BMC Bioinformatics

RESEARCH

Open Access

Conformal novelty detection for multiple metabolic networks

Ariane Marandon[1], Tabea Rebafka[1,2], Nataliya Sokolovska[3*] and Hédi Soula[4]

*Correspondence: nataliya.sokolovska@sorbonne-universite.fr

[1] LPSM, Sorbonne university, 4 place Jussieu, 75005 Paris, France
[2] MaIAGE, INRAE, Domaine de Vilvert, 78350 Jouy-en-Josas, France
[3] LCQB, Sorbonne university, 4 place Jussieu, 75005 Paris, France
[4] NutriOmics, Sorbonne university, 91 boulevard de l'Hôpital, 75013 Paris, France

Abstract

***Background*:** Graphical representations are useful to model complex data in general and biological interactions in particular. Our main motivation is the comparison of metabolic networks in the wider context of developing noninvasive accurate diagnostic tools. However, comparison and classification of graphs is still extremely challenging, although a number of highly efficient methods such as graph neural networks were developed in the recent decade. Important aspects are still lacking in graph classification: interpretability and guarantees on classification quality, i.e., control of the risk level or false discovery rate control.

***Results*:** In our contribution, we introduce a statistically sound approach to control the false discovery rate in a classification task for graphs in a semi-supervised setting. Our procedure identifies novelties in a dataset, where a graph is considered to be a novelty when its topology is significantly different from those in the reference class. It is noteworthy that the procedure is a conformal prediction approach, which does not make any distributional assumptions on the data and that can be seen as a wrapper around traditional machine learning models, so that it takes full advantage of existing methods. The performance of the proposed method is assessed on several standard benchmarks. It is also adapted and applied to the difficult task of classifying metabolic networks, where each graph is a representation of all metabolic reactions of a bacterium and to real task from a cancer data repository.

***Conclusions*:** Our approach efficiently controls — in highly complex data — the false discovery rate, while maximizing the true discovery rate to get the most reasonable predictive performance. This contribution is focused on confident classification of complex data, what can be further used to explore complex human pathologies and their mechanisms.

Keywords: Novelty detection, Conformal prediction, Wrapper method, Metabolic networks, Graph neural networks

Background

With the rise of new sequencing and high-throughput technologies, new data in form of metabolic networks are more and more available to support the study of human pathologies. These datasets are huge and of complex structure requiring the application of appropriate machine learning models. In particular, the interest to apply predictive

models to metabolic information is extremely high in metabolic diseases tasks, such as prediction of obesity and diabetes. These pathologies result from a certain disability of a cell to breakdown or produce some essential substrates. As a result, if an enzyme in one reaction is broken, it may influence subsequent reactions, leading to enormous cascading damages [1, 2].

A metabolic network represents a complete set of metabolic and physical processes describing physiological and biochemical properties of a living cell [3]. Moreover, modern large metabolic databases such as KEGG [4] make it possible to access genomic, enzymatic and metabolic information and to reconstruct interactions. The representation of a graph is an important question. Adjacency matrices are a natural representation of metabolic networks, and the Graph Neural Networks (GNN) which we consider below, cope very reasonably with the complex data analysis. Extraction of the topological properties of graphs is another efficient solution, since it reduces the data dimension and often leads to the state-of-the-art performance [5]. So, strong correlations between phenotypical traits of organisms and the topology of metabolic networks were reported [6, 7], underlining the importance to study metabolic networks and their topology. An efficient classification of patient-specific metabolic networks from a breast cancer study using topological properties was reported by [5], illustrating a strong motivation to make use of metabolic networks in medical tasks.

From a mathematical viewpoint, a metabolic network can be represented by a graph composed of nodes and edges, which connect the nodes. The metabolites and enzymes are the nodes of the graph. Each reaction substrate is linked to the catalysing enzyme and each enzyme is connected to products of the chemical reaction [8]. Modern statistical and machine learning methods can be applied to such data to get more insights in the functioning of living organisms [9].

While the overwhelming majority of existing results concern the analysis of a single network, here we are particularly interested in the classification and comparison of multiple metabolic networks, since it is a step forward to the development of non invasive accurate diagnostic tools. Our specific goal is novelty detection, sometimes also called outlier detection. This amounts to compare metabolic networks to a reference and to decide which of the networks are significantly different from the reference. Detecting metabolic networks whose structure or topology is inherently different from the structure of a set of default or nominal graphs is crucial for the identification of anormal cells and for gaining new insights into the functionalities of different metabolisms.

Network comparison is an inherently involved problem due to the complex structure of graphs. Generally, dimensionality reduction methods are used which provide a graph embedding for every network. This can be achieved by traditional principal component analysis [10, 11] or more recently with graph neural networks [12–15]. In medical and pre-clinical research, novelty detection based on graph embeddings is generally done in a manual way, which is both time consuming for human experts and highly subjective, since a human might take a biased decision instead of using an objective criterion. Moreover, such novelty detection comes without any guarantee on the quality of the results.

In many applications, it is vital not only to make accurate predictions but also to quantify the accuracy and provide explanations on the learned model. Here, we follow the

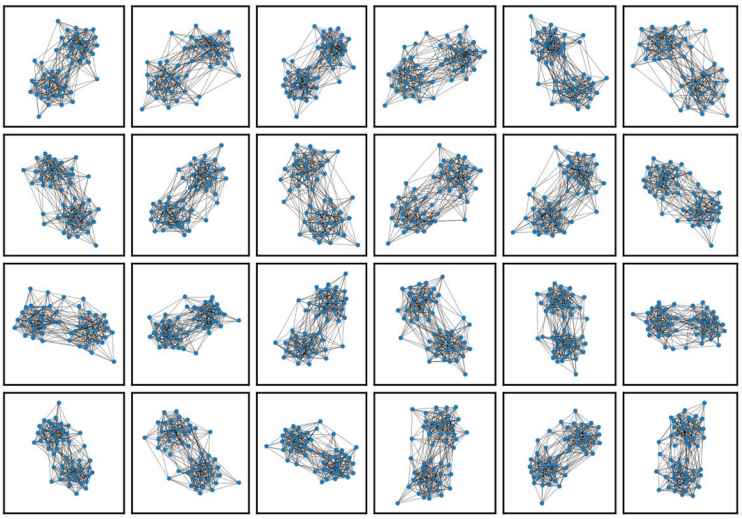
definition proposed by [16, 17] that a novelty is any newly acquired structure or property that permits the assumption of a new function. With the hope that studies about metabolic alterations in organisms help to decipher mechanisms of pathologies, [18] discusses novelty detection and tests several classification methods to detect novel metabolites of a breast cancer-tissue sample that can appear during dynamic evolution of biological cells. It was also recently reported that metabolic changes, i.e., novelties, can help in diagnostic and therapeutic investigations; a repository of tumor metabolic networks derived from Genome-Scale Metabolic Models, called TumorMet [19], is publicly available.

More precisely, any novelty detection method may make mistakes either by declaring observations as novelties while they are not, or by not recognizing a new observation as a novelty. Depending on the difficulty of the underlying problem, that is, whether novelties are completely different from the reference or still share some similarities, the number of errors for a given method may vary greatly. Traditional machine learning methods do not provide any information on the quality of its results. However, recently, new procedures have been developed that come with a statistical guarantee that the *set of detected novelties* contains at most a given percentage of falsely detected novelties, while keeping the number of identified novelties as large as possible. One of such approaches is conformal prediction [20, 21] first introduced in classification and regression settings. Figure 1 illustrates the novelty detection task and the possible error types that a procedure can make.

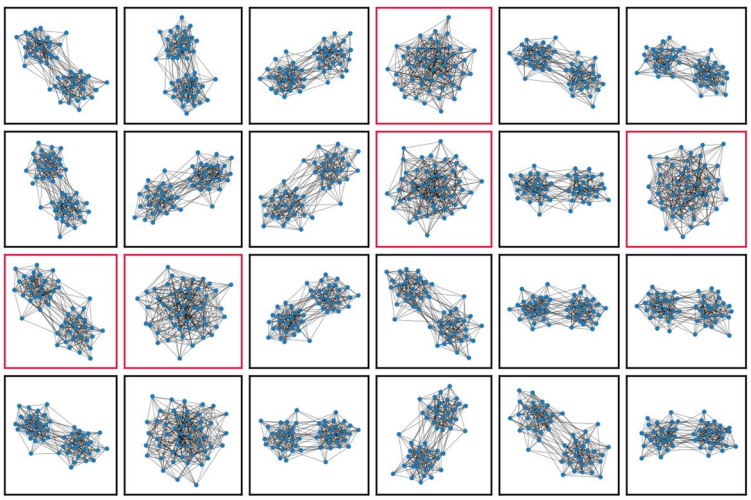
A major advantage of conformal prediction is that it does not make any assumptions on the type of distribution of the observations, which is important for applications where distribution assumptions on the data are hard to verify and/or rarely satisfied. Moreover, the reference or nominal distribution is not assumed to be known, but it is sufficient to dispose of a sample from the reference distribution, that is, a semi-supervised setting is considered. Here, we use the notion of the semi-supervised scenario introduced by [22]. Such a general framework makes conformal prediction highly attractive for uncertainty quantification on complex structures such as networks. Moreover, conformal prediction is known to provide non-asymptotic and distribution-free coverage guarantees for various tasks [23, 24]. Finally, it is extremely noteworthy that conformal prediction is a wrapper around traditional machine learning models, that is, it can be directly applied to the output of existing methods. As such, conformal prediction takes full advantage of existing high-performance machine learning algorithms.

In a recent series of works, conformal procedures for the novelty detection task have been developed [22, 25–29]. The general idea is to, first, learn a non-conformity score for all new observations using some existing machine learning algorithm. Then a comparison of these scores to the scores of the reference observations provides the final set of detected novelties. For the selection of the final set of novelties, tools from multiple testing are used, where finite-sample guarantees on the error rate can be obtained. The procedure AdaDetect proposed by [30] is the most powerful approach up to date, which is based on a particularly efficient way of learning the non-conformity scores, and it is appropriate for a huge variety of settings.

Novelty, or outlier, detection in graphs is particularly challenging due to the complex structure of the data. Some very recent attempts to extend conformal prediction



(a) Reference sample



(b) Test sample with declared novelties

Fig. 1 Illustration of the data sets, the novelty detection task and possible errors. All networks in the reference set are composed of two communities, while the test set also contains networks with one community. The novelty detection method identifies most of the novelties (in red) correctly, but also falsely declares a reference network as a novelty

to graphs concern either link prediction [31] or the prediction of node labels [32, 33] in a single graph. A recently developed procedure for the problem of novelty detection in a collection of networks is given in [34], where a simple outlier detection method is proposed; this approach is applied to graphs in neuroscience and relies on a hierarchical generalized linear model, but not on conformal inference.

Our aim here is to extend AdaDetect to the specific task of novelty detection in a collection of metabolic networks. Our goal is to show its usefulness in practice and illustrate the gain of new insights on the cell metabolism. Our contribution is multi-fold:

- We propose a statistically sound method that identifies the networks in a data set which are significantly different from a set of provided reference observations and that controls the corresponding false discovery rate. Our approach applies to any type of complex networks, not necessarily to metabolic networks. *To our best knowledge, we are the first ones to propose conformal outlier detection procedure for a collection of graphs.*
- We discuss theoretical guarantees of our method, and the ability of the proposed procedure to outperform some existing state-of-the-art baseline methods.
- We show the generalizing performance of the newly introduced model on several benchmarks.
- We validate our method on a real data set of bacteria, where each bacterium is represented by its metabolic network. We also apply the proposed approach to two tasks – Lung and Kidney cancer – of the TumorMet repository.

Note that both novelty and outlier detection are methods to identify anomalies. However, traditionally, in an outlier detection task a dataset contains outliers which should be detected. Therefore, the outlier detection is an unsupervised anomaly detection approach. In a novelty detection task, a learning algorithm has access to the nominal class of observations, and the novelties are not observed. Such a novelty detection scenario is called semi-supervised, since only one class of labeled data is available. More details on the semi-supervised novelty detection framework – and our contribution is in this context – can be found in [35].

Method: AdaDetect for Graphs

Setting and notations

We consider the general setting with observed networks denoted by $G = (A, X)$, where A is the adjacency matrix of the network and X is a matrix of node covariates (if available). Networks may be directed or undirected, binary or valued, share the same nodes over all networks or not, may have varying number of nodes from one network to the other or include node covariates.

In the semi-supervised framework, two sets of networks are observed. First, $\mathcal{G}_{\text{ref}} = \{G_i, i \in [\![1, n]\!]\}$ is a set of networks having the standard or normal behavior, referred to as the *reference sample*. Here $[\![a, b]\!]$ denotes the set of integers from a to b. These networks are assumed to be i.i.d. realizations of some distribution P_{ref}, which is unknown to the user. That is, $G_i \sim P_{\text{ref}}, i \in [\![1, n]\!]$. The second set of observed networks denoted by $\mathcal{G}_{\text{test}} = \{G_i, i \in [\![n+1, n+m]\!]\}$ is the *test sample*, where the observed networks are assumed to be independent, not observed during training. The task is to decide which of them are novelties, that is, which networks do not come from the reference distribution P_{ref}. To put it differently, the aim is to discover the set $\mathcal{I}_{\text{nov}} = \{i \in [\![n+1, n+m]\!], G_i \not\sim P_{\text{ref}}\}$, which is the set of indices of the novelties. Furthermore, denote $\mathcal{I}_{\text{test}} = [\![n+1, n+m]\!]$ the set of indices of networks in $\mathcal{G}_{\text{test}}$ and $\mathcal{I}_{\text{ref}} = \mathcal{I}_{\text{test}} \setminus \mathcal{I}_{\text{nov}}$ the set of indices i of networks in $\mathcal{G}_{\text{test}}$ from the reference distribution, that is $G_i \sim P_{\text{ref}}$.

Now a novelty detection procedure is a (measurable) function R that returns a subset of $\mathcal{I}_{\text{test}}$ corresponding to the indices of the networks declared as novelties. The false

discovery rate (FDR), that is the proportion of falsely declared novelties, and the true discovery rate (TDR), that is the proportion of correctly identified novelties, are defined as

$$FDR(R) = \mathbb{E}\left[\frac{|\mathcal{I}_{ref} \cap R|}{1 \vee |R|}\right], TDR(R) = \mathbb{E}\left[\frac{|\mathcal{I}_{nov} \cap R|}{1 \vee |\mathcal{I}_{nov}|}\right]$$

Our goal is to find a procedure that controls the FDR at a prescribed level α, that is $\mathrm{FDR}(R) \leq \alpha$, and whose power or TDR is as large as possible.

AdaDetect

The general idea of conformal novelty detection [20, 22, 25, 36] is to compare the non-conformity score of a test observation to the scores of the reference observations to decide whether the observation is a novelty or not. In the seminal work of [25], the FDR control is shown to be guaranteed when proceeding as follows: the reference sample $\mathcal{G}_{\mathrm{ref}}$ is split into two parts and the score is trained using one of the parts. The power of the procedure, that is the number of correctly detected novelties, depends on the quality of the scores. In this line, [27] recently achieved an important improvement by training the score not only on the reference sample $\mathcal{G}_{\mathrm{ref}}$ with a one-class classification algorithm, but training the score using a two-class classification method including the test sample $\mathcal{G}_{\mathrm{test}}$. This procedure is called AdaDetect and yields a significant gain in power, while still controlling the FDR. In this work, we work out how to successfully apply AdaDetect to the task of novelty detection in network data.

Algorithm 1 AdaDetect for networks

Input: Set of reference networks $\mathcal{G}_{\mathrm{ref}}$, set of test networks $\mathcal{G}_{\mathrm{test}}$, α desired risk level.
Output: Set of declared novelties $\{j \in \mathcal{I}_{\mathrm{test}}, S_j > \delta_{\mathrm{best}}\}$.

1. Split $\mathcal{G}_{\mathrm{ref}}$ into two parts, $\mathcal{G}_{\mathrm{train}}$ and $\mathcal{G}_{\mathrm{cal}}$.
2. Label the graphs in $\mathcal{G}_{\mathrm{train}}$ as "0" and the graphs in $\mathcal{G}_{\mathrm{cal}} \cup \mathcal{G}_{\mathrm{test}}$ as "1".
3. Learn a graph classifier g of class "0" versus class "1" on all data such that g returns the score or probability of a network to belong to class "1".
4. Let $\mathcal{I}_{\mathrm{cal}} \subset \mathcal{I}_{\mathrm{ref}}$ be the set of indices for which $G_i \in \mathcal{G}_{\mathrm{cal}}$. For every $i \in \mathcal{I}_{\mathrm{cal}} \cup \mathcal{I}_{\mathrm{test}}$, compute the non-conformity score $S_i = g(G_i)$.
5. For $j \in \mathcal{I}_{\mathrm{test}}$, consider the novelty detection procedure R_{S_j} with threshold $\delta = S_j$ and compute the proportion

$$p_j = \frac{|\{i \in \mathcal{I}_{\mathrm{cal}} : S_i > S_j\}| + 1}{|\mathcal{I}_{\mathrm{cal}}| + 1}.$$

6. Determine the smallest threshold S_j such that $p_j < \alpha$, that is,

$$\delta_{\mathrm{best}} = \arg\min\{S_j, j \in \mathcal{I}_{\mathrm{test}}, p_j < \alpha\}.$$

In detail, in AdaDetect the non-conformity score is a classifier trained on the following problem. First, the reference sample $\mathcal{G}_{\mathrm{ref}}$ is split into two subsets, say $\mathcal{G}_{\mathrm{train}}$ and $\mathcal{G}_{\mathrm{cal}}$. The networks in $\mathcal{G}_{\mathrm{train}}$ are labeled as "0" and the elements of $\mathcal{G}_{\mathrm{cal}} \cup \mathcal{G}_{\mathrm{test}}$ as "1". That is, $\mathcal{G}_{\mathrm{train}}$ is a pure class, where all observations come from the reference distribution P_{ref},

while $\mathcal{G}_{\text{cal}} \cup \mathcal{G}_{\text{test}}$ is mixed, containing observations from the reference distribution P_{ref} as well as novelties. A classifier can be trained using any machine learning classifier for graphs. The classifier function, say g, gives the non-conformity score, which generally corresponds to the probability of a network to belong to class "1".

Now denote by $\mathcal{I}_{\text{cal}} \subset \mathcal{I}_{\text{ref}}$ the set of indices for which $G_i \in \mathcal{G}_{\text{cal}}$. For every $i \in \mathcal{I}_{\text{cal}} \cup \mathcal{I}_{\text{test}}$, compute the non-conformity score $S_i = g(G_i)$. Now it is reasonable to declare all networks that have a large score, or more precisely, whose score is larger than some threshold δ, as novelties. That is, we consider the novelty selection procedure R_δ defined as

$$R_\delta = \{j \in \mathcal{I}_{\text{test}}, S_j \geq \delta\}.$$

As the choice of the threshold is crucial for the control of the FDR, we introduce the quantity p_δ defined as the proportion of reference observations declared as novelties by procedure R_δ:

$$p_\delta = \frac{|\{i \in \mathcal{I}_{\text{cal}} : S_i > \delta\}| + 1}{|\mathcal{I}_{\text{cal}}| + 1}.$$

That is, p_δ is an approximation of the FDR of procedure R_δ. Thus, for our procedure we choose the smallest threshold δ such that the corresponding proportion p_δ is lower than α, which is the desired risk level. The justification of this approach is that it is impossible for any classifier to learn to distinguish reference observations in class "0" from reference observations in class "1". Thus, the distribution of the scores $\{S_i, i \in \mathcal{I}_{\text{cal}}\}$ equals the distribution of the score under P_{ref}. Hence, $\{S_i, i \in \mathcal{I}_{\text{cal}}\}$ is appropriate for a comparison of the scores of the test observations for fixing the threshold δ that yields the FDR control.

The procedure is summarized in Algorithm 1. Figure 2 schematically shows the proportion of falsely declared novelties for different thresholds. [27] illustrate that AdaDetect is more powerful than state-of-the art procedures, that is, it detects more novelties than other methods.

Note that while the FDR control holds regardless of sample sizes, in practice, the sample size of the calibration set $\mathcal{G}_{\text{cal}}$ must be large enough to ensure a good power [22, 25, 27]. However, increasing $|\mathcal{G}_{\text{cal}}|$ and hence the proportion of references networks in the mixed set $\mathcal{G}_{\text{cal}} \cup \mathcal{G}_{\text{test}}$ degrades the quality of the scores learned by the classifier in AdaDetect. Based on power results from [22, 27], we recommend to choose $|\mathcal{G}_{\text{cal}}|$ to

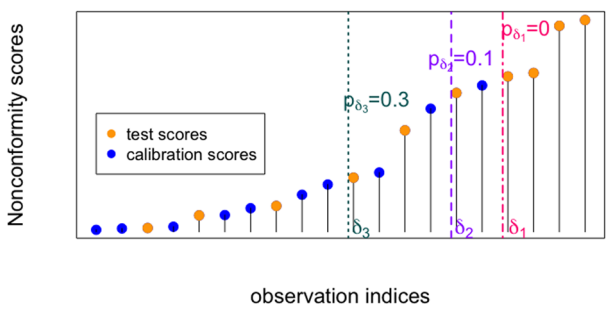


Fig. 2 Choice of threshold δ for the novelty detection procedure R_δ. Illustration of proportion p_δ of falsely declared novelties for three different thresholds

be of the same order as the test sample size $|\mathcal{G}_{\text{ref}}|$. Ideally, $\mathcal{G}_{cal}$ has to be as large as possible. However, we defined $\mathcal{G}_{cal} = \mathcal{G}_{train} = n/2$, where n is the limited number of available observations. If we increase $\mathcal{G}_{cal}$, our $\mathcal{G}_{train}$ will be smaller what will lead to a worse performance.

Machine learning algorithms for graph classification

Graph classification has garnered significant interest in recent years with many new methods, especially in the field of deep learning [37]. A particular feature of graphs is that in general no natural order of the vertices exits, but that classifiers are required to be invariant to the reordering of the nodes. This is one of the main obstacles to extending classical ML methods to graph data. Hence, as networks are complex data objects, their comparison is a challenging task and different methods handle this question in widely different ways. The choice of an appropriate approach also depends on the characteristics of the data at hand such as the availability of node features, the absence or presence of node correspondence, or whether the networks are directed or not. In this section, we present the general approaches to graph classification and we discuss their principal properties. The two main approaches are graph kernels [38] and graph neural networks (GNNs) [37, 39–42].

Graph kernels are the historically dominant approach for graph classification. A graph kernel is a deterministic function defining a similarity measure between a pair of networks, that can be combined with a support vector machine (SVM) classifier for supervised learning. The most popular graph kernel is the Weisfeiler-Lehman (WL) kernel [43], which is suited for networks with discrete node attributes. It is based on the 1-WL or color refinement algorithm, which proceeds as follows: First, for every network a graph embedding is computed according to some message passing mechanism. In detail, for every node its label (or color) is aggregated with the labels of its neighbors yielding a fingerprint. Then all nodes with identical fingerprint are assigned a new common node label (or color).

When this procedure is iterated, say K times, this results in a node embedding describing the structure of the K-hop neighborhood of every node, where nodes with identical K-hop neighborhoods share the same label. In other words, the node embeddings define a node clustering. For the WL subtree kernel [43] the clusterings obtained at all iterations are used to build a graph embedding with multiple resolutions. Finally, the WL kernel of two graphs is defined as the inner product of their graph embeddings. The WL algorithm gives rise to the most powerful existing test to decide whether two graphs are isomorphic, that is, whether one of the graphs can be obtained from the other by a permutation of the nodes [44].

Many other graph kernels have been proposed in the literature, see [38] for a complete review, some of them take into account discrete or continuous node attributes. A general main drawback of graph kernels is the computational burden that comes with computing the kernel function for every pair of networks in the training sample. The running time is $\mathcal{O}(NKn_{\max} + N^2Km_{\max})$, where $n_{\max}$ and $m_{\max}$ are the maximum number of vertices and edges in a collection of N graphs. In our case $N = |\mathcal{G}_{\text{cal}}| + |\mathcal{G}_{\text{test}}|$.

Graph neural networks (GNNs) are recent approaches for graph-based learning, that aim to scale to larger datasets than graph kernels by generalizing neural networks

(NNs) to graph-structured data. Specifically, GNNs are permutation-invariant functions that produce a vector representation of each node in a network, using a combination of linear and non-linear operations. This vector representation is based on a neighborhood aggregation scheme, where, given an initial representation of the nodes, node representations are updated by taking into account their neighbors (e.g. computing the sum, mean or max of the representations). In that view, GNNs can be seen as a neural network version of the 1-WL algorithm. As GNNs produce a matrix of node embeddings, they can be combined with NN layers for node classification [45] or link prediction [46]. Moreover, graph classification can be performed by collapsing the node representations of a network into a single vector representation, and by feeding this graph representation into NN layers for end-to-end learning [40]. More generally, a GNN-based approach for graph classification is a composition of functions (or layers) of two types: 1) GNN layers (also called *graph convolutional* layers), which produce node representations based on the graph structure and node features (or initial node representations), and 2) *pooling* layers that, depending on their role in the architecture, either coarsen the network into a smaller one or produce a graph representation that can be used in a NN (also called a *read-out* layer) for end-to-end learning. Figure 3 provides a schematic illustration.

Here we present several GNN-based approaches suitable to use with AdaDetect, chosen for their popularity, theoretical justification and interpretability.

- **GIN by** [40]. In general, Graph Isomorphism Network (GIN) denotes a type of GNN layer, in which the new node representations are obtained by

$$X' = \mathrm{MLP}\Big((A + I)X\Big),$$

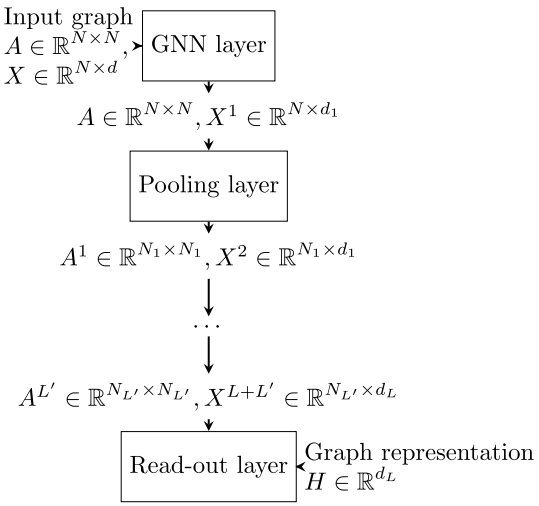


Fig. 3 A typical GNN architecture for graph classification: N denotes the number of nodes of the input graph (A, X), L and L' denote the number of GNN and pooling layers in the architecture, respectively, without the read-out layer. Each GNN layer applies a non-linear transformation on the current graph representation A^l and the current node embedding matrix X^l (initially A, X) and produces an updated node embedding matrix X^{l+1}, whereas each pooling layer coarsens the graph representation A^l (initially A) into A^{l+1}. The final layer (read-out layer) takes the embedding matrix $X^{L+L'}$ and transforms it into a one-dimensional embedding

where MLP is a multilayer perceptron, A an adjacency matrix, I the identity matrix, and X is a node representation. For the task of graph classification, [40] propose to combine several GIN layers with a read-out layer that consists in summing up node representations. Moreover, the authors prove that GIN has favorable theoretical properties, namely that GIN is as powerful as the 1-WL test.

- **DiffPool by** [41]. GIN and most other GNNs are rather flat networks, so that they can only capture local patterns. To learn graph properties on a higher level, differentiable graph pooling (DiffPool) combine GNN layers with pooling layers that successively coarsen the graph. Coarser graphs may represent more global features of the initial graph. In each pooling operation of DiffPool, a new (coarsened) graph and new node representations are obtained by

$$A' = S^T AS, \quad S = \text{softmax}\Big(\text{GNN}_{\text{pool}}(A, X)\Big)$$
$$X' = S^T \text{GNN}_{\text{embed}}(A, X)$$

where GNN_{pool} and $\text{GNN}_{\text{embed}}$ are GNN layers (e.g. GIN layers). The matrix S represents a (differentiable) clustering of the nodes that is used to coarsen the input graph A. A final vector-valued graph representation is obtained using a standard read-out layer.

- **DGCNN by** [39]. To extend convolution neural networks (CNN) to graph-structured input, [39] introduce SortPool, a special kind of read-out layer that also acts as a coarsening operation. The SortPool layer produces a sorted representation of the nodes, such that applying classical one-dimensional CNN layers to these representations makes sense. Moreover, SortPool unifies graph sizes by truncating/extending all sorted representations to the same length, say k (where k is e.g. such that 50% of the graphs have more than k nodes). Dynamic graph CNN (DGCNN) refers to the architecture that results from combining GNN layers with SortPool and one-dimensional CNN layers.

On the one hand, GNNs have more flexibility in that they can easily take into account various characteristics of the network data, such as node attributes of any type, node correspondance, or directed edges. Moreover, graph kernels suffer from a certain computational burden in terms of the number of networks at hand. On the other hand, GNNs inherit from the lack of interpretability of NNs. Finally, while GNNs support end-to-end learning (whereas graph kernels produce fixed embeddings), their expressivity power is limited, since concerning the task of distinguishing networks GNNs cannot be more powerful than the 1-WL algorithm [40, 44].

Results

Real complex graphs

In this section, we apply the method introduced above to real datasets. For the purpose of illustration, we use graph classification datasets, where all data are labeled. A summary of the considered datasets downloaded from [47] is given in Table 1. The datasets DD and PROTEINS encode information about macromolecules. The nodes of

Table 1 Summary of the used datasets

	DD	PROTEINS	AIDS	NCI1
Number of graphs	1178	1113	2000	4110
Average number of nodes	284.32	39.06	15.69	29.87
Average number of edges	715.66	72.82	16.20	32.30
Number of covariates	89	29	4	0

Table 2 Settings

Dataset	*m*	*n*
DD	100	300
PROTEINS	100	300
AIDS	500	1500
NCI1	500	1500

PROTEINS represent secondary structure elements, and an edge exists if they are neighbours along the amino acid sequence. In DD, the nodes are amino acids and the edges refer to the spacial proximity. NCI1 represents chemical compounds where nodes are atoms and edges are bonds between the atoms. This dataset is relative to the cell lung cancer task. AIDS represent molecular compounds from the Antiviral Screen Databased of Active Compounds.

In our novelty detection task, we consider the graphs of one class as anomalies, and the remaining observations as references. In each task, we construct test samples and training samples by subsampling the dataset. We choose the test samples $\mathcal{G}_{\text{test}}$ such that half of its networks are novelties, that is $|\mathcal{I}_{\text{ref}}|/|\mathcal{I}_{\text{nov}}| = 0.5$, and the size of the test samples $m = |\mathcal{G}_{\text{cal}}|$ as given in Table 2. The size of the reference sample is $|\mathcal{G}_{\text{ref}}| = n = 2m$ and for the calibration set $|\mathcal{G}_{\text{cal}}| = m$.

We apply AdaDetect with each of the graph classification methods described in the previous section: the GNN-based approaches GIN, DGCNN, and DiffPool, and one graph kernel-based approach, using the WL kernel, leading to the procedures `AdaDetect-GIN`, `AdaDetect-DGCNN`, `AdaDetect-DiffPool` and `AdaDetect-WL`. For each GNN, the architecture consists of 3 layers and 32 neurons and we train for 10 epochs with a learning rate of 0.001, and the WL kernel is used with 5 iterations. Moreover, we compare our results to the conformal anomaly detection (CAD) procedure proposed by [25], using one-class classification approaches: the one-class classifier given by the Support Vector Data Description (SVDD) method introduced in [48] with a family of functions given by either GIN, DiffPool, or DGCNN, and a one-class SVM using the WL kernel, which gives the procedures `CAD-GIN`, `CAD-DGCNN`, `CAD-DiffPool`, and `CAD-WL`.

The FDR and TDR for the methods are evaluated on 100 subsampled data sets and the results are reported in Table 3. First, we observe that all methods control the FDR at the nominal level $\alpha = 0.2$. Most often the AdaDetect version achieves a larger FDR than its CAD counterpart. Concerning the TDR, values vary a lot over the four settings and

Table 3 Performance of different methods on different data sets in terms of FDR (top) and TDR (bottom) with nominal level is $\alpha = 0.2$. Mean values and standard deviations (in parentheses) over 100 subsampled data sets

		DD	PROTEINS	AIDS	NCI1
		FDR			
GIN	AdaDetect	**0.05** (0.10)	0.04 (0.12)	0.10 (0.10)	**0.04** (0.11)
	CAD	0.04 (0.11)	**0.05** (0.13)	**0.19** (0.08)	**0.04** (0.10)
DiffPool	AdaDetect	**0.02** (0.05)	0.01 (0.05)	**0.06** (0.08)	0.00 (0.01)
	CAD	0.00 (0.02)	**0.03** (0.13)	0.04 (0.08)	0.00 (0.00)
DGCNN	AdaDetect	**0.11** (0.12)	**0.08** (0.12)	**0.19** (0.07)	0.03 (0.08)
	CAD	0.03 (0.09)	0.04 (0.12)	0.10 (0.10)	0.03 (0.11)
WL	AdaDetect	**0.10** (0.12)	**0.08** (0.12)	**0.19** (0.06)	**0.06** (0.11)
	CAD	0.08 (0.04)	0.06 (0.04)	0.09 (0.04)	0.01 (0.05)
		TDR			
GIN	AdaDetect	**0.10** (0.20)	**0.04** (0.10)	0.42 (0.42)	0.00 (0.00)
	CAD	0.04 (0.12)	0.03 (0.09)	**0.87** (0.20)	***0.02*** (0.03)
DiffPool	AdaDetect	**0.04** (0.12)	**0.03** (0.06)	**0.62** (0.42)	0.00 (0.00)
	CAD	0.00 (0.01)	0.01 (0.04)	0.17 (0.24)	0.00 (0.00)
DGCNN	AdaDetect	**0.27** (0.26)	***0.15*** (0.12)	***0.95*** (0.11)	**0.01** (0.02)
	CAD	0.10 (0.17)	0.05 (0.11)	0.27 (0.26)	0.00 (0.01)
WL	AdaDetect	0.22 (0.18)	0.12 (0.12)	**0.89** (0.07)	**0.01** (0.03)
	CAD	***0.29*** (0.10)	**0.15** (0.10)	0.49 (0.01)	0.00 (0.00)

The optimal values are shown in bold

the different procedures, ranging from 0.00 to 0.95. On the data sets DD, PROTEINS and AIDS, AdaDetect with any GNN classifier outperforms the corresponding CAD version (with one exception), illustrating an important gain in power due to the AdaDetect approach. This is in line with the properties of AdaDetect reported in [27]. Concerning WL, depending on the setting, CAD is slightly doing better than AdaDetect in terms of power. The data set NC11 appears to be an inherently difficult setting as none of the procedures detects many novelties and also the FDRs are far below the nominal level α.

Metabolic networks of bacteria

In this section AdaDetect is applied to a database of metabolic networks. To illustrate the performance of AdaDetect, we construct several novelty detection setups using different characteristics of the bacteria as class labels and compute the associated FDR and TDR.

Data description

The data set contains 5610 prokaryotic species from the KEGG database [4]. Among them, there are 301 archaea and 5309 bacteria. The taxonomic information was obtained from the National Center for Biotechnology Information (NCBI) Taxonomy database [49]. The reconstructed networks were provided by [8]. All information on the species was extracted in November/December 2019. The following characteristics of the bacteria are provided, which we use to build different novelty detection tasks:

- Oxygen tolerance: 917 Aerobe, 782 Facultative anaerobe, 532 Anaerobe
- Habitat: 554 Symbionts, 395 Environment, 235 Mixed

In Table 4 we provide the number of nodes, number of edges, degree assortativity, density, and average degree connectivity for the classes of bacteria. We provide the average values with the standard deviation per class. Note that the number of nodes is the same here, since we followed the network reconstruction procedure used in [7, 8] where the networks are reconstructed using the same set (the union) of nodes.

Experimental setup

We use the characteristics provided above to define groups of bacteria, which are then used to construct several novelty detection tasks, by labelling a bacterium as either a reference or a novelty depending on which group it belongs to. Table 5 describes which groups are considered as references or novelties in each setup.

In each task, we construct test samples and training samples by using the complete set of novelties and a random subset of the references as test observations, with the remaining references used for the training sample: the sample sizes $|\mathcal{I}_{\text{ref}}|, |\mathcal{I}_{\text{nov}}|, n$ are given in Table for each scenario. We set $\alpha = 0.1$ and use $|\mathcal{G}_{\text{cal}}| = |\mathcal{G}_{\text{train}}| = n/2$ for splitting the training sample.

When applying AdaDetect to these data, we observed that results are unstable and depend on the random split of the reference set $\mathcal{G}_{\text{ref}}$ into subsets $\mathcal{G}_{\text{train}}$ and $\mathcal{G}_{\text{cal}}$. This indicates that the sample size of the reference set is small compared to the variability of the networks in the reference set. To solve this instability issue, we choose to apply each method 10 times with 10 different splits of $\mathcal{G}_{\text{ref}}$ and consider the union of all detected networks as the final set of detections. We tested different number of layers, neurons, and epochs in our deep architectures. We found out that the optimal values are 3 layers with 32 neurons. We performed 10-fold cross-validation to fix these parameters. Note that we aimed to do sustainable learning, to keep the deep architecture as flat as possible, without degrading its performance, and to avoid unnecessary re-training and excessive number of epochs. We are aware of the stability and variability problems, however, in real applications, we are limited to a number of data provided by a study. Speaking about imbalanced dataset, we are completely aware of this problem. To overcame the problem of the imbalanced data, for each run of our experiments, we select the same number of observations from each class of bacteria.

Note that metabolic networks are too large for the WL algorithm, so we only consider AdaDetect with the three GNNs. The FDR and TDR are displayed in Fig. 4 for 50 randomly constructed samples. We observe that the FDR is controlled (or close to) in all settings for all methods. Concerning the power, GIN makes only very few detections, while AdaDetect with DiffPool and DGCNN are powerful procedures and depending on the setting, one or the other achieves a better TDR. We tested the state-of-the-art GNN. The relatively poor results of the GIN can be explained as follows: the GIN is unstable in tasks with big graphs [50]. The DiffPool and DGCNN are more robust to huge structures, and the explanation is probably in the pooling layers present in both architectures [51].

We also performed the experiments using the topological properties of the graphs. We extracted the topological features from the adjacency matrices, we performed training (using the cross-validation and choosing the optimal hyper-parameters) and we tested on this newly created data set. For the choice of the topological features, we got inspired

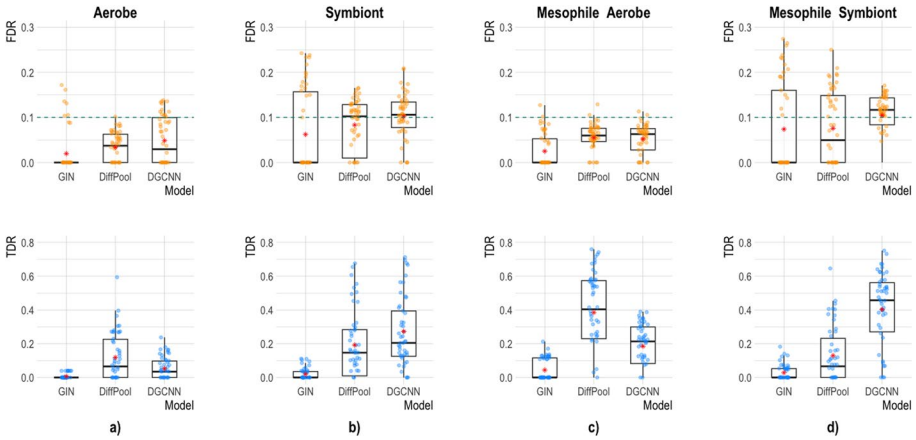


Fig. 4 Data represented as graphs: FDR (upper row) and TDR (lower row) for AdaDetect procedures with $\alpha = 0.1$ (dashed green line) for the setups described in Table . Each boxplot is based on 50 data sets. Red points indicate mean values

by the topological features considered in [5]. Figure 5 illustrates our results using the topological features: average degree connectivity, degree centrality, betweenness centrality, degree assortativity coefficient, edge density, and average hierarchical flow. The classification methods are Random Forest (RF) with 100 estimators and depth of trees equal to 10, MLP (Multilayer Perceptron) with 3 layers and 5 neurons in each layer, and Gradient Boosting (GradBoost) with 100 estimators. We observed that the deep learning architectures, DiffPool and DGCNN reach a better performance. We conclude that

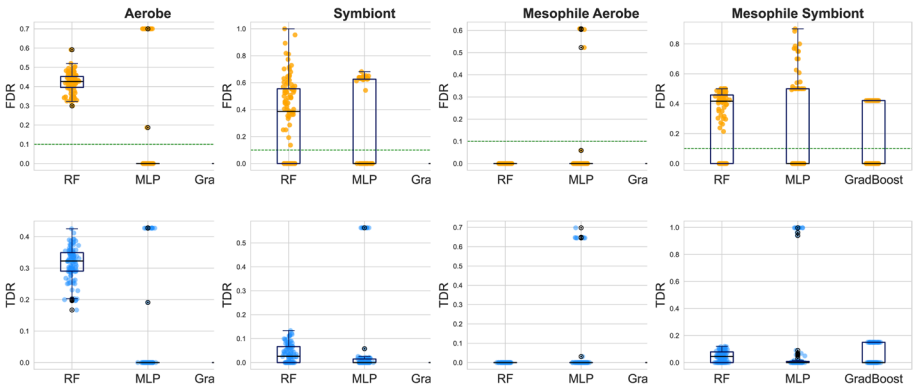


Fig. 5 Results of the experiments based on the extracted topological features: FDR (upper row) and TDR (lower row) for AdaDetect procedures with $\alpha = 0.1$ (dashed green line) for the setups described in Table . Each boxplot is based on 50 data sets

Table 4 Topological characteristics of the bacteria dataset

	Aerobe	Symbiont	Mesophile Aerobe	Mesophile Symbiont
Number of nodes	6554.0	6554.0	6554.0	6554.0
Number of edges	1992.5 ± 584.45	1543.12 ± 741.16	2077.14 ± 558.7	1649.9 ± 691.37
Degree assortativity	-0.06 ± 0.02	-0.03 ± 0.04	-0.06 ± 0.03	-0.04 ± 0.04
Density	$4.6 \cdot 10^{-5} \pm 10^{-5}$	$3.6 \cdot 10^{-5} \pm 4.8^{-5}$	$4 \cdot 10^{-5} \pm 10^{-5}$	$3.8 \cdot 10^{-5} \pm 10^{-5}$
Average degree connectivity	2.14 ± 0.15	2.01 ± 0.19	2.13 ± 0.14	2.02 ± 0.16

Table 5 Description of the novelty detection tasks considered for the metabolic network dataset

	References	Novelties	$\lvert \mathcal{I}_{\text{ref}} \rvert$	$\lvert \mathcal{I}_{\text{nov}} \rvert$	n
a)	Aerobe	Anaerobe, Facultative	417	1314	500
b)	Symbiont	Environment, Mixed	254	630	300
c)	Mesophile Aerobe	Mesophile Anaerobe, Mesophile Facultative	216	874	300
d)	Mesophile Symbiont	Mesophile Environment, Mesophile Mixed	167	324	200

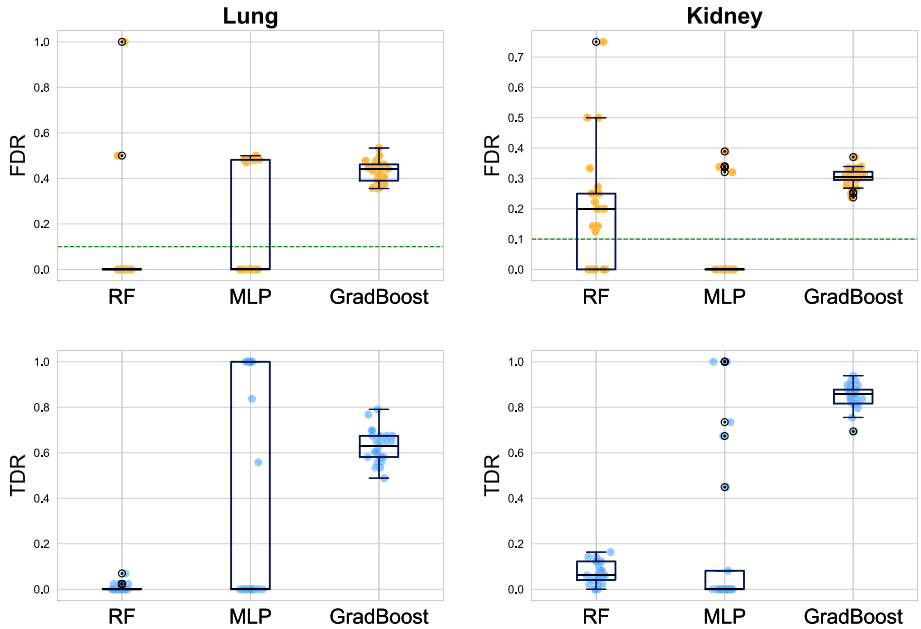


Fig. 6 Results of the numerical experiments on Lung and Kidney tasks from the repository of tumor metabolic networks [19]

taking into account the whole graph is beneficial in the current application, since the resulting estimated probabilities to belong to a class are estimated more accurately compared to a dataset based on the topological features.

Metabolic networks of cancer patients

We test the proposed approach and run several experiments on the rich recent dataset *TumorMet* [19]. In particular, we applied our method on two tasks of the repository, namely, on Lung data (TCGA-LUAD, 428 patients; TCGA-LUSC, 400 patients) and Kidney data (TCGA-KIRC, 483 patients; TCGA-KIRP, 253 patients). We focused on the metabolites-based networks where each network corresponds to a patient, the edges are weighted using the gene expression. Therefore, the graphs are directed and weighted. More detailed description of the data and the topological properties of the graphs can be found in [19]. Figure 6 illustrates the results of the numerical experiments on the Lung and Kidney tasks. We can conclude that although our results are positive, they also confirm that controlling FDR and maximising TDR are challenging problems, whose results

heavily rely on a classification model and on the estimated probabilities to belong to classes.

Discussion and Conclusions

Conformal prediction is an emerging direction in medical and biological applications, since statistical machine learning models have to be developed and applied with caution, especially in high-stake domains.

We propose a powerful tool, applicable to complex structures, to control the desired risk level. To our best knowledge, we are the first ones to challenge this task. Although our method achieves remarkable performance, there is still room for further research. So, our next step is to increase the interpretability of graph embeddings used in the method. Note that different deep or flat models lead to various graph embeddings or representations, and it is an important question how to find an optimal embedding. Particularly, the metabolic networks are huge complex graphs, and their reconstruction and representation can vary according to the scientific aim, e.g., some ubiquitous metabolites can be omitted, enzymes can be included or not, an algorithm of network reconstruction can easily result in different graphs; the direction of reactions is not unique and can also vary according to an ontology used for the graph reconstruction.

The unique role of our approach is the control of the desired risk level of prediction, i.e., our method returns results with guarantees on the classification quality. Such a guarantee is absolutely needed to control uncertainty in the prediction models: if a model is uncertain, it is better to abstain from taking decision. The task to cope with the uncertainty is even more challenging for complex data, such as metabolic networks. In our contribution, we show that our method controls the empirical error at a desired (5%, 10%, or 20%, depending on a task) level.

Note that our conformal prediction approach can be used in combination with any classifier. The reason to consider GNNs here is that graph features are not chosen manually, but graph representations are learned automatically from the data by the GNN.

An open ambitious question is how to relate the data representation and the FDR control, and whether a unified efficient framework can be proposed and developed. Another general question is the anomaly detection with techniques different from the conformal prediction. So, in recent years the detection of adversarial examples was identified as an important issue, and [52] proposed to tackle this problem with ensemble approaches. This setting is similar to ours in some aspects, however, the ensemble method of [52] makes its final decision based on a vector of scores. Note that our method is a statistically sound method based on a non-conformity score. One of the promising research avenues is to study how to combine several scores (e.g., from an ensemble method). The answer to this problem is not straightforward, since it is not clear how to get statistical guarantees with an ensemble approach.

We also plan to apply the method to different tasks and, in particular, to medical challenges. We started to explore the recent rich repository TumorMet [19] dedicated to various pathologies and that contains different metabolic graph representations. Such datasets have to be explored more thoroughly.

Finally, in this contribution is focused on the confident classification. However, much more deeper analysis of the identified novelties in graphs and tight collaboration with

human experts in a specific domain is a further step to understand mechanisms of phenomena, e.g., human pathologies.

Abbreviations

FDR	False Discovery Rate
TDR	True Discovery Rate
ML	Machine Learning
SVM	Support Vector Machines
KEGG	Kyoto Encyclopedia of Genes and Genomes
GNN	Graph Neural Networks
GIN	Graph Isomorph Network
DiffPool	Differentiable Graph Pooling
DGCNN	Dynamic Graph Convolutional Neural Network
WL	Weisfeiler-Lehman

Acknowledgements
Not applicable.

Author Contributions
A.M., T.R., H.S., N.S. developed the concept and wrote the manuscript. A.M. implemented and tested the approach.

Funding
Not applicable.

Availability of data and materials
The proposed method is implemented in Python and publicly available for research purposes (https://github.com/arianemarandon/godconf). PyTorch ($>= 1.11.0$) is required. We run the experiments on Python 3.11.7. The datasets analysed during the current study are available in the following repositories: • Bacteria: https://github.com/AWebZen/FunctionalPrediction5000species • The collection of benchmark datasets for graph classification and regression: https://chrsmrrs.github.io/datasets/ • TumorMet: https://springernature.figshare.com/collections/TumorMet_A_repository_of_tumor_metabolic_networks_derived_from_context-specific_Genome-scale_Metabolic_Models/5931130/1

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
Not applicable.

Received: 25 June 2024 Accepted: 25 October 2024
Published online: 16 November 2024

References

1. Ross R, Dagnone D, Jones PJ, Smith H, Paddags A, Hudson R, Janssen I. Reduction in obesity and related comorbid conditions after diet-induced weight loss or exercise-induced weight loss in men. a randomized, controlled trial. Ann Intern Med. 2000;133(2):92–103.
2. Lee D-S, Park J, Kay KA, Christakis NA, Oltvai Z, Barabasi A-L. The implications of human metabolic network topology for disease comorbidity. Proc Natl Acad Sci. 2008;105(29):9880–5.
3. Jeong H, Tombor B, Albert R, Oltvai ZN, Barabási A-L. The large-scale organization of metabolic networks. Nature. 2000;407:651–4.
4. Kanehisa M, Goto S. KEGG: Kyoto encyclopedia of genes and genomes. Nucleic Acids Res. 2000;28:27–30.
5. Machicao L, Craighero F, Maspero D, Angaroni F, Damiani C, Graudenzi A, Antoniotti M, Bruno OM. On the use of topological features of metabolic networks for the classification of cancer samples. Curr Genomics. 2021;2(22):88–97.
6. Takemoto K, Nacher JC, Akutsu T. Correlation between structure and temperature in prokaryotic metabolic networks. BMC bioinf. 2007;8:1–1.
7. Weber Zendrera A, Sokolovska N, Soula HA. Robust structure measures of metabolic networks that predict prokaryotic optimal growth temperature. BMC bioinf. 2019;20:1–3.
8. Weber Zendrera A, Sokolovska N, Soula H. Functional prediction of environmental variables using metabolic networks. Sci Rep 2021;11(12192)
9. Shah HA, Liu J, Yang Z, Feng J. Review of machine learning methods for the prediction and reconstruction of metabolic pathways. Front Mol Biosci. 2021;8:634141.
10. Pearson K. On lines and planes of closest fit to systems of points in space. Lond, Edinb, Dublin Philos Mag J Sci. 1901;2(11):559–72.
11. Hotelling H. Analysis of a complex of statistical variables into principal components. J Educ Psychol. 1933;24:417–41.
12. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. Neural Netw. 2016;5(1):61–80.

13. Pfeifer B, Saranti A, Holzinger A. GNN-SubNet: disease subnetwork detection with explainable graph neural networks. Bioinformatics. 2022;38:120–6.
14. Long Y, Wu M, Liu Y, Fang Y, Kwon CK, Chen J, Leo J, Li X. Pre-training graph neural networks for link prediction in biomedical networks. Bioinformatics. 2022;38(8):2254–62.
15. Ding K, Wang S, Luo Y. Supervised biological network alignment with graph neural networks. Bioinf. 2023;39:465–74.
16. Mayr E. The Emergence of Evolutionary Novelties, In: Tax S (ed.), Evolution After Darwin, pp. 349–380. Chicago: University of Chicago Press, ??? (1960)
17. Moczek AP. When the end modifies its means: the origins of novelty and the evolution of innovation. Biol J Lin Soc. 2023;139(4):433–40.
18. Migdadi L, Telfah A, Hergenröder R, Wöhler C. Novelty detection for metabolic dynamics established on breast cancer tissue using 2D NMR TOCSY spectra. Comput Struct Biotechnol J. 2022;20:2965–77.
19. Granata I, Manipur I, Giordano M, Maddalena L, Guarracino MR. TumorMet: a repository of tumor metabolic networks derived from context-specific genome-scale metabolic models. Sci Data. 2022;9(1):607.
20. Vovk V, Gammerman A, Shafer G. Algorithmic Learning a Random World. Berlin, Heidelberg: Springer; 2005.
21. Shafer G, Vovk V. A tutorial on conformal prediction. J Mach Learn Res. 2008;9:371–421.
22. Mary D, Roquain E. Semi-supervised multiple testing. Electronic J Stat. 2022;16(2):4926–81.
23. Romano Y, Patterson E, Candès EJ. Conformalized quantile regression. In: NeurIPS (2019)
24. Romano Y, Sesia M, Candès EJ. Classification with valid and adaptive coverage. In: NeurIPS (2020)
25. Bates S, Candès E, Lei L, Romano Y, Sesia M. Testing for outliers with conformal p-values. Ann Stat. 2023;51(1):149–78.
26. Yang C-Y, Lei L, Ho N, Fithian W. Bonus: Multiple multivariate testing with a data-adaptivetest statistic. (2021) arXiv preprint arXiv:2106.15743
27. Marandon A, Lei L, Mary D, Roquain E. Machine learning meets false discovery rate. (2022) arXiv preprint arXiv:2208.06685
28. Liang Z, Sesia M, Sun W. Integrative conformal p-values for powerful out-of-distribution testing with labeled outliers. Journal of the Royal Statistical Society Series B: Statistical Methodology (2024)
29. Bashari M, Epstein A, Romano Y, Sesia M. Derandomized novelty detection with fdr control via conformal e-values. (2023) arXiv preprint arXiv:2302.07294
30. Marandon A, Lei L, Mary D, Roquain E. Adaptive novelty detection with false discovery rate guarantee. Ann Statist. 2042;52(1):157–83.
31. Lunde R, Levina E, Zhu J. Conformal Prediction for Network-Assisted Regression (2023)
32. Huang K, Jin Y, Candes E, Leskovec J. Uncertainty quantification over graph with conformalized graph neural networks. In: NeurIPS (2023)
33. Zargarbashi SH, Antonelli S, Bojchevski A. Conformal prediction sets for graph neural networks. In: International Conference on Machine Learning (2023)
34. Dey P, Zhang Z, Dunson DB. Outlier detection for multi-network data. Bioinf. 2022;38(16):4011–8.
35. Blanchard G, Lee G, Scott C. Semi-supervised novelty detection. J Mach Learn Res. 2010;11:2973–3009.
36. Haroush M, Frostig T, Heller R, Soudry D. A statistical framework for efficient out of distribution detection in deep neural networks. In: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022 (2022)
37. Wu Z, Pan S, Chen F, Long G, Zhang C, Philip SY. A comprehensive survey on graph neural networks. IEEE transact neural netw learn syst. 2020;32(1):4–24.
38. Kriege NM, Johansson FD, Morris C. A survey on graph kernels. Appl Netw Sci. 2020;5(1):1–42.
39. Zhang M, Cui Z, Neumann M, Chen Y. An end-to-end deep learning architecture for graph classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2018;32
40. Xu K, Hu W, Leskovec J, Jegelka S. How powerful are graph neural networks? In: International Conference on Learning Representations (ICLR) (2019)
41. Ying Z, You J, Morris C, Ren X, Hamilton W, Leskovec J. Hierarchical graph representation learning with differentiable pooling. Advances in neural information processing systems 2018;31
42. Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems 2016;29
43. Shervashidze N, Schweitzer P, Van Leeuwen EJ, Mehlhorn K, Borgwardt KM. Weisfeiler-lehman graph kernels. Journal of Machine Learning Research 2011;12(9)
44. Morris C, Ritzert M, Fey M, Hamilton WL, Lenssen JE, Rattan G, Grohe M. Weisfeiler and Leman go neural: Higher-order graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence (2019)
45. Kipf TN, Welling M. Variational graph auto-encoders. NIPS Workshop on Bayesian Deep Learning (2016)
46. Zhang M, Chen Y. Link prediction based on graph neural networks. In: Advances in Neural Information Processing Systems, 2018:5165–5175
47. Kersting K, Kriege NM, Morris C, Mutzel P, Neumann M. Benchmark Data Sets for Graph Kernels. (2016) http://graphkernels.cs.tu-dortmund.de
48. Ruff L, Vandermeulen R, Goernitz N, Deecke L, Siddiqui SA, Binder A, Müller E, Kloft M. Deep one-class classification. In: International Conference on Machine Learning, 2018:4393–4402
49. Federhen S. The NCBI taxonomy database. Nucleic Acids Res. 2012;40:136–43.
50. Abbahaddou Y, Ennadir S, Lutzeyer JF, Vazirgiannis M, Boström H. Bounding the expected robustness of graph neural networks subject to node feature attacks. In: International Conference on Learning Representations (2024)
51. Liu C, Zhan Y, Wu J, Li C, Du B, Hu W, Liu T, Tao D. Graph pooling for graph neural networks: Progress, challenges, and opportunities. In: International Joint Conference on Artificial Intelligence, Survey Track (2023)
52. Craighero F, Angaroni F, Stella F, Damiani C, Antoniotti M, Graudenzi A. Unity is strength: improving the detection of adversarial examples with ensemble approaches. Neurocomputi. 2023;554: 126576.

Publisher's Note