Politecnico di Torino

Ingegneria Matematica

Laurea Magistrale

Network Embedding for brain connectivity

Supervisors:
GIACOMO COMO
SOPHIE ACHARD
MICHEL DOJAT

Presented by:
LUCREZIA CARBONI

October 2020

Abstract

In various contexts, network embedding techniques have been developed for performing analysis on a single graph. This approach has been proven to work for different applications, such as node classification or link prediction. A good network embedding algorithm is capable to capture only the relevant features of the graph and to reproduce them in a low-dimensional Euclidean space.
In the context of neuroscience, networks are currently used for representing the system of connections in the brain with the purpose of determining the characteristics of a pathological brain. However, discriminating a healthy human brain connectivity network from a clinical one using common network descriptors could be misleading. In fact, a difference in the currently used graph measures could not be detected or could be insufficient for the discrimination. For this reason, we investigate network embedding and extend its fields of application to human connectivity network. Moreover, we use the embedding for performing graphs' comparison. Finally, we propose a definition of the representative network of a set of graphs. This representative captures the properties which are in common in the group. In this way, we confirm the existence of a healthy signature, namely a brain structure which is shared by all healthy individuals.

Acknowlegments

Acknowlegments

I need to thank you lot of non-Italian speakers. First of all, I want to thank Sophie and Michel for their support in my internship. In Italy, I never feel my study and my hard work has been appreciated as it has been in Grenoble. Your patient, availability and way of challenging me has been fundamental and I am looking forward to start my PhD under your supervision.

A big thank goes to all my international friends in Grenoble who made this last year an important step in my growing up, opening my mind and sharing their life and experience with me.

Ringraziamenti

È per me doveroso sottolineare la mia gratitudine per quest'ultimo anno a Grenoble che è stato per me fonte di opportunità e di crescita. Per questo, devo dire grazie alla mia famiglia che mi ha visto partire e nonostante la distanza, non mi ha mai fatto mancare l'affetto e il supporto. Grazie a Carlotta e Diletta per i nostri continui incoraggiamenti, per il cercarci quando la casa sembra vuota, per l'affetto e la complicità che ci unisce nonostante le vite in paesi diversi.

Grazie a Fabio che è stato prima un compagno di studi e progetti premuroso e tenace, e che ora lo è anche nella vita.

Vorrei ringraziare tutti gli amici che ho incontrato in questo percorso, quelli di sempre che non si fanno spaventare dai chilometri, quelli che hanno subito reso Torino la mia casa, quelli della smartroom e del progetto, grazie ad Andrea per le ore passate insieme a studiare leggendo novelle e a Silvia e alla sua dose di follia

e pazienza. Ho più volte sentito dire che solo le vere amicizie possono sopportare la lontananza. Penso sia vero.
Grazie a chi continua a farlo.
Ringrazio, infine, il mio relatore per aver accettato di seguirmi in questo progetto di tesi, nonostante le difficoltà del periodo.

Contents

Chapter 1

Introduction

The network model and its natural mathematical representation into graphs find application in many different contexts. Network analysis requires the definition of measure over the constituent elements of the network and the focus on a single feature each time. In that context, tools to represent a graph and to capture at the same time the relevant characteristics of a network have started to be developed. Among all, network embedding technique has shown to be powerful in different types of application.

The topic of this work is to exploit embedding capability in capturing the relevant features of brain connectivity networks. In the majority of cases, the embedding techniques have been developed as a dimensional reduction tool which could be used for analysis in a single huge network, for instance in the task of node classification or link prediction. In the study of brain connectivity networks, we would like to find which are the discriminating properties in the identification of the class of membership of each considered graph. However, it is evident that, because of the subject variability, no network could be exactly equal to another one, even if they belong to the same group. Namely, even under the same healthy conditions, networks of different subjects would be different. From an ideal point of view, we would like to capture in the embedding space this variability, highlighting which are the specific properties of the class. The final goal would be to generate a representative graph (*an average network*) for each of the considered class. Thanks

to this average network, we would like to understand where the dissimilarities between classes are located and how they are affecting the connectivity. For that reason, we do not limit our attention to the embedding computation, but also to the comparison among graphs.

The following report is organized as follows. In the first chapter, a short recall of graph theory necessary to better understand the proposed problem is given together with a general introduction about brain connectivity. In the following chapter, the state of the art in graph embedding is presented, focusing on a special algorithm and in a way to compute graph similarity through embedding method. Next, an original method to tune the embedding parameters is proposed, together with an innovative scheme for the definition of a representative class graph. The mean graph should well represents the variability among individuals of the same group, enlightening the features which make the class different from the other. A presentation on the datasets and on the experiment framework which have been used for the evaluation of our contributions are presented in the following chapter. Finally, the sixth chapter is dedicated to critical discussion of the observed results. As a conclusion, we propose some open problems which could be explored in the future.

Chapter 2

Brain Connectivity Graphs

2.1 Recall of Graph Theory

2.1.1 Notation and nomenclature

Definition 2.1. An unweighted graph $\mathcal{G}$ is a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
Here, $\mathcal{V}$ is a set of vertices or nodes. Each node represents an unit of the network we are considering.
$\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges among the nodes. In this work we will consider graph where each link $e \in \mathcal{E}$ is a pair of nodes $e = \{u, v\}$ $u, v \in \mathcal{V}$ which corresponds to an existing connection among the vertices u, v.
Since the pairs of nodes are not ordered, the graph we consider, is said to be undirected.

Definition 2.2. The adjacency matrix A of an unweighted graph $\mathcal{G}$, is a binary matrix defined as follows:

$$A = (a_{uv})_{u,\, v \in \mathcal{V}} = \begin{cases} 1 & \text{if } \{u, v\} \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}$$

For an undirected graph A is symmetric.

Definition 2.3. A **walk** over a graph is a sequence of nodes where each pair of consequential nodes is connected by an edge.

A **path** is a walk in which each node except for the first and the last one, is present in the sequence only once.
In the case of unweighted graph, the **length of a walk** is defined as the number of edges in the walk itself.

2.1.2 Proximity functions between a pair of nodes

In this section, we introduce the concept of proximity between a pair of vertices in the graph. These definitions could be found in [15, 6, 5]. Whereas the two last definitions are based on [4].

- **I order proximity**

  Definition 2.4. The first order proximity in a graph is the local pairwise proximity between two vertices. For each pair of vertices the presence of an edge, A_{uv} indicates the first-order proximity between u and v. If no edge is observed between u and v, their first-order proximity is 0.
  For each vertex u we denote $p_u^{(I)} = (A_{u1}, \ldots, A_{u|\mathcal{V}|})$ the vector of its first-order proximity.

- **II order proximity**

  Definition 2.5. The second-order proximity between a pair of vertices $\{u, v\}$ in a graph is the similarity between their neighborhood structures. Mathematically, let $p_u^{(I)}$ denotes the first-order proximity of u with all the other vertices, then the second-order proximity between u and v is determined by the similarity between $p_u^{(I)}$ and $p_v^{(I)}$. If no vertex is linked with both u and v, the second-order proximity between u and v is 0. For each vertex u we denote $p_u^{(II)}$ the vector of its second-order proximity. The i^{th} entry corresponds to the scalar product between $p_u^{(I)}$ and $p_i^{(I)}$.

  Example 2.1. Referring to Figure 2.1:

$$
\begin{array}{rcl}
 & & \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
p_1^{(I)} & = & \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \\
p_1^{(I)} & = & p_2^{(I)} = p_3^{(I)} = p_4^{(I)} \\
p_5^{(I)} & = & \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix} \\
p_6^{(I)} & = & \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \\
p_7^{(I)} & = & \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}
\end{array}
$$

$$
\begin{array}{rcl}
 & & \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
p_5^{(II)} & = & \begin{pmatrix} 0 & 0 & 0 & 0 & 4 & 4 & 0 \end{pmatrix} \\
p_6^{(II)} & = & \begin{pmatrix} 0 & 0 & 0 & 0 & 4 & 4 & 0 \end{pmatrix}
\end{array}
$$

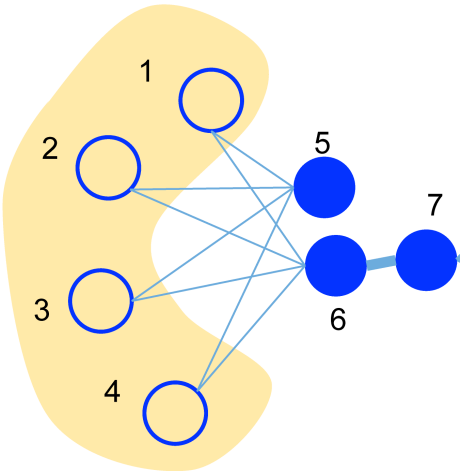
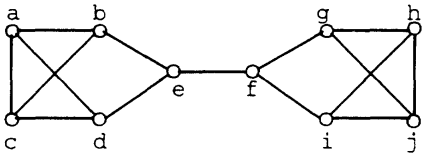
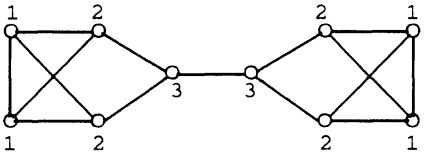


Figure 2.1: In the figure node 6 and 7 are first-order proximal since they are connected one to another. Nodes 6 and 5, instead, are second-order-proximal since they share links with nodes 1, 2, 3 and 4. Figure from [6]



(a) Labels identify nodes



(b) Labels identify positions

Figure 2.2: Top: an example of graph with nodes' labels. Bottom: labels indicate the position which are structurally isomorphic $a \equiv c \equiv h \equiv j$, $b \equiv d \equiv g \equiv i$, $e \equiv f$. Figure from [4]

- **Structural equivalence of nodes**

  The concept of structural equivalence of nodes is related to the way the node is connected with the other nodes in the graph. In the context of social networks, the structural equivalence refers to the *role* the node-associated agent is playing in the networks, especially looking at the type of links with the other equivalence classes. In the case of brain connectivity networks, knowing the role that each node is playing, is fundamental in a neurological interpretation of brain connections. Discursively, two vertices of the same network are said to be structurally equivalent if they have similar connections with the rest of the graph. To give a mathematical definition, we recall the definition of an automorphism.

  Definition 2.6. An automorphism of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a bijection π between $\mathcal{G}$ and itself such that:

$$\forall u, v \in \mathcal{V}, \{u, v\} \in \mathcal{E} \iff \{\pi(u), \pi(v)\} \in \mathcal{E}$$

That means that if two vertices are connected in $\mathcal{G}$ their images through the map π must be connected too. An obvious automorphism is the identity map, however not-obvious automorphism could exist. It is worth to emphasize that the nodes' labels are not taken into account defining an automorphism and therefore, a graph and its automorphism image without labels are indistinguishable.

Definition 2.7. Two nodes $v, w \in \mathcal{V}$ are structurally isomorphic or automorphically equivalent $v \equiv w$ if there exists an automorphism $\pi : \mathcal{G} \to \mathcal{G}$ such that $\pi(v) = w$

In our case, two regions of the brain are said to be structurally equivalent if their representative nodes in the network are structurally isomorphic. An example of automorphically equivalent nodes is shown in Figure 2.2

Proposition 2.1. If two nodes are structurally isomorphic, so are their adjacent nodes.

$$v \equiv w \implies \forall u \in \mathcal{V} \text{ s.t.} \{v, u\} \in \mathcal{E} \; \exists u' \in \mathcal{V} \text{ s.t. } \{w, u'\} \in \mathcal{E}$$

for which it values:

$$u \equiv u'$$

Proof. By hypothesis we have $v \equiv w$, so it exists an automorphism π such that $w = \pi(v)$. By definition of automorphism:

$$\{v, u\} \in \mathcal{E} \iff \{\pi(v), \pi(u)\} = \{w, \pi(u)\} \in \mathcal{E}$$

Let u' equals to $\pi(u)$ then we have $\{w, u'\} \in \mathcal{E}$ for which it values $u \equiv u'$ via π. □

- **Community similarity of nodes**

  As for the structural equivalence, in the case of community similarity of nodes the focus is on the way each node is linked with all the others in the graph. However, there is not an unique way to define community similarity.

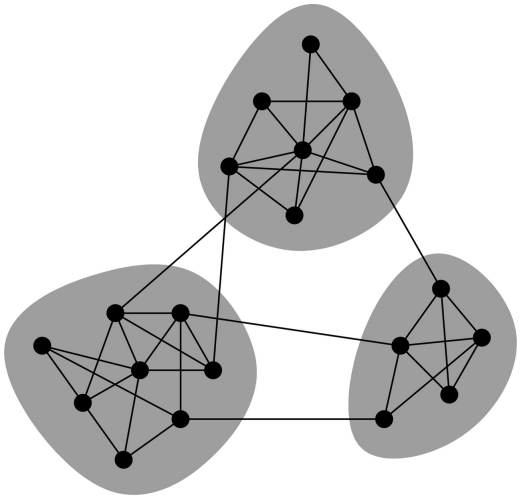


Figure 2.3: Example of graph with strong community structure. The nodes could be divided into 3 groups of nodes strongly interconnected. Figure from [20]

> In fact, the way of define a community of nodes of a network hardly depends on the application. For our purpose, it is sufficient to know that the community structure of a graph is given by the presence of groups of nodes which are strongly connected one to another and, at the same time, present few links with nodes of different group.

2.2 Brain Connectivity

Brain connectivity has multiple meanings, depending on the specific level of brain which is taken into account. In this work, we consider functional connectivity in human brains.

Functional connectivity refers to the way information spreads in the brain. In fact, the brain activity is characterized by neurons and group of neurons which communicate one to another. Using the terms *functional connectivity* we want to refers to this system of interactions. We will use connectivity and brain connectivity as synonyms as-well. Functional neuroimaging methods, such as functional MRI (fMRI), permit to collect data and to analyze brain in this particular context.

In this work, the fMRI data are acquired in a state of resting: the brain imaging

aquisition is performed when the subjects are daydreaming and are not supposed to execute any particular tasks. It is natural, to represent the connections among the brain regions as a graph, the nodes are the specific regions we consider and the edges indicate the presence of a connection between two nodes.
The networks which are considered in the presented report, have been previously built, according to the framework explained in chapter 5.

Chapter 3

State of the art in network embedding

Network embedding can be seen as a dimensionality reduction tool which maps a network into a vector space. The embedding could be performed at different levels, that means it could take as input different parcels of the graph. It could be defined over the nodes, over the edges, over some substructures of the graph or even over the entire graph. In this last case, each graph would be mapped into a single point in the embedded space. We focus on nodal embedding, namely a mapping function which takes as input a single vertex of the graph. In this particular case, the embedding could be defined as follows:

Definition 3.1. For a given network $\mathcal{G}$, a network embedding is a mapping function $\Phi : \mathcal{V} \to \mathbb{R}^{|\mathcal{V}| \times d}$, where $d \ll |\mathcal{V}|$ is the dimension of the embedding space. This mapping Φ denotes the embedding of each node $v \in \mathcal{V}$.
Z denotes the matrix having as rows the embedding vectors of each node.

Embedding algorithms could be classified according to the characteristics they preserve in the embedding space, such as first-order preserving etc. More details about this classification could be found in [6].
Following [5], we briefly present the main steps which could be found in the majority of the embedding algorithms. According to this model, a general algorithm for graph embedding $GEM\text{-}D[h(\cdot), g(\cdot), d(\cdot, \cdot)]$ is characterized by:

- a node proximity function

$$h(\cdot) : \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|} \to \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$$

  which is a function of the adjacency matrix

- a warping function

$$g(\cdot) : \mathbb{R} \to \mathbb{R}$$

  which determines how to measure the distance in the embedding space. The distance between two vertices in the embedding space is given in the entries of the embedded proximity matrix $g(ZZ^T)$. g is applied element-wise and is monotonically increasing.

- a loss function

$$d(\cdot, \cdot) : \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|} \times \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|} \to \mathbb{R}$$

  which is required to the formulation of the embedding problem.

The general formulation of the embedding as an optimization problem is represented by:

$$Z = \arg \min_{Z \in \mathbb{R}^{|\mathcal{V}|\times \mathrm{d}}} d(h(A), g(ZZ^T)) \tag{3.1}$$

$d(\cdot, \cdot)$ is measuring the differences in the nodes proximities between vertices in the graph and their embedding vectors.

3.1 Node2vec

Among the other methods, we focus on a particular embedding algorithm which has been proven of being able of capture the structural equivalence of nodes ([9]). Vectors which correspond to nodes having similar roles are clustered together using k-means algorithm in the embedding space. This algorithm has been proposed in [9]. Moreover, it has recently been used in the context of brain connectivity graph, with good results in the characterization of the correspondence relations among brain regions ([14]). node2vec is based on the Skip-gram architecture which is used to learn a features representation of words based on their context [12]. In the

same way, in the node2vec model, the concept of *context* of a word is translated into the one of *neighborhood* of a node. Precisely, node2vec defines a flexible notion of a node's neighborhood, depending on the characteristics we are interested in. Especially, two opposite characteristics are given by structural or community similarity of nodes.
The neighborhood is not based on a single similarity, but it can be defined through out the interpolation of two searching strategy.
There are two extreme sampling strategies: the breadth-first sampling (BFS) and the depth-first sampling (DFS). For each vertex, the algorithm computes a neighborhood set of a given number of nodes. The BFS strategy assigns to the neighborhood of a node, nodes which are immediate neighbors of the source. Whereas, the DFS neighborhood is composed by nodes which are sequentially sampled at increasing distances from the node itself (Figure 3.1). As said before, node2vec allows to smoothly interpolate between BFS and DFS.

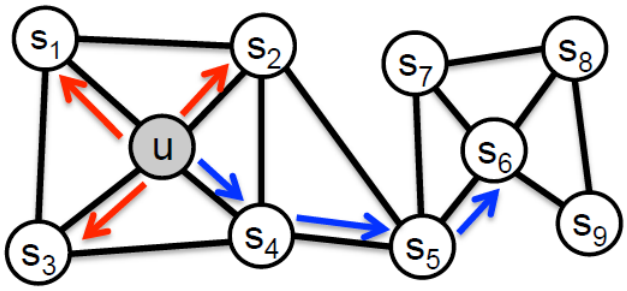


Figure 3.1: Generation of a neighborhood set of given cardinality equals to 3 for node u, according to the two different sampling strategies. In red the BFS-neighborhood (s_1, s_2, s_3). In blue the DFS-neighborhood (s_4, s_5, s_6). Figure from [9]

Definition 3.2. Given a neighborhood sampling strategy S over the nodes, the neighborhood of size k of a node $v \in \mathcal{V}$ is the set $N_S(u)$ composed by a sample of k nodes extracted following the strategy S.

In terms of GEM-D model ([5]), node2vec is composed by the following three building blocks:

Proximity function As a proximity function node2vec uses an approximation obtained through random walk simulation. The proximity depends on the following *memory matrix*. More details could be found in the appendix.

Over the networks nodes, we define a memory matrix as follows:

$$M_{uv} = \begin{cases} \frac{1}{p} & u = v \\ 1 & A_{uv} = 1 \ u \text{ and } v \text{ are adjacent} \\ \frac{1}{q} & l_{uv} = 2, \ \ u \text{ and } v \text{ have minimum path length equals to } 2 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

p and q are the parameters used to interpolate among the two searching strategies. A small p implies breadth-first sampling, while a small q corresponds to depth-first sampling. The proximity matrix $\mathbb{III}^{(L,p,q)}$ is depending on an additional parameter: the length of the generated random walk L. The entry $(\mathbb{III}^{(L,p,q)})_{uv}$ is the expected number of times of visiting node v, starting from u and walks L steps sequentially. $\mathbb{III}^{(L,p,q)}$ is a function of M and A, the adjacency matrix. (See the appendix for the complete definition)

Warping function The warping function used in node2vec, is the exponential

$$g(x) = \exp(x)$$

Loss function node2vec solves the optimization problems of maximizing the log-probability of observing a network neighborhood $N_{pq}(v)$ for a node v conditioned on its feature representation $\Phi(v)$:

$$\max_{\Phi} \sum_{v \in \mathcal{V}} \log P(N_{pq}(v) | \Phi(v)) \tag{3.3}$$

This optimization problem is equivalent to the minimization of the KL-divergence between the two probability distributions obtained after the normalization of $\mathbb{III}$ and $g(ZZ^T)$.

In synthesis, the node2vec embedding function is determined by the following parameters:

d dimension of the embedding space

N number of random walks per node used to estimate the proximity matrix

L random walk length

k size of the neighborhood set for each node

p return parameter, control of the sampling strategy, small values $\rightarrow$ BFS

q in-out parameter, control of the sampling strategy, small values $\rightarrow$ DFS

3.2 Graphs similarity through embedding methods

In our project, we are interested in comparing networks which belong to different classes. For instance, we would like to be able to discriminate a network which belongs to a healthy subject from a pathological one. This comparison could be directly performed in the network space, for example, using one of the measures proposed in [21]. However, our purpose is to explore the embedding potential in the analysis of brain connectivity. For this reasons, we were searching not only an embedding procedure, but also a way to compare graphs in the embedding space. With this goal, we found the work [13] newsworthy. In particular, inspired by the pyramid match kernel used in computer vision, the authors have designed an equivalent version for the graph, named pyramid match graph kernel. Two versions of the kernel are provided, one regarding graph having unlabeled nodes and a second version for the labeled ones. The main idea of the algorithm is to count the matching of vectors in the embedding space at different resolution levels. In their setting, they proposed an embedding based on the eigenvectors of the adjacency matrix and their computed node-representing vectors lay in a d-dimensional hypercube. Given a number of level, a grid of cells having increasing size in one dimension for each level is computed. Two vectors are said to match if they belong to the same cell. Each match is weighted according to the dimension of the region

in which it appears. Counting the matching within two embedded graphs, we are able to compute their proximity. In the case of labeled nodes, the matching is valid only if it appears between two node vectors with the same label. A toy example is shown in Figure 3.2.

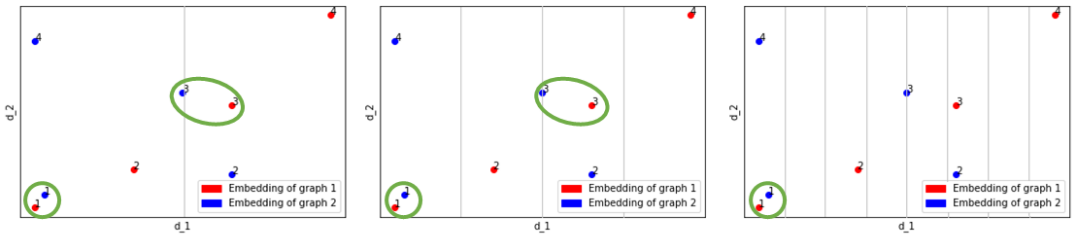


Figure 3.2: A toy example of the pyramid match graph kernel for labeled graph. In two dimensions, we increase the resolution on dimension d_1 from left to right. A matching in the resolution grid is indicated by a green circle. Matching which are present at all resolution levels, like the ones for node having label 1, receive a high weight in the proximity between the two graphs.

Chapter 4

Contributions

The purpose of our work is to determine if a network embedding technique could be used to capture variability among classes and to be used to spot where the differences among healthy and pathological brain connectivity are located. Two points have to be taken into account in this purpose.

First, we assume that the relevant differences in the connectivity graphs are especially given by the changing in the role of some nodes in terms of connectivity structure. Previous works [2] have highlighted, that in connectivity network there are some nodes which are playing a key-role in the spreading of information for consciousness. For example, comparing data of comatose patients with healthy controls, it has been proven that same regions in the brain play different roles in the connection. The nodes which correspond to these regions, are called *hubs*. To be considered a hub, a node has to behave in an unusual way. For example, we could interpret as a hub a node with an unusually high degree or centrality. Thus, in our purpose an ideal embedding would represent a hub faraway to a non-hub node.

Secondly, a comparison through the embedding space would theoretically be able to well-classify data. However, it is not enough to classify. A doctor is usually able to discriminate a brain disease without computing the brain connectivity network. What could be interested in our work is to understand which are the relevant differences for the classification, with the final goal of determine an average graph. If

this average network is representative enough, we would like to classify new data just looking at the distance with the average graph.

4.1 Automatic parameters tuning method

As it has been explained in Chapter 3, node2vec algorithm depends on a substantial number of parameters. We propose a framework to tune these parameters. Every combination of parameters determines a different nodal proximity function in the definition of the embedding algorithm. Thus, it would preserve different features of the graphs in the embedding space. Moreover, it is important to notice that setting the parameters' values referring to a single graph's embedding would result in the necessity of using different parameters for each single graph. However, in this way we lose the possibility of a fair comparison of graphs in the embedding space. Therefore, since our purpose is capturing the class features and not only those of a single graphs, we propose to set the parameters according to the obtained results in the graphs comparison. The obtained embedding, thereby, would eventually be able to capture the shared characteristic of a group of graphs. To better explain, we would like to perform a graph embedding through which the computed kernel proximity in the same class is maximized, while the proximity between different class is minimized.
Since all the graphs in our dataset are comparable from the point of view of number of nodes and edges, we intend tune parameters using a single dataset and apply the same computed values to perform the classification on the other datasets. In this way we expect to find parameters which could be re-used in similar brain connectivity dataset without going through a new parameter tuning process. In addition, this parameter tuning method is general and not related to the accuracy results in classification. However, in our framework it is related to the dataset we decide to use for evaluating the embedding performance.

4.2 Embedding performance evaluation

We define two measures to evaluate the embedding performance for a given set of parameters. With some adjustments, this framework could be applied in dataset having a different configuration of the data classes. We report the case where, each class is composed of a pair of graphs.

Let $\mathcal{D} = \{\mathcal{G}_i\}_{i \leq N}$ be the collection of graphs in the dataset, where each $\mathcal{G}$ has an associated one $\tilde{\mathcal{G}}$ and let be $\mathbf{G}$ the square symmetric matrix of dimension $|\mathcal{D}| \times |\mathcal{D}| = N \times N$, such that $\mathbf{G}_{(i,j)} = \mathrm{K}(\mathcal{G}_i, \mathcal{G}_j)$, where K is the pyramid graph matching kernel. The first step of our parameter tuning framework is to compute the embedding under a given set of parameters. Then, we calculate the kernel matrix over all the data. This matrix $\mathbf{G}$ depends on the configuration of parameters of the embedding. Afterwards, for each row of $\mathbf{G}$, we estimate the frequency distribution of the proximity values f. Hence, we are looking at the proximity's values between a fixed graph and all the other graphs in the collection $\mathcal{D}$.

Considering a graph $\mathcal{G}_i$, we refer to the proximity value between $\mathcal{G}_i$ and $\tilde{\mathcal{G}}_i$, $\mathrm{K}(\mathcal{G}_i, \tilde{\mathcal{G}}_i)$, as its **minimal proximity threshold**.

We say that the graphs having a proximity greater or equal to a fixed threshold are close to the considered one. Theoretically, we would like to find that, pairing each graph to its closest, the resulting sets of two elements are composed by the two graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$. However, we relax this constraint defining a subset of graphs - a **close-set** - which are similar to the considered one and we simply require that $\tilde{\mathcal{G}}$ belongs to the close-set of $\mathcal{G}$.

Definition 4.1. The **close-set** of threshold τ of graph $\mathcal{G}_i$ is

$$C_\tau(\mathcal{G}) = \{\mathcal{G}_j \in \mathcal{D} \text{ s.t. } K(\mathcal{G}_i, \mathcal{G}_j) \geq \tau\}$$

Definition 4.2. The graph $\mathcal{G}_i$ is **well-paired** or **correctly paired** with respect to the threshold value τ, if

$$\tilde{\mathcal{G}}_i \in C_\tau(\mathcal{G}_i) \iff \tau \leq \mathrm{K}(\mathcal{G}_i, \tilde{\mathcal{G}}_i)$$

We would like at the same time to maximize the number of well-paired graphs and their minimal proximity threshold. Since, the proximity distribution differs from a graph to another we want to maximize the minimal proximity threshold looking at its position in the considered distribution. We give, then, the following definitions.

Definition 4.3. Given a close-set $C_\tau(\mathcal{G}_i)$, its **relative dimension** is given by the ratio between its cardinality and $|\mathcal{D}|$:

$$\frac{|C_\tau(\mathcal{G}_i)|}{|\mathcal{D}|}$$

Osservation 1. The **relative dimension** of a close-set $C_\tau(\mathcal{G}_i)$ is equal to $(1 - F_i(\tau))$ where F is the cumulative distribution function of the proximity values for graph $\mathcal{G}_i$.

The relative dimension of the close-set of the minimal proximity threshold, represents the percentage of graphs in the collection which needs to be declared similar to the given graph to correctly pair it.

We evaluate the performance of an embedding looking at two measures.

Definition 4.4. Given a configuration of parameters π and the node2vec embedding Φ_π, its **average cumulative proximity frequency** μ is the average over all the data in the collections of $F_i(\mathrm{K}(\mathcal{G}_i, \tilde{\mathcal{G}}_i))$.

$$\mu = \frac{\sum_{i \leq N} F_i(\mathrm{K}(\mathcal{G}_i, \tilde{\mathcal{G}}_i))}{|\mathcal{D}|}$$

The average cumulative proximity frequency is inversely proportional to the relative dimension of the close-set of minimal proximity threshold in the collection. For a given graph $\mathcal{G}_i$, the greater $F_i(\mathrm{K}(\mathcal{G}_i, \tilde{\mathcal{G}}_i))$ is, the smaller the relative dimension of $C_{\mathrm{K}(\mathcal{G}_i, \tilde{\mathcal{G}}_i)}(\mathcal{G}_i)$ and so are their average.

After having determine the average cumulative proximity frequency of the embedding Φ_π on the collection $\mathcal{D}$, we can define the following well-paired counting with respect this average:

Definition 4.5. Let μ be the average cumulative proximity frequency. For each graph $\mathcal{G}_i$, we set the proximity threshold equal to $F_i^{-1}(\mu)$. The well-paired counting is the number of well-paired graphs in the collection using this value.

Using this framework, the close-sets are not symmetrical and the similarity is not a reciprocal relation. In fact, we are not determining any clusters of the graphs, we are just looking at a possible way to associate them one to another. Moreover, since each close-set is defined using the proximity distribution of the considered graph, it could happened that the graph does not belong to the close-set of some of its similar. However, we consider that for our application it is better to associate each graph to some others instead of setting the same proximity threshold for all the data and finding that some of the graphs could not be associated at all. In fact, an embedding graph could be very close or very far from all the others graphs in terms of absolute proximity value, nevertheless we would like to consider the case in which its higher proximity is with its corresponding.

4.3 Definition of a representative graph

A second goal of our study is to create a representative graph for each class. Previous works have defined a average graph in the network space, taking into account the average of the adjacency matrices of each class or selecting edges which are proven to be present in all the graphs of the class. We propose an innovative approach using a network embedding algorithm. Our proposal requires labeled graphs with labeled nodes. All the graphs have the same number of nodes $|\mathcal{V}|$.

Having at disposal a labeled set of graphs, we consider all the networks which belong to the same class. As we have explained before, any nodes embedding algorithm results into a vertex-associated vector in $\mathbb{R}^d$. Let $\mathcal{D} = \{\mathcal{G}_i\}_{i \leq N}$ be a collection of graphs embedding. For our application, we represent a graph as a bag-of-vectors $\mathcal{G}_i = \{v_{i1}, v_{i2}, \ldots, v_{i|\mathcal{V}|}\}$ where $v_{i1} = (v_{i1_1}, \cdots, v_{i1_d}) \in \mathbb{R}^d$ is the embedding vector of node 1, etc. Each node has a unique label. Let $C_1, C_2, \cdots, C_k$ be a partition of $\mathcal{D}$ where each set C_j corresponds to a class of graphs. For each

class we define a representative graph in the following way. Let's say class j is given by $C_j = \{\mathcal{G}_1, \cdots, \mathcal{G}_n\}$, then its representative graph $\bar{\mathcal{G}_j}$ is

$$\bar{\mathcal{G}_j} = \{\bar{v_1}, \bar{v_2} \cdots, \bar{v_{|\mathcal{V}|}}\} \quad \begin{cases} \bar{v_1} = \frac{1}{n}(\sum_{i=1}^{n} v_{i1_1}, \sum_{i=1}^{n} v_{i1_2}, \cdots, \sum_{i=1}^{n} v_{i1_d}) \\ \vdots \\ \bar{v_k} = \frac{1}{n}(\sum_{i=1}^{n} v_{ik_1}, \sum_{i=1}^{n} v_{ik_2}, \cdots, \sum_{i=1}^{n} v_{ik_d}) \\ \vdots \\ \bar{v_{i|\mathcal{V}|}} = \frac{1}{n}(\sum_{i=1}^{n} v_{i|\mathcal{V}|_1}, \sum_{i=1}^{n} v_{i|\mathcal{V}|_2}, \cdots, \sum_{i=1}^{n} v_{i|\mathcal{V}|_d}) \end{cases}$$

For each node, we consider the set of the embedding vectors obtained through the mapping of all graphs in the class. Then, we average components by components, determine the baricenter. The output of this process is a set of vectors, where each vector is associated to a given node in the graph. This output can be interpreted as a virtual embedding of a representative graph.

A reconstruction of this graph is also possible. Indeed, it is sufficient to invert the optimization problem which has been used to define the embedding map. Knowing the expected number of edges, it is possible to determine a proximity function over the node and to infer a possible adjacency matrix. However, the reconstruction framework is not part of this work.
This proposed method is inspired by the graph kernel which is used to measure the proximity among the graphs. In fact, we are implicitly assuming that the vector position in the embedding space should be related to the associated-node's label. Moreover, since the embedding is supposed to capture the structural equivalence of nodes, we conjecture that for individuals in the same class same brain region behaves in a similar way for the connectivity. Finally, the choice of average each node-embedding vector is justified by the fact that the similarity between two graphs is evaluated according to the vector matching in the embedding space.

Chapter 5

Material and method

5.1 Data

Brain functional networks definition

For this section we refer to [3],[2].

The definition of the functional network is achieved through out different phases. First, the acquired fMRI data are aggregated over regions which are determined according to the anatomical labeling proposed in [17]. Following this parcellation, the brain is partitioned in 45 regions of interest per hemisphere. Thus, in total there are 90 regions which are listed in table B.1. For each zone, a unique time series signal is determined: by averaging the fMRI time series over all voxels in each parcel, weighted by the proportion of gray matter in each voxel. The following stage consists in the application of the discrete wavelet transform to the fMRI time series. Thanks to this procedure, each time series is decomposed into a set of compactly supported basis function, which are uniquely scaled in frequency and located in time. As a results, for each subject, different fMRI time series at distinct scales are at disposal. At each scale, which represents an interval of frequencies, the correlation among regions is estimated. In our study, the focus is given to a low frequency bound, depending on the considered dataset, since it has been observed that the resting state information activity is capture at a frequency

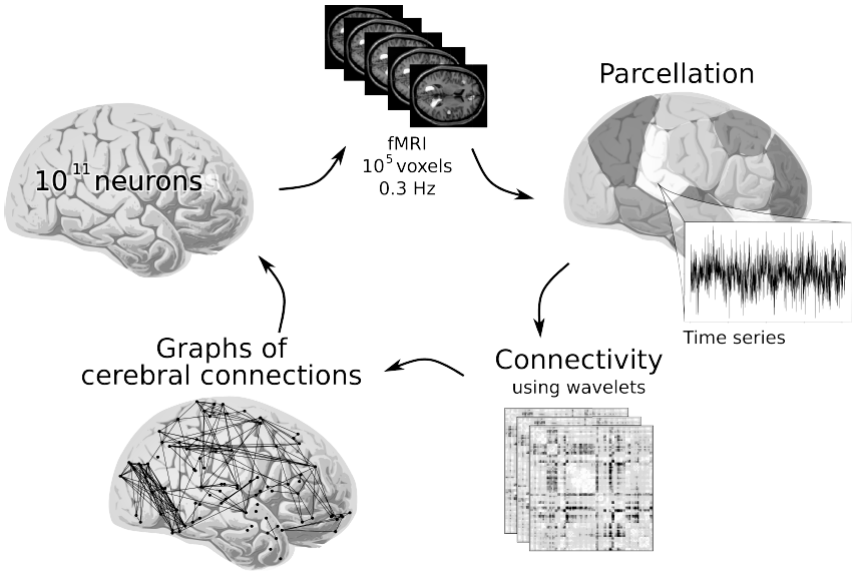


Figure 5.1: Framework for the definition of the graph. Courtesy of Sophie Achard, [http://www.gipsa-lab.grenoble-inp.fr/∼sophie.achard/Tutorial_brainwaver]

lower than 0.1Hz. By thresholding the selected correlation matrix, it is possible to obtain a binary matrix which represents the adjacency matrix of the final graph, whose nodes are given by the parcels. The threshold is tuned for each subject in order to compute a binary matrix with a given non-zeros entries number. In the graph construction, the number of edges of the expected network is previously chosen (we will refer at this number as cost). Thus, for each subject the threshold is different and it is selected to obtain the wanted cost.

Datasets

In our study, we have access to different data. You could find a resume in Table 5.1 (appendix). For each dataset we have at disposal five versions of graphs at different cost: 400, 800, 1200, 1600 and 2000 edges.

1. HCP test retest

   This dataset belongs to the collection of the Human Connectome Project. 100 healthy subjects underwent resting state f-MRI scanned in two different days, providing 200 connectivity graphs, a pair for each subject.

2. Coma and Controls

   This dataset is composed by 37 subject, 20 healthy controls and 17 comatose patients.

3. Young and Elderly

   This dataset is composed by the scans of 26 individuals. 15 are classified as younger and are aged 18–33 years and 11 classified as older participants who are aged 62–76 years.

4. ABIDE

   The Autism Brain Imaging Data Exchange (ABIDE) dataset is composed by 866 graphs. 402 subject with autism spectrum disorders and 464 healthy controls.

5. TBI This dataset is composed by 20 controls and 24 patients who reported traumatic brain injuries. For 15 patients, two scans are provided.

Dataset overview				
Name	Number of Data			Reference
	total	class 1	class 2	
HCP test retest	100 × 2	x	x	[16]
Coma	37	Control : 20	Comatose : 17	[2]
Young and Elderly	26	Young: 15	Elderly: 11	[1]
ABIDE	866	Control: 464	ASD: 402	[7]
TBI	44 - 15× 2	Control : 20	Injured : 24	[7]

Table 5.1

5.2 Experiments framework

5.2.1 Tuning parameters

We decide to use the HCP data for the tuning of the parameters. Referring to the notation introduced in chapter 4, the pair $(\mathcal{G}, \tilde{\mathcal{G}})$ is composed by the graphs of the same subject in the different scans. Using this dataset for the tuning, we expect to maximize the proximity between graphs of the same subject, forcing the embedding to capture the characteristics which are relevant in brain connectivity. Indeed, a parameters configuration which estimates similar embedding for similar graphs, is assumed to reproduce the specificity of brain connectivity network.

In our experiments, we focused on the data of cost = 400. Since the set of parameters and their possible values is not discrete and since, due to the embedding definition, it was not possible to write the tuning of parameters as an optimal problem, our results are expected to correspond to sub-optimal solution. We would evaluate the performance of the embedding in 5 turns, focusing each time on a single parameter. After having chosen the parameter to be tuned, we define a set of its possible values where we would estimate the goodness of the embedding. We fix all the other parameters to some standard values {walk-length = 30, number of walks per node = 18, size of the neighborhood set per node = 3, p = 1, q =

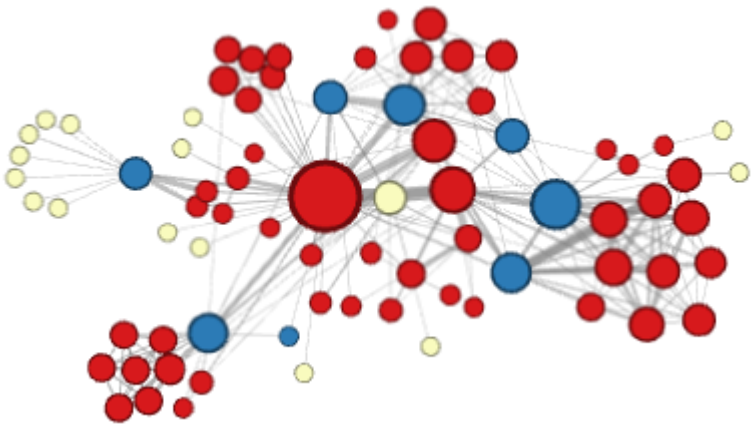


Figure 5.2: Clustering results on the network *Les Misérables*. The embedding is performed using a configuration which preserves structural equivalence and then the nodes are cluster using kmeans. Figure from [9]

2 }. These values are chosen looking at the ones used in the previous works [9, 14]. Especially, in [9] they used {dimension = 16, p = 1, q = 2 } to embed *Les Misérables* network ([10]) which has characteristics similar to the networks in our datasets. Concerning the cost and the node numbers, *Les Misérables* has 77 nodes and 254 edges, while our data have 90 nodes and 400 edges. Using their configuration, they were able to cluster the nodes according to their structural role as it is shown in figure 5.2. We set the remaining parameters (walk-length, number of walks per node and size of the neighborhood set per node) according to the ones used in [14] on the HCP dataset.

Since the embedding depends on a random walk generation process, even if it is performed on the same graph it embeds nodes in different vectors every time. Therefore, we evaluate the results over 50 different runs each time. The tuning order is given by dimension, walk-length, neighborhood size, (p,q) parameters and number of walks. From the second turn on, the already-tuned parameters are fixed to the value which has achieved the best performance in their corresponding turn. For every turn, we select a set of possible values for the considered parameter. This set always includes the values used in [9] and [14].

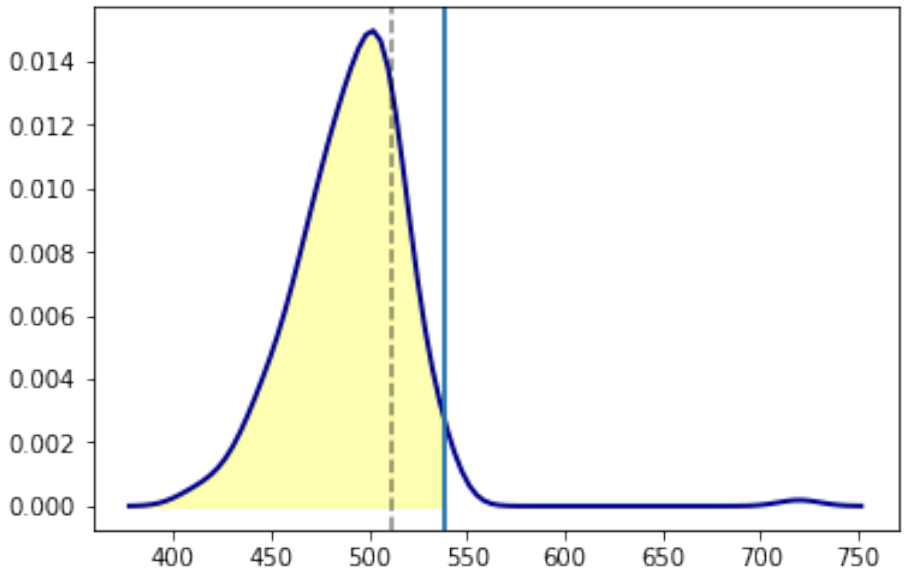


Figure 5.3: X-axis: proximity value Y-axis: frequency density function Example of proximity distribution for network 100307 - first scan. The blue vertical line corresponds to the value of the proximity with network 100307 - second scan. The yellow area is used to evaluated the performance of the embedding. Indeed, this area corresponds to the cumulative distribution evaluated in the proximity with the second scan. Averaging the value of this area over all the data, we determine μ. The final value $F^{-1}_{(1003007,\text{first scan})}(\mu)$ is indicated by the dashed vertical line. All graphs which have a proximity values greater than this value belong to the close-set of $(1003007, \text{first scan})$. Since this value is lower than the one on the blue line, $(1003007, \text{first scan})$ will count as well-classified.

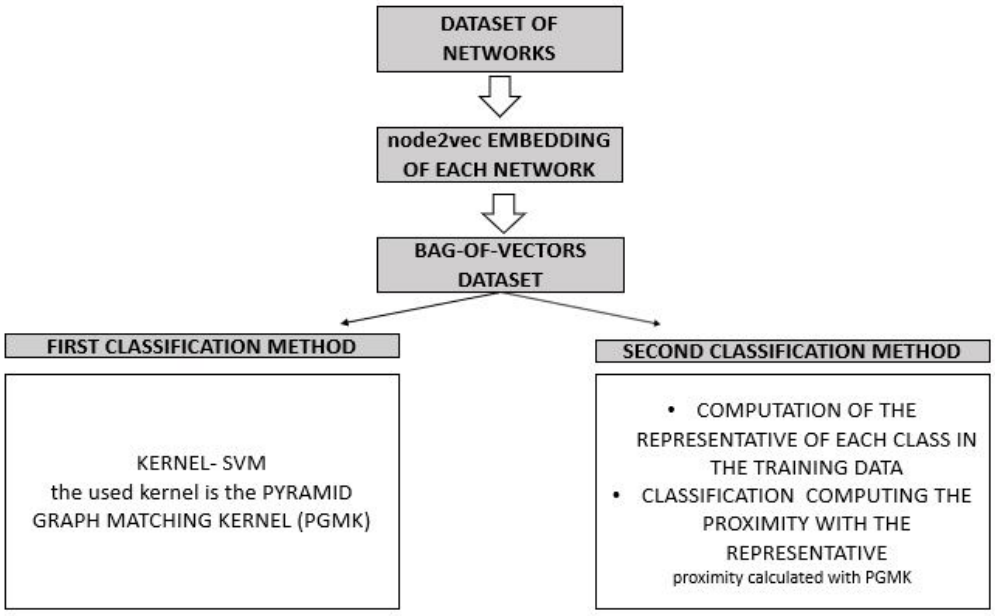


Figure 5.4: Scheme of the two classification procedures

5.2.2 Classification and representative graph

Using the parameters' values we have determined in the previous experiments, we perform the classification on each dataset. We use two procedures: kernel-SVM and comparison through representative.

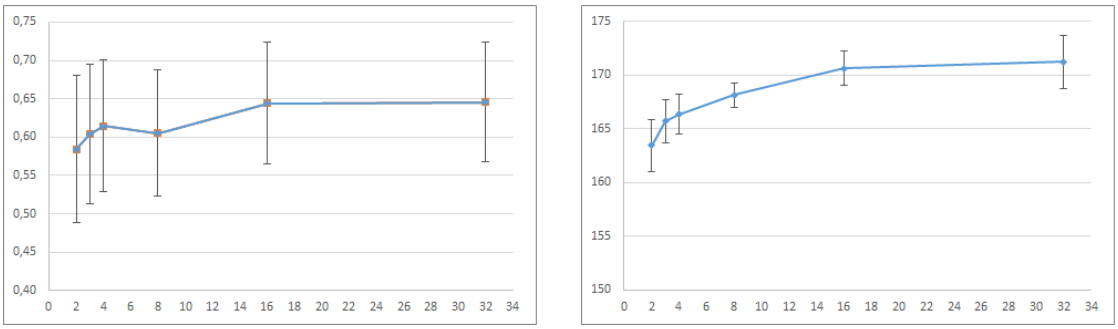
For 100 times, we randomly select a subset of training data, perform the embedding and calculate the gram matrix using the pyramid matching graph kernel. Thus, we use kernel-SVM to predict the labels on the test data.

A similar procedure is used to classify the data using the representative graph. In fact, we determine 100 different subsets of training data. Each training subset is, then, partitioned into the two classes. Afterwards, the representative embedding graph is evaluated for both classes. In order to classify the test data, we compute the proximity between the test and the two representatives and we predict the graph label, according to the one which is closer. A schematic representation of the process for the classification could be found in Figure 5.4.

Chapter 6

Results and Discussion

6.1 Results on the parameters tuning



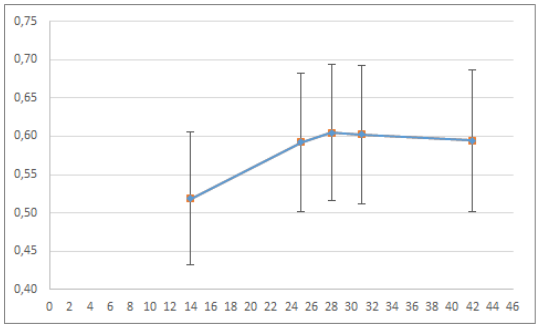
(a) average cumulative proximity frequency

(b) average counting of well-pairs

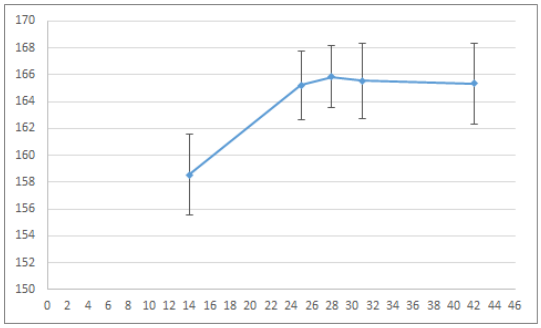
Figure 6.1: dimension tuning results

We first decide to tune the dimension. We would like to use a small dimension for two reasons. First, representing the embedding in a low dimensional space, would allow an easier interpretation and evaluation of the differences among graphs. Moreover, the kernel used for the comparison requires to create 2^4 cells per dimension per each pair of graphs. Therefore, using a low dimension corresponds to a low-computational algorithm in the complete experiments.

We show in Figure 6.1 the observed results for this parameter. The best value is obtained as a trade-off between the two measures used for the performance evaluation of the embedding. For the previous reason, we decide to set the dimension equals to 3. In comparison, with the [14] where the used value is 30, our estimation shows that the increasing between 16 and 30 is not enough for justifying the setting of the dimension to the higher value.

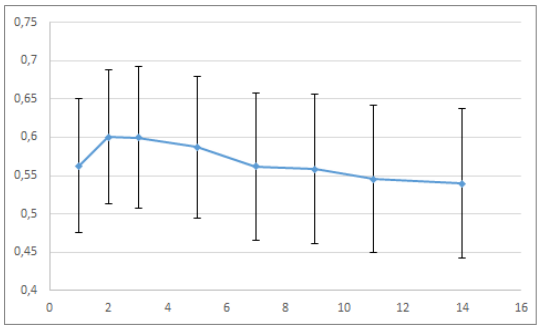


(a) average cumulative proximity frequency

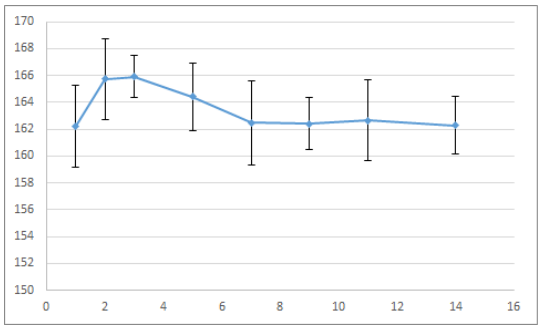


(b) average counting of well-pairs

(c) Top: walk-length's tuning results

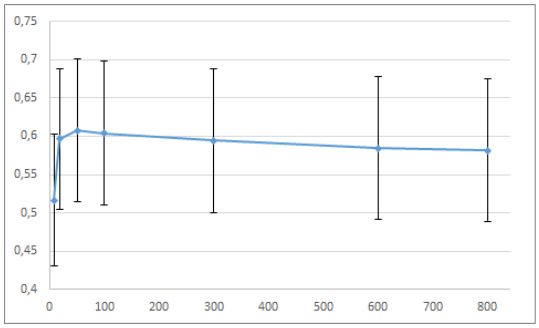


(d) average cumulative proximity frequency

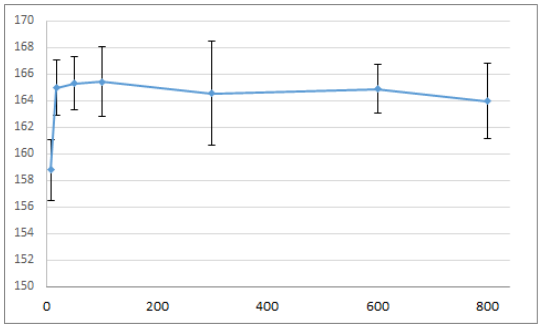


(e) average counting of well-pairs

(f) Top: size of the neighborhood set per node tuning results



(g) average cumulative proximity frequency



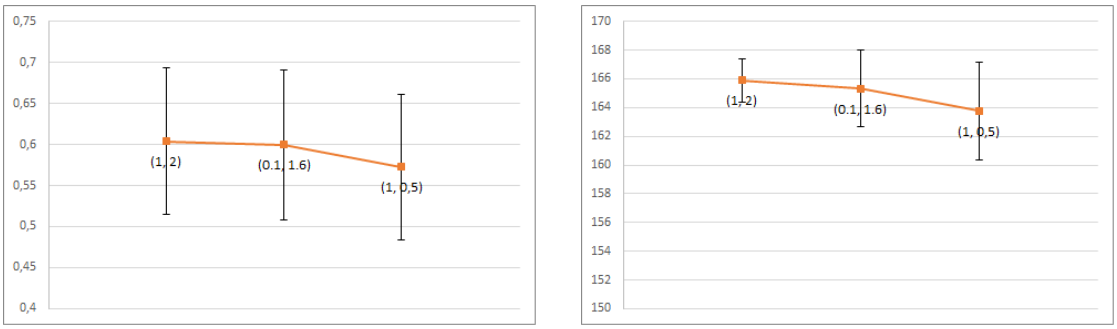
(h) average counting of well-pairs

(i) Top: number of walks' tuning results

Figure 6.2: Tuning results

The second parameter to be tuned is the length of the generated random walk. In [5], it has been proven that the walk length should be related with the diameter of the graph, which is the maximum of the minimal path length between two nodes in the graph. They suggest to use the diameter as walk length. However, our results in Figure 6.2 give a best score for a length of two times the diameter. A possible explanation for the discrepancy between our work and [5], could be related to the type of analysed networks. In fact, in [5], the focus is on a single social network to be clustered in community of nodes. They found that a too small walk length (minor than the diameter) fails to capture the community structure, while a too high length determines the same proximity over all the nodes, failing again. To our intuition, instead, to use twice the diameter length allows to go from a node to any others and to come back. A walk of this type, will represent the way the information is spread in both the directions. In this case, having more than an occurrence of the same node in the same walk could be significant for determining its structural role.
All the results for the other parameters are displayed in Figure 6.2. We compare the value for the number of random walks with the ones used in [14] which is 800. As we can notice in the plot of our results, the performance for 800 is even lower in the counting of well-pairs respect the other values. In [14], there is no explanation in the way they have calibrated the parameters, we limit to report the fact that in our framework their setting is not proven to be the optimal. The only value which is in common is 3 for the neighborhood size. In general, we find that the overall trend for each parameter is meaningful. The plot always reaches a peak and then remains stable. Hence, empirically, after a point there are no better results which could be achieved and we are selecting the best values possible in our experiments.
It is worth to spend some words on the sample strategy's parameters p and q. We evaluate the performance of the embedding in 3 different configurations the two proposed values in the [9] for the embedding of *Les Misérables* network and the pair which is used in [14] on the HCP dataset. We observe, that the results (Figure 6.3) are all very similar. On the other hands, the variance in the average counting of well-pairs is higher for the configuration (1,2) which corresponds to the

sampling strategy capturing the structural equivalence. We decide then to keep them equal to these standard values.



(a) average cumulative proximity frequency

(b) average counting of well-pairs

Figure 6.3: p and q parameters' tuning results

The final parameters' values are then in 6.1:

Parameters		Value
d	dimension of the embedding space	3
N	number of random walks per node	20
L	random walk length	2×diameter
k	size of the neighborhood set per node	3
p	return parameter	1
q	in-out parameter	2

Table 6.1

6.2 Results on the classification

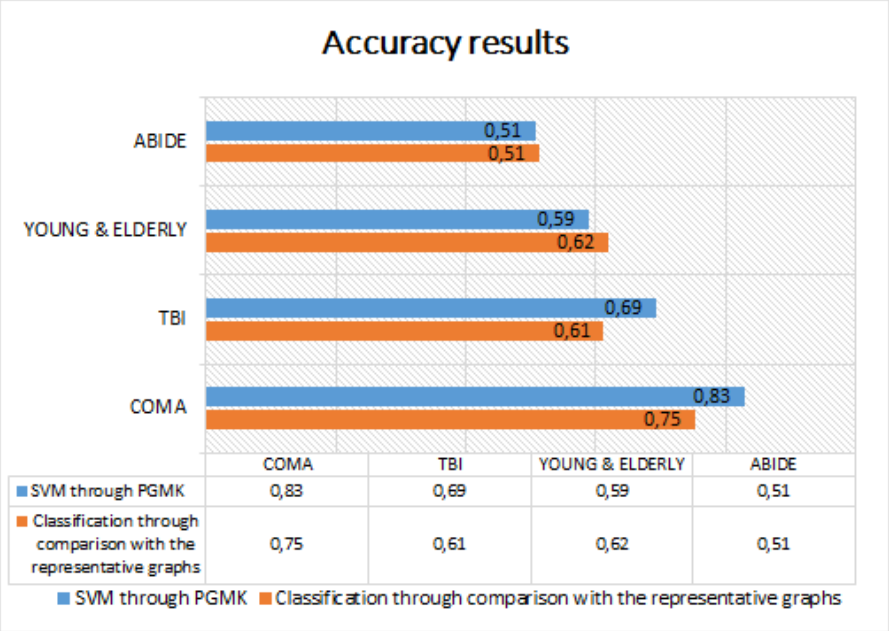


Figure 6.4: Results in the accuracy classification over all the dataset through the two different procedures. In blue the classification through support vector machine is apply on the pyramid graph matching kernel used to compute the graphs' proximity. In orange the classification obtained calculating the proximity given by the pyramid graph matching kernel between the test and the representative graphs of each class.

In terms of accuracy results through the two classification methods, we observe a variability on the datasets. Referring only to the SVM method, we notice that it gives as outcome a very good classification in the data of coma. The accuracy remains appreciable for TBI and Young&Elderly, but is equal to a random classifier for the ABIDE. Before giving a possible explanation for these differences, we would like to stress the encouraging results of the second classification method. In fact, for all the datasets the obtained accuracy is perfectly aligned within the two methods. We interpret this alignment as a validation of the proposal definition of a representative graph. It is important to say that the way our parameters

are calibrated is not in order to achieve the best accuracy, but to obtain a certain pattern on the data. For this reason, we believe that our procedure should not be only evaluated on the classification results. Indeed, the major interest of this work, is to provide an interpretation of the embedding with the purpose of determine which are the property of the brain network. For that reason, even if precedent works in brain networks classification, report better results, we believe that a network embedding procedure provides an additional value: the possibility to determine a signature for the class.
Concerning the coma dataset, previous attempts in the classification did not reach an accuracy value as high as ours. Moreover, using graph's global descriptors has been proven not to be enough in the detection of the difference between the two classes ([2]). Our achievement shows the power of our embedding method for graphs comparison in capturing the meaningful features of the network. Even if further analysis need to be conducted, we read this high result as an indicator of the type of property of the graph which are preserved in the embedding. To better explain, from [2], it has been proven that hubs of brain networks are radically reorganized in comatose patients. Considering the accuracy value, we could presume that in the embedding the structural role of nodes is likely to be preserved. In addition a next result could be analysed. Indeed, we could visualize the dissimilarities between the two representative graphs of coma and control. In Figure 6.5 we overlap the two representatives in the embedding space. The blue dots are the vectors of the control, while the red dots belong to the representative of the comatose. In terms of points' distribution, the two representatives occupy different positions in the embedding space. Accordingly to the way a match is determined in the kernel computation, we know that the vector coordinates associated to each node are significant for the comparison. Hence, these observed dispositions could reveal some important aspects of the brain networks.

Coming to the Young&Elderly dataset, a similar discussion could be conducted. In fact, it has been shown that some nodes of the two classes have a different behaviour in the spreading of the information ([1]). Especially, there are some nodes which are affected by the age in their efficiency. For the reached accuracy,

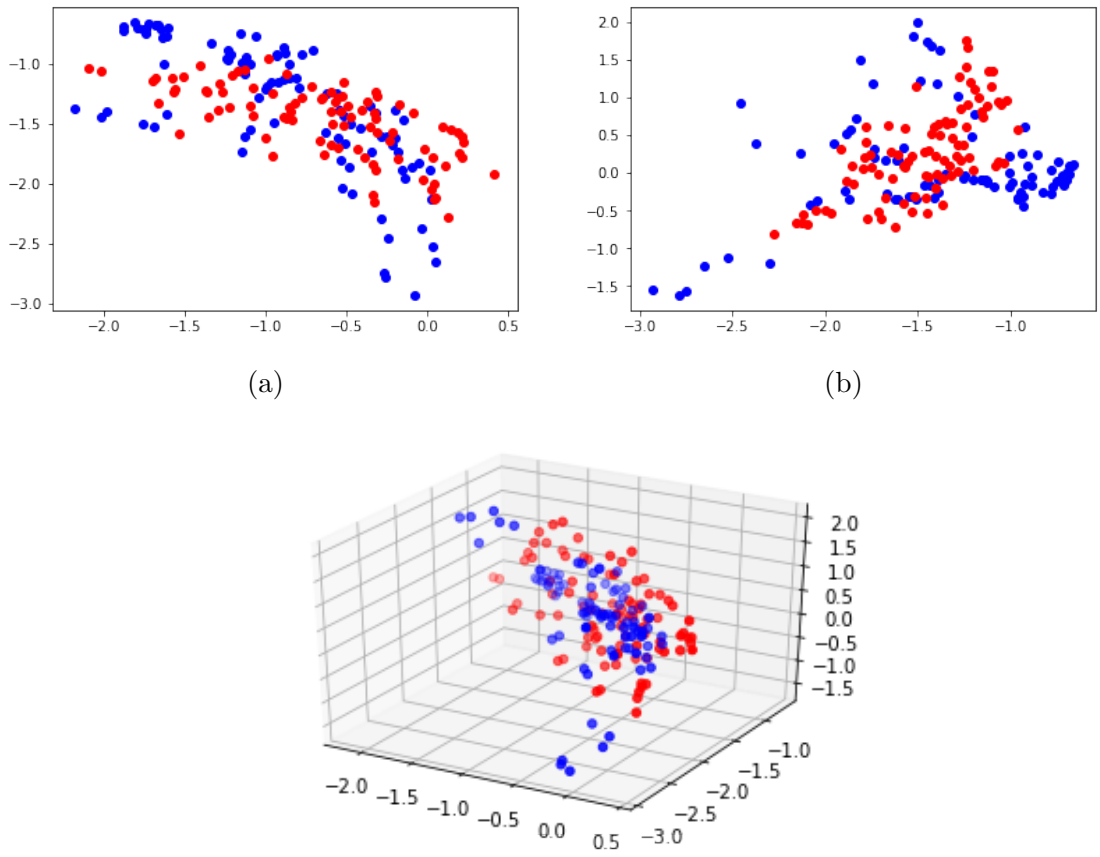
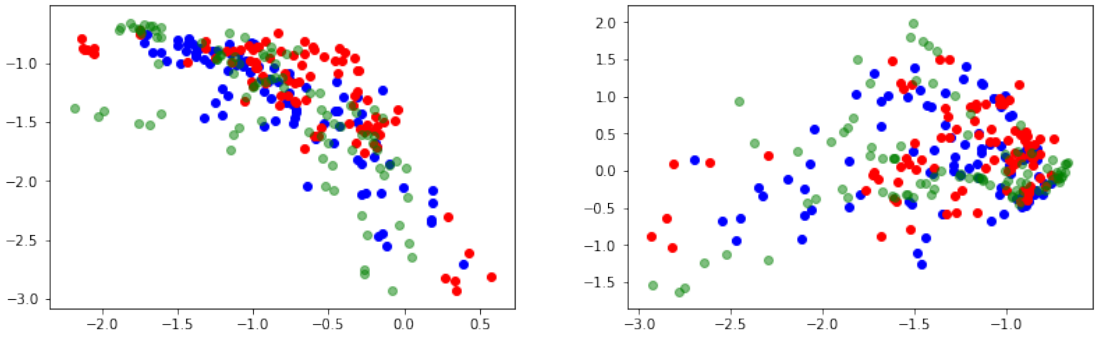


Figure 6.5: A visualization of the representative graphs in the coma dataset. In blue the control, in red the comatose. It could be noticed that the distribution of the vector for the two graphs are different. (a) and (b) are the projections on plane xy and xz



(a) (b)

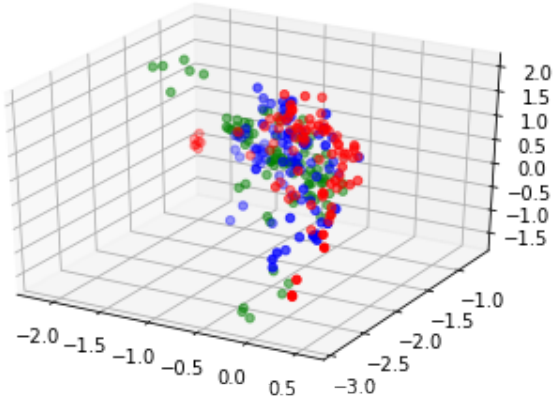
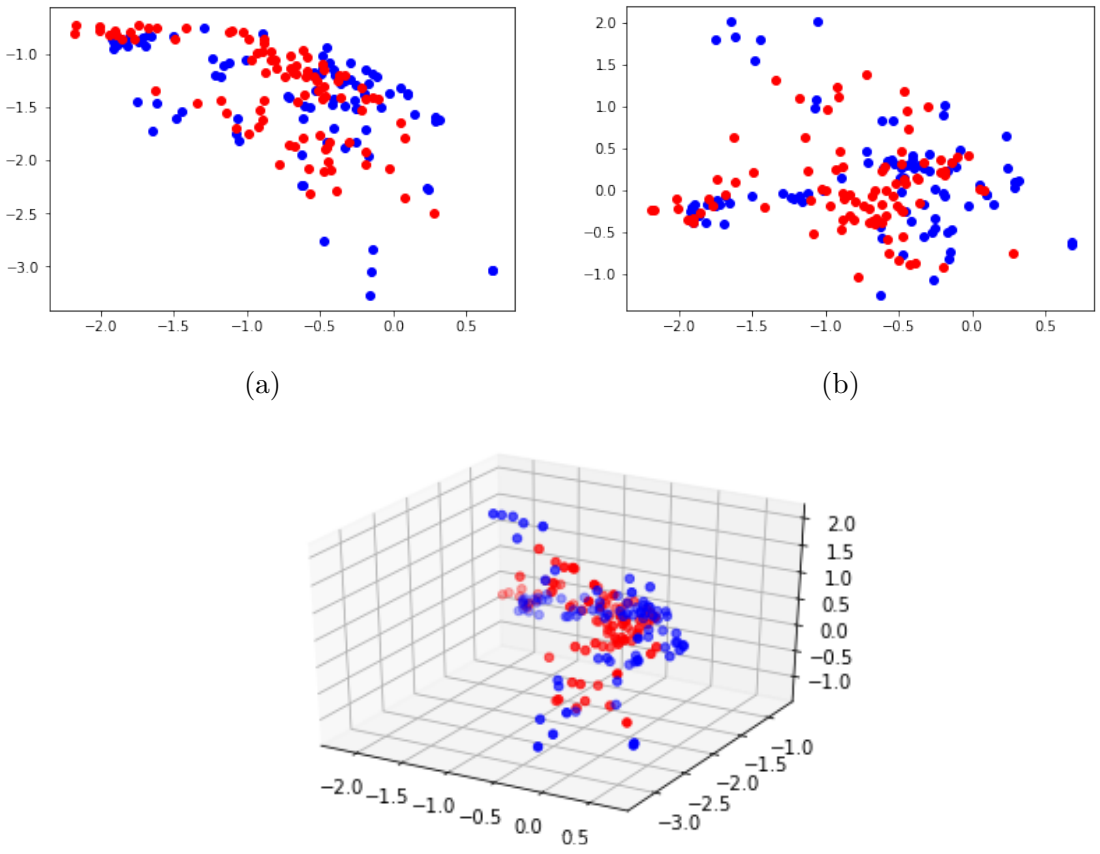


Figure 6.6: A visualization of the representative graphs in the Young&Elderly. In blue the elderly, in red the young. In green the representative of the control in the coma dataset. (a) and (b) are the projections on plane xy and xz,

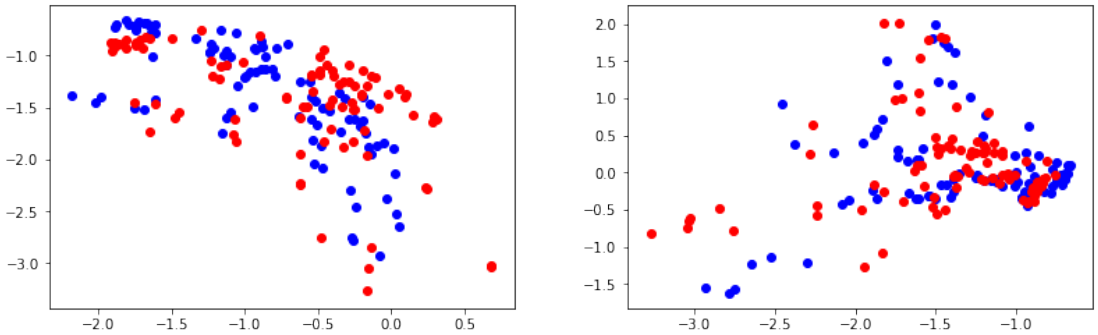
we presume that the embedding has the ability to preserve this difference. Besides, if we look at the representative embedding visualization (Figure 6.6) we notice that the points occupying different positions are few with respect to the previous dataset. It is also interesting to observe that, comparing these two graphs with the control representative in coma dataset, all the three sets of vectors share a similar distribution. Again, this fact can be thought as a validation of this method, since both classes belong to the healthy group.

For the TBI data, the accuracy reported in [18] is equal to 0.80. Even if still appreciable, our result is lower. Nevertheless, the achieved good result suggests



(a) (b)

Figure 6.7: A visualization of the representative graphs for TBI. In blue the control, in red the injured. It could be noticed that the distribution of the vector for the two graphs are different. (a) and (b) are the projections on plane xy and xz



(a) (b)

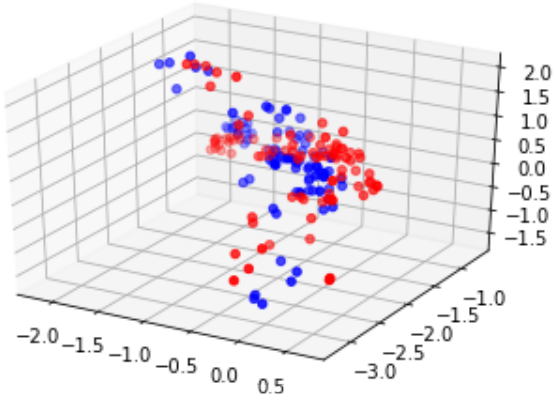


Figure 6.8: A visualization of the representative graphs for the control determined in the coma dataset and on TBI. In blue the first one, in red the second. It could be noticed that the distribution of the vector are perfectly overlaped. (a) and (b) are the projections on plane xy and xz,

again a possible difference on the structure of the connections (Figure 6.7). A further and intriguing result is shown in Figure 6.8, where the two representative graphs for the control class are determined in different dataset (blue for the coma, red for TBI). The two embeddings are perfectly overlapped. This plot confirms the existence of an healthy signature and reveals the possibility in the future, of determine a representative graph for no-pathological brain connectivity which could be used as a comparison with pathological ones.

Coming to the ABIDE dataset, in [7] a classification method is applied with an

accuracy result of 0.75. Our procedure differs from this one in many points: it requires to determine the connectivity graph from the correlation matrices, compute a nodal embedding, estimate a kernel for the comparison, etc. In their framework, the used data are not thresholded to generate an adjacency matrix, but they apply the classifcation directly in the correlation matrices. To compare the two works could then be misleading. In addition, the ABIDE dataset differs from the others in some aspects. For instance, the data acquisition have been obtained in different centers or the evaluated time-series are defined using a lower number of points. Even assuming the irrelevance of these differences in the network determination, the dimension of the dataset could be responsible for the lower accuracy: having more data would give a higher variability in the class and our method could fail in capturing it in its entirety. However, if we look at the computed representative for the two classes in Figure 6.9, we could notice that they are exactly overlying and they are both similar to the representative estimated for the control in the previous datasets. This could be taken as a positive remark and a further validation of our network embedding procedure: in fact, an optimistic explanation could be that the embedding has been forced to focus on some properties of the graph which are irrelevant in ABIDE data. In summary, failing in the classification task in this dataset is not a valid justification for rejecting our proposal. Instead, it is worth to give more space and time to a deeper investigation for determining the possible causes.

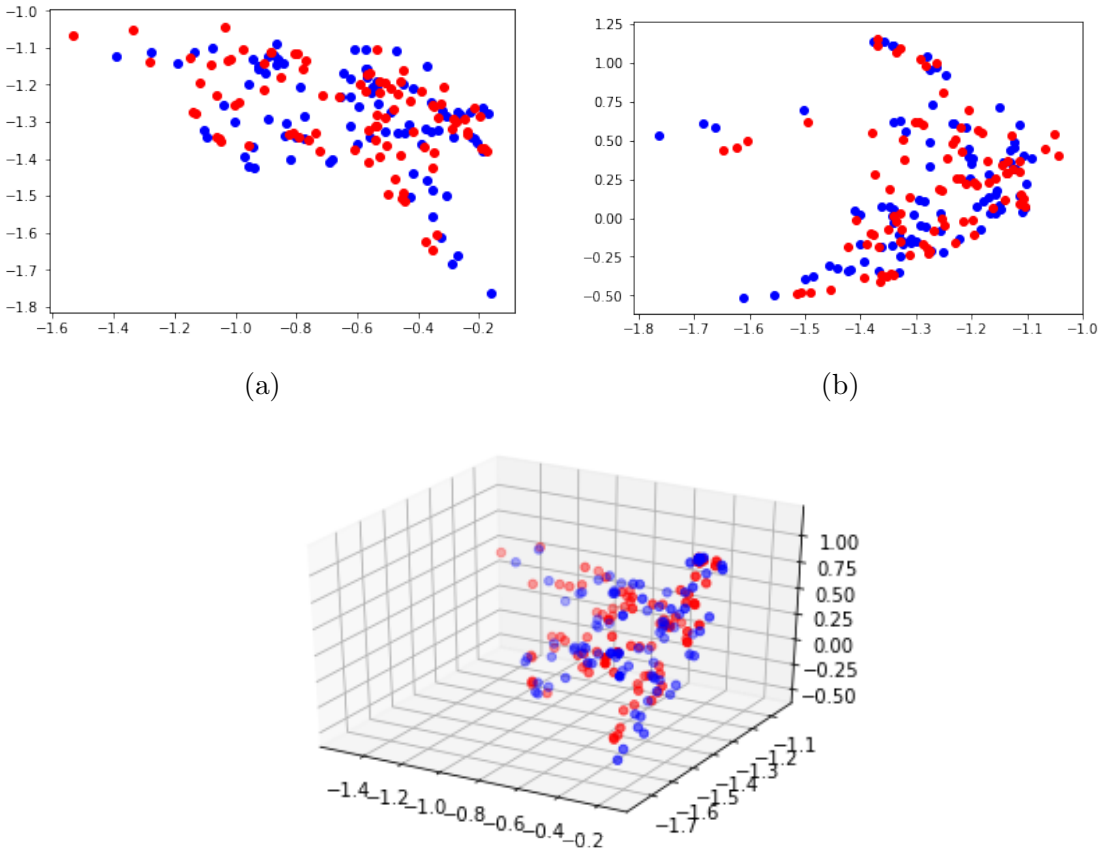


Figure 6.9: A visualization of the representative graphs for the control and the pathological on ABIDE. In blue the first one, in red the second. It could be noticed that the two embeddings are perfectly overlaping. (a) and (b) are the projections on plane xy and xz.

Chapter 7

Conclusion

To conclude, the main goal of this work was to exploit the potential of network embedding in capturing the main properties of brain connectivity network. We have proven that network embedding is a powerful tool not only in the dimension-reduction or for a single network analysis, but it could be applied in the comparison of brain connectivity graphs, too. A first contribution of our work is represented by the proposal of a general framework for the tuning of the parameters which is not related on a specific task, such as classification. The resulting values for the parameters could be applied for the embedding on different datasets, with the only restriction given by the number of nodes and edges of each graph. A second contribution is given by our definition of a representative graphs. The success of this method is validated by the alignment in the accuracy results between two classification procedures in all the datasets. Furthermore, this work opens new questions and needs further studies.

7.1 Future works

First of all, a neuroscience explanation of the difference of connectivity of the pathological brain could be explored. Performing an embedding analysis, node-wise could reveal which nodes are responsible of the variance between the classes. In this direction, a next step would be to construct an associated graph to the

representative network, starting from its embedding. This could lead to relevant results in the interpretation. In addition, in order to determine the best representative graph for an healthy brain, a merging on the control data of different datasets could be performed. In that way, we could investigate how similar the data are and how meaningful are the estimated representative graphs of the different datasets. Moreover, new ways of defining the representative graphs could be proposed. For instance, instead of requiring the representative to have the exact number of nodes as the other graphs in the dataset - as it is in our proposal-, we could examine the possibility of taking into account only the vectors which remain *stable* in the embedding space for a given class. In that way, the representative graphs would focus only on the nodes playing a special role for that class.

In conclusion, this work is not meant to give a definitive solution of the proposed problem, rather it lays the foundations for future explorations in this subject.

Bibliography

[1] Sophie Achard and Ed Bullmore. Efficiency and cost of economical brain functional networks. *PLoS Comput Biol*, 3(2):e17, 2007.

[2] Sophie Achard, Chantal Delon-Martin, Petra E Vértes, Félix Renard, Maleka Schenck, Francis Schneider, Christian Heinrich, Stéphane Kremer, and Edward T Bullmore. Hubs of brain functional networks are radically reorganized in comatose patients. *Proceedings of the National Academy of Sciences*, 109(50):20608–20613, 2012.

[3] Sophie Achard, Raymond Salvador, Brandon Whitcher, John Suckling, and ED Bullmore. A resilient, low-frequency, small-world human brain functional network with highly connected association cortical hubs. *Journal of Neuroscience*, 26(1):63–72, 2006.

[4] Stephen P Borgatti and Martin G Everett. Notions of position in social network analysis. *Sociological methodology*, pages 1–35, 1992.

[5] Siheng Chen, Sufeng Niu, Leman Akoglu, Jelena Kovačević, and Christos Faloutsos. Fast, warped graph embedding: Unifying framework and one-click algorithm. *arXiv preprint arXiv:1702.05764*, 2017.

[6] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2018.

[7] Kamalaker Dadi, Mehdi Rahim, Alexandre Abraham, Darya Chyzhyk, Michael Milham, Bertrand Thirion, Gaël Varoquaux, Alzheimer's Dis-

ease Neuroimaging Initiative, et al. Benchmarking functional connectome-based predictive models for resting-state fmri. *NeuroImage*, 192:115–134, 2019.

[8] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[9] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[10] Donald Ervin Knuth. *The Stanford GraphBase: a platform for combinatorial computing*. AcM Press New York, 1993.

[11] Vito Latora and Massimo Marchiori. Efficient behavior of small-world networks. *Physical review letters*, 87(19):198701, 2001.

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[13] Giannis Nikolentzos, Polykarpos Meladianos, and Michalis Vazirgiannis. Matching node embeddings for graph similarity. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[14] Gideon Rosenthal, František Váša, Alessandra Griffa, Patric Hagmann, Enrico Amico, Joaquín Goñi, Galia Avidan, and Olaf Sporns. Mapping higher-order relations between brain structure and function with embedded vector representations of connectomes. *Nature communications*, 9(1):1–12, 2018.

[15] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077, 2015.

[16] Maïté Termenon, Assia Jaillard, Chantal Delon-Martin, and Sophie Achard. Reliability of graph analysis of resting state fmri using test-retest dataset from the human connectome project. *Neuroimage*, 142:172–187, 2016.

[17] Nathalie Tzourio-Mazoyer, Brigitte Landeau, Dimitri Papathanassiou, Fabrice Crivello, Olivier Etard, Nicolas Delcroix, Bernard Mazoyer, and Marc Joliot. Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the mni mri single-subject brain. *Neuroimage*, 15(1):273–289, 2002.

[18] Davy Vanderweyen, Brent C Munsell, Jacobo E Mintzer, Olga Mintzer, Andy Gajadhar, Xun Zhu, Guorong Wu, Jane Joseph, Alzheimers Disease Neuroimaging Initiative, et al. Identifying abnormal network alterations common to traumatic brain injury and alzheimer's disease patients using functional connectome data. In *International Workshop on Machine Learning in Medical Imaging*, pages 229–237. Springer, 2015.

[19] Jinhui Wang, Xinian Zuo, and Yong He. Graph-based network analysis of resting-state functional mri. *Frontiers in systems neuroscience*, 4:16, 2010.

[20] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. Community preserving network embedding. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[21] Peter Wills and François G Meyer. Metrics for graph comparison: A practitioner's guide. *Plos one*, 15(2):e0228728, 2020.

Appendix A

A.1 Graphs measures of interest: metrics over nodes

We shortly present some metrics which could be associated to each node of the graph. Each of the following measures is a function measure : $\mathcal{V} \to \mathbb{R}$. While the definition of the degree of a node could come with the basics of graph theory, the proposed definition of efficiency and clustering coefficient refer to [2, 11, 19]. Whereas the definition of betweenness centrality could be found in [2, 8].

- **Degree**
  The degree of a node is defined as the number of links in which tue node is involved.

  Definition A.1. The degree of a node $v \in \mathcal{V}$ is given by

  $$\sum_{u \in \mathcal{V}} A_{vu} \tag{A.1}$$

  We can define the degree distribution of a graph $\mathcal{G}$ as the frequency of each degree value in the graph. For each of the possible value of degree, we count the fraction of nodes having that degree and we define the degree distribution of the given graph.

  Definition A.2. $f_k = \frac{1}{|\mathcal{V}|}\{v \in \mathcal{V} \text{s.t. } v \text{ has degree equals to } k\} \quad \forall k \in \mathbb{N}$ The sequences $(f_k)_{k \in \mathbb{N}}$ is called the degree distribution.

- **Efficiency**

  Given a graph, we could find for each pair of nodes in the graph the **minimal path length**; namely, the length of the path which connects the two nodes having the minimal length possible. In some cases, a path between the two nodes could be impossible to be found. In this case, the nodes are said to be *disconnected*. Therefore, for each node we can measure the average minimum length from the node and all the others in the graph. However, to avoid dealing with disconnected nodes, instead of considering the minimum path length, we could define a function which is proportional to the reciprocal average of the minimum path length. Especially, the efficiency measure of a node has been defined as the mean of the inverse of the minimum path length between the node itself and all the other nodes in the graph. The nodal efficiency represents in some way, a measure of the strength of the connection of the node. The closer to 1 the more similar the node behaves as a node in a graph where all possible edges are present. An efficiency close to 0 coincides with a disconnection from the rest of the graph.

  Definition A.3. Calling l_{vu} the minimal path length between nodes u, v, the efficiency of node v is given by

$$E_v = \frac{1}{N-1} \sum_{u \in \mathcal{V}, u \neq v} l_{uv} \tag{A.2}$$

- **Clustering coefficient**

  While the efficiency indicates the quality in the spreading of information in the global graph, the clustering coefficient of a node focus in the way the communication is organized in the closer nodes.

  Definition A.4. Saying that $\mathcal{G}_v$ is the graph obtained considering the neighbour nodes of v in $\mathcal{G}$ and having the same edges in $\mathcal{G}$, the clustering coefficient of v is

$$C_v = \frac{1}{|\mathcal{G}_v|(|\mathcal{G}_v|-1)} \sum_{u,w \in \mathcal{G}_v} \frac{1}{l_{uw}} \tag{A.3}$$

  where l_{uw} is the minimum path length between u and w in $\mathcal{G}_v$

- **Betweenness centrality**

  The betweenness centrality of a nodes is one of the possible measures used to quantify the centrality of a node in a graph. In particular, the betweennes centrality of a node counts the number of shortest paths among a pair of node in the graph which pass through the node itself.

  Definition A.5. The betweenness centrality of node v is given by:

$$B_v = \sum_{u \neq v} \sum_{w \neq u} \frac{|\text{shortest paths from } u \text{ to } w \text{ that pass through } v|}{|\text{shortest paths from } u \text{ to } v|} \tag{A.4}$$

A.2 Nodal proximity in node2vec

Definition A.6. The degree matrix $D \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is the diagonal matrix such that

$$D_{uv} = \begin{cases} 0 & \text{if } u \neq v \\ \sum_{z \in \mathcal{V}} A_{u,z} & \text{if } v = u \end{cases}$$

Definition A.7. The transition matrix $P \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is defined as

$$P = D^{-1}A$$

The entries of the transition matrix P_{uv} could be interpreted as the probability of go directly from node u to node v in a walk. For a given node $v \in \mathcal{V}$, the diagonal matrix $Q_v \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is given by the following formula:

$$(Q_v)_{ij} = \begin{cases} P_{vi} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Its non-zero entries are the probabilities of leaving node v. For each node in $\mathcal{V}$, we build such a matrix and we define $Q \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|^2}$, $Y \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|^2}$ and $W \in \mathbb{R}^{|\mathcal{V}|^2 \times |\mathcal{V}|^2}$ as:

$$Q = [Q_1 \cdots Q_{|\mathcal{V}|}]$$

$$
\mathrm{Y} = \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ |\mathcal{V}| \end{array}
\begin{array}{c}
\begin{array}{ccccccc} 1 \dots |\mathcal{V}| & | & |\mathcal{V}|+1 \dots\ 2|\mathcal{V}| & | & \dots & | & \dots \quad | \quad |\mathcal{V}|^2 - |\mathcal{V}| \dots |\mathcal{V}|^2 \end{array} \\
\left(\begin{array}{ccccccc}
\mathbf{1}_{|\mathcal{V}|} & | & 0 \dots 0 & | & 0 & | & 0 \quad | \quad 0 \dots 0 \\
0 & | & \mathbf{1}_{|\mathcal{V}|} & | & 0 & | & 0 \quad | \quad 0 \dots 0 \\
0 & | & 0 & | & \dots \mathbf{1}_{|\mathcal{V}|} \dots & | & 0 \quad | \quad 0 \dots 0 \\
0 & | & 0 & | & 0 & | & 0 \quad | \quad \mathbf{1}_{|\mathcal{V}|}
\end{array}\right)
\end{array}
$$

$$
W = (W_{ik})_{i,k \in \mathcal{V}} \quad W_{ik} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|} \quad (W_{ik})_{jl} = \begin{cases} \frac{A_{ik} M_{il}}{\sum_v A_{vk} M_{vl}} & \text{if } j = k \\ 0 \quad \text{otherwise} \end{cases}
$$

The matrix M is the memory matrix defined in chapter 3. Then, the nodal proximity $\mathbb{III}^{(L,p,q)} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is:

$$
\mathbb{III}^{(L,p,q)} = \mathrm{Y} \left(\sum_{l=0}^{L-1} W^l \right) Q^T \tag{A.5}
$$

The value $\mathbb{III}_{uv}^{(L,p,q)}$ is the expected number of times to visit node v in a random walk of length L which start in u.

A.3 A schematic on the brain connectivity network definition

Schematic of wavelet correlation analysis, thresholding, and functional network visualization

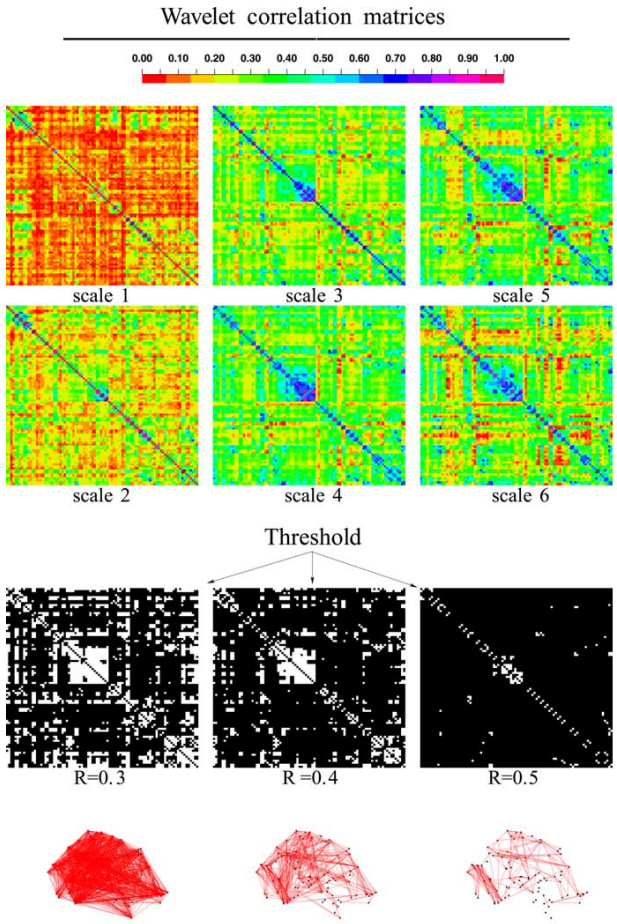


Figure A.1

Schematic of wavelet correlation analysis, thresholding, and functional network visualization. Top, fMRI time series recorded from each of 90 regions in each subject are decomposed using the discrete wavelet transform, and the inter-regional correlation is estimated at each scale for each pair of regions in each subject. Middle, The wavelet correlation matrices are thresholded to generate binary matrices, each element of which is either black (if there is no significant connection between regions) or white (if there is). The sparsity of the obtained adjancency matrix is determined by the choice of the correlation threshold R, as illustrated by applying three different thresholds ($R = 0.3, 0.4, 0.5$) to the scale 4 wavelet correlation matrix. Bottom, Thresholded matrices are visualized in anatomical space by locating

each region according to its y and z centroid coordinates in Talairach space and drawing an edge between regions that are connected. Image and caption from [3]

Appendix B

Tabels

List of anatomical regions in the left hemisphere defined in [17]			
Index	Label	Brain Region	Code
1	FA	Central region precentral	Precentral_L
3	F1	Frontal lobe, laterlar surface, superior frontal, dorsolateral	Frontal_L
5	F1O	Frontal lobe, orbital surface superior frontal, orbital part	Frontal_Sup_Orb_L
7	F2	Frontal lobe,lateral surface, middle frontal	Frontal_Mid_L
9	F2O	Frontal lobe, orbital surface, middle frontal, orbital part	Frontal_Mid_Orb_L
11	F3OP	Frontal lobe, lateral surface, inferior frontal, opercular part	Frontal_Inf_Oper_L
13	F3T	Frontal lobe, lateral surface, inferior frontal, triangular part	Frontal_Inf_Tri_L
15	F3O	Frontal lobe, orbital surface, inferior frontal, orbital part	Frontal_Inf_Orb_L
17	OR	Central region, rolandic operculum	Rolandic_Oper_L
19	SMA	Frontal lobe, medial surface, supplementary motor area	Supp_Motor_Area_L
21	COB	Frontal lobe, orbital surface, olfactory cortex	Olfactory_L
23	FM	Frontal lobe, medial surface, superior frontal, medial	Frontal_Sup_Medial_L
25	FMO	Frontal lobe, orbital surface, superior frontal, medial orbital	Frontal_Med_Orb_L
27	GR	Frontal le, orbital surface, gyrus rectus	Rectus_L
29	IN	Insula	Insula_L
31	CIAN	Limbic lobe, anterior cingulate and paracingulate gyri	Cingulum_Ant_L

33	CINM	Limbic lobe, median cingulate and paracingulate gyri	Cingulum_Mid_L
35	CIP	Limbic lobe, posterior cingulate gyrus	Cingulum_Post_L
37	HIPPO	Limbic lobe, Hippocampus	Hippocampus_L
39	PARA_HIPPO	Limbic lobe, parahippocampal gyrus	ParaHippocampal_L
41	AMYGD	Sub cortical gray nuclei, Amygdala	Amygdala_L
43	V1	Occipital lobe, medial and inferior surfaces, calcarine fissure and surrounding cortex	Calcarine_L
45	Q	Occipital lobe, medial and inferior surfaces, cuneus	Cuneus_L
47	LIN	Occipital lobe, medial and inferior surfaces, lingual gyrus	Lingual_L
49	O1	Occipital lobe, lateral surface, superior occipital gyrus	Occipital_Sup_L
51	O2	Occipital lobe, lateral surface, middle occipital gyrus	Occipital_Mid_L
53	O3	Occipital lobe, lateral surface, inferior occipital gyrus	Occipital_Inf_L
55	FUSI	Occipital love, medial and inferior surfaces, fusiform gyrus	Fusiform_L
57	PA		Postcentral_L
59	P1	Parietal lobe, lateral surface, superior parietal gyrus	Parietal_Sup_L
61	P2	Parietal lobe, lateral surface, inferior parietal gyrus	Parietal_Inf_L
63	GSM	Parietal lobe, lateral surface, supramarginal gyrus	SupraMarginal_L
65	GA	Parietal lobe, lateral surface, angular gyrus	Angular_L

67	PQ	Parietal lobe, medial surface, precuneus	Precuneus_L
69	LPC	Frontal lobe, medial surface, paracentral lobule	Paracentral_Lobule_L
71	NC	Sub cortical gray nuclei, caudate nucleus	Caudate_L
73	NL	Sub cortical gray nuclei, lenticular nucleus, putamen	Putamen_L
75	PALL	Sub cortical gray nuclei, lenticular nucleus, pallidum	Pallidum_L
77	THA	Sub cortical gray nuclei, thalamus	Thalamus_L
79	HESCHL	Temporal lobe, lateral surface, Heschl gyrus	Heschl_L
81	T1	Temporal lobe, lateral surface, superior temporal gyrus	Temporal_Sup_L
83	T1A	Limbic lobe, temporal pole: superior temporal gyrus	Temporal_Pole_Sup_L
85	T2	Temporal lobe, lateral surface, middle temporal gyrus	Temporal_Mid_L
87	T2A	Limbic lobe, temporal pole: middle temporal gyrus	Temporal_Pole_Mid_L
89	T3	Temporal lobe, lateral surface inferior temporal gyrus	Temporal_Inf_L

Table B.1: List of the anatomical regions used