Detection of Autism Spectrum Disorder Using Graph Representation Learning Algorithms and Deep Neural Network, Based on fMRI Signals

Abstract

In this paper, we are going to apply graph representation learning algorithms to identify autism spectrum disorder (ASD) patients within a large brain imaging dataset. Since ASD is characterized by social deficits and repetitive behavioral symptoms, it is mainly identified by brain functional connectivity patterns. Attempts to unveil the neural patterns that emerged from ASD are the essence of ASD classification. We claim that considering the connectivity patterns of the brain can be appropriately executed by graph representation learning methods. These methods can capture the whole structure of the brain, both local and global properties. The investigation is done for the brain imaging worldwide multi-site database known as ABIDE (Autism Brain Imaging Data Exchange). The classifier adapted to the features embedded in graphs is a LeNet deep neural network. Among different graph representation techniques, we used AWE, Node2vec, Struct2vec, multi node2vec, and Graph2Img. The best approach was Graph2Img, in which after extracting the feature vectors representative of the brain nodes, the PCA algorithm is applied to the matrix of feature vectors. Although we could not outperform the previous 70% accuracy of 10-fold cross-validation in the identification of ASD versus control patients in the dataset, for leave-one-site-out cross-validation, we could obtain better results (our accuracy: 80%). It is evident that the effect of graph embedding methods is making the connectivity matrix more suitable for applying to a deep network.

1. Introduction

Autism spectrum disorder (ASD), is a set of clinical presentations, emerging due to neurodevelopmental disorder. ASD symptoms are related to social communication, imagination, and behavior. Early diagnosis and early intervention before age 4 (e.g., 12–48 months) can improve cognition, language, and adaptive behavior significantly (Elder et al., 2017). Accurate and timely diagnoses of ASD significantly improve the quality of life of individuals with ASD. Although there has been widespread research in this field, there is no clear etiology to diagnose ASD.

To date, ASD diagnosis is done based on the behavioral characteristics of children, observed by parents and teachers at home or school (Nickel & Huang-Storms, 2017) (Almuqhim & Saeed, 2021). Since autism is related to abnormal development of the brain, assessing the brain function is at the top of automatic diagnosis and classification research. Resting-state functional MRI has been suitable, due to its spatial resolution. Brain regions interact together to emerge an especial behavior, *i.e.*, a region's function is tightly dependent on its interactions. Various differential observations consider the properties of the brain network in healthy and ASD subjects. However, statistical analysis could lead the researchers to a disorder whose mechanisms vary among patients. In other words, there is no unique fact to announce them as biomarkers of ASD.

By considering brain regions and their connections as a network, autism spectrum disorder detection alternatively could be a network classification task, in which machine learning techniques could help. To efficiently use information hidden in the resting-state fMRI, the connectivity measures obtained from resting-state fMRI are useful to understand the large-scale functional difference between healthy and abnormal brains.

After that, a suitable classifier should be used. Vast number of mental disorder diagnosis studies used traditional classifiers, such as Support Vector Machine (SVM), LASSO, and Bayesian classifier. But, deep learning methods showed a major preference in the case of connectivity matrix, because it is a high-dimensional feature of brain activity. The reason is that, a combination of different functional connectivity metrics increases number of hyperparameters of a machine learning algorithm, in such a way that just deep neural networks can afford it, and learn more complex structures of data. From this point of view, applying fully-connected deep neural networks, and convolutional networks on fMRI volumes and raw connectome data appears to be successful (Heinsfeld et al., 2018) (Sherkatghanad et al., 2020).

In (Heinsfeld et al., 2018) deep learning algorithm using the full connectivity matrix is applied to classify ASD and controls using ABIDE data. They showed anterior-posterior underconnectivity in the autistic brain and surpassed the state-of-the-art classification of autism by achieving 70% accuracy. Also, Convolutional Neural Network (CNN) was used to effectively diagnose Alzheimer's disease (AD) (Sarraf et al., 2016) and mild cognitive impairment (MCI) (Meszlényi et al., 2017). In another study, CNN was used to extract features from fMRI data, and SVM was used for classification (Nie et al., 2016). A Deep Auto Encoder was used to classify fMRI data of MCI (Suk et al., 2016). Furthermore, different hidden layers in between the encoder and the decoder (Patel et al., 2016), were added to afford different tasks, like denoising (Heinsfeld et al., 2018), or generating sparse features (Guo et al., 2017). Other networks, like Radial Basis Function network (RBFN) (Vigneshwaran et al., 2015), Restricted Boltzmann Machine (RBM) (Huang et al., 2016), and Deep Boltzmann Machine (DBM) can be used to extract features from fMRI data, because they can combine the information of different voxels of the region of interest (Zafar et al., 2017).

Different traditional and deep machine learning algorithms reported good performances in ASD classification of ABIDE dataset while considering individual sites (Sherkatghanad et al., 2020). But our main concern is that after intermingling all the sites, or leave-one-site-out cross-validation algorithm, accuracy, *i.e.*, the percent of correctly classified subjects, and the area under ROC is diminished. In other words, yet there is not an algorithm appropriate for clinical usage. In summary, the accuracy of some ASD classifiers using SVM and DNN algorithms, some decomposition techniques like Principal Component Analysis (PCA), and graph neural network ranges from 55% to 70.4% (Kazeminejad & Sotero, 2020; Parisot et al., 2018; Sharif & Khan, 2021; Xing et al., 2019). Thus, still, further experiments are required to be conducted with patients of different ages to ensure the clinical value of these methods (Li et al., 2018).

On the other hand, the benefit of DNN is mainly due to a large number of training examples (Guo et al., 2017; Heinsfeld et al., 2018; Kim et al., 2016; Kuang et al., 2014). Developing deep learning approaches to work with functional connectivity (FC) features using small or at best modest sample sizes of neurological data (A. di Martino et al., 2014; Guo et al., 2017; Heinsfeld et al., 2018; Kim et al., 2016; Kuang et al., 2014) is debatable from the reproducibility and generalizability point of view. One solution is the deep transfer learning neural network (DTL-NN) approach that could achieve improved performance in classification for neurological conditions (70.4% for ASD detection), especially where there are not large neuroimaging datasets available (Li et al., 2018). Other solutions are Synthetic Minority Oversampling Technique (SMOTE) to perform data augmentation to generate artificial data and avoid overfitting (Eslami & Saeed, 2019), and sparse autoencoder (SAENet) that was used for classifying patients with ASD from typical control subjects using fMRI data (70.8% accuracy, and 79.1% specificity) for the whole dataset as compared to other methods (Almuqhim & Saeed, 2021). Another approach is to develop a machine learning approach with a robust training methodology (Li et al., 2018). Machine learning algorithms able to extract replicable, and robust neural patterns from brain imaging data of ASD patients, reach suitable classification results (Pereira et al., 2009).

Another solution in studies with limited sample sizes is the reduction of the size of features indicating useful connectivity properties by network analysis methods. The ease of representing brain connectivity information according to graph theory makes them very valuable tools in this area. Machine learning on graphs finds its importance here: Finding a way to represent, or encode graph structure is the subject of this task. Nowadays, in order to model information underlying the graph structure, there are new ways of representing and analyzing graphs, which afford the complexity of working with big graphs. Referring to these representation algorithms as embedding, applying these approaches to brain networks is named connectome embeddings (CE). These embedding algorithms involve converting graphs into vectors. Network embedding techniques can be divided into three buckets: 1) based on engineered graph features, 2) obtained by training on graph data, 3) obtained by a layer of a deep network. The main drawback of the former is that structural homologies or higher-order relations of the connectivity matrix could not be captured (Rosenthal et al., 2018). Furthermore, these features are not flexible, *i.e.*, they cannot adapt during the learning procedure. In summary, many of these local and global features cannot capture the topological shape of the graph.

In the second bucket, referred to as shallow embedding, network embedding vectors are learned by optimizing different types of objective functions defined as a mapping to reflect geometric information of graph data. This optimum embedded space is the final feature vector. These algorithms involve learning approaches that map nodes to an embedding space. Anonymous walk Embedding (AWE), Node2vec, Struct2vec, Deep Walk, multi node2vec, and graph2image (Grover & Leskovec, 2016; Ribeiro et al., 2017) are some well-known algorithms of this bucket. Representing higher-order features of the connections of a graph is referred to as the Graph2image method, in which the embedded space of the brain network is transformed into an image. This method is helpful to develop an image that is convenient for training a CNN. The advantage of this method is the capability of dimensionality reduction of this image by an algorithm like PCA, and still have an image at hand (Meng & Xiang, 2018). Multi-node2vec was applied on fMRI scans over a group of 74 healthy individuals. Multi-node2vec identifies nodal characteristics that are closely associated with the functional organization of the brain (Wilson et al., 2018).

In the third bucket, referred to as deep embedding, CE and deep learning algorithms are combined to form a single deep network. This combinatory network can exploit the connectome topology. In this category, a Hypergraph U-Net (HUNet), Graph U-Net (GUNet) (Gao & Ji, 2019) and Hypergraph Neural Network (HGNN) (Feng et al., 2019) are proposed in which low-dimensional embeddings of data samples are learnt from the high-dimensional connectivity matrix. Indeed, these networks emerged as a subset of deep graph neural networks (GNN) (Gao & Ji, 2019; Kipf & Welling, 2017; D. Wang et al., 2016) are able to model the deeply nonlinear relationship node connectomic features (Banka & Rekik, 2019; Bessadok et al., 2019; Ktena et al., 2017).

In this paper, we demonstrate the role of the second bucket (CE method) in representing the structure with which brain regions are connected to each other and assess its effect in ASD classification. Based on the ABIDEI and ABIDEII public datasets, recorded at some different sites, we want to investigate whether CE can surpass previous researches or not. Accordingly, by using CNN classifiers, we claim that there is great potential in combining graph representation methods, with deep learning techniques for fMRI-based classification.

In fact, we claim that representation-based features can solve the problem of high-dimensional input of the deep network. The task of classification based on a connectivity matrix is very similar to image classification. In this matrix, each row and column correspond to a brain region. The value of every entry is the connectivity measure between two relevant regions which its row and column stand for. However, the matrix obtained by connectivity embedded patterns is not exactly similar to natural images. In this matrix, any shift of the patterns is interpretable. Thus, the design of the convolutional filters is different from those of natural images, because they should consider the properties of the connectome database to determine different connectivity structures. Therefore, our idea to use the information of node structures as a new and low-dimensional source might increase classification performance. However, such dimension reduction may lead to more ambiguity about the place of alteration in the connectivity matrix.

The structure of the paper is as follows: After describing the network embedding techniques in section 2, suitable embedding-based features are illustrated. In section 3, classification technique using deep network is declared. Afterward, ABIDE database, their pre-processing methods, and the embedded features extracted from them are introduced. These features are applied to deep network to detect ASD subjects. Some evaluation measures like F-score, and accuracy of this classifier is reported in the Results section and is compared to other literature working on ABIDE dataset.

2. Network Embedding Concept

The concept of network embedding date back to the skip-gram model, originally proposed for word embedding. It can be described as follows: Suppose there is a graph $G = (V, E, A)$ with V as the node set, E as the undirected and weighted edge set, and A as the adjacency matrix. We are going to find the optimum function $z = f(v) \epsilon R^d$ that maps each node or sub-graph to a d-dimensional vector disclosing the structure of the graph. These vectors should be representative of the graph and can be used as the feature vectors uncovering the similarities of the graph for machine learning algorithms. At this level, each node corresponds to a d-dimensional embedded vector involving its connections with all other nodes (Hamilton William L. et al., 2017).

Indeed, these low-dimensional embedded vectors can summarize either position of nodes in the graph, or the structure of their neighborhood, and user-specified graph statistics (Hamilton William L. et al., 2017). Most shallow embedding mapping techniques are done based on a look-up table, just like what is occurred in classic matrix factorization for dimensionality reduction (Hamilton William L. et al., 2017). For another part of shallow embedding techniques, learning the embedded vector for each node is in fact training an encoder-decoder system, defined as an optimization method. The decoder maps the similarity of two nodes into a real-valued similarity measure. Different techniques able to afford this job (like Deepwalk, Node2vec, AWE, TSNE, GraRep, and others) are based on a stream of randomly generated walks. Random walk methods lead to better performance, due to their flexible and stochastic nature of the similarity they consider. The resultant vectors can describe the similarities and subgraph membership with relatively few dimensions. These learned embedded vectors can be used as features of the graph.

The core of this relevant optimization problem is to find a mapping such that nearby nodes in short random walks have similar embedding vectors. As a similarity measure, $p_g(v_j|v_i)$ is the probability of visiting v_j on a random walk starting from v_i and with length T. For each vertex v_i, a sequence of vertices is formed, from which the probabilities can be obtained, according to the ratio of weight of edge (v_i, v_j) to the total (positive) edge weight in the r-truncated random walks. Here r varies from 1 till the original walk length T. As seen in Fig. 1, nearby nodes are mapped into vectors with small between angles.

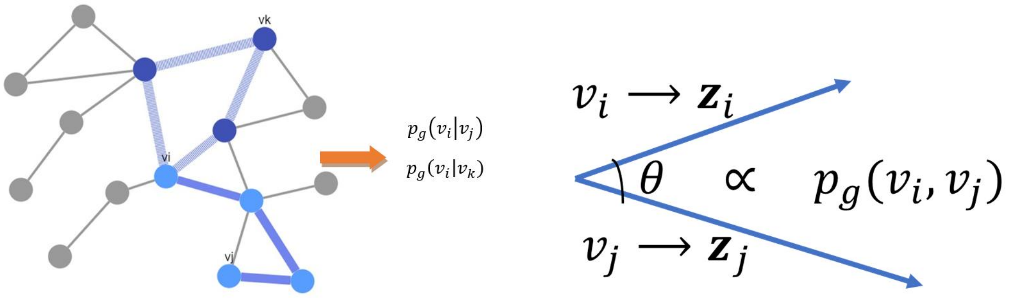
vk
vi
vj
$p_g(v_i|v_j)$
$p_g(v_i|v_k)$
$v_i \longrightarrow \mathbf{z}_i$
$\theta \propto p_g(v_i, v_j)$
$v_j \longrightarrow \mathbf{z}_j$

Figure 1. Schematic of random-walk as the basis of shallow embedding techniques. Possible random walks between every two nodes lead to obtaining the conditional probability of these nodes. This probability is converted into the angle between the embedded vectors of these nodes (Hamilton William L. et al., 2017)

2.1 Node2vec and DeepWalk

Two of the well-known network embedding algorithms defined based on the optimization methods are Node2vec and DeepWalk. For DeepWalk and Node2vec algorithms, decoding is done to obtain vectors z_i, z_j according to an algorithm defined as below (Grover & Leskovec, 2016):

$$DEC(\mathbf{z}_i, \mathbf{z}_j) \triangleq \frac{e^{\mathbf{z}_i^T \mathbf{z}_j}}{\sum_{v_k \epsilon \mathcal{V}} e^{\mathbf{z}_i^T \mathbf{z}_k}} \approx p_{g,T}(v_j|v_i)$$

where $p_{g,T}(v_j|v_i)$ is the probability of visiting v_j on a random walk starting from v_i and with length T. Both Node2vec and DeepWalk algorithms could be easily described by the concept of context graph. This context graph is an auxiliary graph C in which the weight of edge between the source node v_i, and its context v_j, *i.e.*, $c_{i,j}$ is proportional to the similarity between nodes v_i and v_j in graph G. No edge between them, is equivalent with weight -1. For a directed graph G, $c_{i,j}$ determines the role of a vertex as sink or source node. Based on context graph and optimization, $p_{g,T}(v_j|v_i)$ is determined. However, there are different network representation learning algorithm. But, among them Node2vec and DeepWalk are the most popular ones (Khosla et al., 2021).

In Node2Vec and DeepWalk algorithms similarity of nodes $p_{g,T}(v_j|v_i)$ is obtained via assessing the nodes sampled in random walks through graph. In other words, weights of edges of context graph C is the number of walks able to reach node v_i via v_j. Sampling of the nodes on which random walk move, is different in Deep-walk and Node2vec (Khosla et al., 2021). While DeepWalk performs a uniform random walk, Node2vec follows a 2nd order biased random walk. In other words, in Node2vec algorithm, negative sampling (Z. Yang et al., 2020) is used, but in DeepWalk algorithm, hierarchical softmax technique is employed. Distribution function underlying the probability of visiting a node in the random walk differ in these two techniques. Thus, the optimization cost function defined as an expectation of all combinations of elements, and subsequently the results, differs in these two algorithms.

In Fig. 2, bread-first and depth-first search (BFS and DFS) in a graph for node v^* (the algorithms used in random walk) are shown. In BFS the neighborhood is restricted to immediate neighbors of v^*, *i.e.*, for a neighborhood of size $k = 3$ the nodes $v_1, v_2, and\ v_3$. In DFS, the neighborhood consists of nodes sequentially sampled at increasing distances from v^*. In Fig. 2, DFS samples $v_4, v_5, and\ v_9$.

Node2vec does an interpolation between BFS and DFS via two parameters p & q, selected suitably. Hyperparameters (p & q) control likelihood of walks immediately revisiting a node, and revisiting a node's one-hop neighborhood, respectively (Khosla et al., 2021). In a biased random walk, the probability of transition from a node to the next one is different according to the length of the shortest path between them (Grover & Leskovec, 2016). This can be generalized to higher order representation by considering the sequences would be appeared during random walks gathered in set D, including triples of nodes, with parameters p & q (Khosla et al., 2021). Thus, probability of traveling from node u to v and then w indicated by $\underline{P}_{u,v,w}$ can be defined as below.

$$\underline{P}_{u,v,w} = \frac{T_{u\to v\to w}}{\sum_w T_{u\to v\to w}}$$

Where

$$T_{u\to v\to w} = \begin{cases} \frac{1}{p} & if\ (u,v) \in E(v,w) \in E, u = w \\ 1 & if\ (u,v) \in E(v,w) \in E, u \neq w \\ \frac{1}{q} & if\ (u,w) \in E(v,w) \in E, u \neq w, (u,w) \notin E \\ 0 & otherwise \end{cases}$$

where $E(v,w)$ is the shortest path between v and w. This algorithm is summarized in Algorithm 1. As well, DeepWalk algorithm is illustrated in Algorithm 2.

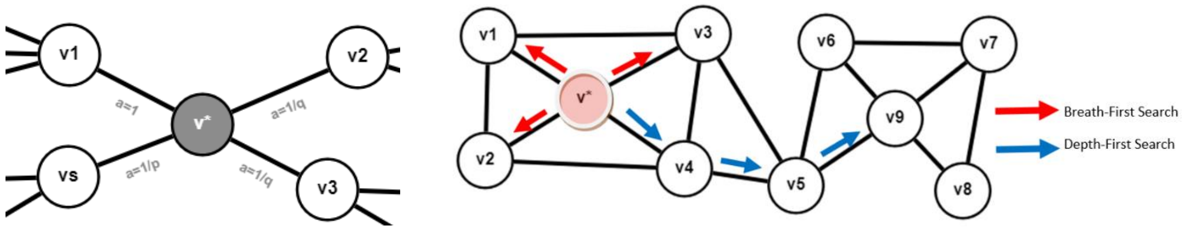

Figure 2. A. The way Node2vec bias the random walk by p and q, B. Difference between Breath-First Search (BFS) that find one-hop neighborhood, and Depth-First Search (DFS) random walks that explore further away, to capture the structure.

Node2vec and Deep-walk approaches lead to a unique embedding vector for every individual node, but have some drawbacks, including woring as a lookup table, its computational cost, fail to leverage attribute information of nodes involving node's position and role, weakness in predict information of unseen nodes. To alleviate the abovementioned drawbacks, two alternatives are arisen: 1) some embedding approaches that enable capturing the structural roles of nodes, have been proposed (Donnat et al., 2018; Ribeiro et al., 2017), 2) Network embedding in a feature-based manner has been proposed.

Algorithm 1. Node2Vec Algorithm (Meng & Xiang, 2018) (Grover & Leskovec, 2016)

```
Input: $G(V,E)$
Walks iteration times: $I$
Walk length: $l$
    For $iter = 1$ to $I$ do
    For each node $v_i \epsilon V$ do
    $j = 0; X_0 = v_i$
    while $j < l - 1$ do
    $\{X_{j+1}$ is a random neigbor of $X_j$ according to $\underline{P}_{u,v,w}$
    Add node to $P_{v_i}^{iter} = (X_0, X_2, \ldots, X_{j+1})$
    $j = j + 1\}$
    For each node we have $I$ random paths: $P_{v_i}^{1}$ to $P_{v_i}^{I}$ in set $D$
    $D$ is complete of $I$ paths for all nodes
    Form the context graph $C$ according to number of edges between each coup
    Form $p_{g,T}(v_j|v_i)$ from $C$
    Define the cost function
    Use gradient descent to solve the optimization problem ***
```

Algorithm 2. DeepWalk Algorithm (Perozzi et al., 2014)

```
Input: $G(V,E)$
Walks iteration times: $I$
Walk length: $l$
    For $iter = 1$ to $I$ do
    For each node $v_i \epsilon V$ do
    $j = 0; X_0 = v_i$
    while $j < l - 1$ do
    $\{X_{j+1}$ is a random neigbor of $X_j$ according to a uniform distribution
    Add node to $P_{v_i}^{iter} = (X_0, X_2, \ldots, X_{j+1})$
    $j = j + 1\}$
    For each node we have $I$ random paths: $P_{v_i}^{1}$ to $P_{v_i}^{I}$ in set $D$
    $D$ is complete of $I$ paths for all nodes
    Form the context graph $C$ according to number of edges between each coup
    Form $p_{g,T}(v_j|v_i)$ from $C$
    Define the cost function
    Use gradient descent to solve the optimization problem
```

2.2 Struct2Vec

As an example of the first alternative, Ribeiro technique referred to as Struc2Vec, generate some new graphs $G_0, \ldots, G_K$, each to capture one kind of k-hop neighborhood structural similarity, from the original graph G. The algorithm is as follows (Ribeiro et al., 2017):

1. For each node v_i, order the sequence of degrees of nodes exactly with distance of k-hops from it: $R_k(v_i)$.
2. Start from a weighted graph G_0 whose edges have zero weights $w_0(v_i, v_j) = 0$.
3. Build a sequence of weighted graphs whose edges varies adaptively by the equation:
$$w_k(v_i, v_j) = w_{k-1}(v_i, v_j) + d\left(R_k(v_i), R_k(v_j)\right)$$
where $d\left(R_k(v_i), R_k(v_j)\right)$ is distance between sequence $R_k(v_i)$ and $R_k(v_j)$, and could be defined with different measures.
4. Run random walk on new graphs $G_0, \ldots, G_K$ to implement Node2vec on them, and learn latents from them, using an algorithm like SkipGram.

2.3 Feature-based embedding:

In the second alternative, namely, feature-based methods, two algorithms referred to as Graph2Img and AWE are considered.

The Graph2Img algorithm, first transfers the original network into feature vectors and then uses clustering methods to group nodes. In other words, after graph node embedding process into a d-dimensional space, representations of nodes are gathered in a matrix of dimension $N \times d$, where $N = |V|$, *i.e.*, number of nodes in the graph. Next, we can decide whether all features are important or not, and determine their priority. In fact, we can use just the most important dimension, the second important one, and so on. Here, principal component analysis (PCA) method is used to transform the d-dimension vector of node v_i into dPCA-dimension vector LPCA, which is a sequential list.

$$LPCA_i = \{f_{1i}, f_{2i}, \dots, f_{dPCA,i}\} \qquad 1 \le i \le N, dPCA < d$$

where N represents the number of nodes in the graph. In such a way, for each component f_1 there are a N-dimension vector. Thus, for each pair of components, we can build a matrix of $2 \times N$ dimension, and interpret it as N ordered pairs, or N points in a 2D plane. As an example, M_{12} is built from components 1 and 2:

$$M_{12} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1,N} \\ f_{21} & f_{22} & \cdots & f_{2,N} \end{bmatrix}$$

Digitizing values of LPCA components to r equally sized bins, this two-dimensional space is partitioned into $r \times r$ bins, i.e., there are $r \times r$ potential states for an ordered pair. 2D histogram of these ordered pairs (namely H), is a $r \times r$ matrix, in which H_{ij} denotes the count of node falling into (i, j) state.

Algorithm 3. Graph2Imag Algorithm

```
Input:
    G(V,E)
    Graph node representation, obtained via Node2Vec
Output:
    Two matrices as image

    1) representations of nodes are gathered in a matrix of dimension $N \times d$, where $N = |V|$
    2) Apply PCA method to transform the $d-$dimension vector of node $v_i$ into $dPCA-$dimension vector LPCA
           $LPCA_i = \{f_{1i}, f_{2i}, \dots, f_{dPCA,i}\} \qquad 1 \le i \le N, dPCA < d$
    3) Build $M_{12}$ from components 1 and 2 of $f$
           $M_{12} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1,N} \\ f_{21} & f_{22} & \cdots & f_{2,N} \end{bmatrix}$
    4) Build $M_{34}$ from components 3 and 4 of $f$
           $M_{34} = \begin{bmatrix} f_{31} & f_{32} & \cdots & f_{3,N} \\ f_{41} & f_{42} & \cdots & f_{4,N} \end{bmatrix}$
    5) Digitize elements of $M_{12}$ and $M_{34}$ to $r$ equally sized levels, i.e., $r * r$ bins
    6) Obtain 2D histogram ($H_{12}$ and $H_{34}$) of the ordered pairs, as a $r \times r$ matrix
    7) These two histograms are the Images
```

Among all dPCA*dPCA matrices M, could be built by this procedure, we just consider M_{12} and M_{34}, *i.e.*, taking into account just four first components of LPCA, which seems to be enough to analyze the brain network. As shown in Fig. 3 (Meng & Xiang, 2018) these two matrices, behaving like images, can be applied as different channels of DCNN. The algorithm pseudo-code is shown in Algorithm 3 (Meng & Xiang, 2018).

As another feature-based network embedding method, Anonymous walk Embeding (AWE) algorithm

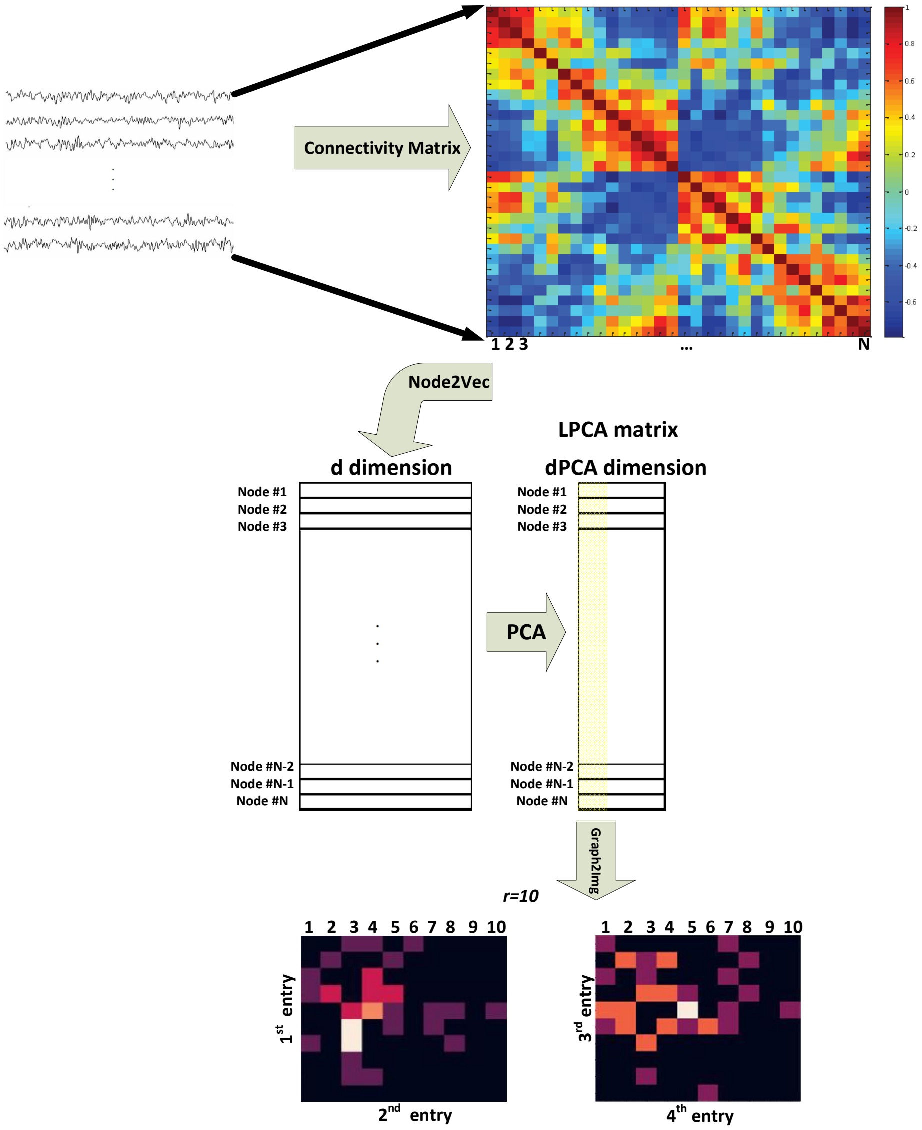

Figure 3. block diagram of the Graph2Img Feature-based embedding algorithm (Meng & Xiang, 2018)

used distribution of anonymous walks. Anonymous walks are the set of walks starting from an initial node u, by length l passing from random nodes, and termination at node v. There are a set of η such random walks $A_l^u = (a_1^u, a_2^u, \dots, a_\eta^u)$. Thus, number of all possible random walks with length l exponentially grows with l. These anonymous walks capture structural information of nodes, because labels of the nodes constituent of a random walk is omitted for them. In fact, corresponded to the random walk: $w = (v_1, v_2, \dots, v_k)$, we can define an anonymous walk involving sequence of integers $a = \left(f(v_1), f(v_2), \dots, f(v_k)\right)$ where $f(v)$ is the minimum place of v in the w random walk (Ivanov & Burnaev, 2018). However, due to huge number of anonymous walks of a large graph, an efficient sampling approach is required to approximate this distribution (Ivanov & Burnaev, 2018). Defining the objective function of similar nodes on local neighborhoods of anonymous walks, improve the structural consideration of the embedding method.

AWE is defined on weighted directed graphs. Given a weighted graph $G = (V, E, A)$, a random walk graph could be constructed $R = (V, E, P)$ in which every edge $e = (u, v)$ has a weight p_e (Ivanov & Burnaev, 2018). A random pair of nodes (u_i, u_{i+1}) can be led to a random walk with probability of $p_e(u_i, u_{i+1})$. Accordingly, it is possible to attribute $p(w)$ to each random walk; as the multiplication of probability of choosing total pairs in that walk, *i.e.*, $p(w) = \prod_{e \in w} p_e$.

Starting from each node, we can obtain marginal probability of selecting a special walk according to

$p(w)$. A probability of seeing anonymous walk a_i^u of length l for a node u is $p(a_i^u) = \sum_{\substack{w\epsilon W^u \\ w\mapsto a_i^l}} p(w)$. where W^u is the set of all walks starting from u, and $w \mapsto a_i^l$ involves the walks with length l. Anonymous walk embedding of this set of all possible anonymous walks of length l of graph G is a vector whose ith component is the probability of that walk. Aggregating probabilities across all vertices in a graph and normalizing them by the total number of nodes N, we get the probability of choosing anonymous walk a_i in graph G:

$$p(a_i) = \frac{1}{N}\sum_{u\in G} p(a_i^u) \ = \frac{1}{N}\sum_{u\in G}\ \sum_{\substack{w\in W_l^u \\ w\rightarrow a_i}} p(w)$$

AWE algorithm includes computation the probability of all random walks (Ivanov & Burnaev, 2018). Since direct computation of AWE requires determining all different random walks in graph G, grow exponentially with the number of steps l, the AWE algorithm is as follows (Ivanov & Burnaev, 2018):

1) Beginning from each node u, sample N random walks starting from u, and map it to an anonymous random walk.
2) Gather all these sampled anonymous random walks for all nodes. These are collection of co-occurring walks.
3) Learn representation vector of these walks, and generate a $\eta * da$ dimensional matrix (W), where η is the number of all possible anonymous walks of length l, and da is the embedding size. This model is used to predict a target vector for anonymous walks in a graph.
4) For a graph vector d, according to sampled co-occurred anonymous walks in W, the model calculates a probability function to predict a target walk among all sampled anonymous walks
5) Now model updates the matrix W and graph vector d via gradient backpropagation, as schematically is shown in Fig. 4.
6) After repeating steps 4 and 5, a learned graph vector d is called anonymous walk embedding.

These four embedding algorithms, Node2Vec, DeepWalk, AWE, and Graph2Img, extract the feature vectors of each node, describing the characteristics and structure of the graph. Thus, the next step of our research is classification of these feature vectors obtained for healthy and ASD subjects.

3. Classification

Graph classification is a task to predict whether a whole graph belong to any class of C pre-defined classes. In other words, the task is to train a classifier based on N graphs $\{G_i\}, i = 1{:}N$ and their corresponding labels $\{L_i\}, i = 1{:}N$, able to classify every new graph $G \rightarrow L$. Graph classification problem can be done using two typical approaches: (1) classification using extended CNNs to be appropriate for the raw graphs (Niepert et al., 2016) and (2) graph kernel methods (Shervashidze et al., 2011), in which graph embeddings $f(G_1)$ are used in conjunction with kernel methods ($K\big(f(G_1), f(G_2)\big)$) to perform classification of new graphs, where $K{:}\,(x, y) \ \rightarrow \ R^n$ is a kernel function, quantifying distance of graphs.

As mentioned earlier, the aim of this paper is kernelized classification of healthy and autistic patients based on functional connectivity matrices. The features extracted from these matrices ($f(G_1)$) are the embedded vectors obtained by using Node2vec, Struct2vec, AWE, and Graph2Img algorithms. To do classification job, we used the DNN classifier. The reason underlying this selection is size of the resultant feature vectors, whose classification requires many parameters to be trained. As well, to validate the performance of our classification task, cross-validation is applied.

Three types of deep network have been considered in this study: LeNet, ResNet, and VGG16. However, finally, we have used LeNet, because of its best performance for our problem. Thus, we just describe it here.

3.1 LeNet

LeNet, one of the first published CNNs in computer vision tasks, was introduced by (and named for) Yann LeCun. In 1989, LeCun published the first study in which he could train CNNs via backpropagation. Then this network was applied in AT&T Bell Labs, for the purpose of recognizing handwritten digits in images (LeCun et al., 1998). LeNet achieved outstanding results comparable with that of support vector machines, and thus became a dominant approach in supervised learning.

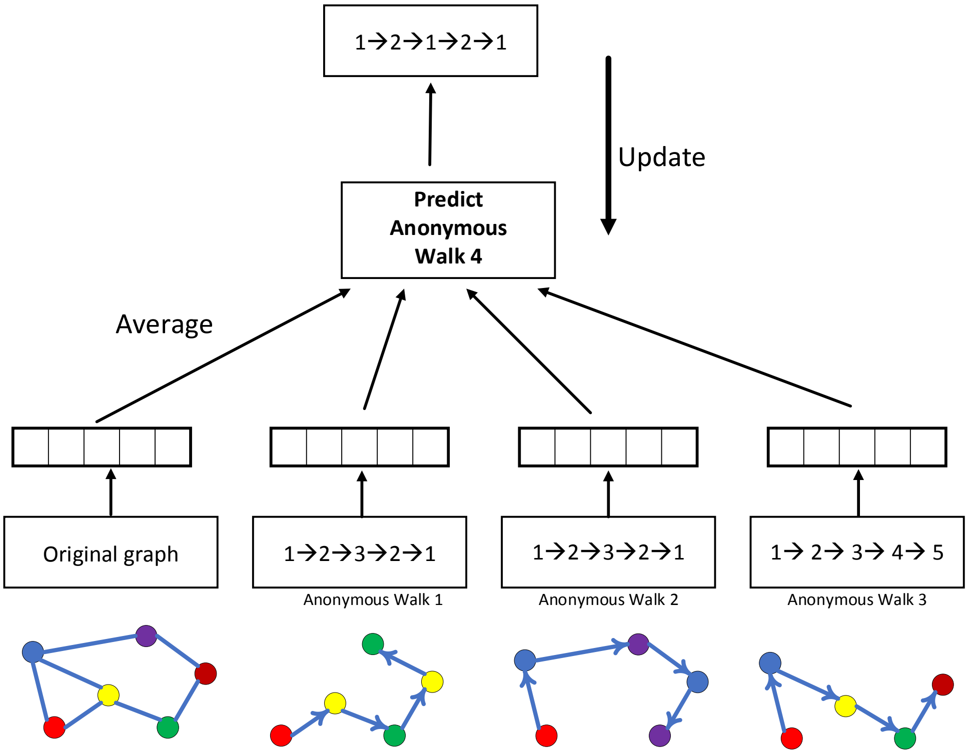

Figure 4. Schematic illustration of learning structure via anonymous random walks (Ivanov & Burnaev, 2018)

LeNet (LeNet-5) consists of two parts (G. Wang & Gong, 2019): *(i)* a convolutional encoder, and *(ii)* a dense block. The former consists of two convolutional blocks, and the latter consists of three fully-connected layers; The architecture is summarized in Fig. 5.

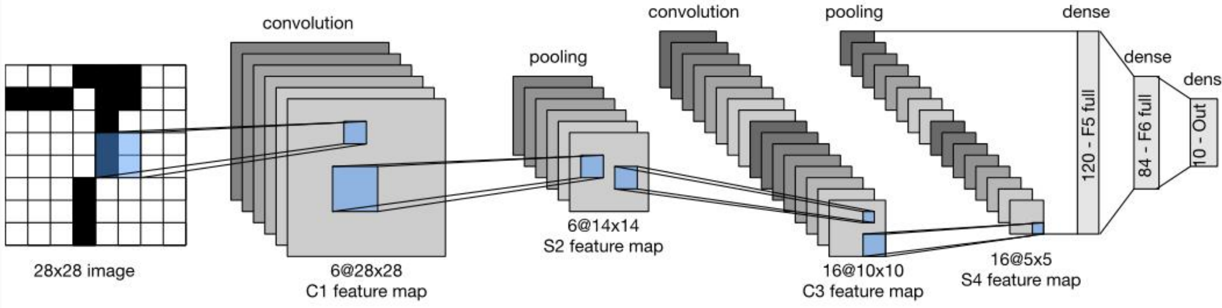

Figure 5. Data flow in LeNet. The input is an image, the output a probability over different possible outcomes (Loey et al., 2016).

Each convolutional block includes a convolutional layer, a sigmoid activation function, and a subsequent average pooling operation. In 1990 ReLUs and max-pooling were discovered to have suitable performance. However, in LeNet, each convolutional layer maps any 5×5 part of the input to a scalar using a kernel and a sigmoid activation function. There is 6 convolutional layers, in such a way that the result is a 6@28*28 tensor. In fact, by these convolutional layers, spatial features of input are mapped to a number of two-dimensional feature maps, namely, channels. Then, a pooling layer sample the channels by a factor of 2, and lead to a 6@14*14 array. Then there is another convolutional layer. Again, this is a convolutional layer with 5*5-dimensional filter. First convolutional layer had 6 output channels, while this second layer has 16 outputs of size 10*10. The output of the convolutional block must be flattened before passed to the dense block. This output is a 16@5*5 vector, created by a pooling layer.

LeNet's dense block has three fully-connected layers, with 120, 84, and 10 outputs, respectively. Because we are still performing classification, the 10-dimensional output layer corresponds to the number of possible output classes. Implementing LeNet models with modern deep learning frameworks is remarkably simple.

4. Implementation and Results

4.1 ABIDE dataset

The rs-fMRI data of ASD and healthy subjects are downloaded from a large multisite data repository ABIDE (Autism Brain Imaging Data Exchange) (Http://Fcon_1000.Projects.Nitrc.Org/Indi/Abide/). The Autism Brain Imaging Data Exchange I (ABIDE I) is a multisite platform gathered from 17 international laboratories, shared some collected resting state functional magnetic resonance imaging (rs-fMRI), anatomical and phenotypic datasets. This dataset includes 1112 patients, from 539 individuals with ASD and 573 from typical controls (ages 7-64 years, median 14.7 years across groups). Till now, this data is used in many researches. The publications have shown its utility for capturing whole brain and regional properties of the brain connectome in Autism Spectrum Disorder (ASD). All data have been anonymized.

Accordingly, ABIDE II was established to further promote discovery science on the brain connectome in ASD. To date, ABIDE II involves 19 sites, overall donating 1114 datasets from 521 individuals with ASD and 593 controls (age range: 5-64 years). All datasets are anonymous, with no protected health information included.

There is not ASD/healthy label for some individuals present in ABIDE database. After removing these cases, 871 individuals of ABIDE I and 910 individuals of ABIDE II would be remained, for investigation in this study (X. Yang et al., 2019).

4.2 Preprocessing and Connectivity Matrix

The rs-fMRI data are slice time corrected, motion corrected, registered and normalized, using FSL software. The steps of preprocessing done for ABIDE I and ABIDE II databases are: 1) AC-PC Realignment, 2) Gray Matter, and White Matter Tissue Segmentation, 3) Non-linear registration to MNI152 space, 4) Normalization, 5) Resampling, 6) modulation, and 7) Smoothing with FWMH=4mm. For the task of brain parcellation, the ICA method is used (F. de Martino et al., 2007; Joel et al., 2011; Smith et al., 2009; Tohka et al., 2008). In other words, instead of obtaining average of the time series (BOLD signal) of some pre-defined regions, spatial maps output from ICA with the specific functional and anatomical interpretation (the locations of brain tissue acting synchronously and with the same activity pattern) are taken into account. ICA is a data-driven model which uses no a priori information about the brain and has been a popular approach in the analysis of fMRI data (Salimi-Khorshidi et al., 2014). In this study ICA decomposed the whole BOLD fMRI data into 39 regions according to MNI ATLAS.

Afterward, the BOLD signal of these 39 ROI is considered to compute their connectivity measures, by the statistical measures such as Pearson correlation, partial correlation (Saad et al., 2009) and tangent correlation (Dadi et al., 2019). The size of connectivity matrix is 39*39, according to number of ROIs. The Pearson correlation coefficient ranges from 1 to -1. 1 indicates that two ROIs are highly correlated, -1 indicates that two ROIs are anti-correlated. This step is done using Nilearn toolbox developed by MIT university, and as well by BrainIAK toolbox (Kumar et al., 2020). Nilearn is a python toolbox for statistical learning on neuroimaging data. In this study, connectivity matrix is obtained via tangent correlation (Pedregosa et al., 2011). See Appendix A for more details. This method is less frequently used, but has solid mathematical foundations and a variety of groups have reported good decoding performances with this framework. Connectivity matrices built with tangent space parametrization give an improvement compared to full or partial correlations.

4.3 Classifying the Graph Embedding Vectors

According to the abovementioned embedding features, we used three scenarios to check whether ASD detection can be improved by graph embedding algorithms or not. In first scenario, features are embedded vectors of the connectivity matrix using each of Node2Vec, Struc2Vec, and AWE method. Accordingly, a deep network with one channel input is used in this scenario. This channel input is an $d \times N$ matrix including the d-dimensional embedded vectors of all $N = 39$ nodes. The embedded vectors obtained by these methods has a dimension of $d = 25$, 64, and 128 respectively. But, in the second scenario, to take different properties of Node2vec algorithm, with different p and q values ($p = 1, q = 1, p = 1, q = 4$, and $p = 4, q = 1$), into account, a 3-channel deep network is applied. In the third scenario, after applying PCA on the result of Node2Vec algorithm, two matrices of the Graph2Img algorithm are considered as input of a 2-channel CNN. These three scenarios are schematically shown in Figure 7.

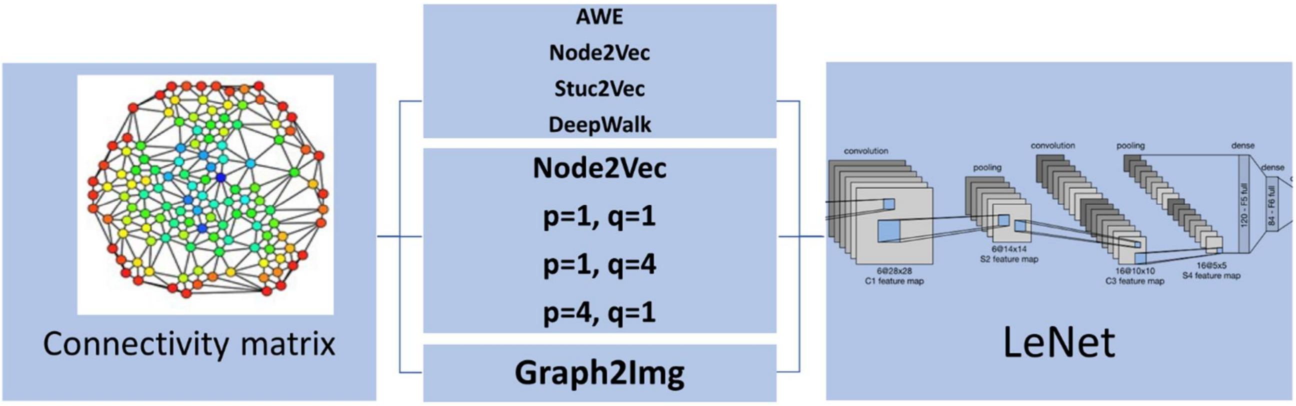

Figure 7. Three Scenarios of applying embedding vectors to detect ASD via LeNet

Indeed, at first, we tried to do the classification job through traditional kernel-base classifiers, like Support Vector Machine (SVM), but satisfactory results could not be obtained. The classifier could not show an accuracy better than chance. The advantage of CNN is that it is composed of an automatic feature extractor, that again extract features from the embedded vectors and, thus is a trainable classifier.

In all three scenarios, we customize the LeNet structure for our problem: In the 1st scenario, there are one channel in the input layer, and the size of the embedded vector in each of Node2vec, Struct2vec, and AWE method determines dimensional of the input. These sizes are respectively equal to 25, 64, and 128.

Thus, in the 1st scenario, the input layer of LeNet is a $39 * d, d = 25,64,128$ image. In the 2nd scenario the network has three channels. Each channel of the deep network consists of $39 * d$ neurons. In the third scenario, there are two channels, each consisting 10*10 neurons ($r = 10$). Lastly, in all three scenarios, there are two output neurons indicating healthy and autism brain.

The default LeNet network was modified according to the abovementioned dimensions of input/output. Furthermore, a dropout layer is employed for regularization at every hidden layer [33] with 0.8 keeping regularity. Another difference is the activation functions we used in LeNet to be ReLU functions, except for the ultimate layer, which uses a softmax function in such a way that a probability distribution over classes would be obtained. For the convolution-pooling block, we employ 64 filters at the first level, and as the signal is halved through the (2,2) max pooling layer, the number of filters in the subsequent convolutional layer is increased to 96 to compensate for the loss in resolution (Tixier et al., 2019). The number of trainable weights in this deep neural network doubles or triples in the 3rd and 2nd scenario.

The illustrated networks are used as the healthy/ASD classifier. Classification results would be reported in section 5 to compare them with previous researches in which deep network is used to classify the raw connectivity matrices.

4.4 Evaluation

To check the performance of our proposed ASD classifier working based on graph embedding techniques and deep machine learning methods, two kinds of cross-validation techniques are used. Indeed, these two techniques depend on how we choose training and test datasets. According to the properties of ABIDE database that consists of different sites we can do three different partitioning jobs: 1) dividing data of each site into N folds, and report accuracy of classification in individual sites, 2) Leave-One-Site-Out validation (distinctly for ABIDE I and II), 3) dividing all data of ABIDE I and II into N folds to report a typical N-fold cross validation. In all three approaches, the classification performance is assessed by accuracy, F-score, recall and precision. To report the accuracy of all data, statistically more reliable, 2nd approach, i.e., Leave-One-Site-Out validation, is the most appropriate one. However, in this paper, the validation type (2) and (3) are considered in the report of the results.

Considering ASD detection as the goal of classifier, True Positive (TP) is defined as the percent of ASD subjects correctly classified as ASD. As well, the percent of ASD subjects, classified as healthy is referred to as False Negative (FN). Similarly, False Positive (FP) is the percent of healthy subjects decided to be ASD. Accordingly, the F-score measure is defined as follows:

$$F-socre=\frac{TP}{TP+\frac{1}{2}(FP+FN)}$$

It is important for the classifier to detect all ASD subjects, so $\frac{TP}{\#ASD\ subjects}$ is referred to as recall. Also, it is expected a classifier to have trusted positive detection, or in other word to be precise. Thus, precision is defined as $\frac{TP}{TP+FP}$. Because the size of subjects of two classes are not necessarily balanced, precision is a better measure of performance. Accordingly, another definition of F-score is based on recall, and precision:

$$F-socre=\frac{2\times recall\times precision}{recall+precision}$$

At last, to check how many subjects are correctly labelled, accuracy is a well-known measure.

$$Accuracy=\frac{TP+TN}{all\ subjects}$$

On the other hand, the time cost of training the classifier is another measure of the method under evaluation.

5. Results

Results of three scenarios for ABIDEI and ABIDE II database are presented in Table 1, using LeNet classifier. In the results of Table 1 validation of type 3 is considered: all subjects of each database are taken into account, then 5-fold and 10-fold cross-validation is applied. The average accuracy of these folds is reported for each scenario. Scenario 2 achieved the best performance in which a mean classification accuracy of 64% (recall 0.77%, precision 0.73%), and 66% (recall 80%, precision 80%) is obtained for ABIDE I and ABIDE II, respectively (in 10-fold cross-validation). The range of accuracy values were between 52% and 69% in individual folds. Based on the literature, this is not better than (Sherkatghanad et al., 2020) in which 70.22% accuracy is reported.

The results of Table 1 show that the type of embedded features is effective in classification. But, as mentioned before, not given here, the results of SVM using embedded features is not better than those of (Sherkatghanad et al., 2020), in which raw connectivity matrix has been used in classification. In other words, it seems that it is the art of deep network classifier in reaching a good separation between ASD and healthy subjects, not the embedding features. So, the question is that whether feature embedding method was effective in ASD/healthy discrimination or not.

Table 1. 5 and 10-fold cross-validation results using different embedding methods and CNN classifier (LeNet)

	Method	ABIDE II (5-fold)	ABIDE I (5-fold)	ABIDE II (10-fold)	ABIDE I (10-fold)
Scenario 1	Struct2vec	54%	56%	56%	58%
Scenario 1	DeepWalk	55%	55%	56%	59%
Scenario 1	Node2vec $p = 1, q = 4$	59%	57%	62%	62%
Scenario 1	AWE	56%	58%	63%	65%
Scenario 2	Node2vec	63%	64%	66%	64%
Scenario 3	Grah2Img	59%	61%	66%	64%

On the other hand, results of the leave-one-site-out cross validation is reported in Tables 2 to 5, respectively for the scenario 1 using AWE, scenario 2 (node2vec), and scenario 3 (Graph2Img). In this validation type, just AWE of scenario 1 is applied, due to its better performance in the k-fold cross validation procedure, against other embedding techniques. For each site, the LeNet CNN classifier is trained by data of other sites in each database, and has been tested on data of that site. Results of the ABIDEI and ABIDEII are distinctly presented. Number of the subjects of each site, number of the ASD subjects, accuracy and F-score of the proposed techniques, as well as those of (Heinsfeld et al., 2018; Sherkatghanad et al., 2020) (just for ABIDE I) are reported in the tables.

As shown in Figure 8, these results in compared to (Heinsfeld et al., 2018; Sherkatghanad et al., 2020), that achieved the best results in the literature so far, depicts that the embedding-based CNN can outperform the other supervised methods. From this point of view, these results are in favor of embedding features, not just the Deep Network.

Therefore, some points worth considering in the results of these two validation methods:

1. Embedding features could not improve the results of the k-fold cross validation, but is able to improve the results of leave-one-site-out one.
2. The accuracy of CNN in classifying ASD subjects of each site is different when using graph embedding methods. In average, all embedding scenarios could improve the results, in comparison to using the raw connectivity matrices, in the leave-site-out validation manner (Heinsfeld et al., 2018; Sherkatghanad et al., 2020). The best embedding technique seems to be Graph2Img that increase the 65% (Heinsfeld et al., 2018) and 63% (Sherkatghanad et al., 2020) results to 80%.
3. Each graph embedding scenario has significantly improved the results of some sites, but not the all of sites. The AWE technique is not successful about the fMRI data of University of Utah School of Medicine (USM), for which (Sherkatghanad et al., 2020) acts well. As well (Sherkatghanad et al., 2020) and/or (Heinsfeld et al., 2018) outperforms the embedding scenario 2 (three channel node2vec with three values of p and q) in Kennedy Krieger Institute, Baltimore (KKI) data, New York University Langone Medical Center (NYU), Ludwig Maximilian University Munich (MAX-MUN), USM, and San Diego State University (SDSU) data. Even, for the embedding scenario 3 (i.e., Graph2Img) there is a site for which the accuracy of ASD classification is lower than (Sherkatghanad et al., 2020).
4. For the ABIDE II database, scenario 1 (AWE method) reached the best mean accuracy. The best individual site result, also is dedicated to AWE method for the KKI database.

However, the most dominant advantage of the proposed algorithm is its training time. Using a system with two Intel Xeon E5-2620 processors with 24 cores running at 2 GHz and 48 GB of RAM. As well, 1 Tesla K40 GPU with 2880 CUDA cores and 12 GB of RAM was used to accelerate training. In such a way, the entire training time took about 200 minutes. In Table 5 the training time of (Heinsfeld et al., 2018; Sherkatghanad et al., 2020) and our proposed method is compared. This achievement is due to the dimension reduction property of the graph embedding methods, decreasing the dimensionality of the CNN input.

The results show that the proposed algorithm, using embedded vectors of connectivity graph, and the CNN classifier, outperforms the previous studies in identification of autism spectrum disorder, from both speed and accuracy point of view.

Table 2. Leave-site-out cross-validation results using scenario 1 just by AWE and CNN classifier

	Sites	**#subjects**	**# ASD subjects**	**Accuracy**	**F-score**	**Accuracy** (Heinsfeld et al., 2018)	**F-score** (Heinsfeld et al., 2018)	**Accuracy** (Sherkatghanad et al., 2020)	**F-score** (Sherkatghanad et al., 2020)
ABIDE I	**UCLA-2**	**21**	**11**	**0.90**	**0.86**	**0.66**	**0.65**	**0.61**	**0.62**
	TRINITY	**49**	**25**	**0.89**	**0.88**	**0.65**	**0.64**	**0.61**	**0.62**
	UM-2	**35**	**13**	**0.79**	**0.73**	**0.64**	**0.66**	**0.66**	**0.56**
	KKI	**33**	**12**	**0.92**	**0.91**	**0.67**	**0.66**	**0.72**	**0.69**
	YALE	**41**	**22**	**0.71**	**0.61**	**0.64**	**0.63**	**0.69**	**0.65**
	PITT	**50**	**24**	**0.76**	**0.73**	**0.66**	**0.65**	**0.69**	**0.73**
	OLIN	**28**	**14**	**0.77**	**0.73**	**0.64**	**0.63**	**0.58**	**0.56**
	LEUVEN-2	**28**	**12**	**0.79**	**0.76**	**0.65**	**0.64**	**0.65**	**0.73**
	STANFORD	**25**	**12**	**0.69**	**0.62**	**0.66**	**0.65**	**0.48**	**0.09**
	NYU	**172**	**74**	**0.76**	**0.73**	**0.66**	**0.65**	**0.65**	**0.73**
	UM-1	**86**	**14**	**0.71**	**0.67**	**0.64**	**0.63**	**0.66**	**0.56**
	UCLA-1	**64**	**37**	**0.74**	**0.72**	**0.66**	**0.65**	**0.69**	**0.65**
	OHSU	**25**	**12**	**0.82**	**0.77**	**0.64**	**0.64**	**0.57**	**0.56**
	MAX-MUN	**46**	**19**	**0.79**	**0.78**	**0.68**	**0.67**	**0.46**	**0.48**
	LEUVEN-1	**28**	**14**	**0.65**	**0.61**	**0.65**	**0.64**	**0.65**	**0.71**
	USM	**67**	**43**	**0.71**	**0.68**	**0.64**	**0.63**	**0.77**	**0.69**
	SBL	**26**	**12**	**0.72**	**0.67**	**0.66**	**0.65**	**0.56**	**0.62**
	SDSU	**36**	**14**	**0.86**	**0.81**	**0.63**	**0.63**	**0.75**	**0.80**
	mean			**0.77**	**0.73**	**0.65**	**0.64**	**0.63**	**0.61**
ABIDE II	BNI	56	29	**0.79**	0.77				
	EMC	54	25	**0.83**	0.82				
	ETH	37	13	**0.82**	0.65				
	GU	106	51	**0.72**	0.71				
	IP	54	21	**0.81**	0.80				
	IU	40	20	**0.91**	0.88				
	KKI	211	56	**0.85**	0.84				
	NYU	77	48	**0.65**	0.61				
	OHSU	93	37	**0.72**	0.69				
	ONRC	59	24	**0.79**	0.79				
	SDSU	58	33	**0.79**	0.73				
	SU	37	21	**0.78**	0.75				
	TCD	42	21	**0.71**	0.69				
	UCD	31	18	**0.74**	0.68				
	UCLA	32	16	**0.85**	0.82				
	USM	33	17	**0.82**	0.79				
	mean			**0.78**	0.75				

Table 3. Leave-site-out cross-validation results using scenario 2 (Node2vec) and CNN classifier

	Sites	**#subjects**	**# ASD subjects**	**Accuracy**	**F-score**	**Accuracy** (Heinsfeld et al., 2018)	**F-score** (Heinsfeld et al., 2018)	**Accuracy** (Sherkatghanad et al., 2020)	**F-score** (Sherkatghanad et al., 2020)
ABIDE I	**UCLA-2**	**21**	**11**	**0.76**	**0.68**	**0.66**	**0.65**	**0.61**	**0.62**
	TRINITY	**44**	**25**	**0.72**	**0.64**	**0.65**	**0.64**	**0.61**	**0.62**
	UM-2	**34**	**13**	**0.71**	**0.63**	**0.64**	**0.66**	**0.66**	**0.56**
	KKI	**33**	**12**	**0.66**	**0.64**	**0.67**	**0.66**	**0.72**	**0.69**
	YALE	**41**	**22**	**0.81**	**0.76**	**0.64**	**0.63**	**0.69**	**0.65**

	PITT	**50**	**24**	**0.76**	**0.74**	**0.66**	**0.65**	**0.69**	**0.73**
	OLIN	**28**	**14**	**0.87**	**0.83**	**0.64**	**0.63**	**0.58**	**0.56**
	LEUVEN-2	**28**	**12**	**0.78**	**0.70**	**0.65**	**0.64**	**0.65**	**0.73**
	STANFORD	**25**	**12**	**0.82**	**0.82**	**0.66**	**0.65**	**0.48**	**0.09**
	NYU	**172**	**74**	**0.63**	**0.58**	**0.66**	**0.65**	**0.65**	**0.73**
	UM-1	**86**	**14**	**0.67**	**0.63**	**0.64**	**0.63**	**0.66**	**0.56**
	UCLA-1	**64**	**37**	**0.65**	**0.54**	**0.66**	**0.65**	**0.69**	**0.65**
	OHSU	**25**	**12**	**0.77**	**0.72**	**0.64**	**0.64**	**0.57**	**0.56**
	MAX-MUN	**46**	**19**	**0.60**	**0.53**	**0.68**	**0.67**	**0.46**	**0.48**
	LEUVEN-1	**28**	**14**	**0.85**	**0.82**	**0.65**	**0.64**	**0.65**	**0.71**
	USM	**67**	**43**	**0.76**	**0.69**	**0.64**	**0.63**	**0.77**	**0.69**
	SBL	**26**	**12**	**0.76**	**0.70**	**0.66**	**0.65**	**0.56**	**0.62**
	SDSU	**27**	**14**	**0.73**	**0.65**	**0.63**	**0.63**	**0.75**	**0.80**
	mean			**0.73**	**0.68**	**0.65**	**0.64**	**0.63**	**0.61**
ABIDE II	BNI	56	29	0.66	0.60				
	EMC	54	25	0.80	0.77				
	ETH	34	13	0.74	0.65				
	GU	106	51	0.64	0.59				
	IP	54	21	0.74	0.69				
	IU	34	20	0.75	0.67				
	KKI	34	56	0.79	0.70				
	NYU	77	48	0.68	0.61				
	OHSU	93	37	0.63	0.56				
	ONRC	49	24	0.80	0.75				
	SDSU	58	33	0.67	0.58				
	SU	54	21	0.70	0.64				
	TCD	39	21	0.76	0.74				
	UCD	31	18	0.77	0.72				
	UCLA	32	16	0.80	0.76				
	USM	33	17	0.70	0.64				
	mean			0.72	0.66				

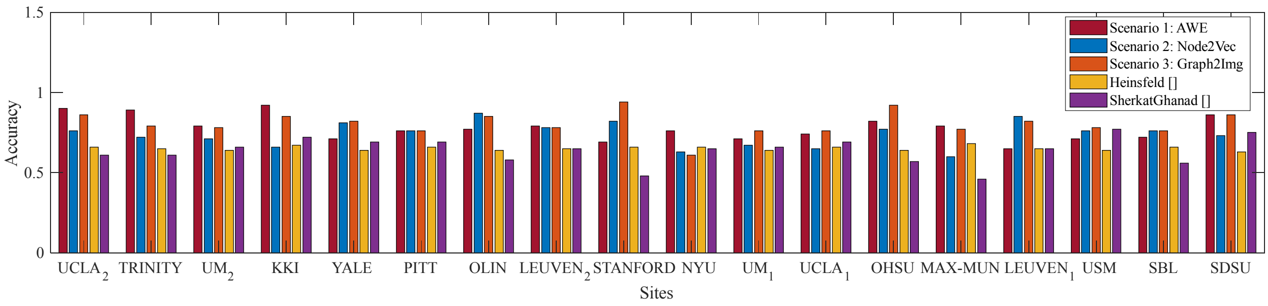

Figure 8. Box plot of leave-out-site accuracy to compare different embedding scenarios and (Heinsfeld et al., 2018; Sherkatghanad et al., 2020) vs. sites, for ABIDE I dataset.

Table 4. Leave-site-out cross-validation results using scenario 3 (Graph2Img) and CNN classifier

	Sites	#subjects	# ASD subjects	Accuracy	F-score	Accuracy (Heinsfeld et al., 2018)	F-score (Heinsfeld et al., 2018)	Accuracy (Sherkatghanad et al., 2020)	F-score (Sherkatghanad et al., 2020)
ABIDE I	**UCLA-2**	**21**	**11**	**0.86**	**0.82**	**0.66**	**0.65**	**0.61**	**0.62**
	TRINITY	**44**	**25**	**0.79**	**0.75**	**0.65**	**0.64**	**0.61**	**0.62**
	UM-2	**34**	**13**	**0.78**	**0.72**	**0.64**	**0.66**	**0.66**	**0.56**
	KKI	**33**	**12**	**0.85**	**0.81**	**0.67**	**0.66**	**0.72**	**0.69**
	YALE	**41**	**22**	**0.82**	**0.81**	**0.64**	**0.63**	**0.69**	**0.65**
	PITT	**50**	**24**	**0.76**	**0.72**	**0.66**	**0.65**	**0.69**	**0.73**
	OLIN	**28**	**14**	**0.85**	**0.81**	**0.64**	**0.63**	**0.58**	**0.56**
	LEUVEN-2	**28**	**12**	**0.78**	**0.72**	**0.65**	**0.64**	**0.65**	**0.73**

	STANFORD	**25**	**12**	**0.94**	**0.92**	**0.66**	**0.65**	**0.48**	**0.09**
	NYU	**172**	**74**	**0.61**	**0.58**	**0.66**	**0.65**	**0.65**	**0.73**
	UM-1	**86**	**14**	**0.76**	**0.71**	**0.64**	**0.63**	**0.66**	**0.56**
	UCLA-1	**64**	**37**	**0.76**	**0.72**	**0.66**	**0.65**	**0.69**	**0.65**
	OHSU	**25**	**12**	**0.92**	**0.90**	**0.64**	**0.64**	**0.57**	**0.56**
	MAX-MUN	**46**	**19**	**0.77**	**0.71**	**0.68**	**0.67**	**0.46**	**0.48**
	LEUVEN-1	**28**	**14**	**0.82**	**0.78**	**0.65**	**0.64**	**0.65**	**0.71**
	USM	**67**	**43**	**0.78**	**0.73**	**0.64**	**0.63**	**0.77**	**0.69**
	SBL	**26**	**12**	**0.76**	**0.71**	**0.66**	**0.65**	**0.56**	**0.62**
	SDSU	**27**	**14**	**0.86**	**0.76**	**0.63**	**0.63**	**0.75**	**0.80**
	mean			**0.80**	**0.76**	**0.65**	**0.64**	**0.63**	**0.61**
EMCABIDE II	BNI	56	29	**0.66**	**0.60**				
	EMC	54	25	**0.80**	**0.77**				
	ETH	34	13	**0.74**	**0.65**				
	GU	106	51	**0.64**	**0.59**				
	IP	54	21	**0.74**	**0.69**				
	IU	34	20	**0.75**	**0.67**				
	KKI	34	56	**0.79**	**0.70**				
	NYU	77	48	**0.66**	**0.61**				
	OHSU	93	37	**0.63**	**0.56**				
	ONRC	49	24	**0.80**	**0.75**				
	SDSU	58	33	**0.67**	**0.58**				
	SU	54	21	**0.70**	**0.64**				
	TCD	39	21	**0.76**	**0.74**				
	UCD	31	18	**0.77**	**0.72**				
	UCLA	32	16	**0.80**	**0.76**				
	USM	33	17	**0.70**	**0.64**				
	Mean			**0.72**	**0.66**				

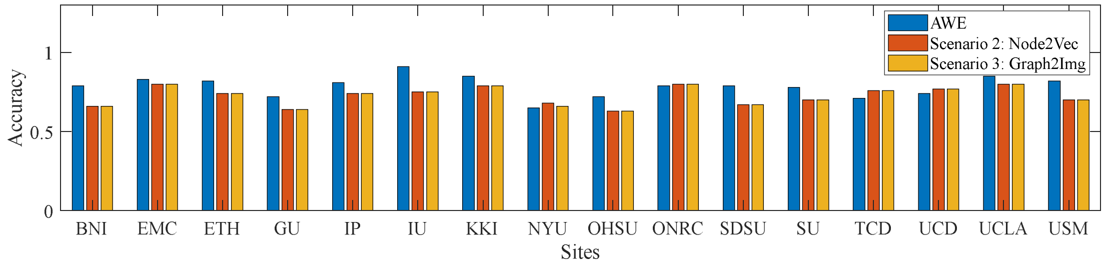

Figure 9. Box plot of leave-out-site accuracy to compare different embedding scenarios vs. sites, for ABIDE II dataset.

Table 5. Summary of best performance values and computational time for ABIDE I, in compared to literature.

	Mean Accuracy (10-fold cross-validation)	**Mean Accuracy (Leave-one-site-out)**	**Computation Time**
Heinsfeld (Heinsfeld et al., 2018)	70%	65%	Over 32h
SherkatGhanad (Sherkatghanad et al., 2020)	70.22%	63%	12h 30m
Proposed method	66%	80%	2h 47m 20s

6. Conclusion and Discussion

Since the functioning of the brain is accompanied by interactions and connection between different functional areas, discrimination of healthy and autism behaviors could be done by assessment of the brain network dynamics (Kim et al., 2017). Indeed, cognitive disorders emerges because of the alteration of dynamic relationships between pairs of specific brain regions. However, we claim that a powerful learning method considering the coupling, similarity, or causality and synchronizing intensity between specific brain regions could be able to detect cognitive impairment.

For a complete comparison, we considered the literature where functional connectivity matrix is used to discriminant healthy and autistic subjects, based on ABIDE database, either using conventional classifiers, or deep networks, or even statistical tests. The result of these comparisons is shown in Figure 8, illustrating that embedded features achieved better results than other feature extraction methods, and, as well, deep neural networks hold much greater promises than conventional classifiers. In other words, since AWE (Scenario 1), Graph2Img (Scenario 3) and Multi-parameter Node2Vec (Scenario 2) algorithms gain better classification results with CNN classifier (in leave-one-site-out validation), we claim that embedded features involving structure of the functional connectivity of brain could be more convenient in ASD detection. The classification results (in k-fold cross validation), although not high (66% and 64% for ABIDEI and II) enough to be appropriate for clinical usage, show that there are strong alterations in brain connections during autism disorder.

Our 10-fold cross-validation best average result is 65% in compared to (Heinsfeld et al., 2018) that is about 65%. Instead, as shown in Tables 2 to 4, and Figure 8, for leave-one-site-out, both the mean accuracy over sites, and the most of individual accuracy of sites, our proposed method is clearly much better than (Heinsfeld et al., 2018) and (Sherkatghanad et al., 2020). These results show that, as the sample size decreases (5-fold, 10-fold cross-validation results, and leave-one-site-out), the gap between performance of the embedding vectors, and the raw connectivity matrix increase. This implies that using embedding vectors is an effective idea, but still needs more investigation to find the more suitable graph representation method. The reason is clearly the intrinsic complexity of brain function.

There are two messages in the obtained results: First, the intrinsic phenotypical properties of subjects within each site, lead to a specific structure in their connectivity graph, in addition to the distinct indicator of ASD/healthy. Different embedding techniques acquire some of these properties. Second, a suitable combination of graph embedding techniques is the alternative approach to take all graph similarities in the ASD group regardless the phenotypes.

The better mean accuracy of leave-one-site-out validation technique compared to that of k-fold cross validation again tells us about the variance of the graph structures between sites due to the within site phenotypes. In such a way, in a random group, finding the common structures just relevant to ASD would be too difficult for an embedding technique. It is the main reason prevents the embedding techniques capture a better result than the raw connectivity matrices.

In this paper, we showed that using structural graph representation algorithms, it is possible to classify subject groups based on connectivity fingerprints of brain regions. However, we did not analyze the results to obtain knowledge about these alterations were of what kind, and where they occur. This is drawback of our proposed algorithm that we could not identify ROIs that alter connectivity strength values. In fact, the main point of a suitable embedding algorithm for brain network is that the representations they emerge would be neurobiologically plausible and meaningful. From this point of view, we can predict the mechanism and causes underlying an impaired brain network during mental disorders. This is our future concern.

7. Appendix: Computing the covariance tangent space

Nilearn Python library provides the tangent-space parametrization of covariance matrices. In this appendix it is described how this parametrization is computed, with required formulation (Dadi et al., 2019). The algorithm is made of two steps: First covariance matrix of BOLD signal of each subject is computed, and the group average matrix over all subjects are obtained. Second, this average matrix is used to transform covariance matrix to a space where ensures a well-conditioned connectivity matrix.

To estimate the average covariance matrix, the Ledoit and Wolf (Ledoit & Wolf, 2004) estimator is a good choice (Brier et al., 2015; Varoquaux & Craddock, 2013). This namely Frechet mean is calculated according to the geometry of covariance matrices (Pennec et al., 2006; Varoquaux et al., 2010), in order to minimize a cost function. This cost function is depicted in algorithm 3 of Fletcher and Joshi (Fletcher & Joshi, 2007). But, Euclidean mean with the following formula give almost the same performance.

$$\text{Euclidean mean}: \Sigma_\star = \frac{1}{n\_train} \sum_{i \in train} \Sigma_i$$

After computing this average matrix $\Sigma_\star$, covariance matrix should be transformed in the tangent-space representation. This is done by first normalizing the covariance matrix of each subject, and then mapping it to a suitable space.

1. After eigenvalue decomposing of $\Sigma_\star = U^T \Delta U$, where U contains eigenvectors and Δ is a diagonal matrix containing eigenvalues, we can compute the normalized matrix of each BOLD signal via

$$\tilde{\Sigma}_i \leftarrow U^T \Delta^{-1/2}\, U \Sigma_i U^T \Delta^{-1/2}\, U$$

2. After eigenvalue decomposing of $\tilde{\Sigma}_i = \tilde{U}^T \tilde{\Delta}_i \tilde{U}$, we can compute the matrix logarithm $logm\tilde{\Sigma}_i = \tilde{U}^T \log(\tilde{\Delta}_i) \tilde{U}$, where the logarithm is applied to the diagonal elements of $\tilde{\Delta}_i$.

Finally, the $logm\tilde{\Sigma}_i$ is the resulting matrix, and can be considered as the tangent connectivity matrix.

8. References

Almuqhim, F., & Saeed, F. (2021). ASD-SAENet: A Sparse Autoencoder, and Deep-Neural Network Model for Detecting Autism Spectrum Disorder (ASD) Using fMRI Data. *Frontiers in Computational Neuroscience*, *15*. https://doi.org/10.3389/fncom.2021.654315

Banka, A., & Rekik, I. (2019). Adversarial connectome embedding for mild cognitive impairment identification using cortical morphological networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11848 LNCS*. https://doi.org/10.1007/978-3-030-32391-2_8

Bessadok, A., Mahjoub, M. A., & Rekik, I. (2019). Symmetric dual adversarial connectomic domain alignment for predicting isomorphic brain graph from a baseline graph. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11767 LNCS*. https://doi.org/10.1007/978-3-030-32251-9_51

Brier, M. R., Mitra, A., McCarthy, J. E., Ances, B. M., & Snyder, A. Z. (2015). Partial covariance based functional connectivity computation using Ledoit-Wolf covariance regularization. *NeuroImage*, *121*. https://doi.org/10.1016/j.neuroimage.2015.07.039

Dadi, K., Rahim, M., Abraham, A., Chyzhyk, D., Milham, M., Thirion, B., & Varoquaux, G. (2019). Benchmarking functional connectome-based predictive models for resting-state fMRI. *NeuroImage*, *192*. https://doi.org/10.1016/j.neuroimage.2019.02.062

de Martino, F., Gentile, F., Esposito, F., Balsi, M., di Salle, F., Goebel, R., & Formisano, E. (2007). Classification of fMRI independent components using IC-fingerprints and support vector machine classifiers. *NeuroImage*, *34*(1). https://doi.org/10.1016/j.neuroimage.2006.08.041

di Martino, A., Yan, C. G., Li, Q., Denio, E., Castellanos, F. X., Alaerts, K., Anderson, J. S., Assaf, M., Bookheimer, S. Y., Dapretto, M., Deen, B., Delmonte, S., Dinstein, I., Ertl-Wagner, B., Fair, D. A., Gallagher, L., Kennedy, D. P., Keown, C. L., Keysers, C., … Milham, M. P. (2014). The autism brain imaging data exchange: Towards a large-scale evaluation of the intrinsic brain architecture in autism. *Molecular Psychiatry*, *19*(6). https://doi.org/10.1038/mp.2013.78

Donnat, C., Zitnik, M., Hallac, D., & Leskovec, J. (2018). Learning structural node embeddings via diffusion wavelets. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/3219819.3220025

Elder, J. H., Kreider, C. M., Brasher, S. N., & Ansell, M. (2017). Clinical impact of early diagnosis of autism on the prognosis and parent-child relationships. *Psychology Research and Behavior Management*, *10*, 283–292. https://doi.org/10.2147/PRBM.S117499

El Gazzar, A., Cerliani, L., van Wingen, G. and Thomas, R.M., (2019). Simple 1-D convolutional networks for resting-state fMRI based classification in autism. In *2019 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-6). IEEE.

Eslami, T., & Saeed, F. (2019). Auto-AsD-Network: A technique based on deep learning and support vector machines for diagnosing autism spectrum disorder using fMRI data. *ACM-BCB 2019 - Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*. https://doi.org/10.1145/3307339.3343482

Feng, Y., You, H., Zhang, Z., Ji, R., & Gao, Y. (2019). Hypergraph neural networks. *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*. https://doi.org/10.1609/aaai.v33i01.33013558

Fletcher, P. T., & Joshi, S. (2007). Riemannian geometry for the statistical analysis of diffusion tensor data. *Signal Processing*, *87*(2). https://doi.org/10.1016/j.sigpro.2005.12.018

Gao, H., & Ji, S. (2019). Graph U-nets. *36th International Conference on Machine Learning, ICML 2019*, *2019-June*.

Grover, A., & Leskovec, J. (2016). Node2vec: Scalable feature learning for networks. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*. https://doi.org/10.1145/2939672.2939754

Guo, Y., Ding, G., Liu, L., Han, J., & Shao, L. (2017). Learning to Hash with Optimized Anchor Embedding for Scalable Retrieval. *IEEE Transactions on Image Processing*, *26*(3). https://doi.org/10.1109/TIP.2017.2652730

Hamilton William L., Ying Rex, & Leskovec Jure. (2017). Representation Learning on Graphs: Methods and Applications. *Arxiv.Org*.

Heinsfeld, A. S., Franco, A. R., Craddock, R. C., Buchweitz, A., & Meneguzzi, F. (2018). Identification of autism spectrum disorder using deep learning and the ABIDE dataset. *NeuroImage: Clinical*, *17*. https://doi.org/10.1016/j.nicl.2017.08.017

http://fcon_1000.projects.nitrc.org/indi/abide/. (n.d.).

Huang, H., Hu, X., Han, J., Lv, J., Liu, N., Guo, L., & Liu, T. (2016). Latent source mining in FMRI data via deep neural network. *Proceedings - International Symposium on Biomedical Imaging*, *2016-June*. https://doi.org/10.1109/ISBI.2016.7493348

Ivanov, S., & Burnaev, E. (2018). Anonymous walk embeddings. *35th International Conference on Machine Learning, ICML 2018*, *5*.

Joel, S. E., Caffo, B. S., van Zijl, P. C. M., & Pekar, J. J. (2011). On the relationship between seed-based and ICA-based measures of functional connectivity. *Magnetic Resonance in Medicine*, *66*(3). https://doi.org/10.1002/mrm.22818

Kazeminejad, A., & Sotero, R. C. (2020). The Importance of Anti-correlations in Graph Theory Based Classification of Autism Spectrum Disorder. *Frontiers in Neuroscience*, *14*. https://doi.org/10.3389/fnins.2020.00676

Khosla, M., Setty, V., & Anand, A. (2021). A Comparative Study for Unsupervised Network Representation Learning. *IEEE Transactions on Knowledge and Data Engineering*, *33*(5). https://doi.org/10.1109/TKDE.2019.2951398

Kim, J., Calhoun, V. D., Shim, E., & Lee, J. H. (2016). Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance: Evidence from whole-brain resting-state functional connectivity patterns of schizophrenia. *NeuroImage*, *124*. https://doi.org/10.1016/j.neuroimage.2015.05.018

Kim, J., Criaud, M., Cho, S. S., Díez-Cirarda, M., Mihaescu, A., Coakeley, S., Ghadery, C., Valli, M., Jacobs, M. F., Houle, S., & Strafella, A. P. (2017). Abnormal intrinsic brain functional network dynamics in Parkinson's disease. *Brain*, *140*(11). https://doi.org/10.1093/brain/awx233

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*.

Ktena, S. I., Parisot, S., Ferrante, E., Rajchl, M., Lee, M., Glocker, B., & Rueckert, D. (2017). Distance metric learning using graph convolutional networks: Application to functional brain networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10433 LNCS*. https://doi.org/10.1007/978-3-319-66182-7_54

Kuang, D., Guo, X., An, X., Zhao, Y., & He, L. (2014). Discrimination of ADHD based on fMRI data with deep belief network. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8590 LNBI*. https://doi.org/10.1007/978-3-319-09330-7_27

Kumar, M., Ellis, C. T., Lu, Q., Zhang, H., Capotă, M., Willke, T. L., Ramadge, P. J., Turk-Browne, N. B., & Norman, K. A. (2020). BrainIAK tutorials: User-friendly learning materials for advanced fMRI analysis. *PLoS Computational Biology*, *16*(1). https://doi.org/10.1371/journal.pcbi.1007549

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11). https://doi.org/10.1109/5.726791

Ledoit, O., & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, *88*(2). https://doi.org/10.1016/S0047-259X(03)00096-4

Li, H., Parikh, N. A., & He, L. (2018). A novel transfer learning approach to enhance deep neural network classification of brain functional connectomes. *Frontiers in Neuroscience*, *12*(JUL). https://doi.org/10.3389/fnins.2018.00491

Loey, M., EL-Bakry, H., & El-Sawy, A. (2016). CNN for Handwritten Arabic Digits Recognition Based on LeNet-5. *Nternational Conference on Advanced Intelligent Systems and Informatics*, *1*.

Meng, L., & Xiang, J. (2018). Brain network analysis and classification based on convolutional neural network. *Frontiers in Computational Neuroscience*, *12*. https://doi.org/10.3389/fncom.2018.00095

Meszlényi, R. J., Buza, K., & Vidnyánszky, Z. (2017). Resting state fMRI functional connectivity-based classification using a convolutional neural network architecture. *Frontiers in Neuroinformatics*, *11*. https://doi.org/10.3389/fninf.2017.00061

Nickel, R. E., & Huang-Storms, L. (2017). Early Identification of Young Children with Autism Spectrum Disorder. In *Indian Journal of Pediatrics* (Vol. 84, Issue 1). https://doi.org/10.1007/s12098-015-1894-0

Nie, D., Zhang, H., Adeli, E., Liu, L., & Shen, D. (2016). 3D deep learning for multi-modal imaging-guided survival time prediction of brain tumor patients. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9901 LNCS*. https://doi.org/10.1007/978-3-319-46723-8_25

Niepert, M., Ahmad, M., & Kutzkov, K. (2016). Learning convolutional neural networks for graphs. *33rd International Conference on Machine Learning, ICML 2016*, *4*.

Parisot, S., Ktena, S. I., Ferrante, E., Lee, M., Guerrero, R., Glocker, B., & Rueckert, D. (2018). Disease prediction using graph convolutional networks: Application to Autism Spectrum Disorder and Alzheimer's disease. *Medical Image Analysis*, *48*. https://doi.org/10.1016/j.media.2018.06.001

Patel, P., Aggarwal, P., & Gupta, A. (2016). Classification of schizophrenia versus normal subjects using deep learning. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3009977.3010050

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*.

Pennec, X., Fillard, P., & Ayache, N. (2006). A riemannian framework for tensor computing. *International Journal of Computer Vision*, *66*(1). https://doi.org/10.1007/s11263-005-3222-z

Pereira, F., Mitchell, T., & Botvinick, M. (2009). Machine learning classifiers and fMRI: a tutorial overview. *NeuroImage*, *45*(1 Suppl). https://doi.org/10.1016/j.neuroimage.2008.11.007

Perozzi, B., Al-Rfou, R. and Skiena, S., (2014) Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

Ribeiro, L. F. R., Saverese, P. H. P., & Figueiredo, D. R. (2017). Struc2vec: Learning node representations from structural identity. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *Part F129685*. https://doi.org/10.1145/3097983.3098061

Rosenthal, G., Váša, F., Griffa, A., Hagmann, P., Amico, E., Goñi, J., Avidan, G., & Sporns, O. (2018). Mapping higher-order relations between brain structure and function with embedded vector representations of connectomes. *Nature Communications*, *9*(1). https://doi.org/10.1038/s41467-018-04614-w

Saad, Z.S., Glen, D.R., Chen, G., Beauchamp, M.S., Desai, R. and Cox, R.W., (2009) A new method for improving functional-to-structural MRI alignment using local Pearson correlation. *Neuroimage*, *44*(3), pp.839-848. doi: 10.1016/j.neuroimage.2008.09.037

Salimi-Khorshidi, G., Douaud, G., Beckmann, C. F., Glasser, M. F., Griffanti, L., & Smith, S. M. (2014). Automatic denoising of functional MRI data: Combining independent component analysis and hierarchical fusion of classifiers. *NeuroImage*, *90*. https://doi.org/10.1016/j.neuroimage.2013.11.046

Sarraf, S., DeSouza, D., Anderson, J., & Tofighi, G. (2016). DeepAD: Alzheimer's Disease Classification via Deep Convolutional Neural Networks using MRI and fMRI. *BioRxiv*. https://doi.org/10.1101/070441

Sharif, H., & Khan, R. A. (2021). A Novel Machine Learning Based Framework for Detection of Autism Spectrum Disorder (ASD). *Applied Artificial Intelligence*. https://doi.org/10.1080/08839514.2021.2004655

Sherkatghanad, Z., Akhondzadeh, M., Salari, S., Zomorodi-Moghadam, M., Abdar, M., Acharya, U. R., Khosrowabadi, R., & Salari, V. (2020). Automated Detection of Autism Spectrum Disorder Using a Convolutional Neural Network. *Frontiers in Neuroscience*, *13*. https://doi.org/10.3389/fnins.2019.01325

Shervashidze, N., Schweitzer, P., van Leeuwen, E. J., Mehlhorn, K., & Borgwardt, K. M. (2011). Weisfeiler-Lehman graph kernels. *Journal of Machine Learning Research*, *12*.

Smith, S. M., Fox, P. T., Miller, K. L., Glahn, D. C., Fox, P. M., Mackay, C. E., Filippini, N., Watkins, K. E., Toro, R., Laird, A. R., & Beckmann, C. F. (2009). Correspondence of the brain's functional architecture during activation and rest. *Proceedings of the National Academy of Sciences of the United States of America*, *106*(31). https://doi.org/10.1073/pnas.0905267106

Suk, H. il, Wee, C. Y., Lee, S. W., & Shen, D. (2016). State-space model with deep learning for functional dynamics estimation in resting-state fMRI. *NeuroImage*, *129*. https://doi.org/10.1016/j.neuroimage.2016.01.005

Tixier, A. J.-P., Nikolentzos, G., Meladianos, P., & Vazirgiannis, M. (2019). Graph classification with 2d convolutional neural networks. *International Conference on Artificial Neural Networks*, 578–593.

Tohka, J., Foerde, K., Aron, A. R., Tom, S. M., Toga, A. W., & Poldrack, R. A. (2008). Automatic independent component labeling for artifact removal in fMRI. *NeuroImage*, *39*(3). https://doi.org/10.1016/j.neuroimage.2007.10.013

Varoquaux, G., Baronnet, F., Kleinschmidt, A., Fillard, P., & Thirion, B. (2010). Detection of brain functional-connectivity difference in post-stroke patients using group-level covariance modeling. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6361 LNCS*(PART 1). https://doi.org/10.1007/978-3-642-15705-9_25

Varoquaux, G., & Craddock, R. C. (2013). Learning and comparing functional connectomes across subjects. *NeuroImage*, *80*. https://doi.org/10.1016/j.neuroimage.2013.04.007

Vigneshwaran, S., Mahanand, B. S., Suresh, S., & Sundararajan, N. (2015). Using regional homogeneity from functional MRI for diagnosis of ASD among males. *Proceedings of the International Joint Conference on Neural Networks*, *2015-September*. https://doi.org/10.1109/IJCNN.2015.7280562

Wang, D., Cui, P., & Zhu, W. (2016). Structural deep network embedding. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*. https://doi.org/10.1145/2939672.2939753

Wang, G., & Gong, J. (2019). Facial Expression Recognition Based on Improved LeNet-5 CNN. *Proceedings of the 31st Chinese Control and Decision Conference, CCDC 2019*. https://doi.org/10.1109/CCDC.2019.8832535

Wilson, J. D., Baybay, M., Sankar, R., & Stillman, P. (2018). Fast embedding of multilayer networks: An algorithm and application to group fMRI. *ArXiv*, *415*.

Xing, X., Ji, J., & Yao, Y. (2019). Convolutional Neural Network with Element-wise Filters to Extract Hierarchical Topological Features for Brain Networks. *Proceedings - 2018 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2018*. https://doi.org/10.1109/BIBM.2018.8621472

Yang, X., Islam, M. S., & Khaled, A. M. A. (2019). Functional connectivity magnetic resonance imaging classification of autism spectrum disorder using the multisite ABIDE dataset. *2019 IEEE EMBS International Conference on Biomedical and Health Informatics, BHI 2019 - Proceedings*. https://doi.org/10.1109/BHI.2019.8834653

Yang, Z., Ding, M., Zhou, C., Yang, H., Zhou, J., & Tang, J. (2020). Understanding Negative Sampling in Graph Representation Learning. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/3394486.3403218

Zafar, R., Kamel, N., Naufal, M., Malik, A. S., Dass, S. C., Ahmad, R. F., Abdullah, J. M., & Reza, F. (2017). Decoding of visual activity patterns from fMRI responses using multivariate pattern analyses and convolutional neural network. *Journal of Integrative Neuroscience*, *16*(3). https://doi.org/10.3233/JIN-170016