# Conformal novelty detection for multiple metabolic networks

Ariane Marandon, Tabea Rebafka, Nataliya Sokolovska, Hédi Soula

# Conformal novelty detection for multiple metabolic networks

Ariane Marandon, LPSM, Sorbonne Univeristé, Université Paris Cité
Tabea Rebafka, LPSM, Sorbonne Univeristé, Université Paris Cité
Nataliya Sokolovska, LCQB, Sorbonne Univeristé
Hédi Soula, NutriOmics, Sorbonne Univeristé

**Abstract**

**Motivation** 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 our 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. We show that 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.

**Availability and implementation** The proposed method is implemented in Python and publicly available for research purposes (`https://github.com/arianemarandon/godconf`).

**Keywords:** Novelty detection, conformal prediction, wrapper method, metabolic networks, graph neural networks

## 1 Introduction

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 [Ross et al., 2000, Lee et al., 2008].

A metabolic network represents a complete set of metabolic and physical processes describing physiological and biochemical properties of a living cell [Jeong et al., 2000]. Moreover, modern large metabolic databases such as KEGG [Kanehisa and Goto, 2000] make it possible to access genomic, enzymatic and metabolic information and to reconstruct interactions. Strong correlations between phenotypical traits of organisms and the topology of metabolic networks were reported [Takemoto et al., 2007, Zendrera et al., 2019], underlining the importance to study metabolic networks.

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 [Zendrera et al., 2021]. Modern statistical and machine learning methods can be applied to such data to get more insights in the functioning of living organisms [Shah et al., 2021].

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 [Pearson, 1901, Hotelling, 1933] or more recently with graph neural networks [Kipf and Welling, 2016a, Pfeifer et al., 2022, Long et al., 2022, Ding et al., 2023]. 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. 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 [Vovk et al., 2005, Shafer and Vovk, 2008] first introduced in classification and regression settings. Figure 1 illustrates the novelty detection task and the possible error types that a procedure can make.

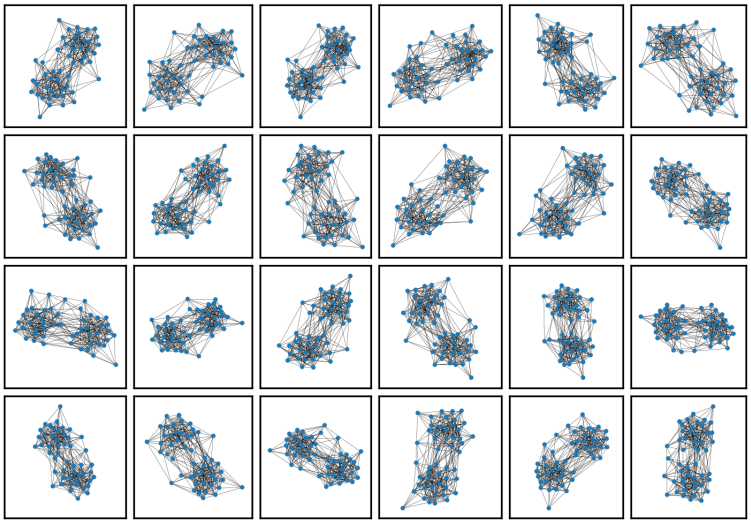
A major advantage of conformal prediction is that it does not make any assumptions on the type of distribution of the observations, which is important for applications where distribution assumptions on the data are hard to verify and/or rarely satisfied. Moreover, the reference or nominal distribution 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 Mary and Roquain [2022]. 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 [Romano et al., 2019, 2020]. 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 [Bates et al., 2023, Mary and Roquain, 2022, Yang et al., 2021, Marandon et al., 2022, Liang et al., 2024, Bashari et al., 2023]. 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 Marandon et al. [2022] 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.

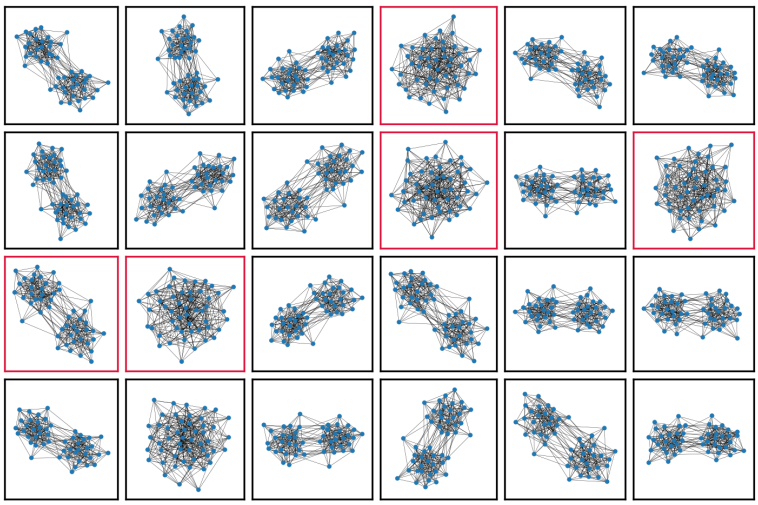
Novelty, or outlier, detection in graphs is particularly challenging due to the complex structure of the data. Some very recent attempts to extend conformal prediction to graphs concern either link prediction [Lunde et al., 2023] or the prediction of node labels [Huang et al., 2023, Zargarbashi et al., 2023] in a single graph. A recently developed procedure for the problem of novelty detection in a collection of networks is given in Dey et al. [2022], 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.*



(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.

- 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.

# 2 AdaDetect for Graphs

## 2.1 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 $P_{\text{ref}}$, 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 $P_{\text{ref}}$. 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

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

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

## 2.2 AdaDetect

The general idea of conformal novelty detection [Vovk et al., 2005, Haroush et al., 2022, Bates et al., 2023, Mary and Roquain, 2022] 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 Bates et al. [2023], the FDR control is shown to be guaranteed when proceeding as follows:

**Algorithm 1** AdaDetect for networks

**Input:** Set of reference networks $\mathcal{G}_{\text{ref}}$, set of test networks $\mathcal{G}_{\text{test}}$, $\alpha$ desired risk level.

**Output:** Set of declared novelties $\{j \in \mathcal{I}_{\text{test}}, S_j > \delta_{\text{best}}\}$.

1. Split $\mathcal{G}_{\text{ref}}$ into two parts, $\mathcal{G}_{\text{train}}$ and $\mathcal{G}_{\text{cal}}$.
2. Label the graphs in $\mathcal{G}_{\text{train}}$ as "0" and the graphs in $\mathcal{G}_{\text{cal}} \cup \mathcal{G}_{\text{test}}$ 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 $\mathcal{I}_{\text{cal}} \subset \mathcal{I}_{\text{ref}}$ be 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 $S_i = g(G_i)$.
5. For $j \in \mathcal{I}_{\text{test}}$, consider the novelty detection procedure $R_{S_j}$ with threshold $\delta = S_j$ and compute the proportion

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

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

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

the reference sample $\mathcal{G}_{\text{ref}}$ 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, Marandon et al. [2022] recently achieved an important improvement by training the score not only on the reference sample $\mathcal{G}_{\text{ref}}$ with a one-class classification algorithm, but training the score using a two-class classification method including the test sample $\mathcal{G}_{\text{test}}$. 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.

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

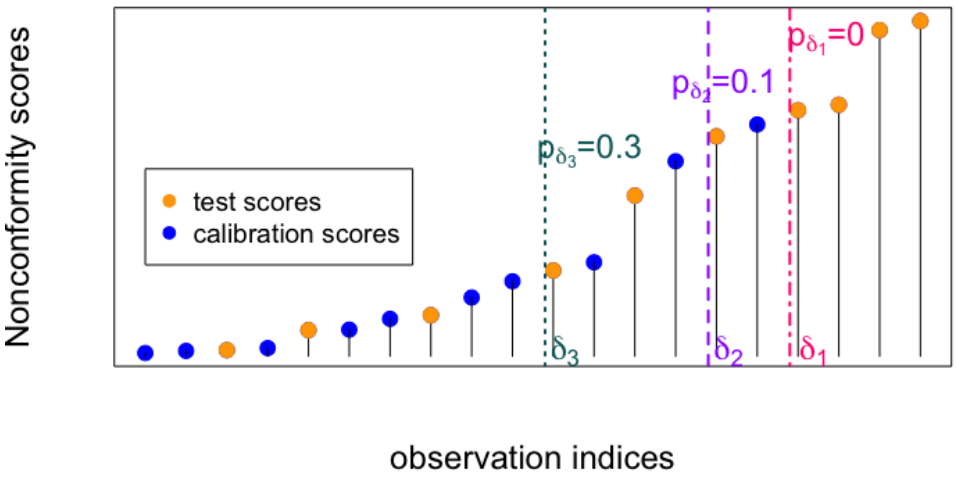



Fig. 2: Choice of threshold $\delta$ for the novelty detection procedure $R_\delta$. Illustration of proportion $p_\delta$ of falsely declared novelties for three different thresholds.

quantity $p_\delta$ defined as the proportion of reference observations declared as novelties by procedure $R_\delta$:

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

That is, $p_\delta$ is an approximation of the FDR of procedure $R_\delta$. Thus, for our procedure we choose the smallest threshold $\delta$ such that the corresponding proportion $p_\delta$ is lower than $\alpha$, 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 $P_{\text{ref}}$. 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 $\delta$ 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. Marandon et al. [2022] 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 [Mary and Roquain, 2022, Bates et al., 2023, Marandon et al., 2022]. 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 Mary and Roquain [2022], Marandon et al. [2022], we recommend to choose $|\mathcal{G}_{\text{cal}}|$ to be of the same order as the test sample size $|\mathcal{G}_{\text{ref}}|$.

## 2.3 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 [Wu et al., 2020]. 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 [Kriege et al., 2020] and graph neural networks (GNNs) [Wu et al., 2020, Zhang et al., 2018, Xu et al., 2019, Ying et al., 2018, Defferrard et al., 2016].

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 [Shervashidze et al., 2011], 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 [Shervashidze et al., 2011] 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 Morris et al. [2019]. Many other graph kernels have been proposed in the literature, see Kriege et al. [2020] 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 [Kipf and Welling, 2016b] or link prediction [Zhang and

Chen, 2018]. 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 [Xu et al., 2019]. 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 Xu et al. [2019].** In general, Graph Isomorphism Network (GIN) denotes a type of GNN layer, in which the new node representations are obtained by

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

  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, Xu et al. [2019] 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 Ying et al. [2018].** 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

$$\begin{aligned} A' &= S^T A S, \quad S = \mathrm{softmax}\Big(\mathrm{GNN}_{\mathrm{pool}}(A, X)\Big) \\ X' &= S^T \mathrm{GNN}_{\mathrm{embed}}(A, X) \end{aligned}$$

  where $\mathrm{GNN}_{\mathrm{pool}}$ and $\mathrm{GNN}_{\mathrm{embed}}$ 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 Zhang et al. [2018].** To extend convolution neural networks (CNN) to graph-structured input, Zhang et al. [2018] 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.

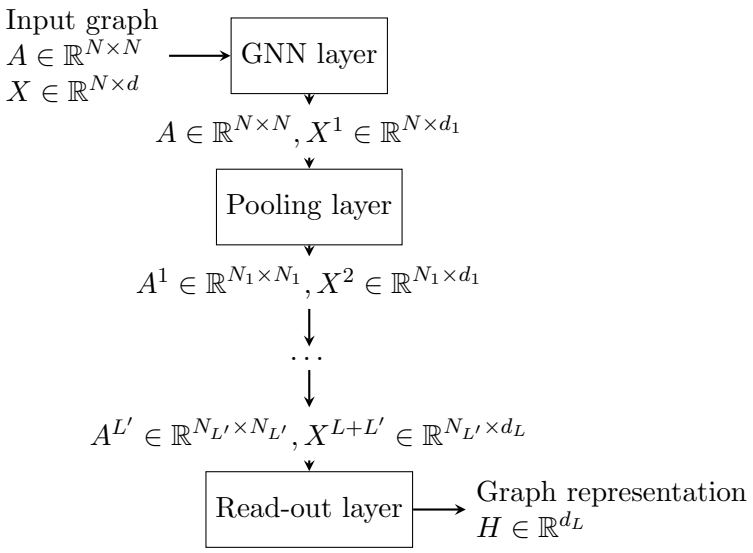



Fig. 3: A typical GNN architecture for graph classification: $N$ denotes the number of nodes of the input graph $(A, X)$, $L$ and $L'$ denote the number of GNN and pooling layers in the architecture, respectively, without the read-out layer. Each GNN layer applies a non-linear transformation on the current 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.

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 [Morris et al., 2019, Xu et al., 2019].

## 3 Numerical results

### 3.1 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 Kersting et al. [2016] is given in Table 1. The datasets DD and PROTEINS encode information

Tab. 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 |

Tab. 2: Experimental settings for numerical study.

| Dataset | $m$ | $n$ |
|---|---|---|
| DD | 100 | 300 |
| PROTEINS | 100 | 300 |
| AIDS | 500 | 1500 |
| NCI1 | 500 | 1500 |

about macromolecules. The nodes of 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 Bates et al. [2023], using one-class classification approaches: the one-class classifier given by the Support Vector Data Description (SVDD) method introduced in Ruff et al. [2018] 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 the different procedures, ranging from 0.00 to 0.95.

Tab. 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) |

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 Marandon et al. [2022]. 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 $\alpha$.

## 3.2 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 [Kanehisa and Goto, 2000]. Among them, there are 301 archaea and 5309 bacteria. The taxonomic information was obtained from the National Center for Biotechnology Information (NCBI) Taxonomy database [Federhen, 2012]. The reconstructed networks were provided by Zendrera et al. [2021]. 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

**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

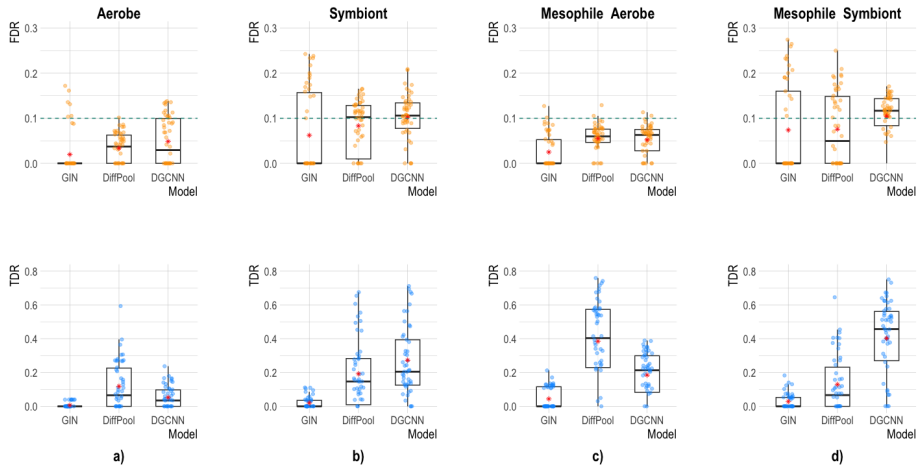



Fig. 4: FDR (upper row) and TDR (lower row) for AdaDetect procedures with $\alpha = 0.1$ (dashed green line) for the setups described in Table 4. Each boxplot is based on 50 data sets. Red points indicate mean values.

Tab. 4: 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 |

belongs to. Table 4 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 $\lvert \mathcal{I}_{\text{ref}} \rvert, \lvert \mathcal{I}_{\text{nov}} \rvert, n$ are given in Table 4 for each scenario. We set $\alpha = 0.1$ and use $\lvert \mathcal{G}_{\text{cal}} \rvert = \lvert \mathcal{G}_{\text{train}} \rvert = n/2$ 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.

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 Figure 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.

## 4 Conclusion

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. 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.

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.

## References

M. Bashari, A. Epstein, Y. Romano, and M. Sesia. Derandomized novelty detection with fdr control via conformal e-values. *arXiv preprint arXiv:2302.07294*, 2023.

S. Bates, E. Candès, L. Lei, Y. Romano, and M. Sesia. Testing for outliers with conformal p-values. *The Annals of Statistics*, 51(1):149–178, 2023.

M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

P. Dey, Z. Zhang, and D. Dunson. Outlier detection for multi-network data. *Bioinformatics*, 38(16):4011 – 4018, 2022.

K. Ding, S. Wang, and Y. Luo. Supervised biological network alignment with graph neural networks. *Bioinformatics*, 39:465 – 474, 2023.

S. Federhen. The NCBI taxonomy database. *Nucleic Acids Research*, 40:D136–D143, 2012.

M. Haroush, T. Frostig, R. Heller, and D. Soudry. 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.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

K. Huang, Y. Jin, E. Candes, and J. Leskovec. Uncertainty quantification over graph with conformalized graph neural networks. In *NeurIPS*, 2023.

H. Jeong, B. Tombor, R. Albert, Z. Oltvai, and A.-L. Barabási. The large-scale organization of metabolic networks. *Nature*, 407:651 – 654, 2000.

M. Kanehisa and S. Goto. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.*, 28:27–30, 2000.

K. Kersting, N. M. Kriege, C. Morris, P. Mutzel, and M. Neumann. Benchmark data sets for graph kernels, 2016. `http://graphkernels.cs.tu-dortmund.de`.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *Neural Networks*, 5(1):61 – 80, 2016a.

T. N. Kipf and M. Welling. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*, 2016b.

N. M. Kriege, F. D. Johansson, and C. Morris. A survey on graph kernels. *Applied Network Science*, 5(1):1–42, 2020.

D.-S. Lee, J. Park, K. Kay, N. Christakis, Z. Oltvai, and A.-L. Barabasi. The implications of human metabolic network topology for disease comorbidity. *Proceedings of the National Academy of Sciences*, 105(29):9880–9885, 2008.

Z. Liang, M. Sesia, and W. Sun. Integrative conformal p-values for powerful out-of-distribution testing with labeled outliers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 2024.

Y. Long, M. Wu, Y. Liu, Y. Fang, C. K. Kwon, J. Chen, J. Leo, and X. Li. Pre-training graph neural networks for link prediction in biomedical networks. *Bioinformatics*, 38(8):2254 – 2262, 2022.

R. Lunde, E. Levina, and J. Zhu. Conformal prediction for network-assisted regression, 2023.

A. Marandon, L. Lei, D. Mary, and E. Roquain. Machine learning meets false discovery rate. *arXiv preprint arXiv:2208.06685*, 2022.

D. Mary and E. Roquain. Semi-supervised multiple testing. *Electronic Journal of Statistics*, 16(2):4926–4981, 2022.

C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, 2019.

K. Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

B. Pfeifer, A. Saranti, and A. Holzinger. GNN-SubNet: disease subnetwork detection with explainable graph neural networks. *Bioinformatics*, 38:120 – 126, 2022.

Y. Romano, E. Patterson, and E. J. Candès. Conformalized quantile regression. In *NeurIPS*, 2019.

Y. Romano, M. Sesia, and E. J. Candès. Classification with valid and adaptive coverage. In *NeurIPS*, 2020.

R. Ross, D. Dagnone, P. J. Jones, H. Smith, A. Paddags, R. Hudson, and I. Janssen. Reduction in obesity and related comorbid conditions after diet-induced weight loss or exercise-induced weight loss in men. a randomized, controlled trial. *Annals of Internal Medicine*, 133(2):92–103, 2000.

L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402. PMLR, 2018.

G. Shafer and V. Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9:371 – 421, 2008.

H. Shah, J. Liu, Z. Yang, and J. Feng. Review of machine learning methods for the prediction and reconstruction of metabolic pathways. *Front. Mol. Biosci.*, 8, 2021.

N. Shervashidze, P. Schweitzer, E. J. Van Leeuwen, K. Mehlhorn, and K. M. Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

K. Takemoto, J. Nacher, and T. Akutsu. Correlation between structure and temperature in prokaryotic metabolic networks. *BMC Bioinform.*, 8(303), 2007.

V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic learning in a random world*. Springer Science & Business Media, 2005.

Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

C.-Y. Yang, L. Lei, N. Ho, and W. Fithian. Bonus: Multiple multivariate testing with a data-adaptivetest statistic. *arXiv preprint arXiv:2106.15743*, 2021.

Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.

S. H. Zargarbashi, S. Antonelli, and A. Bojchevski. Conformal prediction sets for graph neural networks. In *International Conference on Machine Learning*, 2023.

A. W. Zendrera, N. Sokolovska, and H. Soula. Robust structure measures of metabolic networks that predict prokaryotic optimal growth temperature. *BMC Bioinform.*, 20(499), 2019.

A. W. Zendrera, N. Sokolovska, and H. Soula. Functional prediction of environmental variables using metabolic networks. *Sci Rep*, 11(12192), 2021.

M. Zhang and Y. Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, pages 5165–5175, 2018.

M. Zhang, Z. Cui, M. Neumann, and Y. Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.