was able to obtain the first recording of cerebral electrical signals on a man, managing to observe the temporal patterns of the brain's electrical waves.

Thanks to electroencephalographic recordings, classifying the cerebral electrical potentials in three different categories is possible:

- Spontaneous activity;
- Evoked potentials;
- Bioelectric events caused by individual neurons.

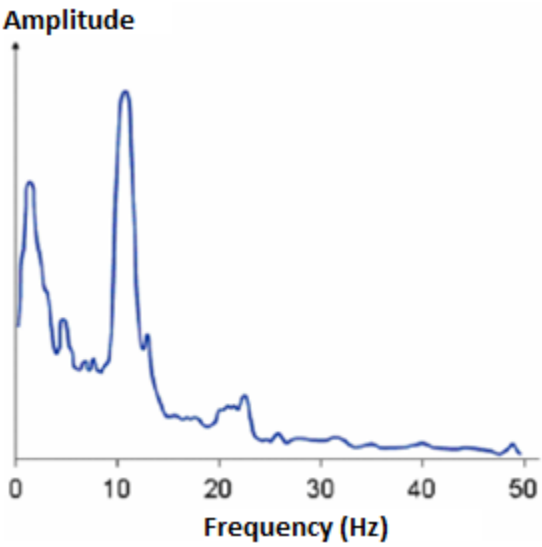


Figure 2: Frequency band of EEG

Spontaneous activity determines the electroencephalogram property, as it is continually present in the brain.

The maximum amplitude of the EEG potentials is close to few tens of μV (10-100 μV). The signal band is included between a few mHz and 50 Hz, as shown in the figure 2, but the greater amount of the content is present in 30 Hz.

Regarding evoked potentials, they are components of the EEG signal that occur in response to an external stimulus (visual, auditory, electrical, etc.). From the amplitude point of view, they are generally much smaller than spontaneous activity, so they tend to be overwhelmed by noise becoming difficult to isolate and measure.

Finally, the bioelectric electrodes caused by the individual neurons can be measured only with the microelectrode implantation in the cells of interest. It is a technique generally used when the recreation of a network model that can reflect the real properties of brain tissue is required.

Understanding the elements of the normal EEG is a prerequisite for the interpretation of abnormal records. Then the frequency bands found in the normal adult EEG are described for both the waking and the sleeping states.

- **Alpha activity**. α waves have frequencies in the range 8-13 Hz. The spontaneous activity with these frequencies is mainly measured in the posterior region of the scalp during the wake phases with closed eyes. It is present in both hemispheres, so its absence on one side is always pathological.
- **Beta activity**. β waves have frequencies in the range 13-30 Hz and they could be distinguished in β_1 and β_2 waves, respectively 13-20 Hz and 20-30 Hz. They are typical of parietal and frontal regions and reflect the drowsiness state or the use of various drugs as the benzodiazepines.
- **Theta activity**. θ waves have frequencies in the range 4-8 Hz and are often present in both first and second sleep state. During the wake state, they are often absent in adult EEG. If theta activity is consistently found in only one location, or is predominant over one hemisphere, it is likely to reflect underlying structural disease.
- **Delta activity**. δ waves have frequencies less than 4Hz and reflect the deep sleeping state. They are not present in the adult during wakefulness, so their presence in wake implies cerebral dysfunction.

1.3 EEG and epilepsy

Epilepsy is the generic name of a brain disorder characterized predominantly by recurrent and unpredictable interruptions of normal brain function, called epileptic seizures. Epilepsy is not

a singular disease entity but a variety of disorders reflecting underlying brain dysfunction that may result from many different causes.[1]

Epilepsy is defined by any of the following conditions:

1. At least two unprovoked seizures occurring >24h apart;
2. One unprovoked seizure and a probability of further seizures similar to the general recurrence risk after two unprovoked seizures;
3. Diagnosis of an epilepsy syndrome.

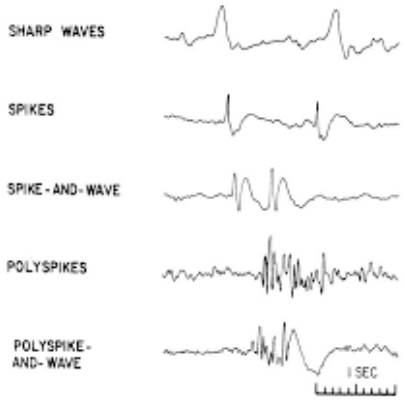


Figure 3: Epileptic waves

Inter-ictal epileptiform discharges (IEDs) in EEG help differentiating between epileptic and other nonepileptic paroxysmal attacks, as shown in figure 3. The following EEG patterns are considered epileptiform:

- Spikes
- Sharp waves
- Benign epileptiform discharges of childhood
- Spike–wave complexes
- Slow spike–wave complexes 3-Hz spike–wave complexes
- Poly-spikes
- Hypsarrhythmia
- Seizure pattern
- Status pattern

There is no objective definition of IEDs. Even experienced electroencephalographers sometimes disagree on the diagnosis of IEDs, and, therefore, EEG interpretation is hampered by poor interobserver reliability. Normal sharp transients have to be differentiated from

epileptiform discharges to avoid misinterpretation of the EEG recording. Some EEG patterns that are not epileptic may be overinterpreted, leading to a misdiagnosis of epilepsy. [2]

1.4 Acquisition of EEG

The system for the measurement of the bioelectrical cortical potential has the aims to capture a weak electrical signal on the scalp, to amplify it, to elaborate it and to record or display it.

Due to the bad conductivity of the external part of the scalp, the skin must be prepared before the application of the electrodes; this preparation consists in a first part of cleaning of the region of interest and a second one in which the electrodes are applied with either a gel or an electrolytic solution. The gel is necessary for two different purposes:

- to create a great skin-electrode interface, promoting the conduction of the signal;
- to maintain a constant adhesion between the skin and the electrode, reducing the movement artifacts.

The presence of the electrolyte is extremely important, because the electric charges are not able to move themselves from the biological tissue to the metal directly; therefore, the couple of electrodes is used as a transductor of electric signals between the ionic conduction of the electrolytic solution and the electronic conduction of the metal.

The ideal electrode should have the characteristics of an equivalent circuit of a metallic cable that ensures the flow of every charge generated by the brain and present on the interface, without limitation of frequency and direction. In practice, the equivalent circuit is more complex, and the values of the components depend on three main factors:

1. the electrolyte;
2. the electrode material;
3. the current density through the junction.

In the derivations on the cranic surface, the used electrolyte must not cause irritation and must be compatible with the chemical substrate of the skin. The active part in most electrodes is a chloride, typically of sodium (Na) or calcium (Ca). There is a large variety of materials used for the construction of the electrodes; silver, stainless steel, gold and tin are the most common.

The material for the sensors shall not be combined, because the presence of different metals can cause an electrolytic reaction, generating an offset potential. All the materials used for sensors, except for the silver chloride (AgCl), generate a metallic connection with the electrolytic solution, and for this reason they are not polarizable.

Ions cannot enter or leave physically from the material of the sensor, so they accumulate on the interface, creating a capacitive layer that hinders the current flow and the low frequencies. Thus, although a lot of metallic sensors are used for clinical EEG, none of them is sufficient for the low frequency work, except for AgCl.

This latter is a material for an ideal sensor, not polarizable, because it is the only one able to exchange ions continuously; in theory it gets a time constant that tends to infinity, so it allows the recorder of weak potentials, without any distortion of the signal.

1.4.1 Placement system standard 10/20

The exact position of the electrodes on the scalp is ruled by the international system 10/20, a standardized method developed in the end of the forties. The system 10/20 involves placing the electrodes according to ideal lines (antero-posterior, medial and lateral sagittal line; frontal, central and parietal coronal line) drawn starting from fixed points: the inion (external protuberance of the occipital bone), the nasion (the upper cluster above the nose) and the preauricular points. The distance between one electrode and another is always 10% or 20% of the total length of the line, hence the name of the system. Every position of one electrode is named with a letter and a number (or a second letter). The numbers refer to the lateralization (even numbers to the right hemisphere, the odd ones to the left one). While the letter refers to the region of the cortex below the electrode and in particular:

- Fp = frontopolar;
- F = frontal;
- C = central;
- T = temporal;
- P = parietal;
- O = occipital;
- Z = median line

Thus, the system includes nineteen sites, eight on the left side, eight on the right one and three on the center. The figure 4 shows the disposition of the electrodes.

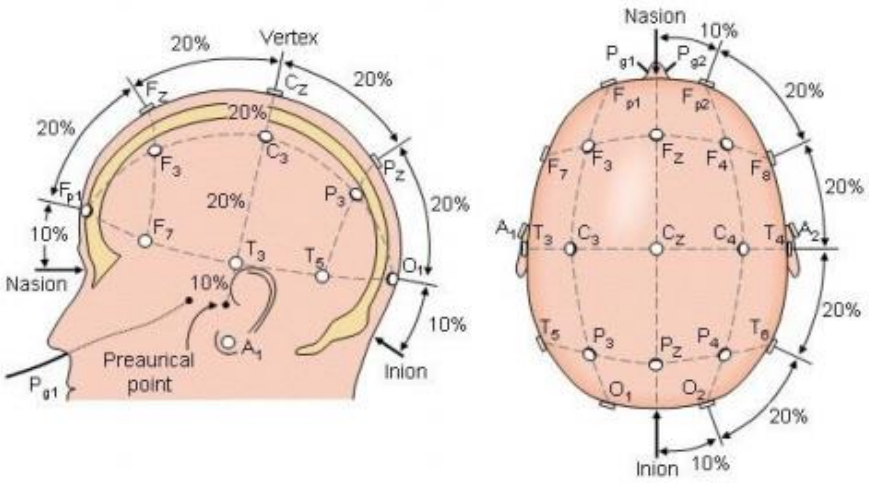


Figure 4: Elecrodes disposition

The monitored signals on the scalp are sent from the electrodes to the differential amplifiers. The latter have both an inverting and non-inverting enter, so electrodes are used couple always.

A couple of electrodes can be positioned according to monopolar derivations and bipolar derivations, depending on specific experimental requirements. On a monopolar derivation, an electrode is located on an electrically active area, while the other one (the reference) is located on an electrically neutral area (as the ear lobe).

The monopolar record represents the absolute level of electrical activity under the active site and is used to acquire signals coming from the deepest areas of the brain mainly.

Instead, during a bipolar record, both electrodes are located on active sites on the region of interest and the acquired signal corresponds to the difference between the active sites. Thus, the acquired signal shows the activity coming from the external layer of the cortex, because the potentials generated by deeper zones are considered as common mode by the differential amplifier.

As well as the two enters, an amplifier requires also a connection with the ground, to allow the current flow between the ground and the active conductor (the reference) and so to permit the correct work of the amplifier.

Therefore, for the single channel work, three sensors are necessary (an active on, a reference and the ground), while for a double channel work are added another active sensor and a second reference, for a total of five sensors.

1.5 Machine learning

Machine learning is a discipline that studies algorithms able to learn from given data how to classify or to make predictions of previously unseen data. Classification refers to supervised learning, while prediction refers to unsupervised learning, because classification is not possible and the model has to automatically extract features from data and use them to insert data in a cluster or in another one.

There are a lot of machine learning methods, but everyone requires a training phase. Using supervised learning, the training phase consists on the building of a model according to given pairs of input and output data. Instead, in unsupervised learning, the construction of the model in the training phase is done purely from input data. Once the model is obtained, it can be used to predict output from previously unseen data.

Even if talking about machine learning and artificial intelligence is normal today, to obtain those results the way has been complex, because it was split in experimentations and scepticism. The first experimentations for the realization of intelligent machines date back to 1950, when mathematics and statisticians proposed to use statistical methods to realize machines able to take decisions considering the probabilities of occurrence of an event. The first name related to machine learning is Alan Turing who realized the need of specific algorithms to make machines able to learn.

In the same period, even studies about artificial intelligence, open system and neural network saw the alternance of growing and abandonment moments, due to both the difficulties met in the realization of the models and the lack of financial allowances.

Luckily during the first years of eighties, a series of interesting results led to the rebirth of this sector in research, and during the end of nineties new techniques related to statistical and probabilistic elements were available. It was an important step that made the machine learning to be high demand in the research nowadays.

Among the most emerging algorithms for both classification and prediction, there are the autoencoder and the convolutional autoencoder.

1.6 Autoencoder and convolutional autoencoder

Despite the recent progress in machine learning and deep learning, unsupervised learning still remains a largely unsolved problem. It is widely recognized that unsupervised learning algorithms that can learn useful representations are needed for solving problems with limited label information.

An autoencoder (AE) is a network with one input layer, one hidden layer and one output layer. The number of neurons in the output layer is equal to the number of neurons in the input layer. During training, the input x is first mapped to the hidden layer to produce hidden output y. Then, y is mapped to the output layer to produce z values. The aim of an AE is to reproduce the input layer, so the produced z values are a representation of the x input values [3]. Less is the error between output and input, more accurate is the network. This type of net is used to compress large amount of data, because, saving the hidden layer (lighter than the input layer), the reproduction of the input is possible.

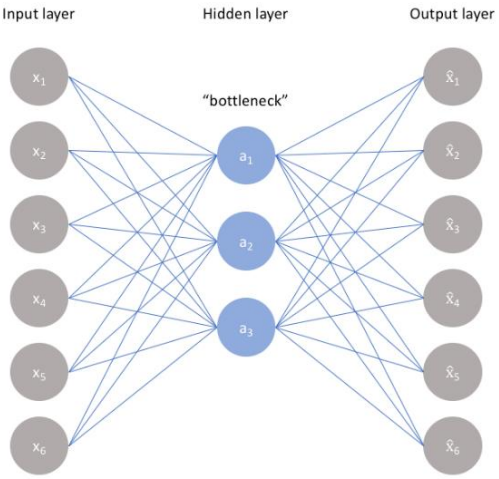


Figure 5: Autoencoder structure

Figure 5 shows the two main phases of an AE:

1. **Encoder**. There is the transition from the input layer to the hidden layer.
2. **Decoder**. There is the transition from the hidden layer to the output.

Both the encoding and the decoding processes use an activation function to analyse data.

From a mathematical point of view, the encoding and decoding phases can be seen as transitions Φ and ψ, such that:

$$\phi : \mathcal{X} \to \mathcal{F}$$
$$\psi : \mathcal{F} \to \mathcal{X}$$
$$\phi, \psi = \arg\min_{\phi,\psi} \|X - (\psi \circ \phi)X\|^2$$

The encoder stage of an autoencoder takes the input $x \in R^d = X$ and maps it to $z \in R^p = F$:

$$z = \sigma(Wx + b)$$

Where:

- σ is the activation function (as a sigmoid function);
- W is a weight matrix;
- b is a bias vector.

Instead, during the decoding phase, z values are mapped to the reconstruction x' of the same shape as x:

$$x' = \sigma'(W'z + b')$$

where σ', W' and b' may differ from the ones used by the encoder.

Weight and bias are iteratively arranged by the net to minimize the reconstruction errors, measured e.g. the squared errors:

$$L(x, x') = ||x - x'||^2 = ||x - \sigma'(W'(\sigma(Wx + b)) + b')||^2$$

where x is usually averaged over some input training set.

The obtained feature space F has lower dimensionality than the input space X, so the vector Φ(x) can be considered as the compressed representation of the input signal.

The reconstruction error is iteratively minimized thanks to some algorithms, known as backpropagation algorithms, combined with optimization methods as the stochastic gradient descendent (SGD). Substantially, the net compares the reconstructed output obtained with the original input signal and distributes the error from the last layer to the first one.

SGD is used, as in a lot of problems in machine learning, when there is the necessity to optimize a function *Q(w)* that has the shape of a sum

$$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i\,(w)$$

where every addend $Q_i(w)$ is the cost function evaluated in every i-th observation.

Each iteration in SGD is made like:

$$w := w - \eta \nabla Q(w) = w - \eta \frac{1}{n}\sum_{i=1}^{n} Q_i(w)$$

In the previous equation, η is the learning rate, a parameter that controls the amplitude of the step for each iteration. Its value can be changed dynamically, typically it is decreased during the iterations.

The parameters are updated until the method converges.

A convolutional autoencoder differs from a common autoencoder due to the multiple layers between both the input layer and the bottleneck and the bottleneck and the output layer (shown in figure 6).

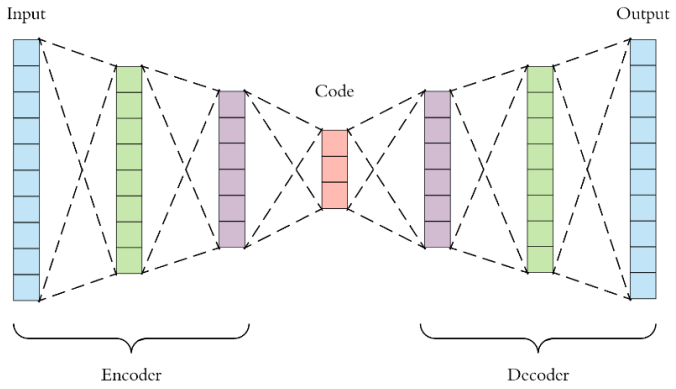


Figure 6: convolutional autoencoder structure

Similar to the simple autoencoder, the size of the input layer is also the same that has the output layer, but the network of decoder changes to convolution layers and the network of decoder change to transposed convolutional layers [4].

The convolutional operator is required because it allows filtering an input signal and extracting some parts of its content. So, the most important concept of a convolutional autoencoder is that it considers a signal as a sum of signals, concept not taken into account in a normal autoencoder.

A convolutional operator is defined in the continue case as the integral of the product of two functions after one is reversed and shifted:

$$f(t) * g(t) = \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d\tau$$

So, if the first function is the signal to encode and the second one is a known filter, the convolution operator could be used to extract features of the signal.

Moreover, this concept could be applied in a generic n-dimensional space, as shown in figure 7.

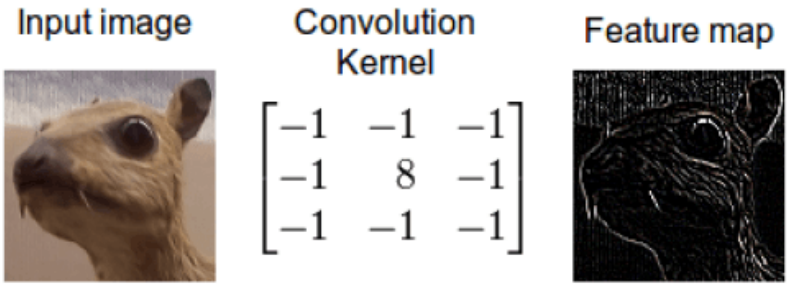


Figure 7: Features extrapolation

Obviously, the result of the convolution depends on the value of the convolutional filter. There are a lot of engineered filters, but the main rule is that it has to have the same number of channels of the signal, it is convolved with.

However, the filter definition task is used in a Convolutional Autoencoder with a different perspective: the filter mask is automatically learnt by the net.

Filtering the signal is the approach used in the encoding part, while in the decoding one the signal is reconstructed with minimal loss.

Deeply in detail, the number of neurons decreases during the encoding phase, thanks to subsampling operation. The signal is analysed in different portions and, for each region, the net learns and saves only a parameter, generally the maximum or the mean value.

A max-pooling layer pools features by evaluating the maximum activity within input feature maps (coming from the previous convolutional layer) and produces:

1. Output features map with reduced size. The output size depends on the size of the pooling kernel used.

2. Complementary switch variables (known as switches) that describe the position of the max-pooling value evaluated.

In parallel, in the the decoding phase, the net is composed of unpooling layers that aim to restore the max-pooled feature either into the correct position thanks to the switches, or into same place within the unpooled output feature map.

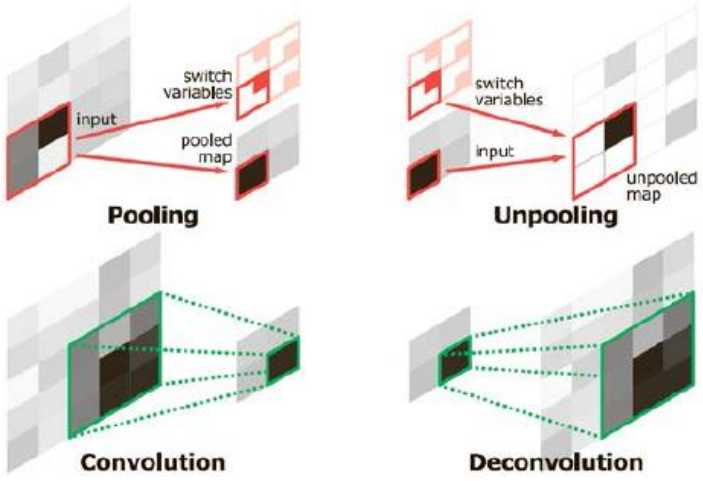


Figure 8: Pooling and unpooling

However, in the theory and in the practice of convolutional neural network (CNN), the use of pooling layers is still object of discussion.

The actual state of the art provides the construction of a model with a max-pooling layer which computes the maximum activation of the units in a small region of the previous convolutional layer. Encoding the result of convolution operation with max-pooling allows higher-layer representations to be invariant to small translations of the input and reduces computational cost [5].

Controversially, recent studies demonstrate that a max-pooling operation could be replaced by a convolutional operation with increased stride. In that way, the net could have a better accuracy.

1.7 Deep infomax network

While many of deep learning successes involve supervised learning, these latter can fail when data annotation is limited or unavailable. On the other hand, an unsupervised approach tends to be either linear or weakly non-linear or is restrictive in parametrization. A different type of algorithm is Deep Infomax (DIM), a promising tool for exploring brain structure in a flexible non-linear way.

DIM works by maximizing the mutual information between a high-level feature vector and low-level feature maps of a highly flexible convolutional *encoder* network, by training a second neural network that maximizes a lower bound on a divergence (probabilistic measure of difference) between the joint or the product of marginals of the encoder input and output.[6]

As shown in the figure 9, the algorithm is based on the following steps:

1. Encode the first-class X into MxM feature map $f_\psi(x)$.
2. Combine the map into feature vector $y= h_\psi(f_\psi(x))$.
3. Combine the same feature vector y.
4. Train a discriminator to tell apart the first class from the second one.

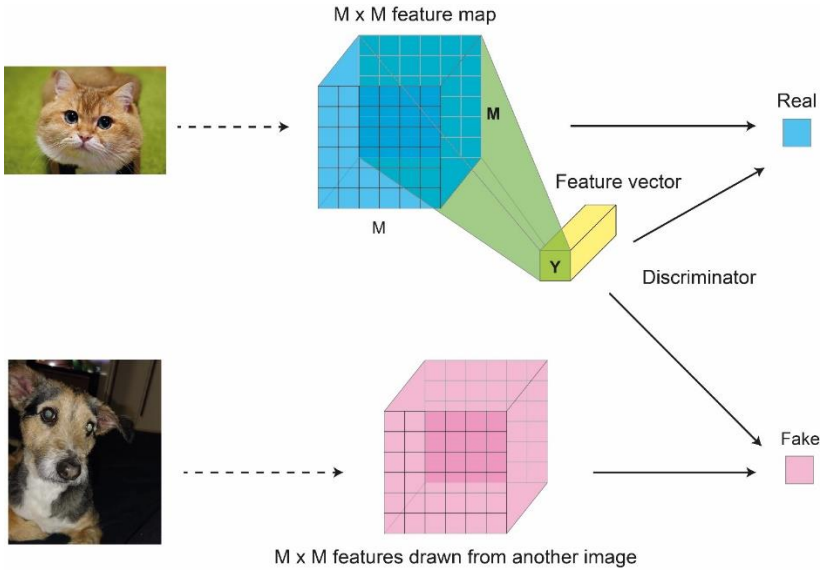


Figure 9: Deep Infomax network structure

Finally, one of the most relevant aspect of DIM is that it does not require a decoder, hence it significantly reduces memory requirements of the model for volumetric data.

From a mathematical point of view, let X := $\{x^{(i)} \in X\}$ and Z := $\{z^{(i)} \in Z\}$ be the input and output variables of a neural network encoder, E_Φ: X -> Z with parameters, where X, and Z are its domain and range.

We wish to find the parameters that maximize the following objective:

$$(\hat{\phi}, \hat{\theta})_G = \arg\max_{\phi,\theta} \hat{\mathcal{I}}_\theta(X; Z)$$

where:

- $\hat{\mathcal{I}}_\theta$ is the mutual information estimate provided by a different network with parameters θ
- Z = E_Φ (X)is the output of the autoencoder.

For the mutual information, a parametric estimator is used. It can be found by training a statistics network to maximize a lower bound based on the Fenchel-dual or the Donsker-Varadhan representation, of the KullbackLeibler divergence D_{KL}. [6]

However, D_{KL} is unbounded, which can be problematic if the above estimators are used for training deterministic neural network encoders. Using an estimator based on the Jensen-Shannon divergence (JSD) (i.e., simple binary cross-entropy), the net is more stable and works well in practice, and it has been shown that this estimator also yields a good estimator for mutual Information.

Mathematically:

$$\hat{I}^{(JSD)}_{(\phi,\theta)}\left(X; E_\phi(X)\right) \coloneqq \mathbb{E}_{\mathbb{P}x}\left[-sp\left(-T_\theta\left(X, E_\phi(X)\right)\right)\right] - \mathbb{E}_{\mathbb{P}x \otimes \mathbb{P}x}\left[sp\left(T_\theta\left(X', E_\phi(X)\right)\right)\right]$$

Where:

- T_θ is a statistics network with parameters θ;
- $sp = \log(1 + e^z)$ is the softplus function;
- X' is another input sampled from the data distribution independently from X.

An alternative estimator could be the Noise-Contrastive:

$$\hat{I}_{(\phi,\theta)}^{(NCE)}\left(X; E_\phi(X)\right) \coloneqq \mathbb{E}_{\mathbb{p}}[\, T_\theta\left(X, E_\phi(X)\right) - \log \sum_{X' \epsilon\, X_b} e^{T_\theta\left(X', E_\phi(X)\right)}]$$

Here, $X_b = \{X\} \cup X_n$ where Xn are a set of *negative samples* drawn from the data distribution, such that there is exactly one positive example in Xb (X occurs exactly once).

1.8 t-distributed stochastic neighbour embedding (t-sne)

T-distributed stochastic neighbour embedding (t-sne) in an algorithm for dimensionality reduction created by Geoffrey Hinton and Laurens van der Maaten, frequently used in research fields as a tool for machine learning. It is a non-linear method, useful for embedding a high dimensionality dataset into a 2D or 3D space. The algorithm models the points making close points of the high dimension close in the reduced space.

The first phase to reduce the dimensionality is to generate a probability distribution that combines each pair of points in the original space into a high probability value if the points are similar, and into a low value if they are not.

Hence, a probability distribution is done in the reduced space in the same way and, after that, the algorithm minimizes the divergence of the two distributions thanks to the gradient descent.

2. Methods

2.1 Matlab

To achieve the goal of this thesis, the use of a programme able to reproduce the experiments is necessary. For that reason, Matlab framework is used. Framework is a term used to indicate an array of libraries that allows to save time, thanks to some functions needed in the development of most programmes.

Every framework gives different advantages on the basis on the scope to achieve, and Matlab allows to manage matrices, analyse both data and graphics. This is the reason why it is used in this work.

An additional concept writing a code is the compilation. When a code is written for the development of a programme, for example in C, it has to be compilated. Before the compilation, there is not a file "readable" by the operative system, but only a series of files with particular extensions of a text files. This means that, loading those files, an editor presents a code and the operative system does not know what to do. Thanks to the compilation process, the textual file becomes interpretable by the system and usable by the user.

Matlab is a framework based on C language created to facilitate data analysis, with both vectors and matrixes, and to increase the speed of creation of graphs. Matlab, as a common framework, provides commands to speed up the creation of the programme that, in the end, is written in C.

In conclusion, using Matlab, a code is written with functions provided. The latter are translated in C language by the system, data are analysed, and results are shown by figures if it is required. This process in shown in figure 10 with an example, where the mean value is easly evaluated with a single command.

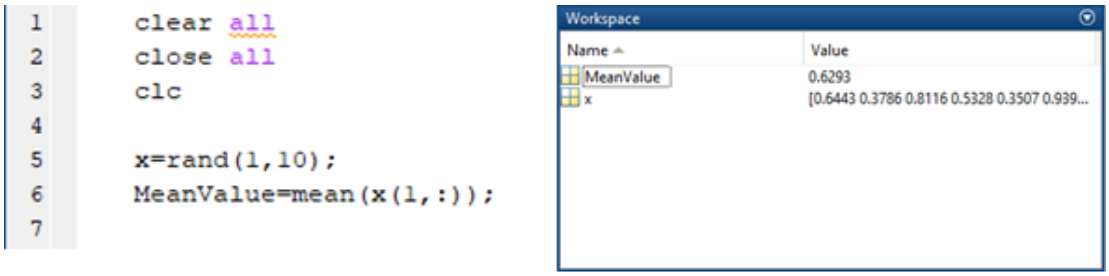


Figure 30: Example of Matlab function

2.2 Signals extraction

The used signals obtained from a database made available by Temple University Hospital of Philadelphia that has the goal to enable deep learning research in neuroscience by releasing the largest publicly available unecumbered database of EEG recordings. This ongoing project currently includes over 30000 EEGs spanning the years from 2002 to present. Data collected can be used for both research and commercialization purposes.

Downloaded files are organized in two folders: epileptic subjects and not epileptic ones. In both folders, recordings are split according to the type of acquisition:

- Linked Ear (LE) are based on the assumption that sites like the ears and mastoid bone lack electrical activity, often implemented using only one ear.
- Averaged Reference (AR) uses the average of a finite number of electrodes as a reference.

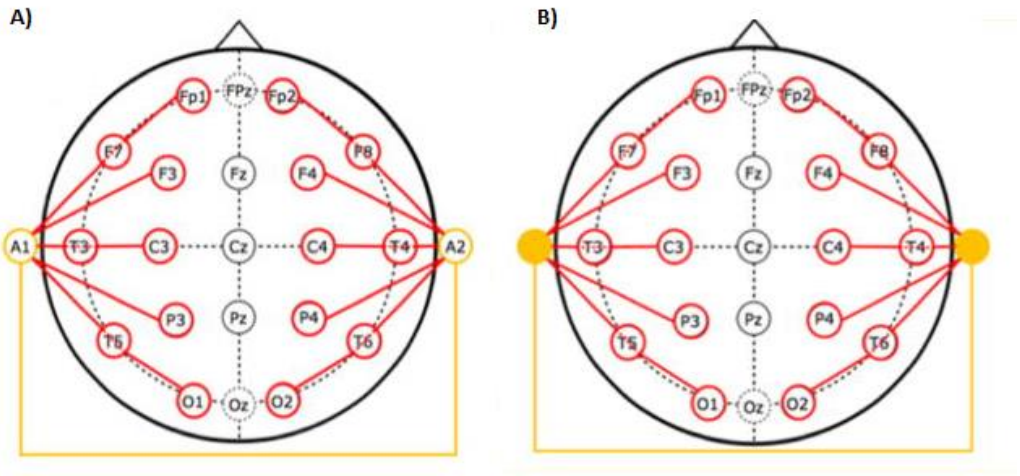


Figure 41: A) Linked Ear reference. B) Averaged reference

For each type of acquisition, the folder contains in turn one folder for each subject. Obviously, in each subject's folder the acquired signals are paired with medical report.

For both convolutional neural network and deep infomax, the extraction of signals of interest is made on the base on some parameters, such as:

- sample frequency 256 Hz;
- epoch length both 1s and 2s (according to the network);
- overlap among epochs 50%;

Once data are uploaded on Matlab, the extrapolation of one or two channels is important. In this study the channel T3 (left temporal) is always isolated, while the T4 one (right temporal) is considered in subsequent tests as well.

The choice of temporal channels is made because most epileptic waves are shown in this area of the cortex.

Before splitting the signals in epochs, the normalization is important. According to theoretical studies, Copula transformation is used.

As can be seen in figure 12, Copula transformation allows to redistribute the signal in a range [0,1], to compare signals of the same amplitude range but with a different wave form. The theory about Copula transformation is discussed below, in section 2.2.2

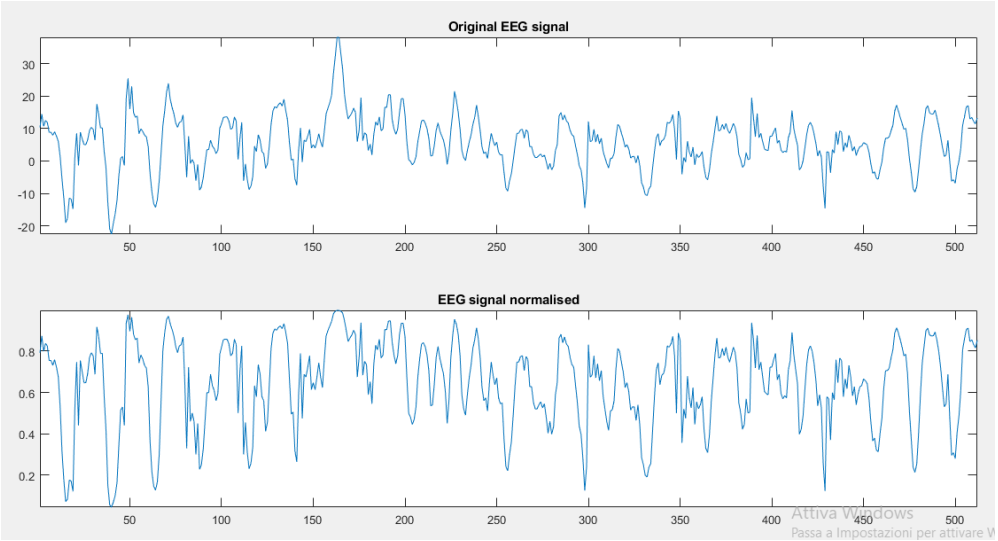


Figure 12: signal pre and post Copula transformation

Once normalised, the signals are splitted in different epochs according to their length. So, the code evaluates the number of epochs and multiplies it for two. In that way, the overlap equal to 50 % is considered.

This procedure is shown in figure 13.

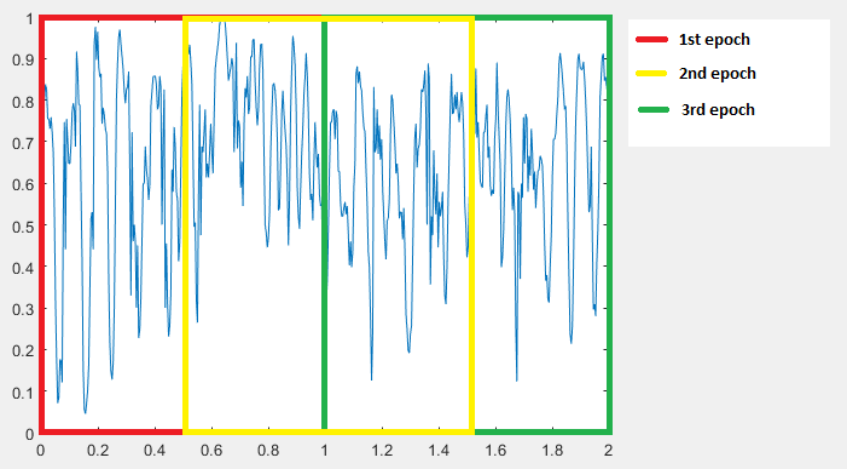


Figure 13: 50% overlap

2.2.1 Copula transformation

In probability theory and statistics, a copula is a multivariate cumulative distribution function for which the marginal probability distribution of each variable is uniform. Copulas are used to describe the dependency between random variables.

Copulas are popular in high-dimensional statistical applications as they allow one to easily model and estimate the distribution of random vectors by estimating marginals and copula separately. There are many parametric copula families available, which usually have parameters that control the strength of dependency.

Two-dimensional copulas are known in some other areas of mathematics under the name permutons and doubly-stochastic measures.

From a mathematical point of view, it is supposed that $(X_1, X_2, \dots, X_d)$ is a random vector and that its marginal are continuous, so the margianal CDFs (Continuous distribution functions)

$$F_i = \Pr[X_i \leq x]$$

are continuous too.

Once the probability integral transform is applied to each component, the random vector

$$(U_1, U_2, \dots, U_d) = (F_1(X_1), F_2(X_2), \dots, F_d(X_d))$$

has uniformly distributed marginals.

So, the copula of $(X_1, X_2, \dots, X_d)$ is defined as the joint cumulative distribution function of $(U_1, U_2, \dots, U_d)$:

$$C(u_1, u_2, \dots, u_d) = \Pr[U_1 \leq u_1, U_2 \leq u_2, \dots, U_d \leq u_d]$$

The copula *C* contains all information on the dependence structure between the components of $(X_1, X_2, \dots, X_d)$ whereas the marginal cumulative distribution functions F_i contains all information on the marginal distributions.

An important aspect of what explained, is that the reverse steps can generate pseudo-random samples. Mathematically, the sample (U1, U2, …, Ud) can be estimated from the copula distribution thanks to:

$$(X1, X2, \dots, Xd) = (F_1^{-1}(U_1), F_2^{-1}(U_2), \dots, F_d^{-1}(U_d))$$

Finally, the copula transformation can be rewritten on the base of the above as:

$$C(u_1, u_2, ..., u_d) = \Pr [X_1 \leq (F_1^{-1}(u_1), X_2 \leq (F_2^{-1}(u_2), ..., X_d \leq (F_d^{-1}(u_d)]$$

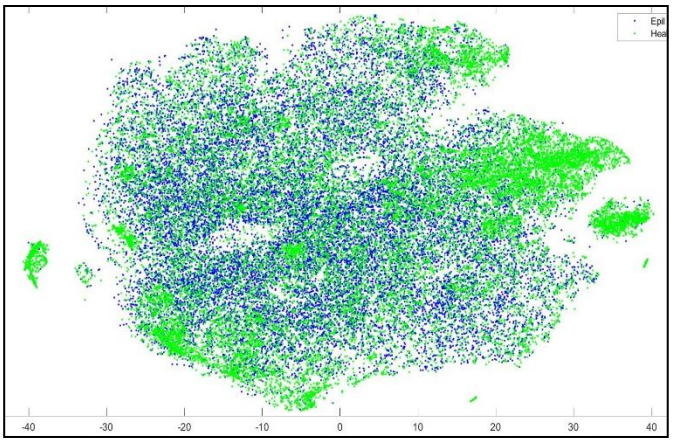
2.3 Convolutional autoencoder code

To use signals in Matlab with a convolutional autoencoder network, the reshaping of data is needed, passing to a 4D space where the dimensions are [1, epoch length, 1, number of epochs].

Thus, the creation of the net is possible.

Our net is composed by:

- Encoding phase
  - convolutional layer with a filter size [1,5] and a number of filters equal to 32
  - ReLu layer
  - max pool layer with a pool size equal to [1,2]
  - convolutional layer with a filter size [1,5] and a number of filters equal to 32
  - ReLu layer
  - max pool layer with a pool size equal to [1,2]
  - convolutional layer with a filter size [1,5] and a number of filters equal to 64
  - ReLu layer
  - max pool layer with a pool size equal to [1,2]
  - convolutional layer with a filter size [1,5] and a number of filters equal to 128
  - ReLu layer
  - max pool layer with a pool size equal to [1,2]
  - a layer for data reshaping to work with a single vector and not with matrix
  - a fully connected layer with output size equal to 16
- Decoding phase
  - a fully connected layer with output size equal to 16*128
  - a layer for data reshaping to work with matrix and not with vector
  - a transposed convolutional layer with filter size [1,2] and number of filters equal to 128
  - ReLu layer



- a transposed convolutional layer with filter size [1,2] and number of filters equal to 64
- ReLu layer
- a transposed convolutional layer with filter size [1,2] and number of filters equal to 32
- ReLu layer
- a transposed convolutional layer with filter size [1,2] and number of filters equal to 128
- ReLu layer
- convolutional layer with a filter size [1,5] and a number of filters equal to 1
- a Clipped Rectified Linear Unit (ReLU) layer

ReLu layers are needed to perform a threshold operation to each element of the input, where any values less than zero is set to zero, as shown in figure 14.

Mathematically:

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$$

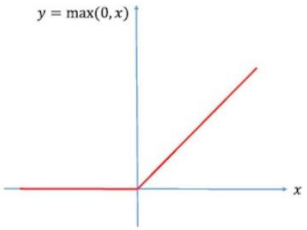


Figure 14: ReLU function

Instead, the fully connected layers are used to change the number of elements. In fact, during the encoding phase, the net shifts from 16*128 elements to 16. In the decoding phase the inverse operation is done: the number of elements change from 16 to 16*128.

Obviously, parameters of decoding phase are chosen to restore the original signals, so they are opposite to the ones of the encoding phase.

The table 1 contains a summary of the parameters used by the net.

	Name	Type	Activations	Learnables
1	imageinput 1x256x1 images...	Image Input	1×256×1	-
2	conv_1 32 1x5x1 convol...	Convolution	1×256×32	Weights 1×5×1×32 Bias 1×1×32
3	relu_1 ReLU	ReLU	1×256×32	-
4	maxpool_1 1x2 max pooling...	Max Pooling	1×128×32	-
5	conv_2 32 1x5x32 conv...	Convolution	1×128×32	Weights 1×5×32×32 Bias 1×1×32
6	relu_2 ReLU	ReLU	1×128×32	-
7	maxpool_2 1x2 max pooling...	Max Pooling	1×64×32	-
8	conv_3 64 1x5x32 conv...	Convolution	1×64×64	Weights 1×5×32×64 Bias 1×1×64
9	relu_3 ReLU	ReLU	1×64×64	-
10	maxpool_3 1x2 max pooling...	Max Pooling	1×32×64	-
11	conv_4 128 1x5x64 con...	Convolution	1×32×128	Weights 1×5×64×128 Bias 1×1×128
12	relu_4 ReLU	ReLU	1×32×128	-
13	maxpool_4 1x2 max pooling...	Max Pooling	1×16×128	-
14	layer_1 personalLayer3	personalLayer3	2048	-
15	fc_1 30 fully connect...	Fully Connected	16	Weights 16x2048 Bias 16x1
16	fc_2 2048 fully conne...	Fully Connected	2048	Weights 2048x16 Bias 2048×1
17	layer_2 personalLayer4	personalLayer4	1×16×128	-
18	transposed-c... 128 1x4x128 tra...	Transposed Convol...	1×32×128	Weights 1×4×128×128 Bias 1×1×128
19	relu_5 ReLU	ReLU	1×32×128	-
20	transposed-c... 64 1x4x128 tran...	Transposed Convol...	1×64×64	Weights 1×4×64×128 Bias 1×1×64
21	relu_6 ReLU	ReLU	1×64×64	-
22	transposed-c... 32 1x4x64 trans...	Transposed Convol...	1×128×32	Weights 1×4×32×64 Bias 1×1×32
23	relu_7 ReLU	ReLU	1×128×32	-
24	transposed-c... 32 1x4x32 trans...	Transposed Convol...	1×256×32	Weights 1×4×32×32 Bias 1×1×32
25	relu_8 ReLU	ReLU	1×256×32	-
26	conv_5 1 1x5x32 convol...	Convolution	1×256×1	Weights 1×5×32 Bias 1×1
27	clippedrelu Clipped ReLU w...	Clipped ReLU	1×256×1	-
28	regressionout... mean-squared-e...	Regression Output	-	-

Table 1: layers parameters

However, the layers' conformation is not the only choice to create the net.

In fact, Matlab allows to select training options thanks to the homonym command. In particular, the proposed net has:

- Adam optimizer;
- Maximum number of iteration epochs equal to 5 and 15 according to different test;
- Initial learning rate equal to 10^{-4}.

This specific command allows to impose other parameters, but in this study, default parameters have been used.

Another important aspect of this function offered by Matlab is the possible visualization of the training of the net thanks to a plot, recallable with the pair "plots-training-progress".

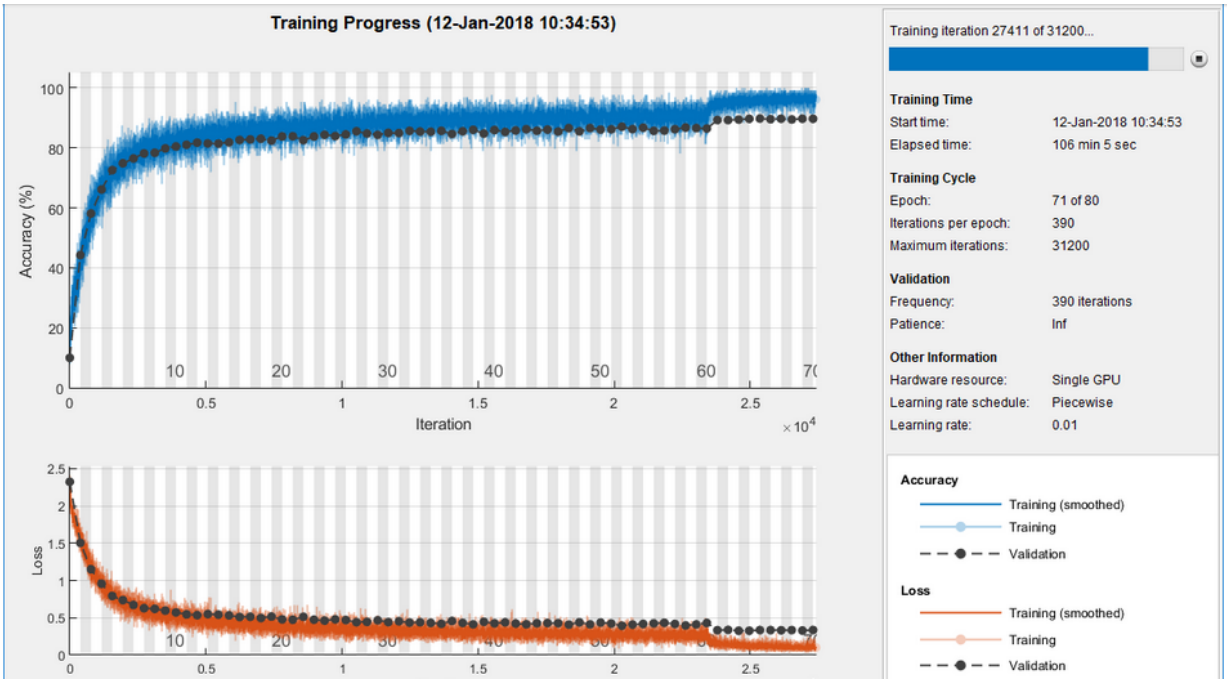


Figure 15: example of training progress plot

Once the net is constructed, it is trained by the command 'trainNetwork', using the following parameters:

- Training set as input data to encode;
- The same training set as output data to restore;
- Sequence of layers;
- Training options.

The test phase is now possible. If the training set corresponds to the 90% of the whole healthy data set, the test one is composed by the 10% of healthy data set and the same number of epileptic signals.

The test phase is possible with the simple command "predict" that takes as input an external signal not used in the training phase e tries to reconstruct it with the parameters of the net.

Finally, it is important to underline that, with this network, three different tests have been done:

1. One channel (T3) used and 15 as maximum number of iteration epochs of the net;
2. Two channels (T3 and T4) used and 5 as maximum number of iterations epochs of the net;
3. One channel (T3) used with 100 healthy subjects and 100 epileptic subjects for training phase and a maximum number of iterations of the net equal to 5.

Once matrixes are obtained, hidden layer of dimension equal to 16 is extrapolated thanks to "activation" command. To this layer, that is the representation of the 216 sample of each epoch in only 16 values, t-sne is applied. The aim of this technique is to distinguish in a 2D space epileptic epochs. This concept is shown in figure 16.

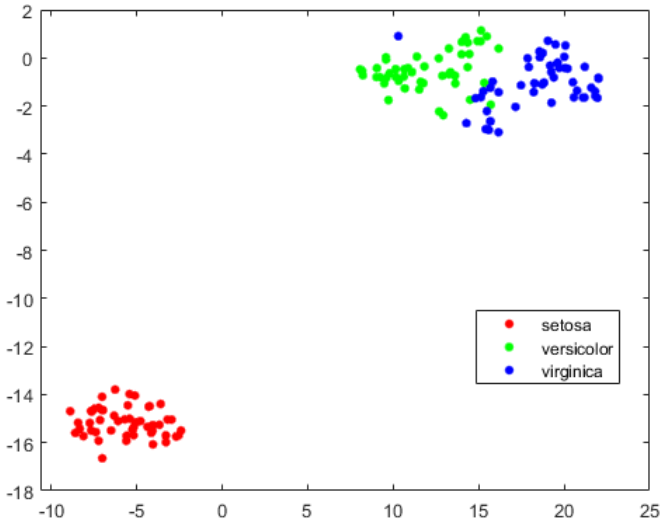


Figure 16: scatter plot of t-sne

2.4 Deep infomax network code

Even in this case, the training phase is made by the analysis of healthy subjects to discriminate epileptic ones in a second phase.

Obviously, the first step is to create the net structure, imposing following parameters:

- Adam optimizer;
- Learning rate equal to 0.001;
- Regularization of L2 norm equal to 0.

Figure 17 describes the blocks used during the development of the net. It is important to remember that this type of net has the encoding part only.

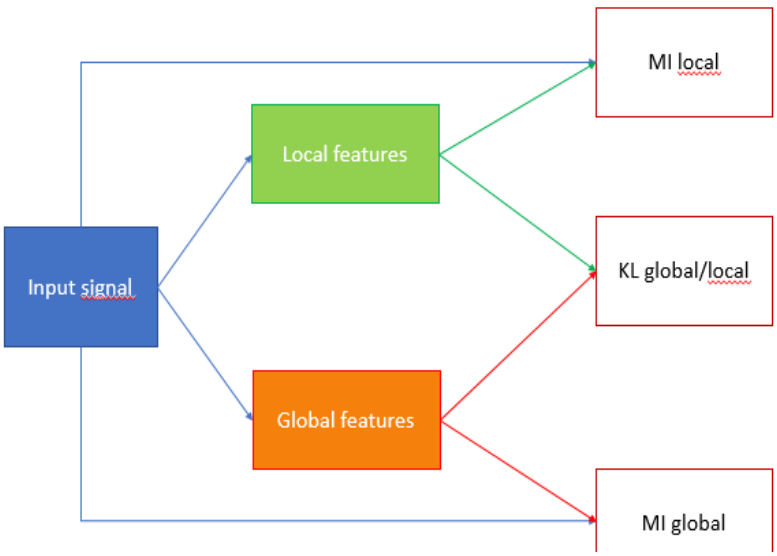


Figure 17: block diagram of the net

It is now possible to analyse each block, describing parameters used in the experiments.

1. **Extrapolation of local features.** It takes as input the signal and encodes it in a feature map reflecting some aspects of the data, such as spatial locality. The obtained map is the concatenation of those feature vectors evaluated for each location.
2. **Extrapolation of global features.** As it is shown in figure 18, the extrapolation of global features consists in the summarization of local vectors into a single feature vector, Y. Our goal is to train this network such that useful information about the input is easily extracted from the high-level features.

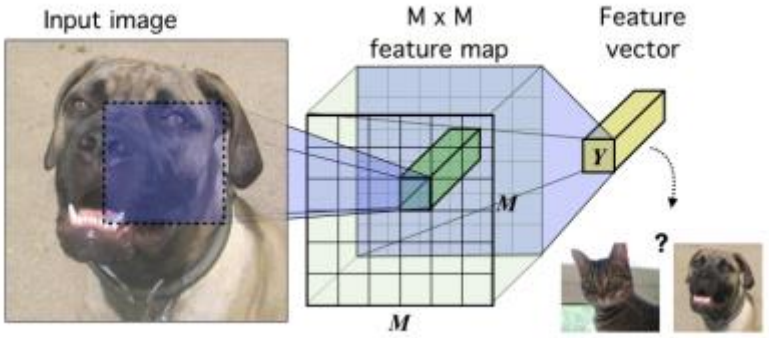


Figure 18: Extrapolation of global features

3. **Optimization of global mutual information.** This block has the aim to combine the lower-level MxM features map and the high-level feature vector, Y, thanks to a discriminator to get a score. The aim is to maximize E_ψ: $X \rightarrow Y$, where X and Y are domain range of E, that is a continuous and differentiable parametric function with parameters ψ.

$$(\hat{\omega}, \hat{\psi})_G = \arg\max_{\omega,\psi} \hat{\mathcal{I}}_\omega(X; E_\psi(X))$$

4. **Optimization of local mutual information.** The equation above could be used to maximize mutual information between input and output in the most of cases, but everything depends on the task. For example, the presence of noise is useless for signal classification, so its representation is not a benefit and the net has to try not to encode it. One alternative is to maximize the average of mutual information between the high-level representation and local patches of the image.
   After input encoding into a feature map,

$$C_\psi(x) := \{C_\psi^i\}_{i=1}^{MxM}$$

   it is summarized into a global feature

$$E_\psi(x) = f_\psi \circ C_\psi(x)$$

   It is now possible to define the mutual information estimator on global/local pairs, maximizing the average MI previously estimated.
   Mathematically:

$$(\hat{\omega}, \hat{\psi})_L = \arg\max_{\omega,\psi} \frac{1}{M^2} \sum_{i=1}^{M^2} \hat{\mathcal{I}}_{\omega,\psi}(C_\psi^{(i)}(X); E_\psi(X))$$

5. **Evaluation of Kullback- Leibler divergence.** In probability theory and in information theory, the Kullback-Leibler divergence (named also divergency of information or relative entropy) is an asymmetric measure of the difference between two probability distribution P and Q. More detailed, the KL divergency from Q to P, represented by $D_{KL}(P||Q)$, is the measure of lost information when Q is used to approximate P.

Once the net is created, its training is possible using healthy subjects. The aim is to obtain 16 samples from the encoder and use them to distinguish healthy epochs from epileptic ones. The latter are present in testing phase only.

Different tests have been done using this net:

- Learning rate equal to 10^{-3} and decay of learning rate equal to 1;
- Learning rate equal to 10^{-4} and decay of learning rate equal to 0.5;
- Learning rate equal to 10^{-4} and decay of learning rate equal to 0.625;
- Learning rate equal to 10^{-4} and decay of learning rate equal to 0.75;
- Learning rate equal to 10^{-4} and decay of learning rate equal to 0.875;
- Learning rate equal to 10^{-4} and decay of learning rate equal to 1;
- Learning rate equal to 0.0032 and decay of learning rate equal to 0.5;
- Learning rate equal to 0.0032 and decay of learning rate equal to 0.625;
- Learning rate equal to 0.0032 and decay of learning rate equal to 0.75;
- Learning rate equal to 0.0032 and decay of learning rate equal to 0.875;
- Learning rate equal to 0.0032 and decay of learning rate equal to 1;
- Learning rate equal to 0.1 and decay of learning rate equal to 0.5;
- Learning rate equal to 0.1 and decay of learning rate equal to 0.625;
- Learning rate equal to 0.1 and decay of learning rate equal to 0.75;
- Learning rate equal to 0.1 and decay of learning rate equal to 0.875;
- Learning rate equal to 0.1 and decay of learning rate equal to 1;
- Learning rate equal to 3.16 and decay of learning rate equal to 0.5;
- Learning rate equal to 3.16 and decay of learning rate equal to 0.625;
- Learning rate equal to 3.16 and decay of learning rate equal to 0.75;
- Learning rate equal to 3.16 and decay of learning rate equal to 0.875;
- Learning rate equal to 3.16 and decay of learning rate equal to 1;
- Learning rate equal to 100 and decay of learning rate equal to 0.5;
- Learning rate equal to 100 and decay of learning rate equal to 0.625;
- Learning rate equal to 100 and decay of learning rate equal to 0.75;
- Learning rate equal to 100 and decay of learning rate equal to 0.875;
- Learning rate equal to 100 and decay of learning rate equal to 1.

Using those networks and testing them with both healthy and epileptic subjects, matrixes composed by 16 samples for each epoch are ready.
They are used as input of t-sne to decrease further the dimension from 16 to 2 as in the previous net.

3. Results

The results about every single test will be discussed and shown in the following. It is important to underline that effectuated tests have not led to hoped results.
Probable causes and solutions will be discussed in chapter 4.

3.1 Convolutional autoencoder results

For the first test effectuated with convolutional autoencoder, that is the use of this net with one channel and 15 epochs available (in terms of iterations of net training), the main square error has been evaluated to analyse its actual functioning.

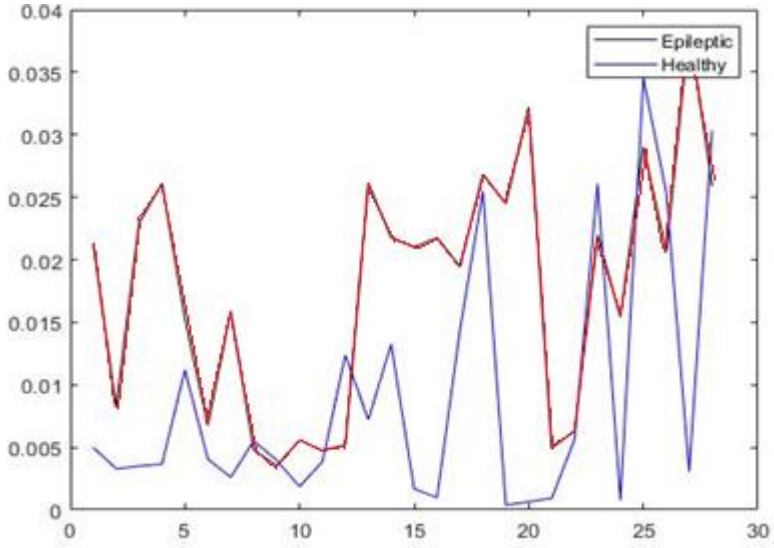


Figure 19: MSE one channel and 15 epochs

As shown in figure 20, the reconstruction of the original signal is quite good. However, it is possible to see that the net reconstructs well both the healthy epochs (coloured in blue in the graph) and the epileptic one (in red), so the evaluation of MSE is not a good discriminator to differentiate epileptic subjects.

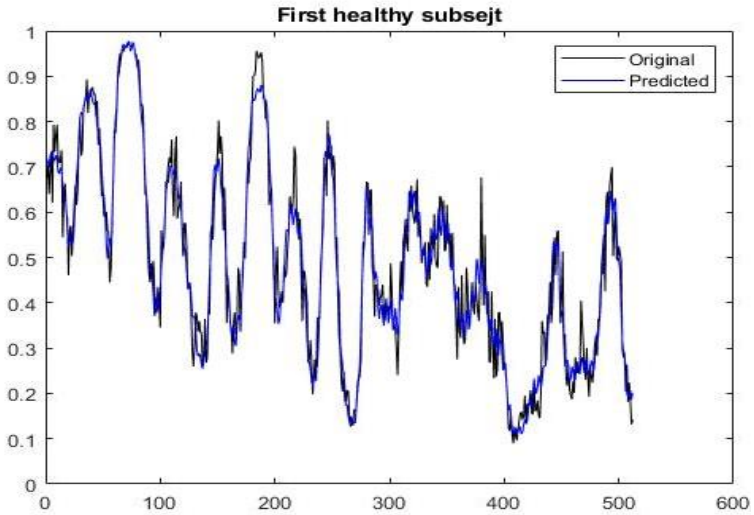


Figure 20: first epoch of healthy subject

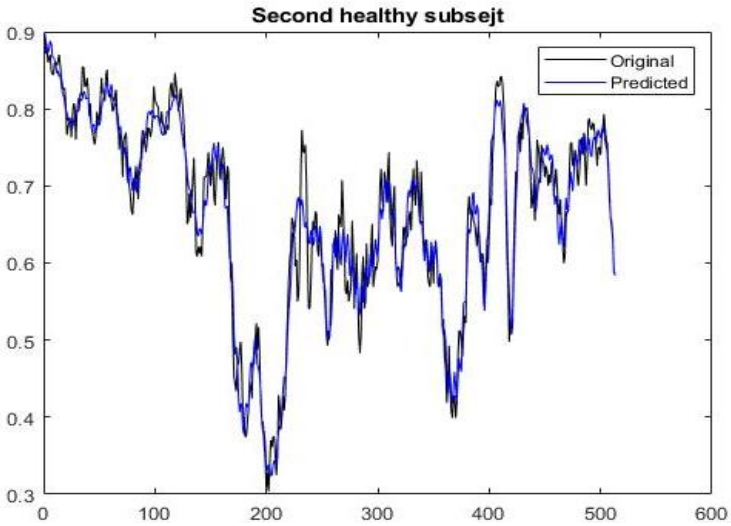


Figure 21: second epoch of healthy subject

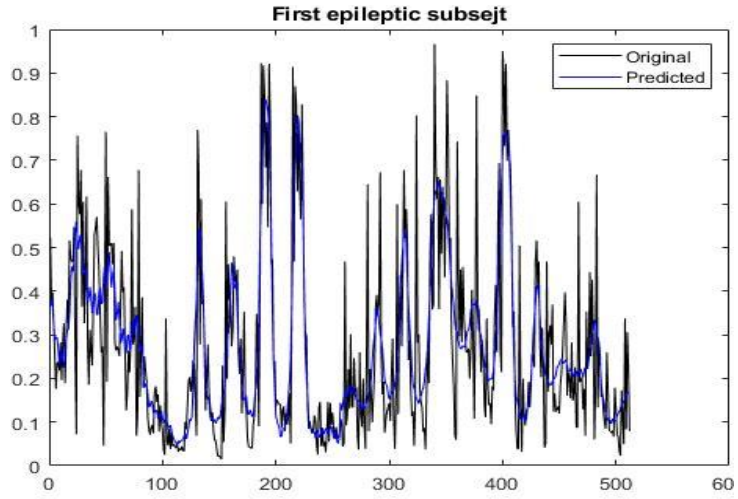


Figure 22: first epoch of epileptic subject

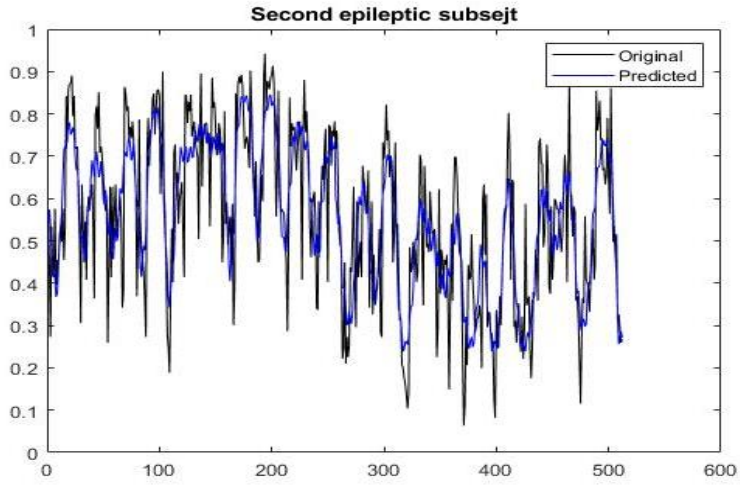


Figure 23: second epoch of epileptic subject

Figures 24,25,26,27 show the effective goodness of MSE. Each plot shows a random epoch extrapolated from random signal, both healthy and epileptic. Even if the amplitude is sometimes decreased, the wave form is always reproduced and in most of cases it is the base of the diagnosis of epilepsy.

As described in previous chapter, starting from the smallest hidden layer of each net, t-sne is applied to reduce further the dimension and to represent each epoch in a 2D space. Here, scatter plots are presented.

T-sne allows to evaluate the distance between point with different techniques. For our test, four different methods have been used:

1. Chebychev distance

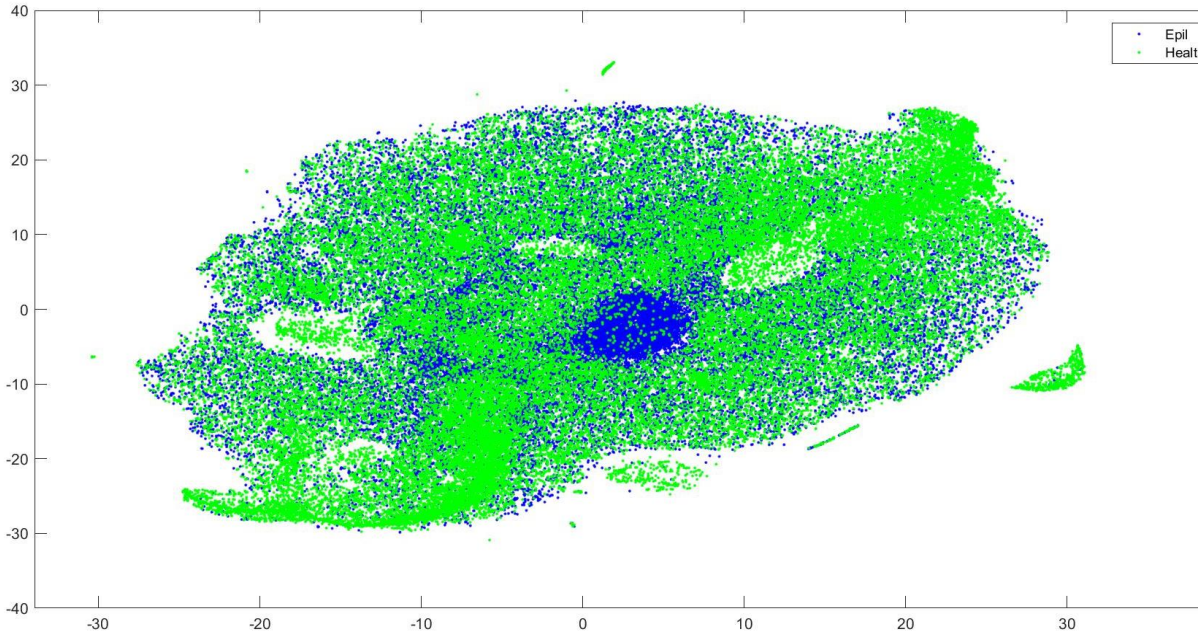


Figure 24: scatter plot with Chebychev distance

2. Euclidean distance

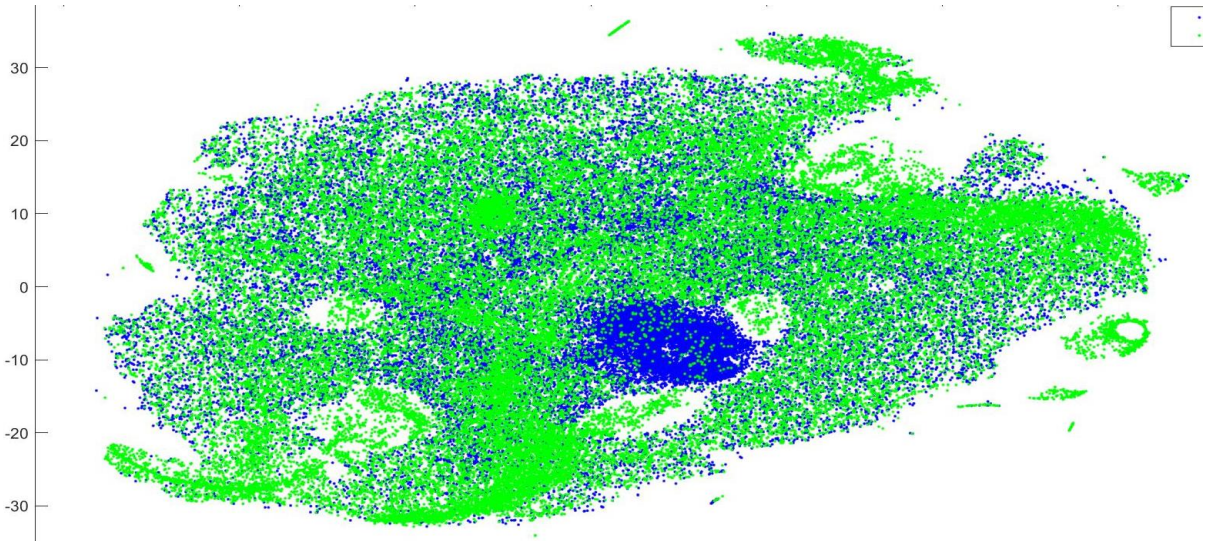


Figure 25: scatter plot with Chebychev distance

3. Cityblock distance

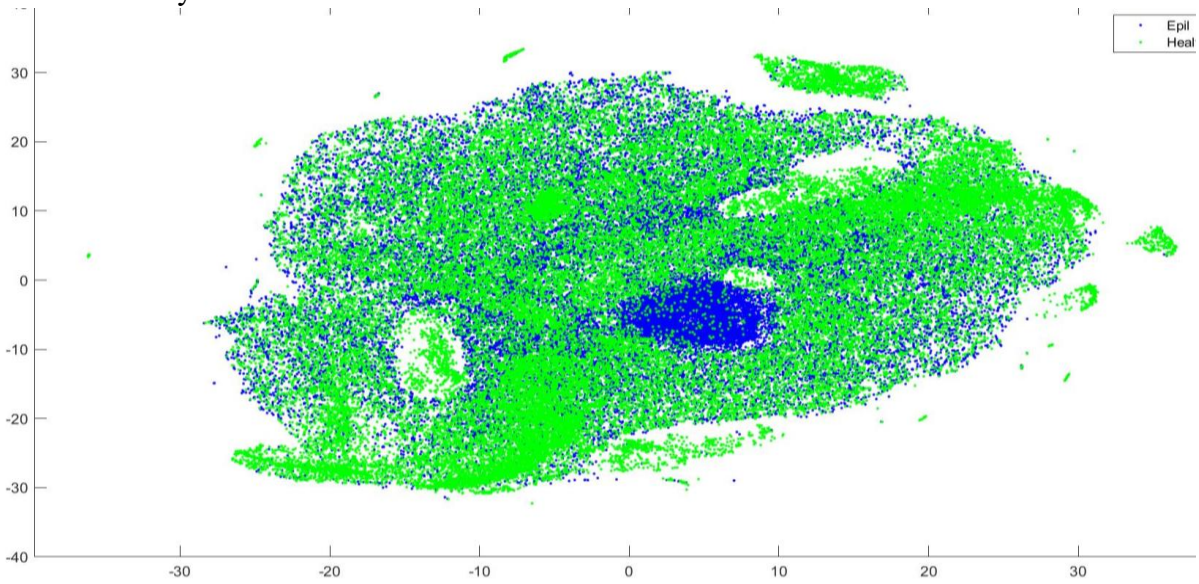


Figure 26: scatter plot with Cityblock distance

4. Mahalanobis distance

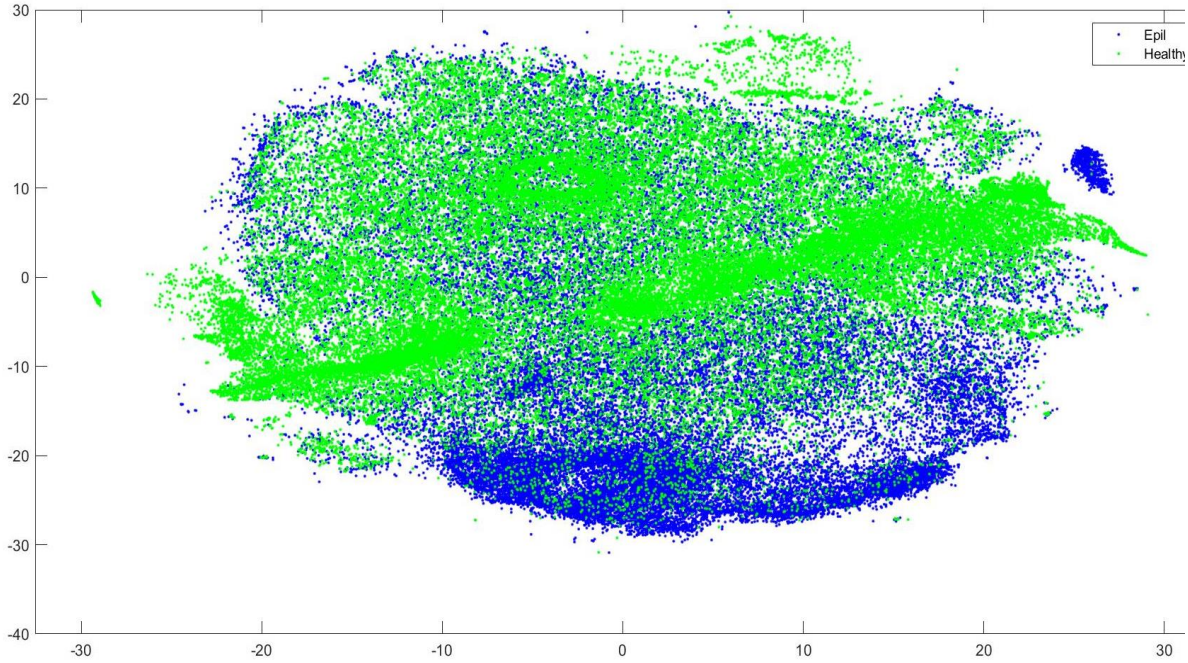


Figure 27: scatter plot with Mahalonobis distance

Starting from those scatter plots, where blue points represent epochs of epileptic signals and the green ones represent epochs of healthy signals, and considering that in epileptic signals not every epoch shows seizure, the next step has been to compare epileptic epochs in green region and analyse if blue region are characterized by actual signs of epilepsy.

Following plots show three different type of epochs:

- Epileptic epoch in healthy region;

- Epileptic epoch in epileptic region;
- Healthy epoch;

In every plot below, the periodicity of epileptic wave is evident. This could be an important result but is not sufficient to discriminate an epileptic subject from a healthy one.

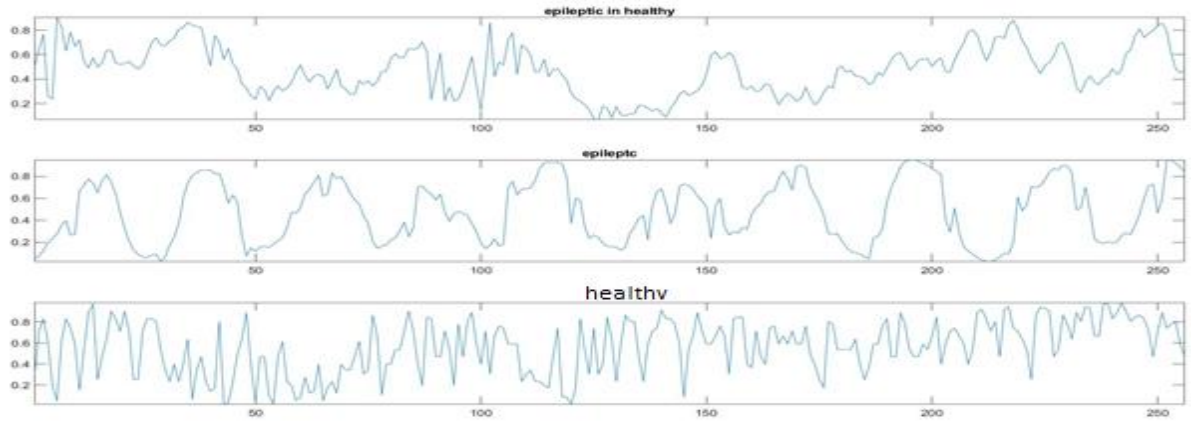


Figure 28: Signals from Chebychev scatter plot

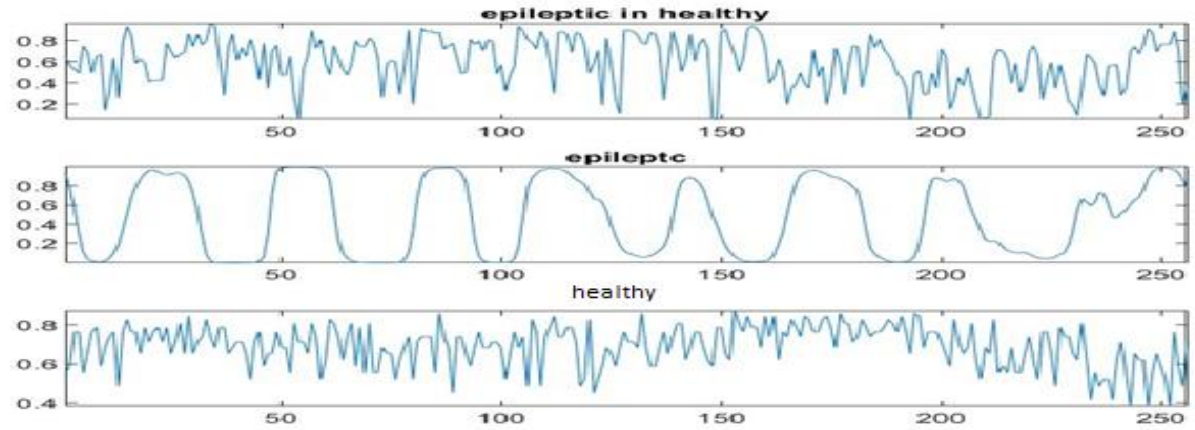


Figure 29: Signals from Euclidean scatter plot

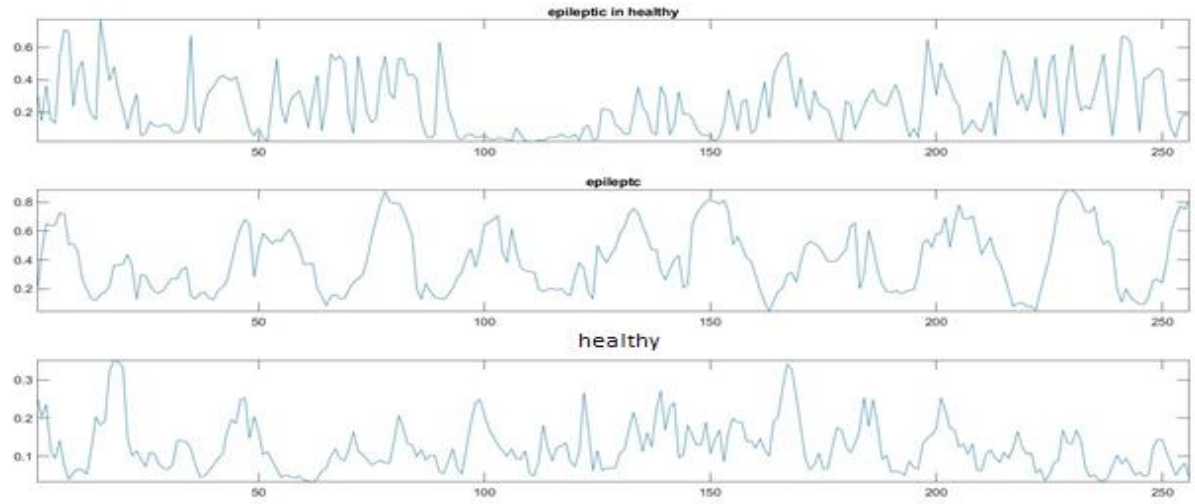


Figure 30: Signals from Cityblock distance

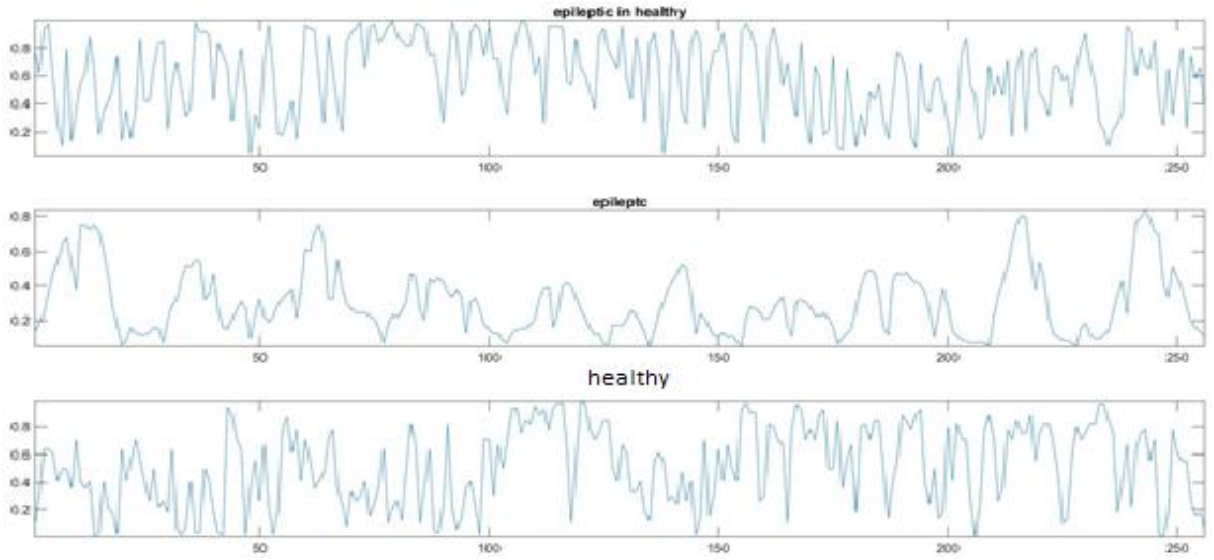


Figure 31: Signals from Mahalanobis distance

As mentioned before, the second test consists on the use of two channels (T3 and T4) as input signals and one epoch, intended as epoch of iteration of the net during the training phase.

The use of two channels instead only one is made because some epileptic seizures could not be seen in the left temporal region. It is important to underline that now the quantity of input signals is twice, and so also computational time is significantly increased.

As before, the firs parameter evaluated is MSE. As it is possible to see in figure 32, this time the difference between the two lines is clear, however, even in this case, this result is not so strong to discriminate an epileptic subject.

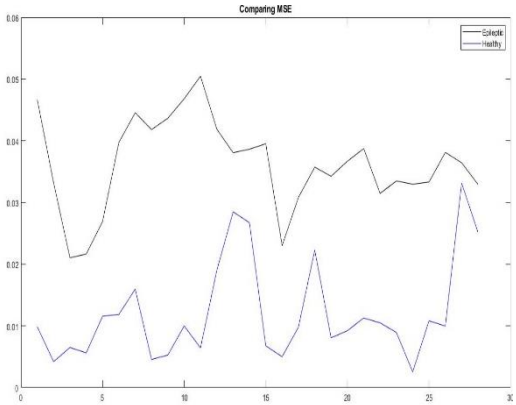


Figure 32: MSE using two channels and one epoch

On the base of previous graph, where the black line characterizes epileptic epochs and the blue one describes healthy subjects, even if the distinction is clear, MSE values are too close to each other and are too much like the ones obtained in the first experiment.

Another daunting parameter is similar to the original signals and those reconstructed by the net. In fact, as shown in following picture, the reconstruction is not as good as in the first test. Probably, this depends also on the random extraction of epochs to compare, because even if the MSE is comparable, in this case not only the amplitude of the signal is compromised, but also the wave form.

Unluckily, this aspect emerges for healthy subjects, as shown in figures 33 and 34, and for the

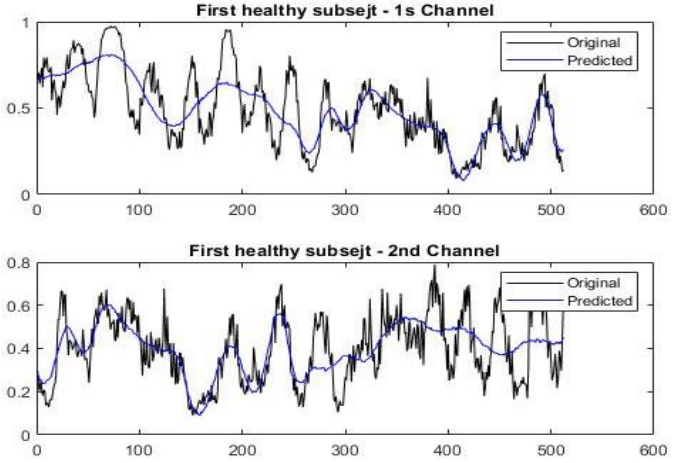


Figure 33: First example of healthy epoch reconstructed

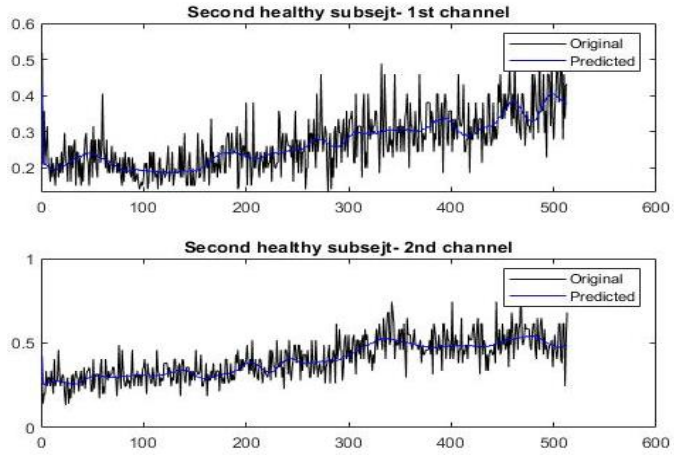


Figure 34: Second example of healthy epoch reconstructed

epileptic ones, as in figures 35 and 36.

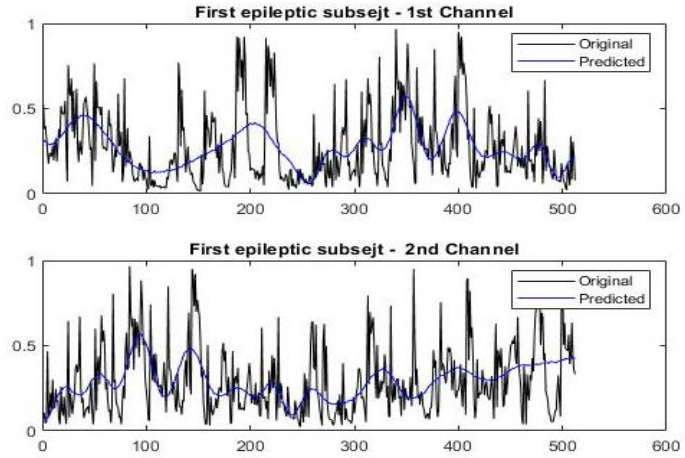


Figure 35: First example of epileptic epoch reconstructed

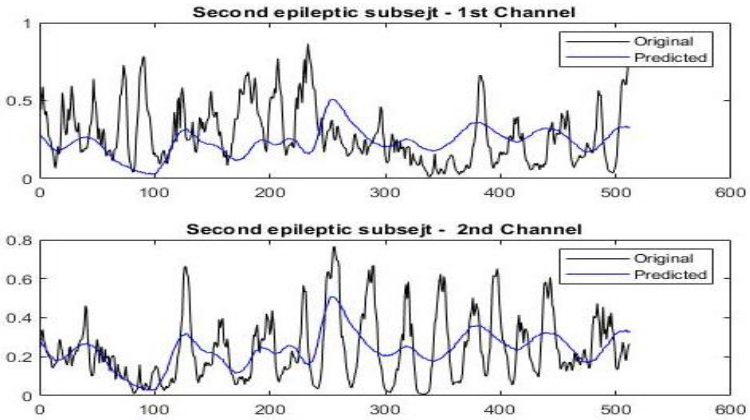


Figure 36: Second example of epileptic epoch reconstructed

Exactly as it has been done for the first test, even in this test the second step has been the application of t-sne to the shortest hidden layer (final encoding layer) to represent in a 2D space each epoch.

Figure 37 and 38 show results obtain by the application of Euclidean and Mahalanobis distances. However, this test has not led to a positive result. There is not a region characterized only by epileptic blue epochs for both methods.

Even Cityblock and Chebychev have been applied and they do not exhibit significant result as well.

According to figures above, it has not been necessary to compare signals coming from different regions of the graphs.

Analysing previous results, the main idea has been that the number of epileptic epochs was too small, so the third test consist in the use of a training set composed by 100 healthy signals and 100 epileptic ones.

However, the main problem is that we have signals of epileptic patients, but it is not sure that in each signal there is seizure. Another problem is that, even if in the signal seizures are present, most of tracks do not show signs of epilepsy, so the effective number of epileptic epochs is always smaller.

Figure 37: t-sne of second test with Euclidean distance

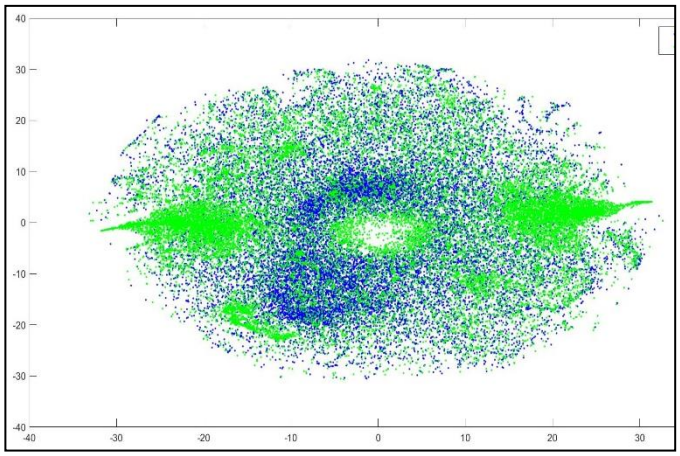


Figure 38: t-sne of second test with Mahalanobis distance

Probably this is the reason why also in this case results are not very satisfactory.

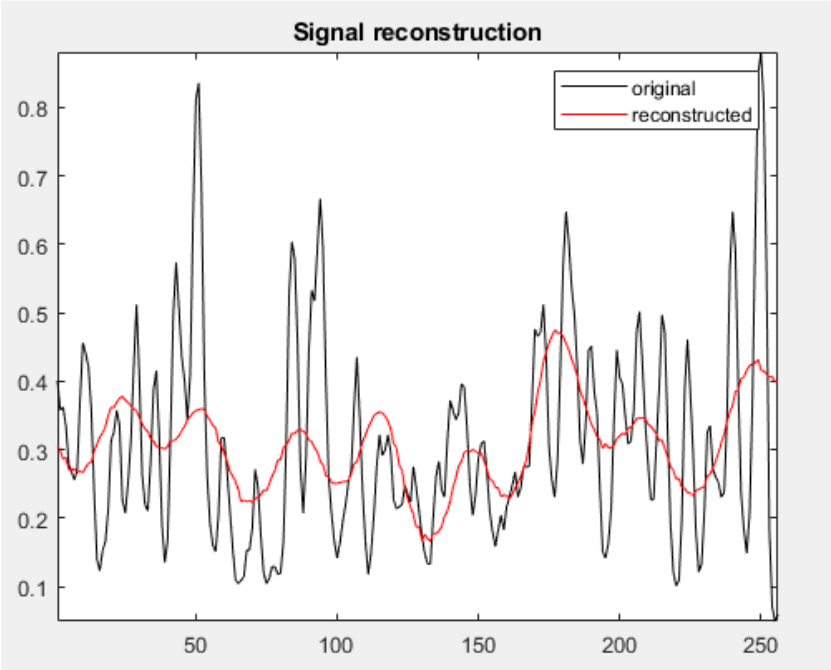


Figure 39: Signal reconstruction third test

Figure 39 shows the reconstruction of the signal with this third net. Red line indicates the reconstructed signal, while the black one is the original one. Even in this case both amplitude and wave form are not respected, so the goodness of the net is not so high.

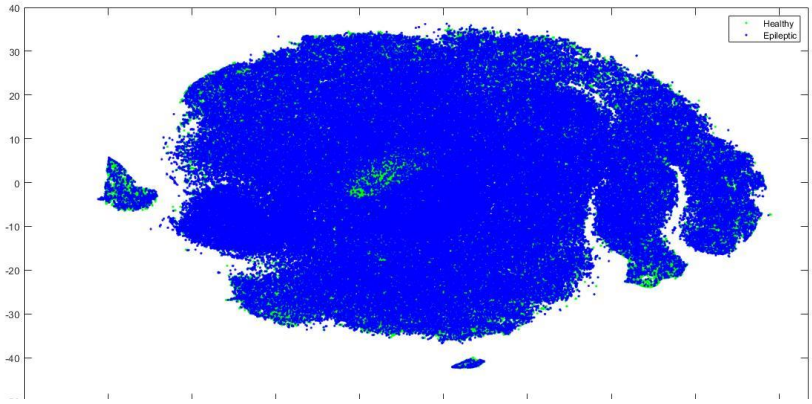


Figure 40: Scatter plot of third test

It is clear from figure 40, where blue points represent epochs from epileptic signals and green points represent epochs from healthy signals, this type of training do not lead to significant results, because there is not a single region characterized by epileptic epochs. This scatter plot has been obtained thanks to Mahalanobis distance, however with every other method the result is quite the same.

3.2 Deep Infomax results

Later, results about deep infomax net are presented. For each test scatter plots regarding both global and local features are presented with three different distances:

- Cityblock
- Euclidean
- Mahalanobis

However, satisfactory results have been obtained, because there is not a region characterized by the only epileptic epochs.

Possible reasons and possible solutions will be discussed in chapter 4.

1. Test with learning rate equal to 10^{-3} and decay of learning rate equal to 1

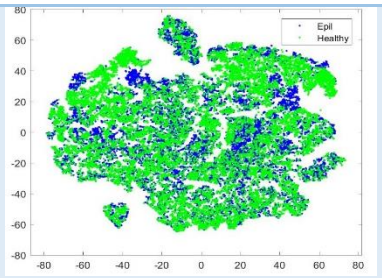
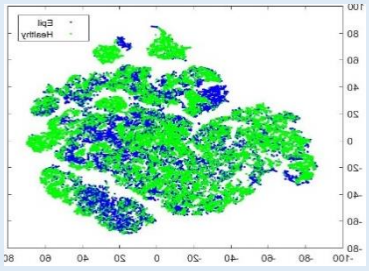
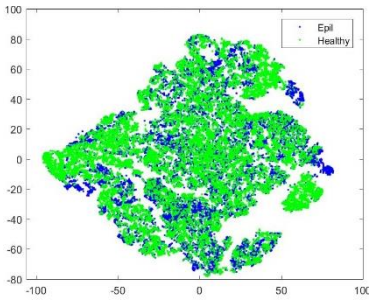
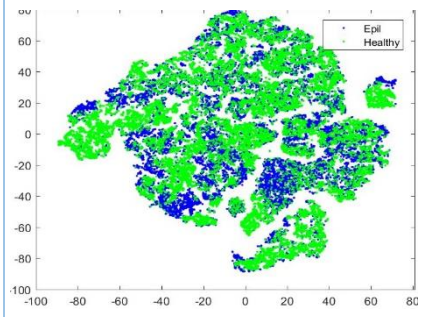
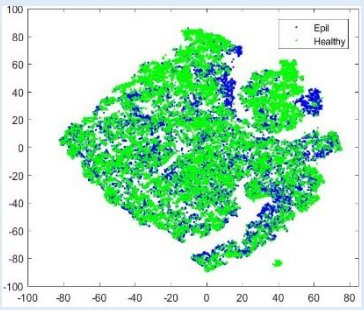
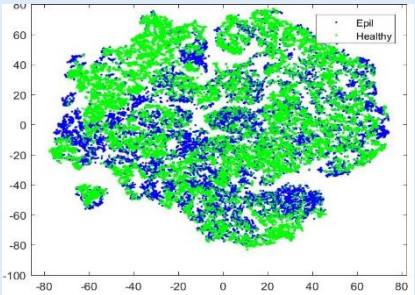
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 2: learning rate equal to 10^{-3} and decay of learning rate equal to 1

Table above shows results obtained from the first test with this net. In particular, columns contain information regarding local and global features that, as analysed in previous chapter, described signals trying to analyse epochs in detailed and to describe them in general respectively.

As clear from scatter plots in the table 1, in which blue point represent epochs from epileptic signals and the green one the epochs from healthy tracks, there are not blue big region. However, from the two scatter plots of the last row, where Mahalonobis distance has been used

for t-sne evaluation, there are two little regions full of only blue point. Starting from the letter, epochs in evidence as epileptic have been extracted and compared with the healthy ones.

Unluckily, results are not optimal, because the differences among tracks are not attributed to the presence of pathology, but also to noise, patient's movement, etc.

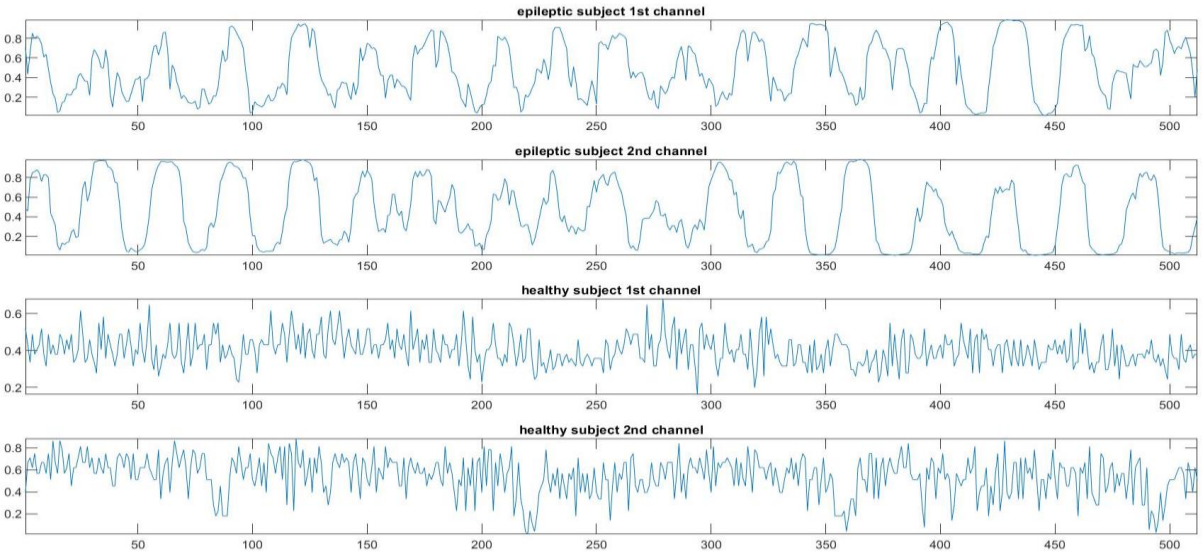


Figure 41: Signals from global features

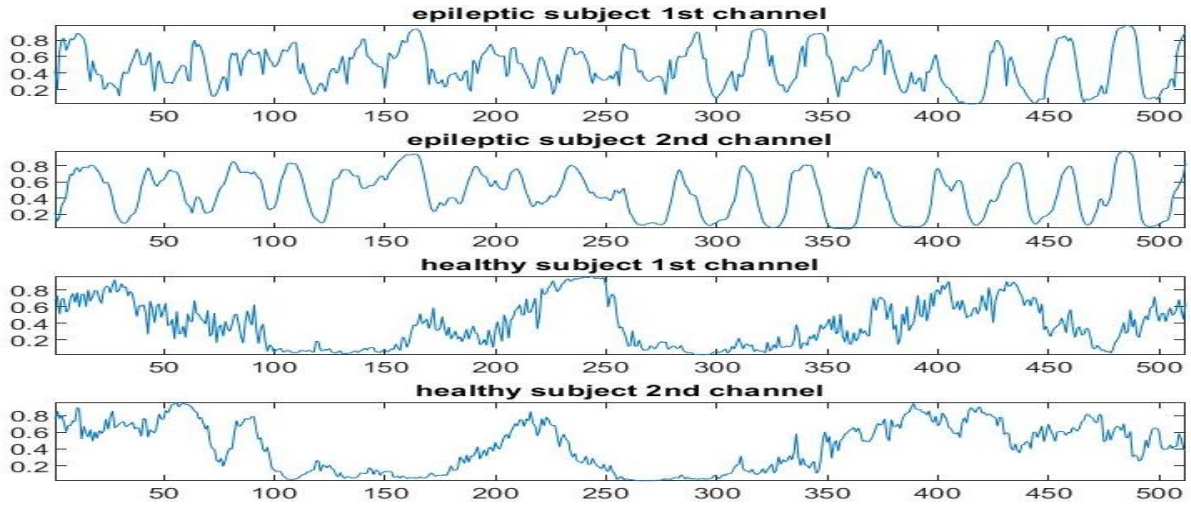


Figure 42: Signals from local features

Figures 41 and 42 confirm concept described above.

2. Test with learning rate equal to 10^{-4} and decay of learning rate equal to 0.5

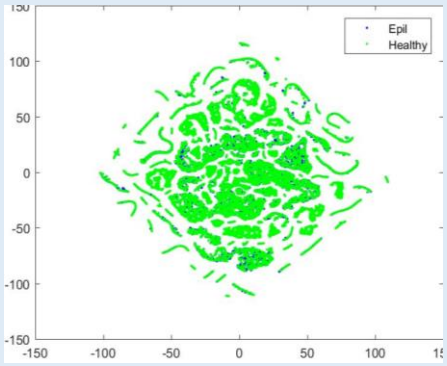
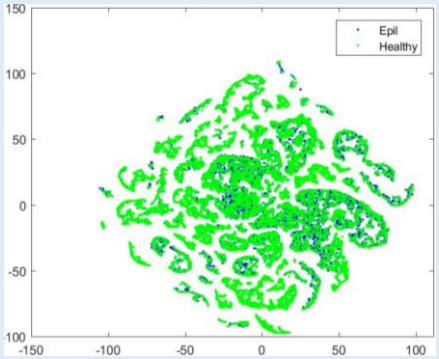
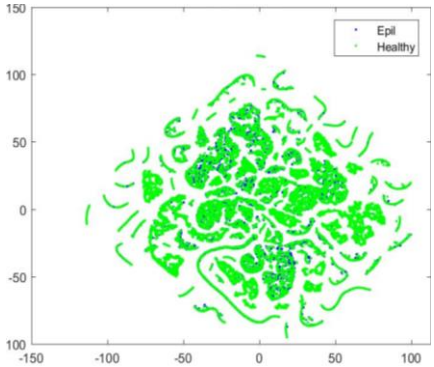
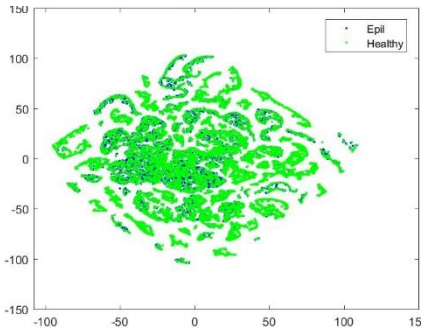
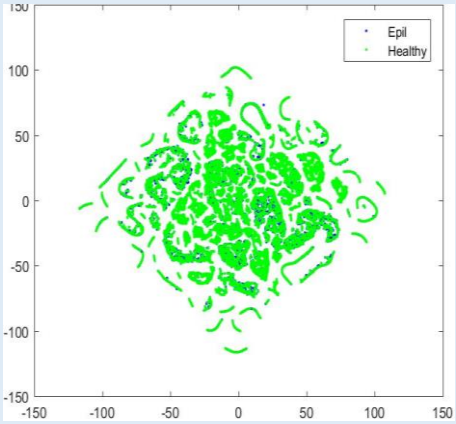
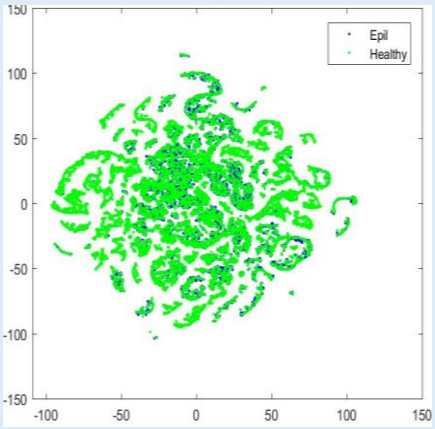
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 3: results learning rate equal to 10^{-4} and decay of learning rate equal to 0.5

3. Test with learning rate equal to 10^{-4} and decay of learning rate equal to 0.625

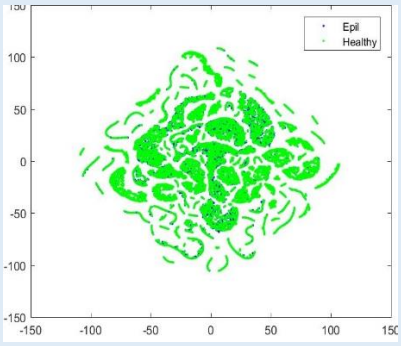
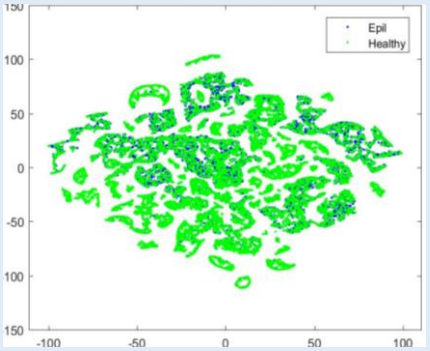
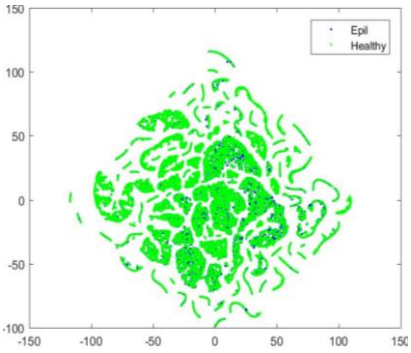
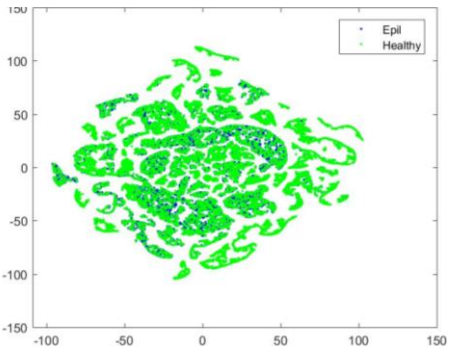
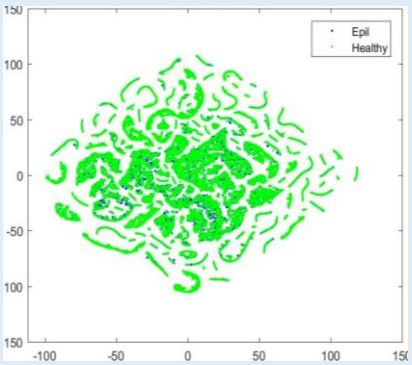
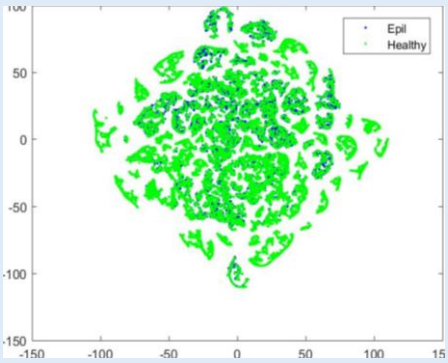
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 4: results learning rate equal to 10^{-4} and decay of learning rate equal to 0.625

4. Test with learning rate equal to 10^{-4} and decay of learning rate equal to 0.75

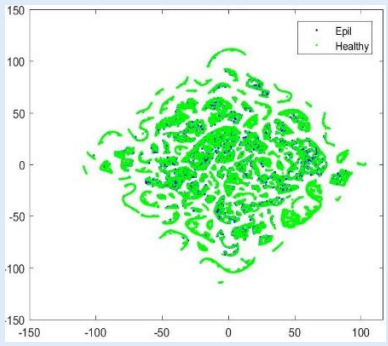
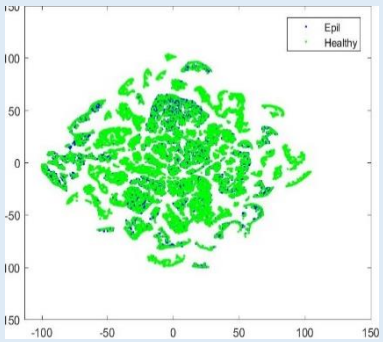
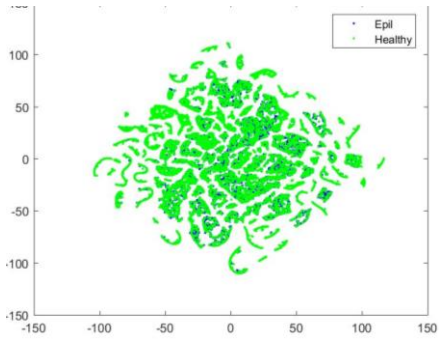
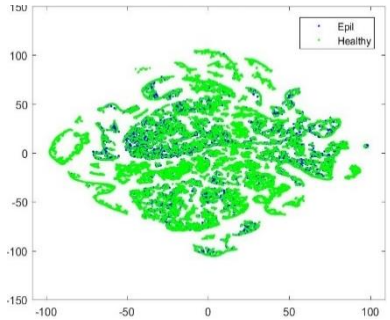
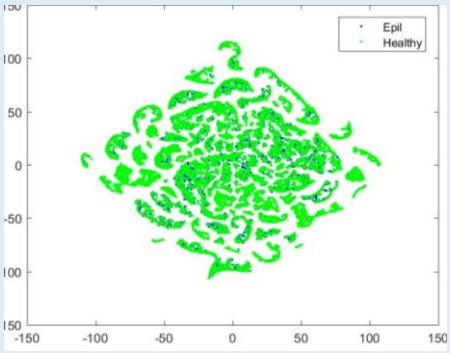
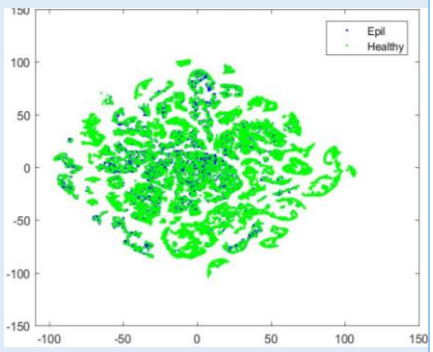
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 5: reults learning rate equal to 10^{-4} and decay of learning rate equal to 0.75

5. Test with learning rate equal to 10^{-4} and decay of learning rate equal to 0.875

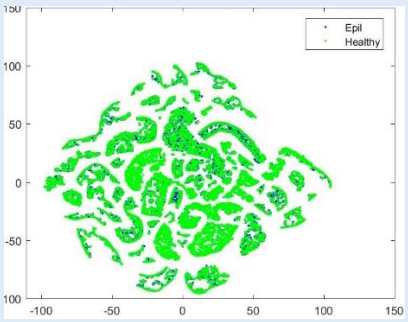
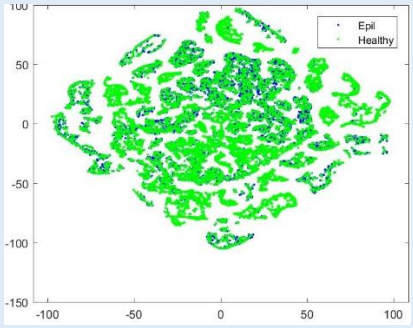
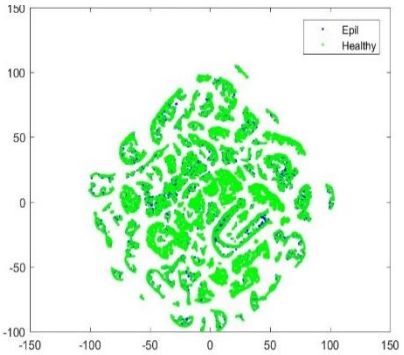
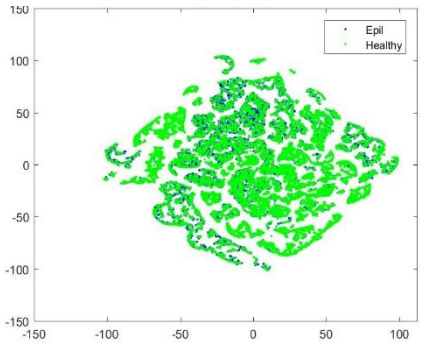
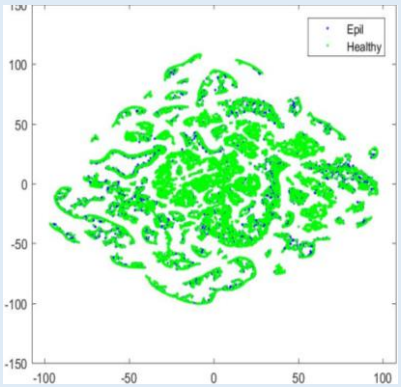
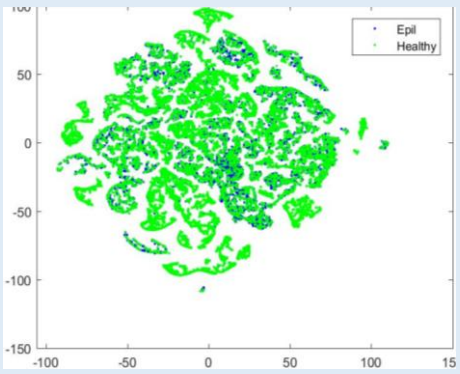
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 6: results learning rate equal to 10^{-4} and decay of learning rate equal to 0.875

6. Test with learning rate equal to 10^{-4} and decay of learning rate equal to 1

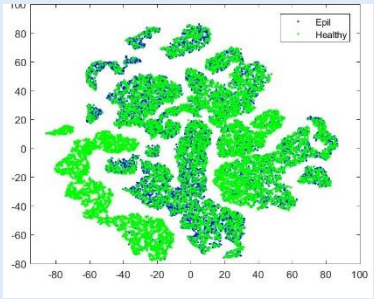
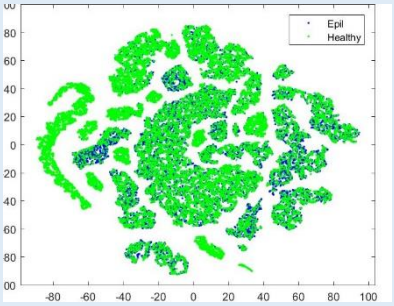
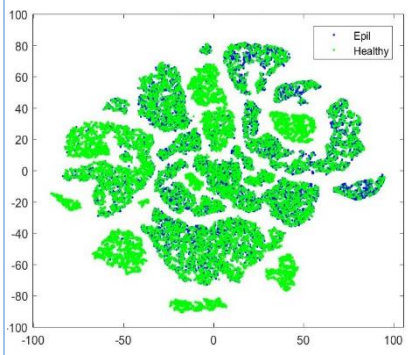
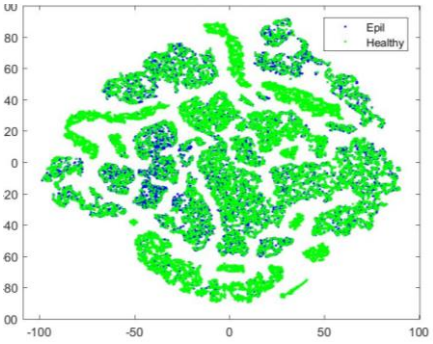
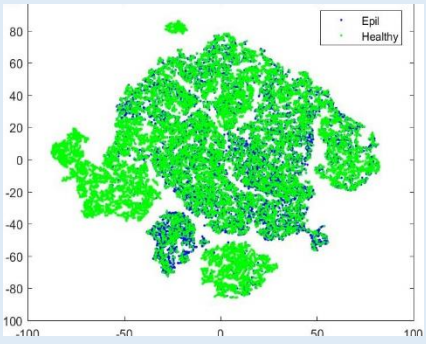
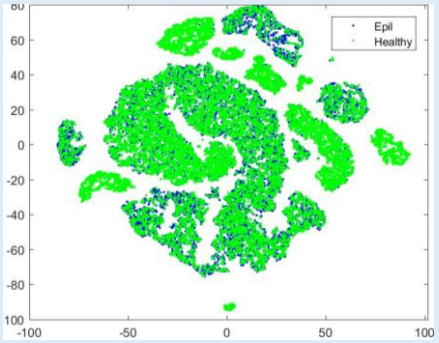
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 7: results with learning rate equal to 10^{-4} and decay of learning rate equal to 1

7. Test with learning rate equal to 0.0032 and decay of learning rate equal to 0.5

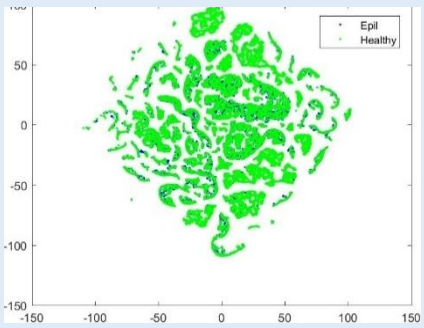
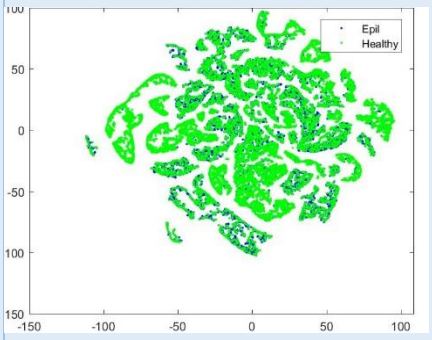
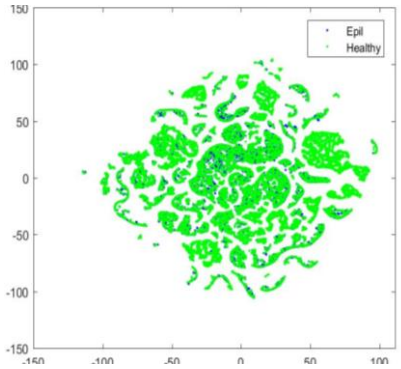
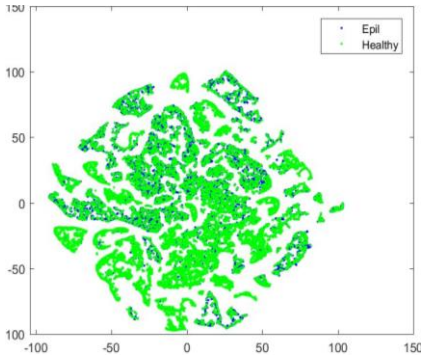
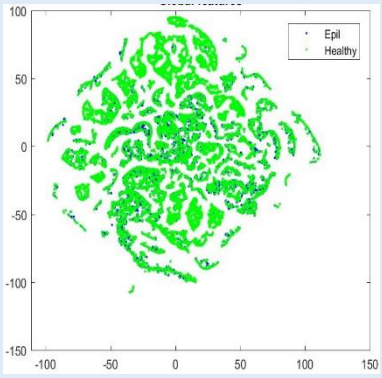
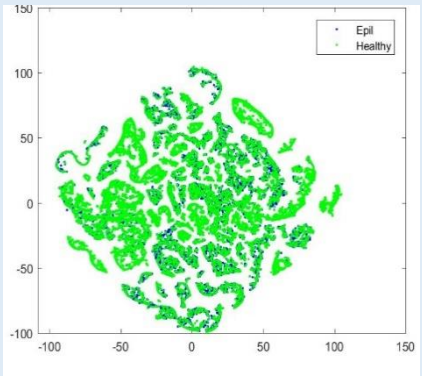
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 8: results learning rate equal to 0.0032 and decay of learning rate equal to 0.5

8. Test with learning rate equal to 0.0032 and decay of learning rate equal to 0.625

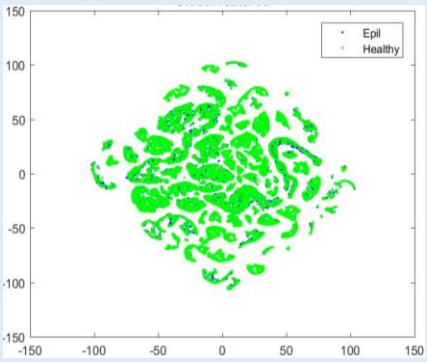
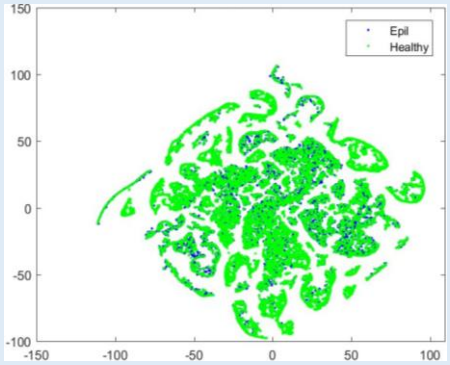
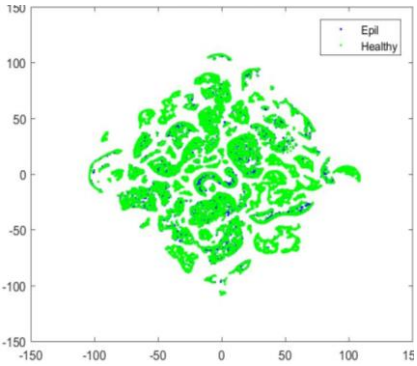
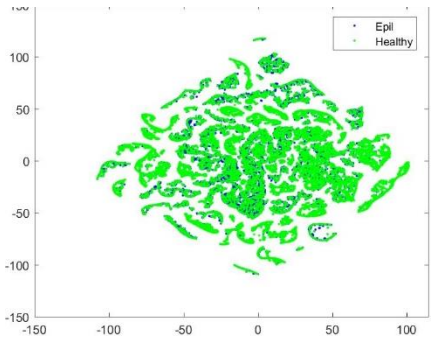
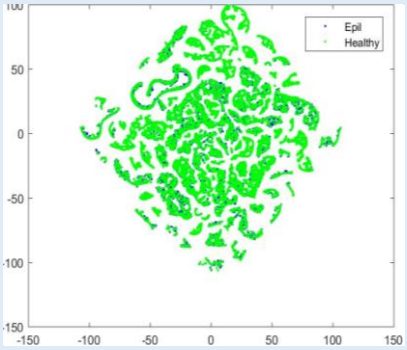
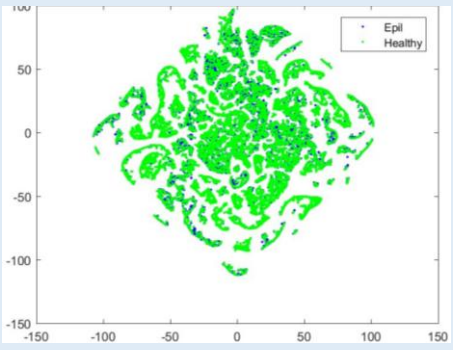
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 9: results learning rate equal to 0.0032 and decay of learning rate equal to 0.625

9. Test with learning rate equal to 0.0032 and decay of learning rate equal to 0.75

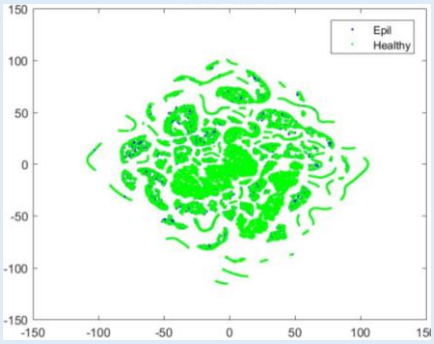
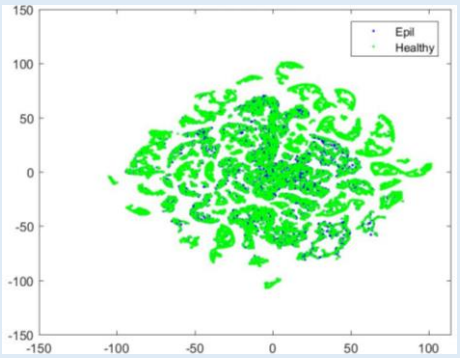
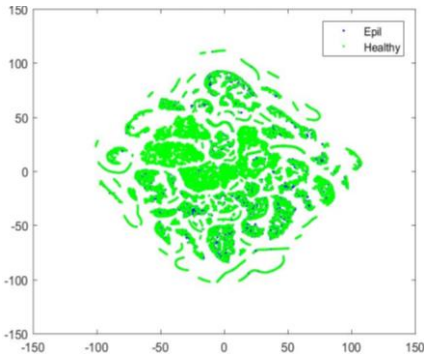
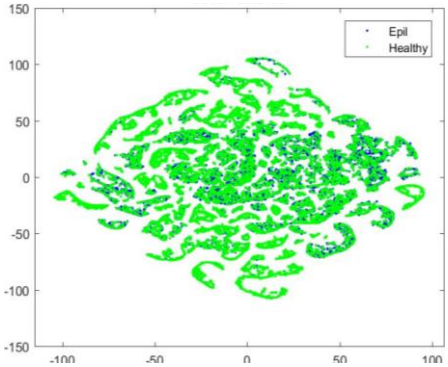
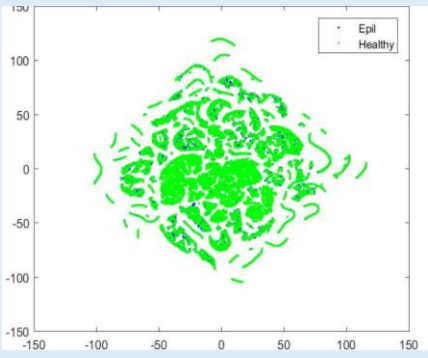
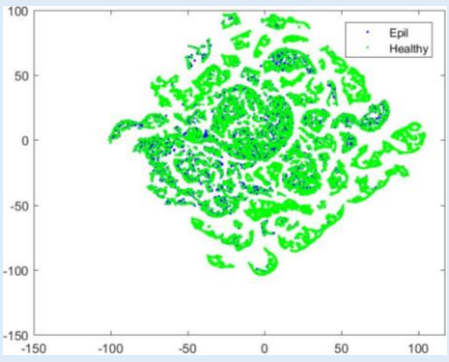
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 10: results learning rate equal to 0.0032 and decay of learning rate equal to 0.75

10. Test with learning rate equal to 0.0032 and decay of learning rate equal to 0.875

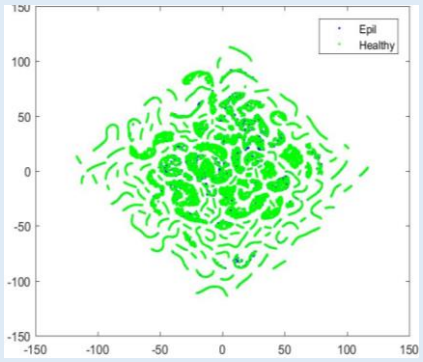
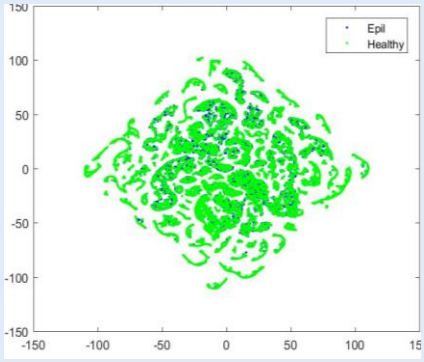
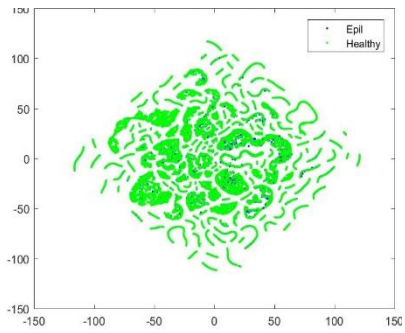
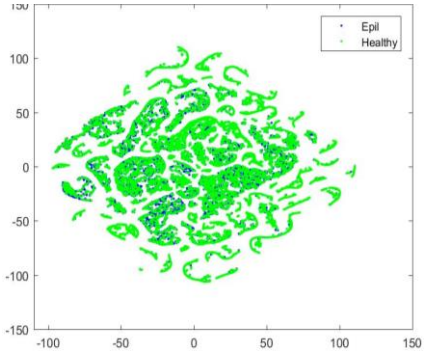
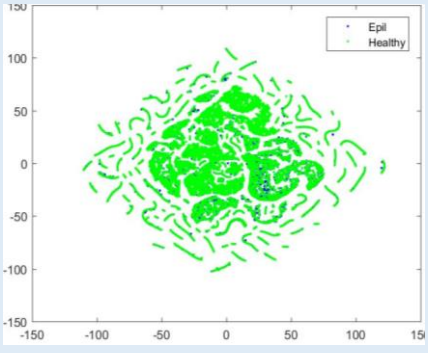
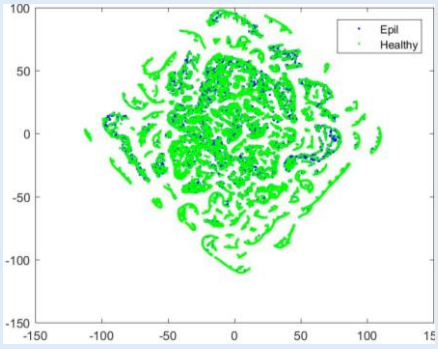
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 11: results learning rate equal to 0.0032 and decay of learning rate equal to 0.875

11. Test with learning rate equal to 0.0032 and decay of learning rate equal to 1

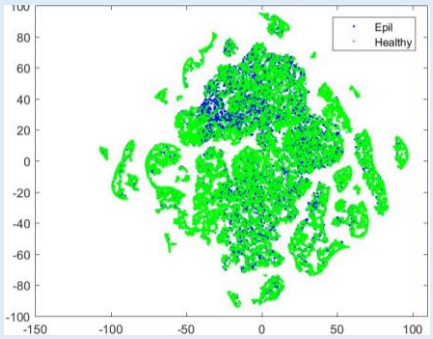
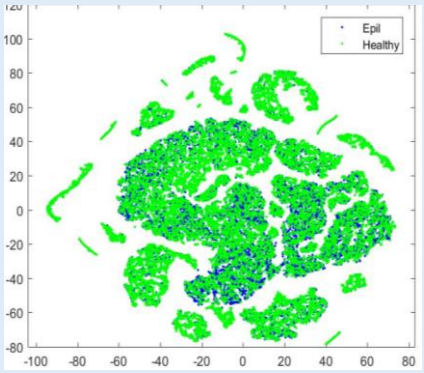
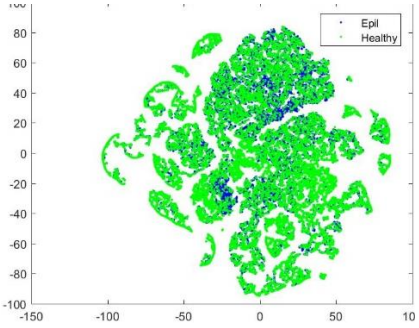
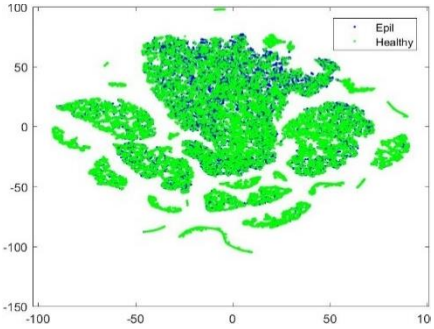
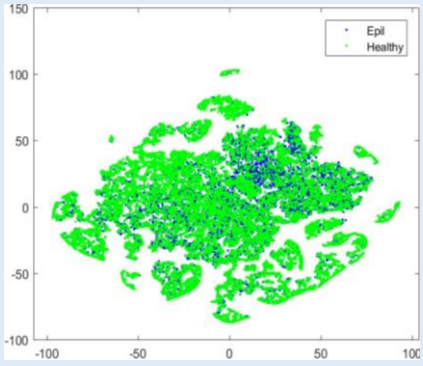
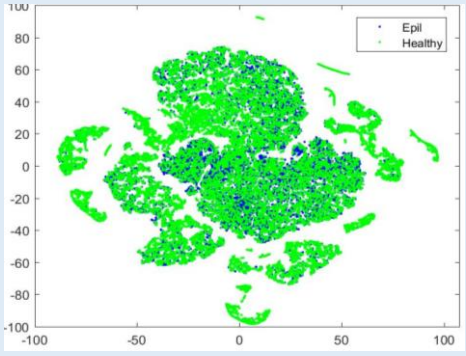
DISTANCE	GLOBAL FEATURES	LOCAL FEATURES
CITYBLOCK		
EUCLIDEAN		
MAHALANOBIS		

Table 12: results learning rate equal to 0.0032 and decay of learning rate equal to 1

Tables 2 to 11 show the inefficiency of Deep Infomax in this particular application, due to the missing of an isolated blue region characterized by the only epileptic epochs. For this reason, the comparison among different epochs from different type of signals has not been done.

As described in chapter 2, tests have also been done with:

- Learning rate = 0.1;
- Learning rate = 3.16;
- Learning rate = 100.

However, using those parameters, net converged too deeply, giving as output matrixes full of NaN.

4. Conclusions and further development

As deeply discussed before, networks proposed in this thesis are not yet usable for the diagnosis of epilepsy.
Possible reasons and solutions are now listed, giving some starting points for further works.

1. **Framework used**
   During this work of thesis framework of 1s and 2s have been used. The choice has been taken studying a series of works available on literature [8], however there are some studies in which different frameworks lengths have been used [9]. So a possible further work could consist on using proposed nets with different framework.

2. **Pre-processing of data.**
   The real nature of data is not clear. During this thesis, it has been decided to do not pre-process signals, for two main reasons:

   - There is not a document that affirms if signals have been denoised before the uploading on the database.
   - Denoising signals some epileptic features could be eliminated or, vice-versa, healthy features could be attenuated and considered as pathologic.

   A possible solution could be to contact Temple University Hospital of Philadelphia to discover if signals have been denoised or not.

   If no denoising method has been applied, some techniques are proposed by literature to do not alter the real nature of data.

   As proposed by [7], Discrete Wavelet Function (DWT) is a good alternative to do not eliminate important information from data.

   Obviously, more than one denoising technique could be applied. Another noise source could be the movement of the patients or the eye-artefact. In that case, having an EMG signal and a video about patient during the EEG recording, those type of noises could be removed, and the quality of both Deep InfoMax and Convolutional autoencoder for epilepsy clustering could be improved.

3. **Percentage of epileptic epochs too low.**
   Even if the original idea was training the net with only healthy epochs to discriminate the ones with epileptic signs, in the third test effectuated using convolutional autoencoder, the net has been trained with both healthy and epileptic signals. The problem is that epileptic signals present most of epochs as healthy, because seizures attack the brain just for few moments and sometimes our signals have duration of hours.

   Thus, another recommended test is to train the net with an equal percentage of healthy epochs and epochs that contain seizures of epileptic signs effectively.

4. **Effective epilepsy seizure in tracks.**
   Even if every signal offered by Temple University Hospital is commented by a text file offered by neurologists, it is not possible to read every single text file of 8GB of data and to extract from epileptic folder signals that contain epileptic seizures.
   In fact, the folder contains signals from epileptic subjects, but:
   - It is not sure that there is a seizure in each signal;
   - Maybe the signal contains a form of periodicity of the wave and for that reason the subject is considered as epileptic. However, in that case everything is based on expert's experience;
   - A signal could be belonging to an epileptic subject, but in the specific track there is not either seizure or attenuation of the brain activity (creating a form of periodicity of the wave).

   A valid solution could be to use tracks containing seizures in tests proposed.

5. **Is Copula transformation the best choice?**
   As discussed in chapter 2, signals have been treated with Copula transformation and, as seen, it does not alter the nature of the signals.

   However, changing the real amplitude of samples in a range [0,1] and averaging "unexpected" peaks, epileptic signs could be mitigated.

   Further studies could investigate if other normalization methods are a better solution.

6. **Are T3 and T4 channels enough?**
   During this work of thesis only temporal channels have been used because they reflect most of epileptic form.
   However, to increase the accuracy of work, a valid propose is either to select only signals in which epilepsy is evident in temporal channels or, on the bases of medical diagnosis, to give as input of the net the correct channel.

7. **Excessive reduction of dimension.**
   The last problem observed during this work is that the dimension is reduced too much, and this aspect emerges in two different moments:
   - Passing from 256 sample of each epochs to 16;
   - Passing from 16 sample of each hidden layer to 2 applying t-sne technique.

   It would not be surprising discovering that the quantity of information in those two steps is too much. Other tests could be effectuated either changing the dimension of hidden layers, or showing results in a space 3D.

Appendices

Appendix A

A.1 Signals extraction Matlab code

```
clear all
close all
clc

fs=256;
dur_ep=1; %epochs length 1
n_samp=fs*dur_ep; %number of samples for each epoch
overlap=n_samp/2; %overlap 50%
time_v=[1:n_samp].';

%% File loading and signals extraction

v=[];
indici=[]%number of epochs for each signal
myFiles = dir(fullfile('**/*.edf*'));
for idx = 1:numel(myFiles)
    currentFile = fullfile(myFiles(idx).folder,myFiles(idx).name);
    [hdr, record] = edfread(currentFile);
    current_file=record(13,:); %Extraction of 13 channel
    durata=length(current_file);
    xx=getRank(current_file);%, Copula transformation
    n_ep=fix(length(xx)/n_samp);
    offsets=overlap*[0:(n_ep*2)-2];
    v=horzcat(v,xx(time_v+offsets));
    indice=(durata/n_samp)*2; %true if overlap is 50%
    indici=horzcat(indici,indice);
end

save('HealthyMatrix','v');
save('index','indici');

```

A.2 Convolutional autoencoder Matlab code

```
%% Training
load TrainingSet
TrainSet= trainingSet;
[n_epoch,epoch_len]=size(TrainSet');
dataSet=reshape(TrainSet,[1,epoch_len,1,n_epoch]);
epochLayer = imageInputLayer([1,epoch_len,1]);

encodingLayers = [convolution2dLayer([1,5],32,'Padding','same'),...
    reluLayer,maxPooling2dLayer([1,2],'Padding','same','Stride',2),...
    convolution2dLayer([1,5],32,'Padding','same'), reluLayer,...
    maxPooling2dLayer([1,2],'Padding','same','Stride',2),...
    convolution2dLayer([1,5],64,'Padding','same'), reluLayer,...
    maxPooling2dLayer([1,2],'Padding','same','Stride',2),...
    convolution2dLayer([1,5],128,'Padding','same'), reluLayer, ...
    maxPooling2dLayer([1,2],'Padding','same','Stride',2)];
b=16;
c=128;
personal1=personalLayer3(b,c);
f1=fullyConnectedLayer(16);
f2=fullyConnectedLayer(b*c);
personal2=personalLayer4(b,c);

decodingLayers = [createUpsampleTransponseConvLayer([1,2],128),reluLayer,..
    createUpsampleTransponseConvLayer([1,2],64),reluLayer,...
    createUpsampleTransponseConvLayer([1,2],32),reluLayer,...
    createUpsampleTransponseConvLayer([1,2],32),reluLayer,...
convolution2dLayer([1,5],1,'Padding','same'), clippedReluLayer(1.0),...
regressionLayer];

layers = [epochLayer,encodingLayers,personal1,f1,f2,personal2,...
    decodingLayers];

options = trainingOptions('adam', ...
    'MaxEpochs',1,...
    'InitialLearnRate',1e-4, ...
    'Verbose',false, ...
    'Plots','training-progress');

net = trainNetwork(dataSet,dataSet,layers,options);

save('healthynet','net')

%% Testing

[n_epoch,epoch_len]=size(TestSet');
TestSet=reshape(TestSet,[1,epoch_len,1,n_epoch]);
YPred = predict(net,TestSet);
save('YPredSani','YPred');
```

Bibliography

[1] Robert S. Fisher ,Walter van Emde Boas , Warren Blume, Christian Elger, Pierre Genton, Phillip Lee, Jerome Engel Jr. **Epileptic Seizures and Epilepsy: Definitions Proposed by the International League Against Epilepsy (ILAE) and the International Bureau for Epilepsy (IBE)**

[2] Epilepsy Center, Department of Neurology, University of Munich, Marchioninistrasse 15, 81377 Munich, Germany. **The role of EEG in epilepsy: A critical review**

[3] Yousef Rezaei Tabar and Ugur Halici. **A novel deep learning approach for classification of EEG motor imagery signals**

[4] Yifei Zhang. **A Better Autoencoder for Image: Convolutional Autoencoder**

[5] Volodymyr Turchenko, Eric Chalmers, Artur Luczak. **A Deep Convolutional Auto-Encoder with Pooling - Unpooling ayers in Caffe**

.[6] Alex Fedorov, R Devon Hjelm, Anees Abrol, Zening Fu, Yuhui Du, Sergey Plis, Vince D. Calhoun. **Prediction of Progression to Alzheimer's disease with Deep InfoMax**

.[7] Md. Mamun, Mahmoud Al-Kadi, Mohd. Marufuzzaman. **Effectiveness of Wavelet Denoising on Electroencephalogram Signals**

[8] Yash Paul. **Various epileptic seizure detection techniques using biomedical signals: a review.**

[9] Turky N. Alotaiby, Saleh A. Alshebeili, Fathi E. Abd El-Samie, Abdulmajeed labdulrazak, Eman Alkhnaian. **Channel selection and seizure detection using a statistical approach.**

Ringraziamenti

Vorrei ringraziare prima di tutti le Professoresse Gabriella Olmo e Monica Visintin, che hanno mostrato fiducia nei miei confronti prestandosi come relatrici di questa tesi, sia per la pazienza che hanno avuto nei miei confronti che per l'aiuto indispensabile ai fini della stesura di questo elaborato.

Non posso non rivolgere un mio pensiero anche al Dottor Giulio Franzese, che quotidianamente ha supervisionato il mio lavoro insegnandomi tanto.

Vorrei ora dire un grazie speciale alle persone più importanti della mia vita, i miei genitori, senza i quali probabilmente non sarei la persona che sono ora. Grazie a mio padre per avermi insegnato a non mollare mai e per avermi supportata in ogni mia scelta con pazienza e dolcezza; grazie a mia madre per avermi regalato consigli saggi e per avermi spronata in ogni momento.

Ovviamente un grazie speciale va a mio fratello Gianluca, il quale mi ha sempre stimolata a dare di più e mi ha sempre ricordato chi sono.

Grazie a Federica, ormai parte della famiglia. Non posso che dirle grazie per essermi sempre stata accanto, nei momenti belli e soprattutto in quelli brutti. So che non ci perderemo a prescindere da tutto e che sarà sempre un riferimento importante nella mia vita.

Grazie a Simone e Vincenzo, per aver reso la mia adolescenza meravigliosa e per farmi sentire sempre "a casa" anche a tanti chilometri di distanza. Probabilmente senza di loro avrei un fegato ancora sano, ma niente sarebbe stato così bello.

Un grazie lo meritano le mie amiche da sempre: Alessia, Alessia e Chiara. Grazie per aver condiviso così tanto con me.

Un ringraziamento speciale lo meritano le coinquiline che ho avuto nel corso degli anni: Serena, Eleonora e Franca. Ognuna di loro mi ha aiutata a superare le piccole difficoltà quotidiane con dolcezza.

Passerei ora a ringraziare quella che ormai è una seconda famiglia, ovvero tutti i miei amici conosciuti a Torino. Grazie a Mattia e Gera, con i quali ho condiviso ogni giorno di questi anni. Probabilmente questo traguardo non sarebbe così bello se non l'avessi condiviso con voi. Grazie a Carlo, Valerio e Isabella, che sono stati salvezza appena arrivata in una nuova città e che non mi hanno mai fatta sentire sola.

Grazie a Marta, Alice e Clelia, con le quali anche un pomeriggio di studio diventava divertente. Grazie per avermi fatta ridere e per avermi regalato abbracci sinceri quando ne avevo bisogno. L'unico rimpianto è non averle conosciute prima.

Un ringraziamento sincero lo meritano Annalisa e Martina, conosciute quasi per caso, che da semplici compagne di gruppo son diventate amiche e compagne di avventure.

Infine, vorrei ringraziare tutti gli amici di Freccitalia, ognuno dei quali mi ha lasciata qualcosa e probabilmente questi anni non sarebbero stati così belli senza di loro.