

Counterfactual Graphs for Explainable Classification of Brain Networks

Carlo Abrate

Sapienza University, Rome, Italy

ISI Foundation, Turin, Italy

carlo.abrate@isi.it

Francesco Bonchi

ISI Foundation, Turin, Italy

Eurecat, Barcelona, Spain

francesco.bonchi@isi.it

ABSTRACT

Training graph classifiers able to distinguish between healthy brains and dysfunctional ones can help identifying substructures associated to specific cognitive phenotypes. In this regard, the mere predictive power of the graph classifier is of limited interest to the neuroscientists, which have plenty of tools for the diagnosis of specific mental disorders. What matters is the interpretation of the model, as it can provide novel insights and new hypotheses. In this regard, we propose *counterfactual graphs* as a way to produce local post-hoc explanations of any black-box graph classifier. Given a graph and a black-box, a counterfactual is a graph which, while having high structural similarity with the original graph, is classified by the black-box in a different class. We propose and empirically compare several strategies for counterfactual graph search. Our experiments against a white-box classifier with known optimal counterfactual show that our methods, although heuristic, can produce counterfactuals very close to the optimal one. Finally, we show how to use counterfactual graphs to build global explanations correctly capturing the behaviour of different black-boxes classifiers and providing interesting insights for the neuroscientists.

ACM Reference Format:

Carlo Abrate and Francesco Bonchi. . Counterfactual Graphs for Explainable Classification of Brain Networks . In *Proceedings of (KDD’21)*. ACM, New York, NY, USA, 10 pages.

1 INTRODUCTION

In recent years *brain connectomics* [36], a field concerned with providing a network representation of the brain by comprehensively mapping the neural elements and their connections, has emerged as an important paradigm promising to help understanding mental processes and brain diseases [12]. Thanks to the network representation of the brain, many interesting neuroscience challenges can be tackled as graph mining problems [6]. A prominent example is that of *classification of brain networks*: given two groups of individuals, a *condition group* (class 1) and a *control group* (class 0), where each individual is represented as a brain network, i.e., a graph defined over the same set V of vertices corresponding to the brain *regions of interest* (ROIs), the goal is to infer a model that, given an unseen graph defined over the same vertices V , it is able to predict whether the new individual belongs to class 0 or class 1 [17, 23, 40, 42].

In this setting, building accurate classifiers is important, but not the main goal. In fact, the mere predictive power is of limited interest to the neuroscientists, which have plenty of tools for the diagnosis of specific mental disorders. What matters is the ability of discovering discriminative substructures that might be associated

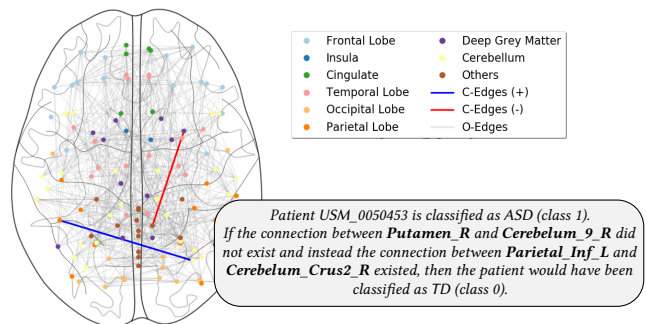


Figure 1: An example of counterfactual graph for a patient from the ABIDE dataset [7]. The patient is classified by a black-box (see Appendix A) in class ASD (Autism Spectrum Disorder), but with just two edge changes could have been classified as TD (Typically Developed). In gray the edges of the original graph, in red the edge to be removed and in blue the one to be added, so to form the counterfactual graph. In the box, the counterfactual expressed in natural language.

to specific cognitive phenotypes or mental dysfunctions [10], so to provide insights and new hypotheses for the neuroscientists to further investigate. Surprisingly, not much attention has been devoted in the literature to produce explanations for graph classification. On the one hand, most of the methods proposed for graph classification are not very transparent and enjoy little interpretability, as they are based either on kernels [33], embeddings [1, 15, 25], or deep learning [19, 41, 43]. On the other hand, although providing *post-hoc explanations of a black-box classifier* is a very active research area [13, 14], most of the proposed methods focus on tabular data [22, 27], images [32, 34, 35] or time series [3].

In this paper we propose *graph counterfactuals* as a way to produce local post-hoc explanations of any black-box graph classifier. Counterfactuals are borrowed from causal language to create contrastive example-based explanations, similar to the following statement (borrowed from [39]):

“You were denied a loan because your annual income was £30,000. If your income had been £45,000, you would have been offered a loan.”

Intuitively, a counterfactual for a given data point x is another data point x' which is as close as possible to x , yet classified in a different class. The rationale for finding a counterfactual instance which is as close as possible to the original example is twofold: firstly, it ensures that the counterfactual explanation is compact, thus much easier to communicate and comprehend; secondly, it guarantees that the produced counterfactual is a realistic example, being not too far from a real data point.

In the context of brain network classification, we create a counterfactual graph, by changing the links (either adding or removing) of a given graph. Figure 1 provides an example of such counterfactual graph. Patient USM_0050453 from the ABIDE dataset [7] (more information in Section 5 and Appendix A) is predicted in class ASD (Autism Spectrum Disorder, class 1) by a black-box graph classifier. By just removing an edge and adding another non-existing edge, we can obtain a new graph, which is very similar to the original one, but classified differently by the same black-box classifier.

Paper contributions and roadmap. Besides the importance of explainability, classification of brain networks distinguishes from other graph classification tasks for other aspects. As discussed by other authors, e.g., [15, 17], brain networks enjoy *node identity awareness*, i.e., the fact that a specific vertex id is unique in a graph and corresponds to the same brain ROI through all the input networks, which are all thus defined over the same vertices V . This is different from other graph classification tasks and other application domains (e.g., prediction of functional and structural properties of molecules in cheminformatics) where vertices can share the same labels and the vertex identity is not a relevant information.

In this specific context, the roadmap and contributions of this paper are as follows:

- We define the COUNTERFACTUAL GRAPH SEARCH problem (§3.2). As the search space of the problem is exponential and we do not know anything about the black-box classifier, we can not hope for a polynomial-time exact algorithm.
- We thus propose heuristic-search methods (§4), distinguishing between two cases: when we do not have any data (oblivious), or when we have access to a dataset (data-driven).
- Our empirical comparison on different brain network datasets and with different black-boxes (§5.2) show that both the oblivious and the data-driven approaches can produce good quality counterfactual graphs, but that the data-driven approach can do so faster (i.e., with less calls to the black-box classifier).
- Our experiments against a *white-box classifier with known optimal counterfactual* (§5.3) show that our heuristic methods can produce counterfactuals very close to the optimal one.
- We finally show how counterfactual graphs can be used as basic building blocks to produce LIME-like [27] (SHAP-like [22]) local explanations (§6.1), as well as global explanations (§6.2 and §6.3) correctly capturing the behaviour of different black-boxes classifiers and which are meaningful from the neuroscience standpoint.

2 RELATED WORK

As discussed in Section 1 not much effort has been devoted to explainability in graph classification. Explanation methods for node classification and link prediction have been proposed in [16, 44, 45], but this is a different problem from the graph classification task in which the aim is to classify the whole graph and not its nodes.

In the context of brain networks, Yan et al. [42] propose a deep learning approach for explainable classification, using a node-grouping layer before the convolutional layer: such node-grouping layer provides insight in the most predictive brain subnetworks.

Still in the context of brain network classification, Lanciano et al. [17] propose a method based on *contrast subgraph*, i.e., a set of vertices whose induced subgraph is very dense in one class of graphs and very sparse in the other class. This approach produces very simple classifiers which are transparent and self-explanatory.

While [17, 42] propose methods that have some explainability *by-design*, here we study how to produce local *post-hoc* explanations of any black-box graph classifier, by means of graph counterfactuals.

In *cheminformatics*, Numeroso and Bacciu [4] propose molecule counterfactuals for explaining *deep graph networks* for the prediction of functional and structural properties of molecules. Their proposal is an explanatory agent based on *reinforcement learning* which has access to the internal representation of the black-box and leverages domain knowledge to constrain the generated explanations to be valid molecules. The main difference with our work is in the type of graph classification assumed: our graphs, in fact, enjoy *node identity awareness* [15, 17], i.e., the fact that a specific vertex id is unique in a graph and corresponds to the same brain ROI through all the input networks. Conversely, molecules are small graphs where the vertices are atoms, the same vertex can thus appear many times in the same graph, and the links are labeled with the type of bond. Moreover, our approach provides counterfactual explanations for any black-box classifier, not only for a specific one; it does not assume any knowledge of the internals of the black-box, nor it needs any background knowledge by the domain expert.

3 PRELIMINARIES

3.1 Brain network construction

Magnetic resonance imaging (MRI) has played a central role in the development of connectomics, by allowing relatively cost-effective in vivo assessment of the macro-scale architecture of brain network connectivity. A connectome, or brain network, is created by linking the brain regions of interest (ROIs), either by observing anatomic fiber density (structural connectome), or by computing pairwise correlations between time series of activity associated to ROIs (functional connectome). The latter approach, known as fMRI, exploits the link between neural activity and blood flow and oxygenation, to associate a time series to each ROI. A brain network can then be defined by creating a link between two ROIs that exhibit co-activation, i.e., strong correlation in their time series.

ROIs are defined by aggregating adjacent *voxels*, i.e., the most fine-grain units of the 3-dimensional image which correspond to a very large number of brain cells (in the order of millions). This is done as a dimensionality reduction step and it is justified by the fact that spatially adjacent voxels typically play their role on the same activities, acting as a unique unit. For this voxel-aggregation task, named *parcellation* of the brain, several atlases are well established in neuroscience [8, 38].

Finally, as MRI signals are heavily subject to noise caused by different confounding factors, a plethora of signal pre-processing strategies (e.g., filtering, signal correction, thresholding etc.) are typically adopted in the data preparation pipeline (see Lang et al. [18] for a comprehensive survey).

For the purposes of this paper, we abstract from all the pre-processing details (except when providing the experimental setting

information needed for repeatability), and just assume that each brain is given to us as an undirected unweighted simple graph. More formally we consider a set of vertices V , i.e., the ROIs, as defined by the selected parcellation atlas. We let V^2 denote the set of all possible pairs of elements of V , i.e., $V^2 = \{(u, v) | u, v \in V \wedge u \neq v\}$. As we consider only graphs defined over the same set of vertices, in the rest of this paper each graph is simply identified by its set of edges $E \subseteq V^2$. The set of all possible graphs defined over V is then the powerset of V^2 , that we denote as $\mathbb{G}(V) = 2^{V^2}$. Given two groups of such graphs, a condition group (class 1) and a control group (class 0), a graph classification model can be trained to discriminate among the two classes, i.e., to predict the class for a new graph $E \in \mathbb{G}(V)$, by using any graph classification method.

3.2 Problem statement

We are given a *black-box graph classifier*, denoted as a function $f : \mathbb{G}(V) \rightarrow \{0, 1\}$. We assume that we do not know anything about f and we can only query it as an oracle, i.e., giving it a graph in input and getting as output the predicted class.

Given a specific graph $E \in \mathbb{G}(V)$, a *counterfactual graph* is another graph $E' \in \mathbb{G}(V)$, such that $f(E') = 1 - f(E)$. As discussed in Section 1, we are not interested in just any counterfactual instance, but we want to find a E' that is as close as possible to E . As distance measure we adopt a simple *edit distance*: $d(E, E') = |E \Delta E'|$ where Δ denotes the *symmetric difference* of the two sets, i.e., $E \Delta E' = (E \setminus E') \cup (E' \setminus E)$. This is the set of edges that need to be removed and the edges that need to be added to go from E to E' .

PROBLEM 1 (COUNTERFACTUAL GRAPH SEARCH). *Given a black-box graph classifier $f : \mathbb{G}(V) \rightarrow \{0, 1\}$ and a graph $E \in \mathbb{G}(V)$, find a counterfactual graph $E^* \in \mathbb{G}(V)$ such that:*

$$E^* = \arg \min_{\substack{E' \in \mathbb{G}(V): \\ f(E') = 1 - f(E)}} d(E, E').$$

As discussed previously, having a counterfactual graph which is as close as possible to the original graph, allows to produce a concise explanation, as the *edit distance corresponds exactly to the dimension of the explanation*.

Given that the search space has size $|\mathbb{G}(V)| = 2^{\frac{n(n-1)}{2}}$ where $n = |V|$, thus exponential in the number of vertices, and that we do not know anything about the black-box classifier f , we cannot hope for a polynomial-time algorithm that solves Problem 1 exactly. What we can do is heuristically explore the search space, querying the black-box classifier f as an oracle over new instances $E' \in \mathbb{G}(V)$. The number of calls to the oracle f is an important measure when assessing search strategies, as different black-box classifiers might have different time requirements to produce the classification for a new instance E' . Therefore, in our framework, we will always generate the closest counterfactual that we can find during a search with a number of calls to the oracle f , limited by a user-defined parameter $\eta \in \mathbb{N}^+$.

4 COUNTERFACTUAL GRAPH SEARCH

As we can not hope to have a polynomial-time exact algorithm for COUNTERFACTUAL GRAPH SEARCH, we have tried several simple heuristic-search methods, all having comparable performances. In

the rest of this section we present one of these approaches: a *bidi-rectional local-search heuristic* which, while being simple, provides good performance as we will show in Section 5.

Figure 2 depicts an high-level overview of the proposed method, structured in two phases. In the first phase, we move away from the original graph E looking for a first counterfactual graph E_c .

In the second phase, we iteratively try to produce a better counterfactual, by exploring the space in between the original graph E and the current best counterfactual E_c . The exploration is performed by adding or removing one or more edges from a pool of candidates, i.e., the symmetric difference of the original graph E and the current best counterfactual E_c .

Figure 2: Abstraction of our method.

We propose two main variants of this general heuristic search schema.

Oblivious: the algorithm is only given the starting graph E and the possibility of querying the black-box.

Data-driven: the algorithm has access to a dataset of brain networks D , which could be, e.g., the training set used for building the black-box, a subset or a superset of it.

We next describe in more details our heuristic search approaches.

4.1 Phase 1: Finding the first counterfactual

Oblivious Forward Search. As we have access only to the given graph E and the black-box, all we can do is iteratively edit E (adding or removing edges) until we produce a counterfactual graph. The pseudocode of this procedure, dubbed Oblivious Forward Search (OFS), is given in Algorithm 1.

More in details, at each step (lines 2 to 8) we select k edges to be changed: either edges in $V^2 \setminus E$ to be added to E , or edges in E to be removed. As the former pool of candidate is typically much larger than the latter one, in order to avoid biasing the search in favor of edge additions over removal, before each edit operation we flip an unbiased coin ($\text{rnd}()$ line 6) to decide whether to add (lines 6–7) or remove (line 8) an edge. Then one edge e to be changed is picked from the corresponding pool of candidates uniformly at random (pick function). A list L of already used edges is maintained, to avoid adding and removing the same edge multiple times.

Algorithm 1 Oblivious Forward Search (OFS)

Input: $E \in \mathbb{G}(V)$, $\eta \in \mathbb{N}^+$, $k \in \mathbb{N}^+$, black-box $f : \mathbb{G}(V) \rightarrow \{0, 1\}$

Output: $E_c \in \mathbb{G}(V)$ such that $f(E_c) = 1 - f(E)$ or fail

```

1:  $i \leftarrow 0$ ;  $E' \leftarrow E$ ;  $L \leftarrow \emptyset$ 
2: while  $f(E') = f(E)$  and  $i < \eta$  do
3:    $i++$ ;  $j \leftarrow 0$ 
4:   while  $j < k$  do
5:      $j++$ 
6:     if  $\text{rnd}() < 0.5$  then  $e \leftarrow \text{pick}(1, V^2 \setminus (E \cup L))$ ;
7:        $E' \leftarrow E' \cup \{e\}$ ;  $L \leftarrow L \cup \{e\}$ ;
8:     else  $e \leftarrow \text{pick}(1, E \setminus L)$ ;  $E' \leftarrow E' \setminus \{e\}$ ;  $L \leftarrow L \cup \{e\}$ ;
9:   if  $f(E') = 1 - f(E)$  then return  $E'$  else fail
```

The search is called “forward” because it starts from the original graph E and it moves away from it. In particular, at each step, the

edit distance $d(E', E)$ between the current graph E' and the original graph E increases of k edges.

Data-driven Forward Search. We next consider the case in which we have access to a database of brain networks $D = \{(E_i, y_i)\}_{i=1}^n$ s.t. $E_i \in \mathbb{G}(V)$, $y_i \in \{0, 1\}$, which could be, e.g., the publicly available training set used for building the black-box, a subset or a superset of it, or even a different dataset but having the same structure of the training set, with a condition and a control group for the same mental disease. In this setting, we can use the distribution of edges among the two classes to bias the random selection (i.e., the *pick* function at lines 6 and 8 of Algorithm 1), *to favor edges which are better at discriminating between the two classes*. Specifically, we define a weighting function $w : V^2 \rightarrow \mathbb{Z}$ that assigns a weight to each edge based on its occurrences in the two classes as follows. For each $e \in V^2$ we define $D^+(e) = \{(E_i, y_i) \in D | e \in E_i \wedge y_i = f(E)\}$ the set of graphs in the database that contain e and have label concordant with the classification of the original graph $f(E)$, and similarly, we define $D^-(e) = \{(E_i, y_i) \in D | e \in E_i \wedge y_i = 1 - f(E)\}$ the set of graphs in the database that contain e and have label discordant with the classification of the original graph.

Then we define:

$$w(e) = \begin{cases} |D^+(e)| - |D^-(e)| & \text{if } e \in E \\ |D^-(e)| - |D^+(e)| & \text{if } e \in V^2 \setminus E \end{cases}$$

The idea is that we want to remove edges from E that are strongly characterizing of the class $f(E)$, so to more easily produce a counterfactual with small edit distance, or alternatively, add to E edges that are strongly characterizing of the opposite class $1 - f(E)$.

Therefore, in the setting in which we have access to D , we substitute *pick*() at lines 6-8 of Algorithm 1 with a function that picks edges with probability proportional to $\max(\epsilon, w(\cdot))$, where ϵ is a very small positive constant, used to avoid null probabilities while making the edges with negative weights unlikely to be picked. The resulting method is called Data-driven Forward Search (DFS).

4.2 Phase 2: Finding a better counterfactual

In the second phase, we start from the first counterfactual graph produced by phase 1, let us denote it E_c^1 , and we go back towards the original graph E . We do so by randomly picking edges in the symmetric difference $E \Delta E_c^1$ and modifying E_c^1 consequently. The approach is justified by the key observation that modifying E_c^1 by adding or removing edges from the symmetric difference with E , we obtain a graph E' which is guaranteed to have a smaller edit distance from E , i.e., $d(E', E) < d(E_c^1, E)$.

Oblivious Backward Search. Our heuristic search method reported in Algorithm 2, adapts the number of edges k that are changed in each iteration, so to try to reach the classification border faster. If in one iteration i , the candidate graph E_c^i turns out to be a counterfactual (line 7), we increase k (line 8) as we are not yet at the classification border, if instead, it is not a counterfactual graph (meaning that the backward search has crossed the classification border), we reduce the value of k for the next iteration (line 9). At every iteration, E_c keeps the current best counterfactual graph, while E_d keeps the symmetric difference, i.e., the pool from which to select the edges to add or remove.

Algorithm 2 Oblivious Backward Search (OBS)

Input: $E, E_c^1 \in \mathbb{G}(V)$, $\eta, k \in \mathbb{N}^+$, black-box $f : \mathbb{G}(V) \rightarrow \{0, 1\}$

Output: $E_c \in \mathbb{G}(V)$ such that $f(E_c) = 1 - f(E)$

```

1:  $E_c \leftarrow E_c^1; i \leftarrow 0;$ 
2:  $E_d \leftarrow E \Delta E_c$ 
3: while  $i < \eta$  and  $|E_d| > 0$  do
4:    $i++; k \leftarrow \min(k, |E_d|);$ 
5:    $E_d^k \leftarrow \text{pick}(k, E_d)$ 
6:    $E_c^i \leftarrow (E_c \cup (E_d^k \setminus E_c)) \setminus (E_d^k \cap E_c)$ 
7:   if  $f(E_c^i) = 1 - f(E)$  then
8:      $k++; E_c \leftarrow E_c^i; E_d \leftarrow E \Delta E_c$ 
9:     else if  $k > 1$  then  $k--$ 
10:    else  $E_d \leftarrow E_d \setminus E_d^k;$ 
11: return  $E_c$ 
```

In line 5, the function *pick*(k, E_d) picks uniformly at random k edges from the pool of candidate and then add or remove them from E_c accordingly (line 6) to produce the candidate graph E_c^i . When k reaches the value of 1, we stop decreasing k and start removing the edges that have been already tested from the pool of candidate edges E_d : as we are trying edges one by one, it does not make sense to try them multiple times. Eventually, when E_d is emptied, the algorithm terminates (line 3). The other termination condition is when the algorithm has reached the maximum allowed number η of calls to the black-box.

Data-driven Backward Search. A key part of the second phase of counterfactual graph search is the selection of the edges to change at each iteration, which, in the oblivious case, is implemented by a uniform random selection (function *pick*(k, E_d), in line 6 of Algorithm 2). In the case in which we have access to a database of brain networks $D = \{(E_i, y_i)\}_{i=1}^n$, similarly to what done for DFS, we can use the distribution of edges among the two classes to bias the random selection, favoring edges which are better at discriminating between the two classes. In this case we define the weight as:

$$w(e) = \begin{cases} |D^+(e)| - |D^-(e)| & \text{if } e \in E \setminus E_c \\ |D^-(e)| - |D^+(e)| & \text{if } e \in E_c \setminus E \end{cases}$$

where $D^+(e)$ and $D^-(e)$ are defined as in §4.1. By substituting the function *pick*(k, E_d), in line 6 of Algorithm 2 with a function that picks edges from E_d with probability proportional to $\max(\epsilon, w(\cdot))$, where ϵ is a very small positive constant, we obtain the method called Data-driven Backward Search (DBS).

4.3 A simple baseline

When we have access to the dataset D we can also consider a baseline that simply searches in D for the counterfactual graph with the minimum edit distance from the original graph E . This baseline, called Dataset Search (DS) and described in Algorithm 3, is interesting because it answers the question of finding the *best counterfactual graph among the real networks* available.

As we have access to the real labels of the graphs in D , we avoid checking those graphs which have the same class label as E (line 2). For the other ones, we first check the edit distance from E , and only if it is smaller than the current best one δ (line 3), we call the

Algorithm 3 Baseline: Dataset Search (DS)

Input: $E \in \mathbb{G}(V)$, black-box $f : \mathbb{G}(V) \rightarrow \{0, 1\}$,
dataset $D = \{(E_i, y_i)\}_{i=1}^n$ s.t. $E_i \in \mathbb{G}(V), y_i \in \{0, 1\}$
Output: $E_c \in \mathbb{G}(V)$ such that $f(E_c) = 1 - f(E)$ or fail

```

1:  $E_c \leftarrow \emptyset; \delta \leftarrow |V|^2$ 
2: for all  $(E_i, y_i) \in D$  s.t.  $y_i = 1 - f(E)$  do
3:   if  $d(E, E_i) < \delta$  then
4:     if  $f(E_i) = 1 - f(E)$  then
5:        $E_c \leftarrow E_i; \delta \leftarrow d(E, E_i)$ 
6: if  $E_c \neq \emptyset$  then return  $E_c$  else fail

```

black-box (line 4). If the graph is in the counterfactual class and it is correctly classified so by the black-box, then it becomes the current best counterfactual graph (line 5).

5 ASSESSMENT ON BRAIN NETWORKS

In this section we assess the performance of the counterfactual graph search methods introduced in Section 4.

5.1 Experiment setting

Human-brain datasets. Our experiments are performed on two publicly-available datasets. The first dataset, about *Autism Spectrum Disorder* (ASD), is taken from the Autism Brain Image Data Exchange (ABIDE)¹ [7]. In particular, we focus on the portion of dataset containing children below 9 years of age [17]. These are 49 individuals in the condition group, labeled as *Autism Spectrum Disorder* (ASD) and 52 individuals in the control group, labeled as *Typically Developed* (TD). The second dataset, about *Attention Deficit Hyperactivity Disorder* (ADHD), is taken from the USC Multimodal Connectivity Database (USCD)² [5], in particular from the ADHD200_CC200 study. This contains 190 individuals in the condition group, labeled as ADHD and 330 individuals in the control group, labeled as TD. Both datasets study brain functional connectivity by means of functional magnetic resonance imaging (fMRI), that allows to measure the statistical association between ROIs, as reviewed in §3.1. For the parcellation of the brain we use the AAL [38] atlas for the ASD dataset ($|V| = 116$) and the Craddock 200 (CC200) [8] for the ADHD dataset ($|V| = 190$). The pre-processing needed to go from the time series to the correlation matrices is performed in accordance with the literature.³

The final data-preparation step is to go from correlation matrices to networks. In accordance with large portion of the literature, e.g., [17, 21, 30], we transform a correlation matrix $C = \{c_{ij}\}$ into an adjacency matrix $A = \{a_{ij}\}$ by setting $a_{ij} = 1$ when c_{ij} is larger than a given threshold, and setting $a_{ij} = 0$ otherwise. The threshold is selected for both datasets as the 90th percentile of the distribution of the correlation matrix values.

Black-box classifiers. Our method is totally agnostic of the black-box classifier under analysis: it just sees the black-box as a function $f : \mathbb{G}(V) \rightarrow \{0, 1\}$, i.e., an oracle that can be queried with a graph and which gives back a label in $\{0, 1\}$. As such the specific black-box

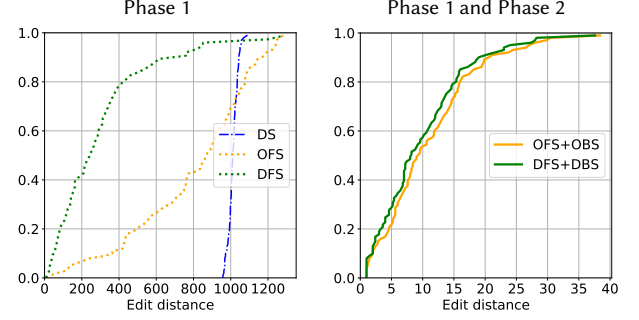


Figure 3: Cumulative distribution of $d(E, E_c)$ between the input graph E and its counterfactual E_c , on the ASD dataset using Contrast Subgraphs + SVM as black-box. One experiment (average over 5 runs) for each graph in the dataset.

we use is not so important in our empirical assessment. Nevertheless, for variety sake, we use 4 different black-box classifiers for our brain network context: Contrast Subgraphs [17], Graph2Vec [25], Sub2Vec [1], and Autoencoders [15]. These methods are very diverse and cover a wide spectrum of complexity for the explanation task. Each of these methods (the details about their parameters settings are provided in Appendix) produces a vector for each graph in the training set. Then a k -nearest neighbors (KNN) or a Support Vector Machine (SVM) is adopted to produce the final classification.

5.2 Methods assessment

We first assess the counterfactual graph search algorithms w.r.t. (1) the quality of the counterfactual produced, measured in terms of edit distance from the original graph, and (2) the number of calls to the black-box. We run an experiment for each graph in the training set. Each experiment is run 5 times and we report average results over the 5 runs. For each phase of the counterfactual graph search, we allow a maximum number of calls to the black-box $\eta = 2000$. We set $k = 5$ in all experiments.

Figure 3 reports the cumulative distribution of the edit distance between the input graph and its counterfactual, on the autism dataset using as black box Contrast Subgraphs. The left plot reports the distribution after the first phase and includes the baseline Dataset Search (DS). We can observe that, as expected, the data-driven approach DFS is much more effective than the oblivious OFS in finding the first counterfactual. In 80% of the cases, the edit distance of the first counterfactual is below 400, while the average counterfactual distance in the dataset is approximately 1000.

However, after phase 2, the difference in quality between the data-driven approach and the oblivious one get much smaller, indicating that both approaches converge very close to the best possible counterfactual graph. A large difference between the oblivious and the data-driven case is instead observed in the number of calls to the black-box, which is reported in Figure 4.

From this preliminary assessment, we conclude that: (1) both oblivious and data-driven approaches produce similar results in term of quality, but the data-driven case converges to a good counterfactual graph faster (with less calls to the black-box); (2) the counterfactual graph produced by the proposed methods is much better in quality than what we could find by searching among the real graphs in the dataset.

¹http://fcon_1000.projects.nitrc.org/indi/abide/

²http://umcd.humanconnectomeproject.org/umcd/default/browse_studies

³See <http://preprocessed-connectomes-project.org/abide/dparsf.html> for ASD dataset and https://ccraddock.github.io/cluster_roi/atlas.html for ADHD dataset.

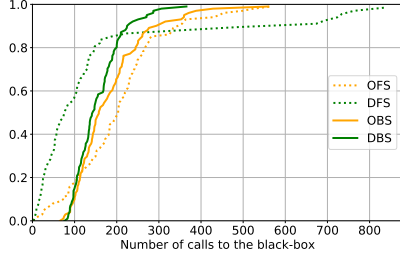


Figure 4: Cumulative distribution of the number of calls to the black-box (same setting as Figure 3).

In Table 1 we report the same statistics for all the black-boxes and on both datasets, using only the oblivious counterfactual search approach (OFS+OBS). We can observe that the black-boxes that explicitly keep in consideration the node identity awareness property of brain networks (i.e., Contrast Subgraphs [17] and Autoencoders [15]) typically get counterfactual graphs with smaller edit distance than the other two models. It is worth mentioning that in average, the best counterfactuals that one can obtain by using real examples in the dataset (i.e., by means of DS baseline), have edit distance from their original graphs of approximately 1000 in the ASD dataset and 2250 in the ADHD dataset.

While there is no shared consensus on how to evaluate post-hoc explanations, when counterfactuals come into play we argue that the smaller the edit distance the better the explanation. When counterfactuals are systematically closer to the original data points, the black boxes are easier to interpret. We can use the minimum edit distance as a measure to select among different classification methods, in domain such as brain networks in which post-hoc explainability is a selection criterion as important as the accuracy.

5.3 Assessment against a white-box

Since in general, it is not possible to compute the optimal counterfactual exactly, our methods are heuristic in nature. It would be thus interesting to compare our counterfactual solution with the optimal counterfactual, in a setting in which this is doable.

We achieve this by means of a very simple white-box, i.e., a classifier in which we can see the inner logic. In particular, we train a linear classifier built on a 2-dimensional embedding. This gives us a cartesian (x, y) -plane as the one in Figure 5, in which it is possible to identify the optimal counterfactual *geometrically*. In fact, if we consider the original graph E as a point (x_o, y_o) on the (x, y) -plane and the classifier f as a line $r_f : y = mx + c$, we can obtain the optimal counterfactual graph E_c^* as the point (x_c, y_c) as follows:

- (1) Generate the line r_p perpendicular to the classifier line r_f and passing through (x_o, y_o) , with equation $y = -\frac{1}{m}(x - x_o) + y_o$;
- (2) get the point (x_c, y_c) obtained by the intersection of r_p with r_f , with the following coordinates $x_c = \frac{(x_o + my_o - mc)}{m^2 + 1}$ and $y_c = \frac{m(x_o + my_o - mc)}{m^2 + 1} + c$.

The 2-dimensional embedding we use in Figure 5 is obtained by means of the Contrast Subgraphs method [17], on the ASD dataset. By applying this method we extract a contrast subgraph ASD_TD, i.e., a set of vertices whose induced subgraph is very dense among ASD individuals and very sparse among TD individuals, and similarly, another contrast subgraph TD_AS, i.e., a set of vertices

Table 1: Counterfactual graph search statistics on for all the black-boxes and on both datasets, using only the oblivious counter-factual search approach (OFS+OBS). Same experimental settings as before: One experiment (average over 5 runs) for each graph in the dataset, $\eta = 2000$ maximum calls to the black-box for each phase of the counterfactual search. We report the 10th, 25th, 50th, 75th, and 90th percentiles.

	ASD: Edit Distance					ASD: Calls to Oracle				
	10-p	25-p	50-p	75-p	90-p	10-p	25-p	50-p	75-p	90-p
Cont. Sub.	2	5.4	9	15.2	19.8	100	119.2	157.4	214.6	276.2
Sub2Vec	4	47.5	134	285	476	110.5	175.5	299.5	492	617.5
Graph2Vec	1.5	6.8	33.5	119	207.4	112.2	131	181.6	346.3	445
Autoenc.	1.6	2.4	3.8	6	9.2	82.2	101	141	209.4	275.4

	ADHD: Edit Distance					ADHD: Calls to Oracle				
	10-p	25-p	50-p	75-p	90-p	10-p	25-p	50-p	75-p	90-p
Cont. Sub.	9	20.6	43.8	70	100.2	173.2	249.2	406	578.2	764.7
Sub2Vec	8.8	56.2	185.3	367.2	742.8	283.88	474.5	780.56	1807.1	2000
Graph2Vec	1	2.2	49.3	166	411.6	135	156.6	324.4	547	1293.2
Autoenc.	1	1.8	5	15.4	25.1	129	137.8	172.5	273.8	367.5

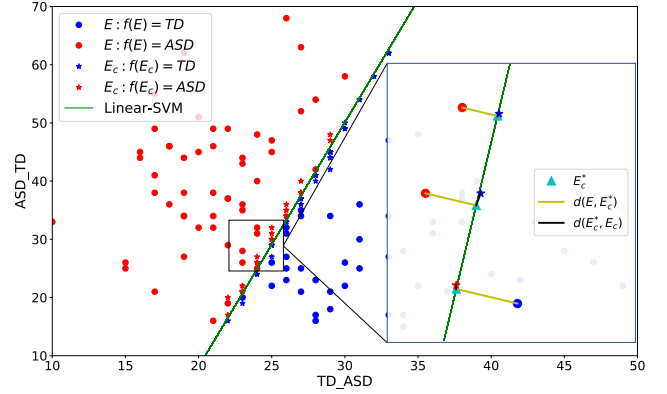


Figure 5: Real setting (ASD dataset): using Contrast Subgraphs to produce a 2-dimensional embedding on which we learn a linear classifier (green line) by means of SVM. For each individual E (dot) we produce one counterfactual E_c (star) using OFS+OBS with $\eta = 2000$ and $k = 5$. On the right-hand side we show a zooming on the classification border where we can appreciate how close the produced counterfactuals are to the optimal ones E_c^* (cyan triangles).

whose induced subgraph is very dense among TD individuals and very sparse among ASD individuals. Then for each graph in the dataset, we compute the two dimensions which are the number of edges in the subgraph induced by the contrast subgraph ASD_TD (y -axis), and the number of edges in the subgraph induced by the contrast subgraph TD_AS (x -axis). Over this embedding, we use a linear Support Vector Machine (SVM) to build the white-box classifier, represented by the green line in Figure 5 (which has an accuracy of 0.78 on the training set). Hence we can reconstruct the optimal solution geometrically and measure the error of the counterfactual graph produced (i.e., its distance from the optimal).

As measure of error of the counterfactual search method, we use the average edit distance between the optimal counterfactual graphs E_c^* and the estimated counterfactual graph E_c for all the graphs in the dataset D . In the setting reported in Figure 5 (i.e., ASD dataset, Contrast Subgraphs + SVM classifier), the average edit distance $d(E_c^*, E_c)$ between the optimal counterfactual graphs E_c^* and the estimated counterfactual graph E_c is 1.83.

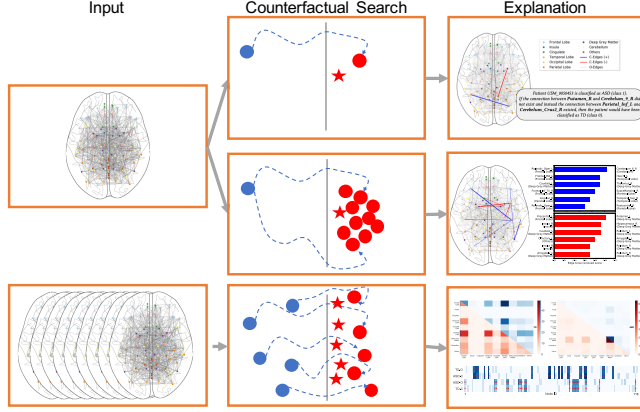


Figure 6: Schematization of the three different types of explanations we can produce by means of graph counterfactuals: Contrastive case-based explanations against counterfactuals (first row), LIME-like (SHAP-like) local explanations (second row), and global explanations (third row).

6 LOCAL AND GLOBAL EXPLAINABILITY

In this section to showcase the versatility and usefulness of our proposal, we show how counterfactual graphs can be used as basic building blocks to produce various type of explanations. In particular, we consider three different types of explanations, as summarized in Figure 6:

Contrastive case-based explanations against counterfactuals:

this is the direct presentation of a counterfactual graph as a local post-hoc explanation, as done in the example in Figure 1 in §1.

LIME-like (SHAP-like) explanations: LIME [27] and SHAP [22] are prominent examples of general explanation methods for classifiers (on tabular data), that explain the prediction for a given example by ranking the input features by their importance in the prediction. We can obtain a similar type of explanation by producing, for a given graph, many counterfactual graphs and collecting statistics about the edges that appear more frequently.

Global explanations: finally, by producing counterfactual graphs for many input examples and collecting statistics we can produce meaningful global explanations of any black-box graph classifier.

In the rest of this section we will elaborate further on the latter two points providing concrete examples on our brain network datasets.

6.1 LIME-like (SHAP-like) local explanations

Given a counterfactual graph E_c of a graph E , the explanation provided by the counterfactual graph is the symmetric difference $E \Delta E_c = (E \setminus E_c) \cup (E_c \setminus E)$, i.e., the set of edges $E_-^c = E \setminus E_c$ to be removed from E , and the set of edges $E_+^c = E_c \setminus E$ to be added to E , so to transform the original graph E in its counterfactual graph E_c . From this perspective, edges in E_-^c are edges whose *presence* in E is important to explain the classification $f(E)$. Similarly, edges in E_+^c are edges whose *absence* from E is important to explain $f(E)$.

As for a graph E its counterfactual is not unique, we can produce many counterfactual graphs (say 1000) and collect statistics about the number of times edges appear in E_-^c or in E_+^c . This statistic has a direct interpretation as importance of the edges in explaining $f(E)$.

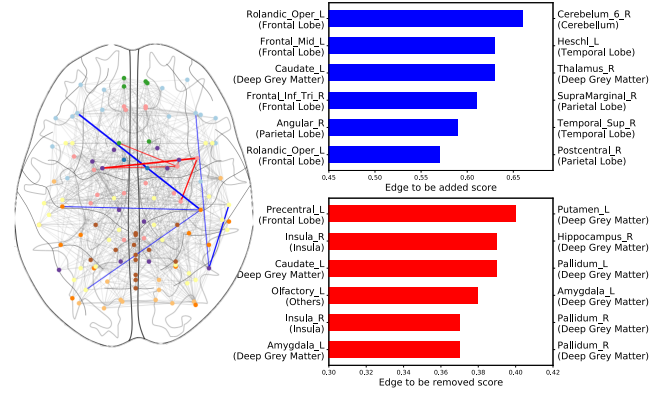


Figure 7: Visualization of LIME-like (SHAP-like) local explanation for the individual MaxMun_d_0051353 from the ASD dataset, which is classified in the autism class by the classifier of Figure 5. In the bar-plots on the right-hand side, we represents the top-6 edges to be added (blue bar) and the top-6 edges to be removed (red bard). These important edges can also be represented on the brain network, with the same colors, and with a thickness proportional to their importance.

In Figure 7 we provide an example of visualization of this local LIME-like (SHAP-like) explanation for a given individual which is classified in the ASD class. We can see that, among the edges whose absence is important in explaining the classification of this specific patient, there are many edges in the *frontal lobe* region, while among the edges whose presence is important for this classification, there are many in the *deep grey matter*. These observation are consistent with some neuroscience literature, e.g., [28].

6.2 Global explanations: edge-level aggregation

Similarly, we can aggregate the counterfactual graphs produced for many input graphs of both classes, so to produce statistics usable as a global explanation of the whole behaviour of a given black-box classifier. Given a dataset of brain networks $D = \{E_1, \dots, E_n\}$ all defined over the same set of ROIs V , similarly to what done previously in §6.1, we can create one or more counterfactuals for each $E_i \in D$, then collect statistics about the occurrences of edges in the symmetric differences between the graphs and their counterfactuals. More formally, for each edge $e \in V^2$ we can define 4 counters:

- $C_0^+(e) = |\{(E, E_c) | f(E) = 0 \wedge e \in E_+^c\}|$, i.e., the number of pairs (original graph, counterfactual graph), such that the fact that $e \notin E$ seems to be important to explain $f(E) = 0$;
- $C_0^-(e) = |\{(E, E_c) | f(E) = 0 \wedge e \in E_-^c\}|$, i.e., the number of pairs (original graph, counterfactual graph), such that the fact that $e \in E$ seems to be important to explain $f(E) = 0$;

and similarly $C_1^+(e)$ and $C_1^-(e)$ for class 1. Once we have collected these statistics we can combine or plot them in several ways to provide meaningful global explanations. In Figure 8 we present one such possible visualizations by means of heatmaps, aggregating ROIs in coarser-grain brain regions as provided by the AAL [38] parcellation atlas, for three different black-box classifiers. In particular, assuming that ASD is class 1 and TD is class 0, in the heatmaps

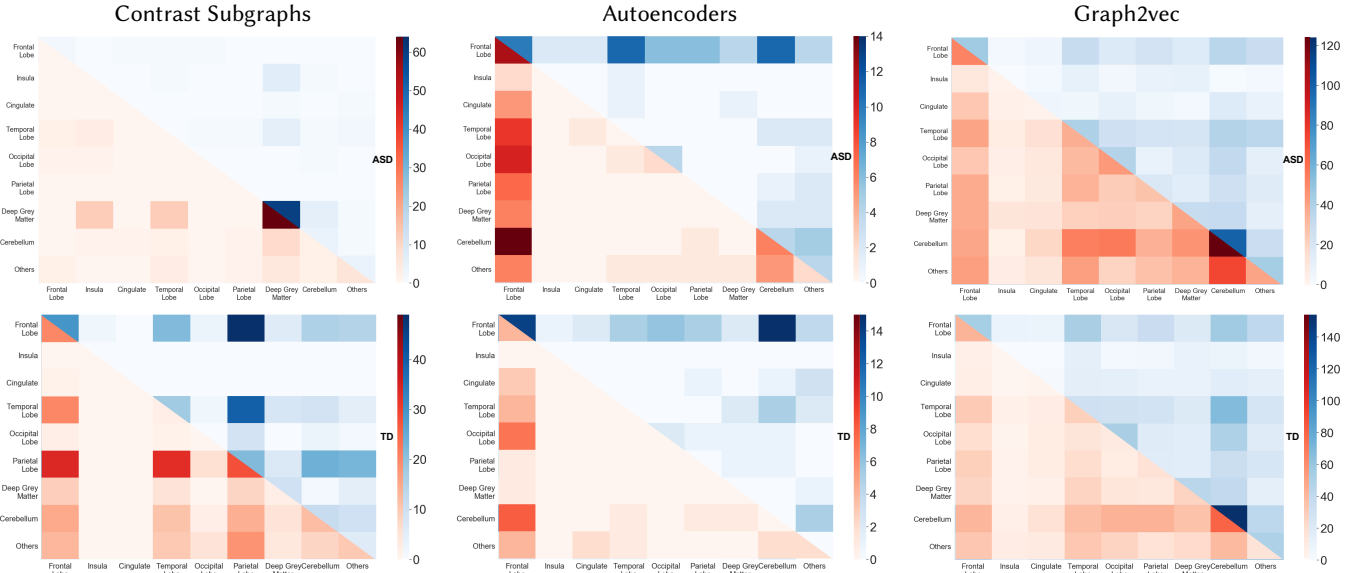


Figure 8: Heatmaps representing global explanations for three different black-box classifiers on the ASD dataset, obtained by aggregating edges importance at the level of coarser-grain brain regions.

marked ASD (top) we report in the upper-triangle (blue) the aggregation of counters $C_0^+(e)$ and in the lower-triangle (red) the counters $C_1^-(e)$. In other terms, both in the blue part that in the red part, the darker cells are those whose edges presence is important for the ASD class. For the way the 4 counters are defined, we expect to see a high level of symmetry in these heatmaps, as this is indeed the case in all 6 heatmaps in Figure 8.

A bird-eye view of the heatmaps highlights the fact that, *although trained on the same training set with the same class labels, the three models have a very different inner logic, as unveiled by occurrences of the edges in the counterfactual explanations.*

In the case of the Contrast Subgraphs model, we can appreciate the relevance of *deep grey matter* for the ASD class, consistently with the example in Figure 7 and with neuroscience literature [28]. *Parietal lobe*, *temporal lobe* and *frontal lobe* are instead the most important regions for the classification in the TD class. In fact, in the latter, the researchers found that the connectivity of *Fusiform Gyrus*, that is in the temporal lobe, captures the risk of developing autism as early as 1 year of age and provides evidence that abnormal fusiform gyrus connectivity increases with age [26, 31]. Also in the Autoencoders model, the *frontal lobe* region is the most important to explain the classification. Finally, the heatmaps for the classifier based on Graph2Vec highlight the relevance of the *cerebellum*, whose role in cognitive impairments in ASD has been observed in a plethora of studies, e.g., [2, 9, 20].

6.3 Global explanations: ROI-level aggregation

Finally, global explanations can also be produced in terms of importance of the ROIs for the classification, where the importance of a vertex can be computed by aggregating the same 4 counters defined in §6.2 (i.e., C_0^+ , C_0^- , C_1^+ , and C_1^-), for all its incident edges.

In Figure 9 we provide two examples of explanations based on ROIs' importance.

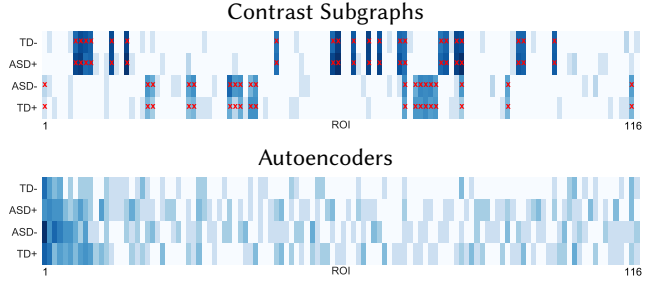


Figure 9: Global explanation based on ROIs' importance: Contrast Subgraphs black-box (top) and Autoencoders black-box (bottom) on the ASD dataset (same setting as in Figure 8). On x-axis the 116 ROIs of the AAL atlas ordered by id, and on the y-axis the aggregation of the 4 counters defined in §6.2 for all the edges incident on a given ROI.

The final “litmus test”. As we have a way to measure the importance of each single vertex in a black-box classifier, we can use the same idea we used in §5.3: i.e., to apply our counterfactual graphs framework to explain a “white-box” – a classifier of which we know, as a sort of ground truth, the inner logic – and check whether the most important vertices turn out to be the expected ones.

Therefore, we consider again the model based on Contrast Subgraphs on the ASD dataset. We recall that, by applying the method in [17], we extract a contrast subgraph ASD_TD , i.e., a set of vertices whose induced subgraph is very dense among ASD individuals and very sparse among TD individuals, and similarly, another contrast subgraph TD_ASD , i.e., a set of vertices whose induced subgraph is very dense among TD individuals and very sparse among ASD individuals. Then each graph in the dataset is mapped to two dimensions: the number of edges in the subgraph induced by each of the contrast subgraphs. Essentially the classification model is entirely defined by two sets of vertices and the classification hyperplane.

In Figure 9 (top) we mark with red crosses the ROIs belonging to the TD_ASF contrast subgraph on the first two rows, and the ROIs forming the ASF_TD contrast subgraph on the third and fourth rows. The red crosses fall exactly on the darker cells of the heatmap: *the global explanation built on counterfactual graphs is capturing exactly the (known) logic of the classifier.*

7 CONCLUSIONS AND FUTURE WORK

This paper introduces counterfactual graphs as a way to provide explanations of any black-box graph classifier, in the setting of graphs with node identity awareness, such is the case with brain networks. Counterfactual graphs can be used as basic building blocks to produce various type of explanations: they can be used directly as contrastive case-based explanations, they can be aggregated to provide LIME-like (SHAP-like) local explanations, as well as to produce global explanations.

We define the COUNTERFACTUAL GRAPH SEARCH problem and propose heuristic-search methods for it, distinguishing between two cases: when we do not have any data (oblivious), or when we have access to a dataset (data-driven). Our empirical assessment on different brain network datasets confirms that our methods produce good counterfactual graphs and that counterfactual graphs are a useful tool for explainability of brain network classification.

The language of explanations adopted in this paper is the most fine-grained possible: the edges whose addition or removal are important for a graph classification. In our future work, we plan to consider counterfactual graphs built over a richer vocabulary, including graph properties and higher-order structures, such as, e.g., motifs. On-going collaboration with neuroscientists is aimed at validating the usability of counterfactual graphs as a tool for explorative differential case-based reasoning for the domain expert.

REFERENCES

- [1] B. Adhikari, Y. Zhang, N. Ramakrishnan, and B. A. Prakash. Sub2vec: Feature learning for subgraphs. In *PAKDD*, 2018.
- [2] E. Anagnostou and M. Taylor. Review of neuroimaging in autism spectrum disorders: What have we learned and where we go from here. *Molecular autism*, 2:4, 04 2011.
- [3] R. Assaf and A. Schumann. Explainable deep neural networks for multivariate time series predictions. In *IJCAI*, 2019.
- [4] D. Bacciu and D. Numeroso. Explaining deep graph networks with molecular counterfactuals. *arXiv:2011.05134*, 2020.
- [5] J. Brown, J. Rudie, A. Bandrowski, J. Van Horn, and S. Bookheimer. The ucla multimodal connectivity database: A web-based platform for brain connectivity matrix sharing and analysis. *Frontiers in neuroinformatics*, 6:28, 11 2012.
- [6] E. Bullmore and O. Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature reviews neuroscience*, 10(3):186–198, 2009.
- [7] C. Cameron et al. The neuro bureau preprocessing initiative: open sharing of preprocessed neuroimaging data and derivatives. *Front. Neuroinform.*, 7, 2013.
- [8] R. C. Craddock et al. A whole brain fmri atlas generated via spatially constrained spectral clustering. *Human Brain Mapping*, 33(8):1914–1928, 2012.
- [9] A. D’Mello. Cerebellar gray matter and lobular volumes correlate with core autism symptoms. *NeuroImage: Clinical*, 7:631–639, 02 2015.
- [10] Y. Du, Z. Fu, and V. D. Calhoun. Classification and prediction of brain disorders using functional connectivity: Promising but challenging. *Frontiers in Neuroscience*, 12:525, 2018.
- [11] H. Engeland. Review on structural neuroimaging findings in autism. *Journal of neural transmission*, 111:903–29, 08, 2004.
- [12] A. Fornito and E. T. Bullmore. Connectomics: A new paradigm for understanding brain disease. *European Neuropsychopharmacology*, 25(5):733 – 748, 2015.
- [13] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal. Explaining explanations: An overview of interpretability of machine learning. In *DSAA*, 2018.
- [14] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.
- [15] L. Gutiérrez-Gómez and J.-C. Delvenne. Unsupervised network embeddings with node identity awareness. *Applied Network Science*, 4(1), 2019.
- [16] Q. Huang, M. Yamada, Y. Tian, D. Singh, D. Yin, and Y. Chang. GraphLIME: Local interpretable model explanations for graph neural networks. *arXiv:2001.06216*, 2020.
- [17] T. Lanciano, F. Bonchi, and A. Gionis. Explainable classification of brain networks via contrast subgraphs. In *KDD*, 2020.
- [18] E. W. Lang, A. M. Tomé, I. R. Keck, J. M. Górriz-Sáez, and C. G. Puntonet. Brain connectivity analysis: A short survey. *Computational Intelligence and Neuroscience*, 2012:1–21, 2012.
- [19] J. B. Lee, R. Rossi, and X. Kong. Graph classification using structural attention. In *KDD*, 2018.
- [20] J. Li, X. Chen, R. Zheng, A. Chen, Y. Zhou, and J. Ruan. Altered cerebellum spontaneous activity in juvenile autism spectrum disorders associated with cognitive functions. preprint, October 2020.
- [21] A. Lord, D. Horn, M. Breakspear, and M. Walter. Changes in community structure of resting state functional connectivity in unipolar depression. *PLOS ONE*, 7(8):1–15, 08 2012.
- [22] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In *Adv Neural Inf Process Syst*, 2017.
- [23] L. Meng and J. Xiang. Brain network analysis and classification based on convolutional neural network. *Frontiers in Computational Neuroscience*, 12, 2018.
- [24] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal. graph2vec: Learning distributed representations of graphs, 2017.
- [25] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv:1707.05005*, 2017.
- [26] K. Nickel et al. Inferior frontal gyrus volume loss distinguishes between autism and (comorbid) attention-deficit/hyperactivity disorder—a freesurfer analysis in children. *Frontiers in Psychiatry*, 9, 10 2018.
- [27] M. T. Ribeiro, S. Singh, and C. Guestrin. “Why should I trust you?” Explaining the predictions of any classifier. In *KDD*, 2016.
- [28] R. Richards et al. Increased hippocampal shape asymmetry and volumetric ventricular asymmetry in autism. *NeuroImage: Clinical*, 26:102207, 2020.
- [29] R. Rossi, R. Zhou, and N. Ahmed. Deep inductive graph representation learning. *TKDE*, 11, 2018.
- [30] M. Rubinov et al. Small-world properties of nonlinear brain activity in schizophrenia. *Human brain mapping*, 30:403–16, 02 2009.
- [31] B. Scherrer et al. The connectivity fingerprint of the fusiform gyrus captures the risk of developing autism in infants with tuberous sclerosis complex. *Cerebral cortex*, 30, 12, 2019.
- [32] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. GradCam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.
- [33] N. Shervashidze, P. Schweitzer, E. J. v. Leeuwen, K. Mehlhorn, and K. M. Borgwardt. Weisfeiler-lehman graph kernels. *JMLR*, 12:2539–2561, 2011.
- [34] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. *arXiv:1704.02685*, 2017.
- [35] A. Singh, S. Sengupta, and V. Lakshminarayanan. Explainable deep learning models in medical image analysis. *Journal of Imaging*, 6(6), 2020.
- [36] O. Sporns, G. Tononi, and R. Kötter. The human connectome: a structural description of the human brain. *PLoS Comput Biol*, 1(4), 2005.
- [37] C. Stoodley. The cerebellum and neurodevelopmental disorders. *Cerebellum*, 15, 08, 2015.
- [38] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. 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.
- [39] S. Wachter, B. Mittelstadt, and C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard journal of law & technology*, 31:841–887, 04 2018.
- [40] S. Wang, L. He, B. Cao, C.-T. Lu, P. S. Yu, and A. B. Ragin. Structural deep brain network mining. In *KDD*, 2017.
- [41] J. Wu, J. He, and J. Xu. Demo-net: Degree-specific graph neural networks for node and graph classification. In *KDD*, 2019.
- [42] Y. Yan, J. Zhu, M. Duda, E. Solarz, C. Sripada, and D. Koutra. Groupinn: Grouping-based interpretable neural network for classification of limited, noisy brain data. In *KDD*, 2019.
- [43] P. Yanardag and S. Vishwanathan. Deep graph kernels. In *KDD*, 2015.
- [44] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec. GNNExplainer: Generating explanations for graph neural networks. In *Adv Neural Inf Process Syst*, 2019.
- [45] Y. Zhang, D. Defazio, and A. Ramesh. RelEx: A Model-Agnostic Relational Model Explainer. *arXiv:2006.00305*, 2020.

A REPRODUCIBILITY

Our Python code and the notebooks of our experiments are available at: <https://github.com/carlo-abrate/CounterfactualGraphs>.

The main pipeline of our experiments of Section 5 is as follows:

Input: The input of the experiments is a dataset of labeled networks $D = \{(E_i, y_i)\}_{i=1}^n$ s.t. $E_i \in \mathbb{G}(V)$, $y_i \in \{0, 1\}$.

Embedding training: We use multiple embedding techniques, whose main parameters are discussed in the rest of this Appendix.

Classifier training and testing: The black-box oracle function f is built by training a classifier on the embeddings. In particular, we tested support-vector machine (SVM), k-nearest neighbors (KNN), and Random Forest on each combination of embeddings parameters setting, using a 5-fold cross validation. Finally, we selected the best dimension of the embedding and the classification algorithm based on accuracy. In the paper we presented results based on KNN in combination with all the embedding methods, with the exception of Contrast Subgraphs where we use SVM.

As discussed in Section 4, counterfactual search methods iteratively call the oracle function for new candidate counterfactual graphs. The oracle function is composed of the embedding and the classifier part, at each call the candidate counterfactual graph is considered by the embedding as a new and unseen data. Some transductive methods do not naturally generalize to unseen data, due to non-determinism. However, for both Graph2Vec and Sub2Vec we force the determinism as explained in <https://github.com/RaRe-Technologies/gensim/issues/447>.

In the following we report the parameters settings used in the paper.

Contrast Subgraphs [17]. The code can be found at <https://github.com/tlancian/contrast-subgraph>. We use the parameter $\alpha = 0.025$ to generate the contrast subgraphs for ASD dataset and $\alpha = 0.01$ for the ADHD dataset.

Graph2Vec[25]. The code can be found at <https://github.com/benedekrozemberczki/graph2vec>. The main parameters used in the experiments of Table 1 for both ASD and ADHD datasets, are as follows:

- $node_degree = 2$;
- $workers = 1$;
- $epochs = 100$;
- $min_count = 5$;
- $learning_rate = 0.025$;
- $down_sampling = 0.0001$;
- vector space dim = 50 for both the datasets.

Sub2Vec [1]. The code can be found at <https://goo.gl/Ef4q8g>. The main parameters used in the experiments of Table 1 for both ASD and ADHD datasets, are as follows:

- $type = Neighborhood$;
- $walk_length = 100000$;
- $model = DBON$;
- $iteration = 50$;
- vector space dim = 50 for both the datasets.

Autoencoders [15]. The code can be found at <https://github.com/leoguti85/GraphEmbs>. The main parameters used in the experiments of Table 1 for both ASD and ADHD datasets, are as follows:

- $noise = 0.05$;
- $nb_epoch = 150$;
- $batch_size = 128$;
- $lr = 0.001$;
- vector space dim = [200, 400] for ASD and ADHD respectively.

We finally summarize report the main parameters and settings used to generate the figures of the paper:

Fig. 1 Dataset: ASD; Patient: USM_0050453; Classifier: Contrast Subgraphs with SVM; Counterfactual Search: OFS+OBS with $k = 5$, $\eta = 2000$.

Fig. 3 and 4 Dataset: ASD; Classifier: Contrast Subgraphs with SVM; Counterfactual Search: DS, OFS+OBS and DFS+DBS, with $k = 5$, $\eta = 2000$.

Fig. 5 Dataset: ASD; Classifier: Contrast Subgraphs with SVM; Counterfactual Search OFS+OBS, with $k = 5$, $\eta = 2000$. Patients in the zoomed section are: UM_1_0050366, Stanford_0051164, UCLA_1_0051226.

Fig. 7 Dataset: ASD; Patient: MaxMun_d_0051353; Classifier: Contrast Subgraphs with SVM; Counterfactual Search: OFS+OBS with $k = 5$, $\eta = 2000$.

Fig. 8, 9 Dataset: ASD; Classifier: Contrast Subgraphs with SVM, Autoencoders, Graph2Vec with KNN; Counterfactual Search: OFS+OBS, with $k = 5$, $\eta = 2000$.