Applied Soft Computing 118 (2022) 108489

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Community-based anomaly detection using spectral graph filtering

Rodrigo Francisquini [a], Ana Carolina Lorena [b], Mariá C.V. Nascimento [a,b,*]

[a] Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo (UNIFESP), Av. Cesare M. G. Lattes, 1201, Eugênio de Mello, São José dos Campos-SP, CEP: 12247-014, Brazil

[b] Divisão de Ciência da Computação (IEC), Instituto Tecnológico de Aeronáutica (ITA), Praça Marechal Eduardo Gomes, 50, Vila das Acácias, São José dos Campos-SP, CEP 12228-900, Brazil

ARTICLE INFO

Article history:
Received 22 September 2021
Received in revised form 14 January 2022
Accepted 15 January 2022
Available online 29 January 2022

Keywords:
Spectral filter
Anomaly detection
Community detection
Graph Fourier transform
COVID-19

ABSTRACT

Several applications have a community structure where the nodes of the same community share similar attributes. Anomaly or outlier detection in networks is a relevant and widely studied research topic with applications in various domains. Despite a significant amount of anomaly detection frameworks, there is a dearth on the literature of methods that consider both attributed graphs and the community structure of the networks. This paper proposes a community-based anomaly detection algorithm using a spectral graph-based filter that includes the network community structure into the Laplacian matrix adopted as the basis for the Fourier transform. In addition, the choice of the cutoff frequency of the filter considers the number of communities found. In computational experiments, the proposed strategy, called *SpecF*, showed an outstanding performance in successfully identifying even discrete anomalies. *SpecF* is better than a baseline disregarding the community structure, especially for networks with a higher community overlapping. Additionally, we present a case study to validate the proposed method to study the dissemination of COVID-19 in the different districts of São José dos Campos, Brazil.

1. Introduction

The explosive growth of technology has led to a substantial increase in the amount of data collected from several applications. They include sensor measurements, transportation, internet, biological data, financial transactions, among others. Therefore, data analysis and processing techniques to manage massive amounts of data are of paramount importance. Data structure is usually irregular and relational data is commonly represented through graphs or networks [1,2].

A great number of applications consists of networks with community structure. Some examples are in internet of things (IoT) [3], protein–protein interactions [4] and COVID-19 related data [4]. Anomaly detection in this type of network is a specially relevant task. Anomaly detection in networks with community structure consists of identifying nodes, called anomalous nodes, that significantly differ from the standard observed in their community they belong to [5]. These are also known as context anomaly.

Despite being the target of intense investigation, there are networks for which graph anomaly detection tools are limited. There is a dearth of literature on data anomaly approaches that deal with data with relational properties and temporal information. Moreover, the topological graph structure is not usually taken into consideration on time series graph anomaly detection algorithms. According to Chen et al. [3], applications such as sensor networks, for which there is a strong geographical and temporal dependency, can benefit from frameworks that consider the topological graph structure.

Anomaly detection in networks has attracted a lot of attention in the last five years with a soar on the number of studies [1]. The reason behind this phenomena is not only the increase on the amount of applications but also the advent of sophisticate deep learning tools to perform such a task [1,3]. Most of the existing deep learning-based methods to time series anomaly detection uses semi-supervised learning, requiring some labeled data. In addition, according to Choi et al. [6], the existing methods are too case-specific, demanding domain knowledge.

The goal of this paper is to give new insights into anomaly detection in networks by designing a more generic unsupervised anomaly detection tool to approach attributed networks with community structure. These networks model a wide range of applications as the attributes of the networks are not limited to time series. As a result, for example, networks representing biological processes can be considered by the proposed method [4].

* Corresponding author at: Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo (UNIFESP), Av. Cesare M. G. Lattes, 1201, Eugênio de Mello, São José dos Campos-SP, CEP: 12247-014, Brazil.

E-mail addresses: rodrigo@francisquini.com (R. Francisquini), aclorena@ita.br (A.C. Lorena), mcv.nascimento@unifesp.br, mariah@ita.br (M.C.V. Nascimento).

https://doi.org/10.1016/j.asoc.2022.108489
1568-4946/

Moreover, this paper introduces a framework that considers the topological graph structure to describe the anomalies. To this end, the proposed algorithm is founded on a recent signal processing concept which has also been drawing the attention of the data analysis research community, the *graph signal processing* [7].

Graph signal processing (GSP) techniques extend concepts from classical signal processing to signals indexed by generic graphs [8,9] and seek to analyze the data considering its underlying relational structure. Data from various domains, such as sensor networks, molecular network interactions, financial transactions, can be modeled as graph-indexed signals. For example, graphs can represent data collected from sensor networks, where sensors correspond to vertices, and edges connect sensors close to each other. The signals on the graph nodes correspond to the set of values measured by the sensors at a given time. Thus, GSP tools are used for many purposes, such as fault diagnosis, signal denoising, signal compressing, and anomaly detection [9–11].

Two approaches are commonly employed for processing signals indexed by graphs. The first considers the Laplacian matrix and is based on the spectral graph theory [12]. The second relies on the adjacency matrix and is based on algebraic signal processing theory [13,14]. Both approaches generalize classical signal processing operations, such as filtering and Fourier transform, to the graph domain, by defining the concept of graph filters and graph Fourier transform, respectively [15].

This paper proposes a method to detect anomalies in signals indexed by graphs using GSP theory and spectral graph theory. In comparison to the related literature, the introduced method, named *SpecF*, not only considers the adjacency relationships between vertices but also takes into account the community structure in the graph Fourier transform. This is achieved by incorporating the network community structure into an expanded adjacency matrix. The expanded adjacency matrix can be understood as a modification of the original unweighted graph to a weighted network through the inclusion of new edges. The method marks vertices whose signal values are outside the expected behavior for the community they belong as potential anomalies.

Computational experiments comparing the accuracy of the novel method with a counterpart using the classical adjacency matrix evidence the superiority of our community detection-based strategy in recovering anomalies, even for the most discrete cases. In addition to experiments with labeled artificial networks proposed in this paper, a comparative analysis with state-of-the-art time series anomaly detection algorithms on two IoT databases publicly available is performed. This paper also shows an experiment with the proposed strategy on a COVID-19 dataset. The results of this experiment indicates an anomalous growth in the number of COVID-19 cases in the different districts of an upstate city of São Paulo, Brazil. The main contributions of this paper are presented next.

- We propose an anomaly detection algorithm that addresses attributed networks for which the literature is scarce of anomaly detection algorithms;
- We introduce an anomaly detection algorithm, *SpecF*, based on graph signal processing theory which defines anomalous objects as those nodes whose signals required the most significant correction by a low-pass filter;
- We propose an anomaly detection algorithm, *SpecF*, which differs from other community-based anomaly detection methods by explicitly using the community structure in an extended adjacency matrix;
- We introduce a methodology to add normal and anomalous signals to networks with community structure, to better assess the proposed framework;
- We apply *SpecF* to COVID-19 data to analyze the dissemination of the COVID-19 virus by investigating the anomalous districts of an upstate city from Brazil with approximately 730 thousand inhabitants.

The remainder of this paper is organized as follows. Section 2 presents fundamental concepts of the GSP theory relevant for the understanding of the proposed tool. Section 3 shows a brief literature review on related anomaly detection algorithms. Section 4 introduces the proposed anomaly detection algorithm, *SpecF*. Section 5 presents the computational experiments, including a thorough analysis of the anomaly detection algorithm in the COVID-19 dataset compiled in this study. Finally, Section 6 presents final remarks and future research directions.

2. Graph signal processing

Data from several applications can be represented by graphs. Let $G = (V, E, A)$ be a weighted graph, where V and E are its respective sets of n nodes and m edges, and $A \in \mathbb{R}^{n\times n}$ is the weighted adjacency matrix. Each node $v_i \in V$ describes an instance of the dataset, and the weight a_{ij} of an undirected[1] edge carries the strength of the relation between a pair of vertices v_i and v_j. The degree of a vertex v_i is quantified by the sum of the weights of the edges incident to it. Let $D \in \mathbb{R}^{n\times n}$ be a diagonal matrix, called degree matrix, where its ith diagonal element d_{ii} receives the degree of node v_i. Moreover, we denote here the set of neighbors of a vertex v_i by $\mathcal{N}_i$, which means that this set contains all vertices adjacent to v_i.

The graph signal, defined by function $f : V \to \mathbb{R}$, is represented by a vector $\mathbf{f} \in \mathbb{R}^n$, where each element f_i corresponds to the signal of node $v_i \in V$, i.e, $f(v_i)$.

2.1. Graph fourier transform

The graph Laplacian, also known as the non-normalized graph Laplacian, is the matrix $L = D - A$, corresponding to a real-valued symmetric matrix [16]. Let $U = [u_{ij}]_{n\times n}$ be the set of eigenvectors of L. Without loss of generality, let G be a connected component. Therefore U is orthonormal, since L is a real-valued symmetric matrix. Moreover, the associated non-negative eigenvalues are referred here to as $\{\lambda_l\}_{l=0,1,\ldots,n-1}$, where $0 = \lambda_0 < \lambda_1 \leq \lambda_2 \ldots \leq \lambda_{n-1} \coloneqq \lambda_{max}$. As U is orthonormal, $U^T U = I$ holds.

The graph Fourier transform $\hat{\mathbf{f}}$ of a signal $\mathbf{f} \in \mathbb{R}^n$ given a symmetric shift operator $S = R\Lambda R^*$ is $\hat{\mathbf{f}} = R^*\mathbf{f}$ [9]. As $L = U\Lambda U^T$, $U^* = U^T$, the graph Fourier transform given L is calculated by $\hat{\mathbf{f}} = U^T\mathbf{f}$. In addition, the lth row of U^T corresponds to the eigenvector associated with λ_l and each component of $\hat{\mathbf{f}}$ can be written as:

$$\hat{f}(\lambda_l) = \sum_{i=1}^{n} u_{il} f(v_i), \quad \forall l \in \{0, \ldots, n-1\} \tag{1}$$

Let $\hat{\mathbf{if}} = U\hat{\mathbf{f}}$ be the inverse graph Fourier transform of $\hat{\mathbf{f}}$ considering the same shift operator L. It is possible to return to the signal $\mathbf{f}$ by the inverse graph Fourier transform of $\mathbf{f}$, since

$$\hat{\mathbf{if}} = U\hat{\mathbf{f}} = UU^T\mathbf{f} = I\mathbf{f} = \mathbf{f}$$

Therefore, we have that

$$f(v_i) = \sum_{l=0}^{n-1} u_{il}\hat{f}(\lambda_l), \quad \forall i \in \{1, \ldots, n\}. \tag{2}$$

[1] This paper assumes that G is an undirected graph.

2.2. Frequencies on graphs

In classical Fourier analysis, eigenvalues $\{(2\pi\xi)^2\}$ carry the notion of high and low frequencies. Low frequencies are associated with smooth complex exponential eigenfunctions that oscillate slowly, whereas high frequencies are related to complex exponential eigenfunctions that oscillate more rapidly.

The definition of high and low frequencies for signals indexed by graphs takes into account graph Fourier theory (GFT). According to the GFT, the eigenvectors of graph Laplacians associated to the first eigenvalues vary more sharply. Consequently, the eigenvectors of end vertices of heavier edges are more likely to be similar.

2.3. Low-pass graph filter

The process of frequency filtering transforms and input signal into a linear combination of complex exponentials. As a consequence, filtering amplifies or attenuates the contributions of some frequencies. Graph spectral filtering can be directly generalized as

$$f_{out}(\lambda_l) = f_{in}(\lambda_l)h(\lambda_l) \tag{3}$$

where $h(\bullet)$ is the filter transfer function and f_{in} is the Fourier transform of an input signal function. Several well-known continuous filtering techniques can be implemented as discrete, considering filtering functions that satisfy Eq. (3), such as Gaussian smoothing, bilateral filtering and non-local means filtering [17].

A filter is said to be low-pass if it does not significantly affect the frequency content of low-frequency signals but attenuate the magnitude of high-frequency signals. An ideal low-pass filter keeps the magnitude of the spectrum at low frequencies unchanged and attenuates it at high frequencies. The frequency response of these filters is defined as

$$h(\lambda_l) = \alpha_l = \begin{cases} 1, & \lambda_l < \lambda_{cut} \\ 0, & \lambda_l \geq \lambda_{cut} \end{cases} \tag{4}$$

Sandryhaila and Moura [15] demonstrated that the design of these filters is a linear problem and the construction of a filter with frequency response $h(\lambda_l) = \alpha_l$ corresponds to solving a system of n_l non-linear equations:

$$h_0 + h_1\lambda_0 + \cdots + h_{d_p}\lambda_0^{d_p} = \alpha_0, \tag{5}$$

$$h_0 + h_1\lambda_1 + \cdots + h_{d_p}\lambda_1^{d_p} = \alpha_1, \tag{6}$$

$$\vdots \tag{7}$$

$$h_0 + h_1\lambda_{n_l-1} + \cdots + h_{d_p}\lambda_{n_l-1}^{d_p} = \alpha_{n_l-1} \tag{8}$$

where d_p is the degree of the polynomial. We can find an approximate solution by, for example, the least squares method, to get around the over-determination of the system when $n_l \geq d_p + 1$. The complexity of the least squared method for determining a triangular n_l-dimensional matrix inverse is $O(n_l^2)$.

3. Related works

An important task in data mining is finding instances with unexpected behavior, which are more likely to be anomalous observations. Although several techniques have been developed over the last years to identify anomalies in data [18], there are few techniques capable of efficiently dealing with graph-structured data. Graph structured data have complex correlations and require techniques capable of analyzing not only the data itself but the relations between the elements. This paper is particularly interested in graphs with node attributes to find community outliers. A community outlier is a node whose attribute values deviate significantly from its community members.

In a recent survey on graph-based anomaly detection, Akoglu et al. [5] pointed out only two community-based methods for detecting anomalies in graphs with attributes. The first, introduced in [19], is a probabilistic model that considers both relational and raw data information to find more meaningful outliers. On the one hand, the information regarding the relation is obtained from the topological structure of the network and describes the relationships that exist between instances of the dataset. On the other hand, the network node attributes store the data. According to the authors, the algorithm, called community outlier detection algorithm (CODA), can identify meaningful community outliers. The second method is a node outlier ranking technique in attributed graphs, called GOutRank, developed by Müller et al. [20]. GOutRank ranks the graph nodes according to their degree of deviation in both graph-relational data and node attribute properties.

In a particular case of attributed graphs, where the attributes of the nodes can be interpreted as a signal indexed by the graph, techniques that extend signal processing concepts to the graph domain can be used to identify anomalies. For example, attributed graphs can represent wireless sensor networks, where each (sensor) node stores the value measured by the corresponding sensor in an instant of time. In this case, to identify an anomaly, it is necessary to consider the geographical proximity between the sensors and the values sensed by them. A sensor measurement significantly different from neighboring sensors' represents a potential anomaly. For this type of application, classical signal processing techniques, such as filters, can be extended to the graph domain.

Several studies use graph-based filters for data compression, data recovery, classification, noise removal, signal recovering, among others. However, to the best of our knowledge, few studies have adopted graph-based filtering to detect anomalies. Sandryhaila and Moura [15], for example, introduced the concept of total variation to frequency sorting. The authors developed a filter using the proposed ordering method to identify malfunctions in sensor networks by extracting high-frequency components. Egilmez and Ortega [21] introduced a spectral anomaly detection method that also uses a graph-based filter. They considered collective anomalies which occurred locally in time and space and applied the proposed method to sensor networks.

Both [15] and [21] adopted spectral filters and presented experiments with sensor network data where the adjacency relationships may be sufficient to detect nodes whose attributes deviate from the attributes of other nodes from the same community. However, in applications where the adjacency relationships are not only based on the physical distance, identifying a community outlier may consider the network's community structure. Neither of the previous work takes such information into account in their analysis.

In protein–protein interaction networks, for example, adjacency relationships define the biological strength of the interaction between a pair of proteins. Proteins, represented by the graph nodes, may have quantitative attributes that correspond to the expression level of the protein in the network. In these networks, a community can be interpreted as a group of functionally related proteins [4] whose attributes have similar characteristics. For this type of application, analyzing only the adjacency affinities may not reveal relevant information. Then, the data analysis must explicitly consider the community structure. To the extent of our knowledge, there is no graph-based filter method to detect anomalies that also takes the community structure into account, as proposed here.

4. Anomaly detection

The anomaly studied in this paper is defined as a vertex v_i whose signal $f(v_i)$ is far from the expected standard found in the community to where v_i belongs. In this case, most of the signal's energy is concentrated in the low-frequency signals of the Fourier transform. A node with a signal value that considerably differs from the neighborhood average will probably be anomalous.

This type of anomaly is observed, for example, in temperature sensor networks, in which neighboring sensors are expected to have close measured values. In this case, a potential anomaly is a sensor measurement that deviates significantly from the values sensed by neighboring nodes, such as a sensor failure. In protein–protein interaction networks, expression levels of neighboring nodes that represent proteins belonging to the same biological process usually vary proportionately. For example, if the expression level of a protein increases, other proteins that belong to the same biological process usually have their expression level increased or decreased accordingly. In this case, where proteins from the same biological process generally belong to the same community, a potential anomaly is a node with a variation of expression level significantly different from the nodes in the same community, e.g., a protein involved in cancerous processes.

The reasoning behind the proposed strategy is that the attenuation of the magnitude of the signal spectrum at high frequencies can correct the abrupt variations that define the studied anomalous behavior. Thus, the anomaly detection method introduced in this paper, *SpecF*, relies on the idea that vertices whose signal value needed a substantial correction after the filtering process are more likely to be anomalous. For this, *SpecF* uses a low-pass filter to attenuate the magnitude of the high frequencies of the spectrum $\hat{\mathbf{f}}$ of a signal $\mathbf{f}$. The inverse Fourier transform, defined in Eq. (2), is applied to the filtered spectrum to obtain a filtered signal $\mathbf{f}'$. The filtered signal $\mathbf{f}'$ is compared to the original signal $\mathbf{f}$ to determine to which vertices the signal value has been severely attenuated, to obtain the set of potentially anomalous vertices. Algorithm 1 presents a basic pseudocode of *SpecF*, whose main steps will be thoroughly described in the next sections. Besides the graph and its signals, the input data required by this algorithm are a matrix representing G, which can be the adjacency matrix, and a partition $\mathcal{C} = \{C_1, C_2, \ldots, C_{|\mathcal{C}|}\}$ representing the set of communities of G.

Algorithm 1: *SpecF*

Data: A graph G, a matrix M_G representing G, signal B, partition $\mathcal{C}$
Result: A list with the anomalous nodes *PAN*
$L := D - M_G$
Calculate matrix U, the set of eigenvectors of L
Calculate the Fourier transform using Eq. (1) to obtain the spectrum $\hat{B}$ of B: $\hat{B} = U^T B$
$B' \leftarrow$ Low-pass filter $(G, L, U, \hat{B})$ – Algorithm 2 discussed in Section 4.1
$PAN \leftarrow$ Potentially anomalous nodes$(G, B, B', \mathcal{C})$ – Algorithm 3 discussed in Section 4.2

To evaluate the proposed strategy, this paper also introduces an approach to generate synthetic anomalous signals similar to the signals considered in the hypothesis. The next section discusses the details of the proposed anomaly detection algorithm.

4.1. Low-pass filter and the cut-off frequency

In GSP theory, the first k eigenvalues of L correspond to the k-lowest frequencies in the spectrum of a signal. Low frequencies carry the information of signals that vary slightly across the nodes of the network. Moreover, the proposed strategy focuses on a signal whose intra-community variation is expected to be low. Therefore, the introduced method considers the number of communities k in the network as the cut-off frequency, so that $\lambda_{cut} = \lambda_k$. A low-pass filter, as defined in Section 2.3, attenuates the magnitude of high-frequency signals and keeps low frequencies unchanged. In this case, a high-frequency is defined as a frequency λ_l higher than the cut-off frequency λ_{cut}, that is, $\lambda_l > \lambda_{cut}$.

In cases where the expected partition is known beforehand, the choice of λ_k is trivial, since k is the number of communities. On the other hand, when the number of communities is unknown, two approaches to estimate the number of communities in the network can be used. The first approach consists of applying a community detection algorithm to the network to estimate the number of communities k – algorithms that do not require the number of communities to be informed *a priori*. The second approach comes from graph theory and consists of finding the value of k by analyzing the eigenvalues of L and choosing k so that $\lambda_1, \ldots, \lambda_k$ are very small, but λ_{k+1} is relatively large. In spectral graph theory, when a graph has k completely disconnected components, the first k eigenvalues of L will have the value 0 and then there is a gap to the eigenvalue in position $k+1$ [16]. Algorithm 2 shows a pseudocode to determine the low-pass filter for the proposed anomaly detection algorithm. The complexity of Algorithm 2 is dominated by the least square method to the system (5), which has been discussed earlier, $O(n^2)$.

Algorithm 2: Low-pass filter

Data: A graph G, the Laplacian matrix L, matrix U, spectrum $\hat{B}$
Result: Filtered signal B'
Estimate the number of communities k of the input graph G by analyzing the eigenvalues of L (second approach)
Find the h_l values by solving the system (5), where n_l and d_p are n
Define a diagonal matrix $\mathcal{F}$ where its i-th diagonal element is the approximate α_i – according to system (5)
The $\hat{B}$ spectrum is submitted to the proposed low-pass filter to obtain a filtered spectrum $\hat{B}'$: $\hat{B}' = \mathcal{F}\hat{B}$
Then, the inverse Fourier transform is applied in $\hat{B}'$ to obtain a filtered signal B': $B' = U\hat{B}'$

Fig. 1 plots the filtered signal B' of the signal presented in Fig. 4. By comparing the signal B' to the signal B, it is possible to observe that the groups of vertices are more cohesive in signal B', demonstrating that the filter brought the signal values even closer to nodes from the same community. In addition, the anomalies, which previously diverged from other nodes in the same community, are now within the expected standard.

4.2. Potentially anomalous nodes

Fig. 1 shows the ability of the *SpecF* to correct anomalies and normalize the values of a signal according to the network's community structure. The filtered signal B' is contrasted to the original anomalous signal B to identify in which nodes the correction of the signal was more significant. The intuition behind this idea is that, if anomalous nodes differ from what is expected for their community, the normalization applied to the signal considering the filter will be more intense in these nodes. Thus, when identifying the anomalous nodes, a set of nodes with the greatest anomalous potential is also detected. For such, let $Y = |B - B'|$ be a signal, and its ith element be referred to as y_i, corresponding to the difference signal at vertex v_i. In general, vertices v_i with a

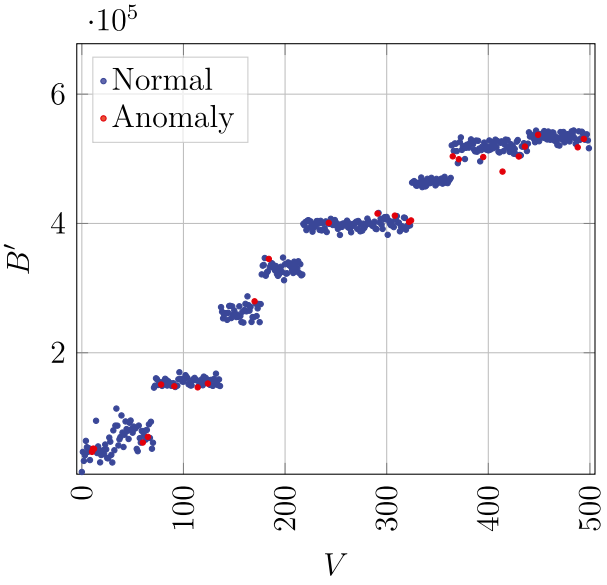

Fig. 1. Example of a filtered signal B'.

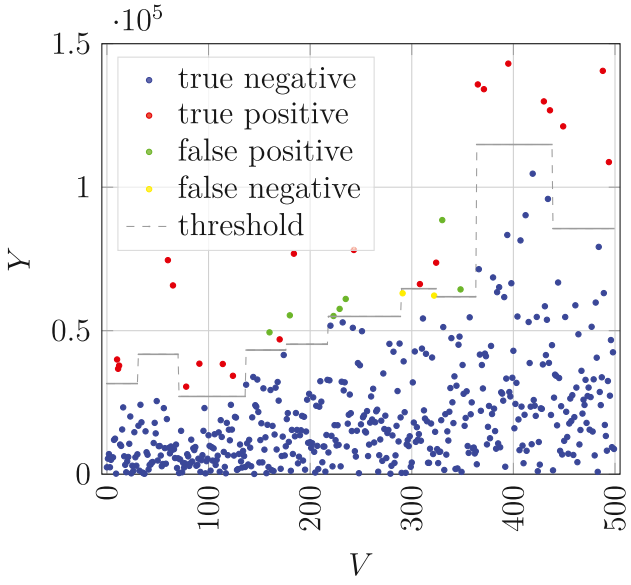

Fig. 2. Example of a signal Y and the threshold.

high y_i value are more likely to be anomalous and the vector Y is deemed an abnormality quantifier.

To classify the nodes as anomalous or normal in a binary way, *SpecF* applies to Y a threshold to distinguish which y_i values are considered normal and which are identified as abnormal observations. The threshold values employed in *SpecF* are regarded for each community, taking the mean and standard deviation of Y into account, as presented in Eq. (9).

$$TD(Y, C_k) = \frac{mean(Y, C_k) + 2std(Y, C_k)}{max(Y, C_k)} \tag{9}$$

$mean(Y, C_k)$, $std(Y, C_k)$ and $max(Y, C_k)$ are, respectively, the mean, standard deviation and maximum of the values of y_i's that represent the nodes of community C_k. Algorithm 3 presents a pseudocode of the strategy that defines the potentially anomalous nodes. In this algorithm, let c_{v_i} be the community C_k where vertex v_i belongs to. The complexity of Algorithm 3 is $O(n)$.

Algorithm 3: Potentially anomalous nodes

Data: A graph G, signal B, filtered signal B', partition $\mathcal{C}$
Result: A list *PAN* with the potentially anomalous nodes
$Y = |B - B'|$
$PAN = \emptyset$
Insert in *PAN* every node $v_i \in V(G)$ whose $TD(Y, c_{v_i})$ is lower than y_i

Fig. 2 illustrates the relation between y_i values and the vertices of a network. Moreover, it presents the threshold values $TD(Y, C_k)$ – dotted curve – that separate anomalous nodes from normal nodes. It is possible to notice that the vast majority of nodes above the red curve are true positives and, therefore, are in the set of anomalous nodes defined by the generator. On the other hand, most of the nodes below the red curve are true negatives and, thus, are labeled normal. There are also false positives and false negatives and they are usually close to the threshold curve.

The computational complexity of *SpecF*, described in Algorithm 1, is $O(n^2)$ since to calculate the Fourier transform a matrix multiplication operation is required.

4.3. Expanded adjacency matrix

To embed to a matrix information about its community structure, this paper proposes the use of an expanded adjacency matrix to represent a graph G – one of the forms to define matrix M_G.

Let $W \in \mathbb{R}^{n \times n}$ be the so-called expanded matrix that incorporates the community structure of a graph to represent the pairwise relationship between vertices v_i and $v_j \in V$, referred to as w_{ij}. The values w_{ij} are defined in Eq. (10), where c_{v_i} is the community where vertex v_i belongs to.

$$w_{ij} = \begin{cases} 5, & \text{if } v_i \text{ and } v_j \text{ are neighbors and } c_{v_i} = c_{v_j} \\ 3, & \text{if } v_i \text{ and } v_j \text{ are neighbors and } c_{v_i} \neq c_{v_j} \\ 1, & \text{if } v_i \text{ and } v_j \text{ are not neighbors and } c_{v_i} = c_{v_j} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

According to the definition of the expanded matrix, if vertices v_i and v_j are adjacent and in the same community, w_{ij} will receive the highest weight, the value 5. If they are adjacent but belong to distinct communities, w_{ij} will receive the intermediary value 3. If the two vertices are not adjacent but are in the same community, w_{ij} will receive the value of 1. As a consequence, an explicit affinity between these vertices is defined in such a matrix. Other edges assume null weight and are disregarded.

4.4. Attributed networks generator

This section introduces the methodology to generate anomalous and normal signals in networks with community structure.

4.4.1. Normal signal generator

Let G^c be a graph whose nodes v_i^c represent communities C_i of G and the edge weights w_{ij}^c are defined as the number of edges between communities C_i and C_j. To define a signal S^c, the nodes of G^c are sorted according to the sum of the edges' weights w_{ij}^c and s_i^c is then defined as

$$s_i^c = \sum_{\forall v_j^c \in \mathcal{N}_i^c} w_{ij}^c \times (i+1) \tag{11}$$

where $\mathcal{N}_i^c$ is the set of nodes adjacent to v_i^c.

To properly evaluate the anomaly detection strategy proposed in this paper, we developed an artificial signal generator. The generator produces a synthetic signal S similar to the signal observed in the investigated applications. As a result, it creates signals whose values for nodes of the same community are similar.

Algorithm 4 describes the process employed by the generator to obtain the signal S from a graph G and a given signal S^c, with elements defined by Eq. (11). First, all nodes are marked and an auxiliary n-dimensional vector S^x is initialized as empty. For every community C_k of G, the algorithm assigns the value of s_k^c to the position of S^x that corresponds to the highest degree node in the community C_k. In the case of a tie, a node is randomly selected among the highest degree nodes. These nodes are regarded as community heads. Then, the algorithm starts a propagation process from the community heads. Each node propagates a percentage of its value to its neighbors. This percentage is lower (10%) if the nodes involved belong to different communities, and higher (at least 25%) if they are in the same community. For nodes from the same community, this percentage also depends on their degree, so that nodes with higher degrees have a greater influence on lower degree nodes. After the propagation process, every node that propagated values have their value reduced by 5%. This process is repeated until all nodes have propagated a number of their signal values. The last step consists of a normalization process to define the signal of each node as that considers the weight of the edge between neighbor nodes to carry out a weighted average.

Algorithm 4: Normal Signal Generator

Data: A graph G, a signal S^c, a partition $\mathcal{C}$
Result: A signal S
Mark all nodes;
$S^x \leftarrow \emptyset$;
forall *community* $C_k \in \mathcal{C}$ **do**
  $v_i \leftarrow$ the highest degree node of C_k;
  $s_i^x \leftarrow s_k^c$;
  Unmark v_i;
Create a list F with all unmarked nodes sorted by index values;
while *there is a node in* F **do**
  Select the first node v_i from F;
  forall $v_j \in \mathcal{N}_i$ **do**
    $t \leftarrow 0.1$;
    if v_i *and* v_j *are in the same community* **then**
      $mt \leftarrow \frac{degree(v_i)}{degree(v_j) + degree(v_i)}$;
      $t \leftarrow max(0.25, mt)$;
    $s_j^x \leftarrow s_j^x + (s_i^x \times t)$;
    $s_i^x \leftarrow s_i^x \times 0.95$;
    if v_j *is marked* **then**
      Unmark v_j and include v_j at the end of F;
  Remove v_i from F;
forall $v_i \in G$ **do**
  $s_i \leftarrow \frac{\sum_{j=1}^{n} w_{ij} \times s_j^x}{\sum_{j=1}^{n} w_{ij}}$;

Fig. 3 presents box-plots of the values of the S signal, obtained through Algorithm 4 applied to a 500-node LFR network [22] with the parameters defined in Table 1 and low overlapping between communities (mixture degree 0.1). The network has 10 planted communities. Section 5.1 describes the parameters and the software to generate the networks. We show a box-plot for each of the expected communities, sorted by increasing order of intra-community average signal. Fig. 3 shows that, although the average value of signal S in each community is different, vertices in the same community have similar values.

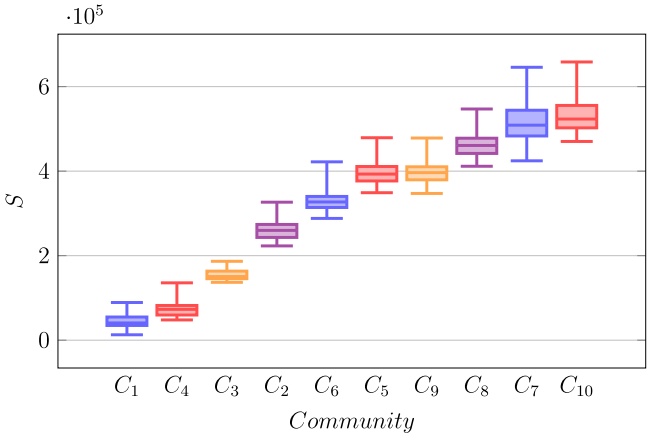

Fig. 3. Box-plots of S for each community of G.

4.4.2. Anomalous signal generator

In addition to the normal signal generator, a strategy to create anomalous signals in an attributed network is introduced in this paper.

For such, consider a signal S of a graph G. We generate an anomalous signal B from signal S by increasing the value of s_i in some vertices of G. This process is detailed in Algorithm 5, which randomly selects a set of nodes to corrupt. The anomaly intensity θ defines how much greater the value of an anomalous node will be when compared to the rest of the community. An anomaly intensity value of 0.1, for example, means that an anomalous node will have a signal value between 5% and 10% higher than the largest signal value of that community.

Algorithm 5: Anomalous Signal Generator

Data: A graph G, a normal signal S, percentage of anomalies AN, anomaly intensity θ
Result: An anomalous signal B
$B \leftarrow$ Create a copy of S;
$P \leftarrow$ Randomly select a set with AN% of distinct vertices of G;
forall *vertex* v_i *in* P **do**
  $max \leftarrow$ Highest value of S among nodes in c_{v_i};
  $tax \leftarrow$ Random value between $\frac{1}{2}\theta$ and θ;
  $b_i = max \times (1 + tax)$;

Fig. 4 illustrates the values of an anomalous signal B at each node. Anomalies are highlighted (when 'Anomaly' is 1). Moreover, the vertices are represented in the x-axis, sorted according to the intra-community average signal, as in Fig. 3. The vertices within communities are ordered by index. It is possible to see that, when sorted by the community average, the anomalous nodes become more evident. However, they are within the mean and standard deviation values when considering the complete signal, which makes them difficult to detect using techniques that do not consider the network community structure.

The next section presents the computational experiments performed to evaluate the efficacy of *SpecF*.

5. Computational experiments

This section presents five experiments carried out to attest the efficiency of the anomaly detection method proposed in this paper. The first three experiments were carried out with artificial networks, whereas the fourth and fifth consist of tests performed with real-world datasets. The first experiment assesses the behavior of *SpecF* by varying the number of anomalies present in the

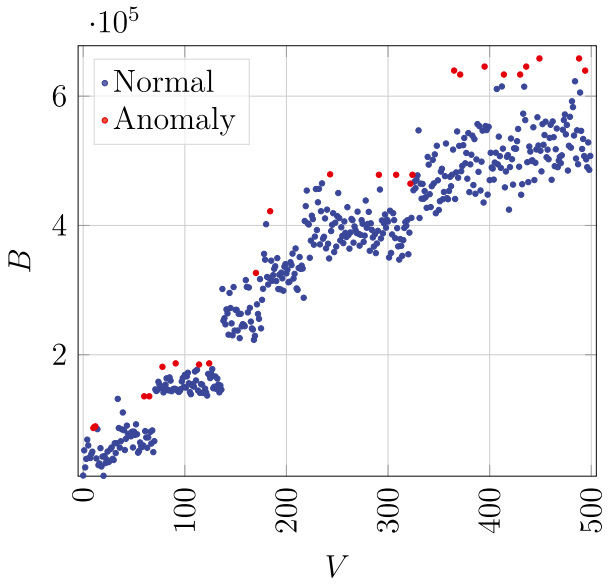

Fig. 4. Example of an anomalous signal *B*.

network. The second experiment analyzes *SpecF* when faced with variations in the intensity of the anomalies. The third experiment evaluates the effectiveness of the strategy in multiple executions with varied parameter values. Also an experiment with labeled data, the fourth experiment employs two publicly available IoT datasets, where a comparative analysis with state-of-the-art algorithms is performed. The last experiment presents a thorough analysis of an unlabeled COVID-19 dataset.

Before going into detail about the experiments with artificial networks, the generated networks and employed evaluation metrics are discussed.

5.1. Artificial networks

A set of artificial networks was generated using the software introduced in [22], referred to as LFR networks. By using this software, a set of undirected and unweighted benchmark graphs, with heterogeneous distributions of node degree and community sizes is created. The nodes of the generated LFR networks have an average degree d_G of 10 and a maximum degree $max\ d_G$ of 50. The parameters related to the exponent of the distribution of degrees (neg. exp. d_G) and community vertex count (neg. exp. $|\ C\ |$) are 2 and 1, respectively. The mixing parameter (μ) was set to be valued between 0.1 and 0.8, with a step size of 0.1. The mixture degree of the communities reflects how well separated the communities are since μ specifies the amount of inter-community edges. Therefore, low values for the mixture parameter produce networks with a more evident division in communities. The strategy introduced to generate normal and anomalous signals discussed in Section 4.4 was applied to the LFR networks. More details regarding the anomaly intensity (θ) and percentage of anomalies (AN) values are approached in the experiments. Table 1 summarizes the LFR and signal parameters used to generate the networks.

Fig. 5 illustrates an LFR network with 1000 vertices colored according to the signal values. The bluer a node, the higher its signal value. On the other hand, the redder the nodes, the lower their signal values. Larger nodes are those with a higher number of neighbors. The vertices are separated into different groups that represent the communities to which they belong. One may observe that intra-community vertices have similar colors, for example. However, there are some anomalous vertices, like the blue ones, that subtly clash with the standard of the vertices of their community.

Table 1
Parameters employed to generate the LFR networks and the normal/anomalous signal.

Type	Parameter	Values
LFR	n	500 and 1000
	μ (mixture parameter)	0.1, 0.2, ..., 0.8
	av. d_G	10
	max d_G	50
	neg. exp. d_G	2
	neg. exp. $\|\ C\ \|$	1
	min/max vertex count in communities	20/100
Signal	θ, AN (%)	1, 5, 10, 15, 20

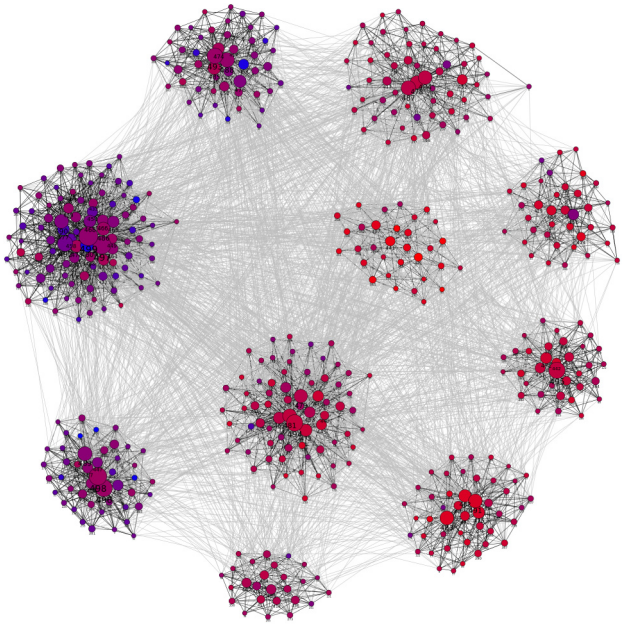

Fig. 5. Example of an LFR network with anomalies.

5.2. Evaluation metrics

Classic measures were used to evaluate and interpret the results obtained by *SpecF*.

The Receiver Operating Characteristic curve, or ROC curve, summarizes the trade-off between the false positive rate and true positive rate (also known as recall). The ROC curve allows the comparison of different models directly, and the area under the curve (AUC) can be used as a model quality quantifier. On the one hand, a random classifier, for example, could be represented by a diagonal curve that has an area of 0.5, starting at the bottom left and ending at the top right of the ROC space. On the other, a curve that starts in the lower-left corner, moves up to the upper left corner, and then advances to the upper right corner, adding up to an area of 1 would represent an ideal classifier.

In cases of binary classification problems with a skewed distribution of numbers of observations per class, Saito and Rehmsmeier [23] point out the precision–recall curve as a more informative metric. Since anomaly detection is often characterized by a skewed distribution (there are far more normal cases than anomalies), the precision–recall curve in the evaluation of the anomaly detection performance is also presented here. The precision–recall curve is similar to the ROC curve, but it

Table 2
Results of experiment I using the standard adjacency matrix.

AN	AUC-ROC	AP
1%	0.899 ± 0.042	0.511 ± 0.099
5%	0.936 ± 0.029	0.639 ± 0.098
10%	0.928 ± 0.023	0.704 ± 0.067
15%	0.953 ± 0.017	0.808 ± 0.050
20%	0.949 ± 0.023	0.793 ± 0.066

Table 3
Results of experiment I using the expanded adjacency matrix.

AN	AUC-ROC	AP
1%	0.931 ± 0.016	0.513 ± 0.050
5%	0.958 ± 0.014	0.682 ± 0.068
10%	0.975 ± 0.008	0.785 ± 0.053
15%	0.969 ± 0.015	0.810 ± 0.037
20%	0.978 ± 0.008	0.826 ± 0.048

Table 4
Results of experiment II using the standard adjacency matrix.

θ	Mean AUC ROC	Mean AP
1%	0.925 ± 0.057	0.427 ± 0.169
5%	0.923 ± 0.035	0.628 ± 0.080
10%	0.900 ± 0.020	0.681 ± 0.023
15%	0.868 ± 0.016	0.673 ± 0.032
20%	0.857 ± 0.021	0.681 ± 0.024

Table 5
Results of experiment II using the expanded adjacency matrix.

θ	Mean AUC ROC	Mean AP
1%	0.957 ± 0.030	0.431 ± 0.179
5%	0.946 ± 0.017	0.648 ± 0.074
10%	0.939 ± 0.015	0.716 ± 0.037
15%	0.919 ± 0.017	0.730 ± 0.039
20%	0.887 ± 0.014	0.697 ± 0.018

compares precision with recall for different thresholds. Precision corresponds to the proportion of true positives concerning the sum of true positives and false positives. A random classifier, in this case, is represented by a horizontal line with a value proportional to the number of positive cases in the dataset. An efficient classifier, on the other hand, would be represented by a line that approaches the upper right point of the plot. Again, the Area Under the Precision–Recall Curve (AUC-PR) can be used as a quantifier of the classification model's ability, also called Average Precision (AP) [23].

5.3. Experiment I

The first experiment seeks to evaluate the performance of *SpecF* by varying the number of anomalies and keeping the intensity θ at 5%. To assess the robustness of the method, this experiment employs a total of 250 networks generated by considering the following methodology:

- Generate five different LFR networks with $\mu = 0.1$, 500 nodes and the remaining LFR parameters at the values presented in Table 1;
- Apply the normal signal generator algorithm (Algorithm 1) once to each of the five networks;
- Apply the anomalous signal generator algorithm (Algorithm 4) to each network ten times for each of the *AN* values presented in Table 1 and fixing θ at value 5%. Therefore, this step produces 50 different networks (one for each execution of the anomalous signal generator) for each *AN* value, totaling 250 networks.

Table 2 presents the mean and standard deviation of the AUC-ROC and AP values considering the 50 networks per *AN*, when a standard adjacency matrix is used in *SpecF*. These results show that the performance of *SpecF* improves as *AN* increases. Similarly, Table 3 presents the results of the same experiment, but using the expanded adjacency matrix W instead of the adjacency matrix A to calculate the Fourier transform defined in Section 2.1. A comparison of the results with both matrices reveals that in all cases the anomaly detection strategy performed much better using the expanded adjacency matrix W, for all *AN* values and both AUC-ROC and AP metrics.

5.4. Experiment II

The second experiment compares the accuracy of *SpecF* considering different anomaly intensities and keeping *AN* fixed at 5%. This experiment was carried out using the same networks generated in the second step of the methodology presented in the earlier section. Therefore, for this experiment, the third step of the methodology, the one that produces the anomalous signals, consists in following the procedure:

- Apply the anomalous signal generator algorithm (Algorithm 4) to each network ten times for each of the θ values presented in Table 1 and fixing *AN* at 5%.

Table 4 reports the results of *SpecF* when the standard adjacency matrix is used. In cases where the anomaly is extremely hard to identify, as in cases where the value of the signal at the anomalous nodes are only 1% greater than the maximum signal in their community, the accuracy of the model is poor. In the case with anomaly intensity $\theta = 1\%$, for example, the mean AP was approximately 0.43, which is very close to the values obtained by a random classifier. Table 5 presents the results of this experiment using the expanded adjacency matrix W instead. Again, the use of the matrix W improves the results in anomaly detection in all scenarios, despite the anomaly intensity value and performance metric considered.

5.5. Experiment III

This experiment evaluates the performance of *SpecF* in a considerably larger set of networks. The methodology to generate the set of 7200 networks had the following steps:

- Generate five different networks for each pair (μ, n) of values presented in Table 1, totaling 80 different networks;
- Apply the normal signal generator (Algorithm 4) once to each of the 80 networks (40 networks with 500 nodes and 40 networks with 1000 nodes);
- Apply the anomalous signal generator (Algorithm 5) to each network ten times for each of the nine possible pairs (θ, AN), where $\theta, AN \in \{1, 5, 10\}$, totaling 400 networks for each triple (n, θ, AN).

Tables 6 and 7 report the mean AUC-ROC and AP for each triplet (n, θ, AN), considering *SpecF* with the adjacency (A) and expanded (W) matrices. Therefore, each row of these tables corresponds to the average results of 400 different networks. It is possible to observe that, in all cases, better results are obtained when the expanded adjacency matrix W is used instead of the adjacency matrix A in *SpecF*.

Fig. 6 illustrates the mean AP values for different values of μ and θ comparing *SpecF* with A and W as input matrices. It is evident that, in all cases, *SpecF* performed better when it adopted matrix W instead of matrix A. Besides, the results are better for

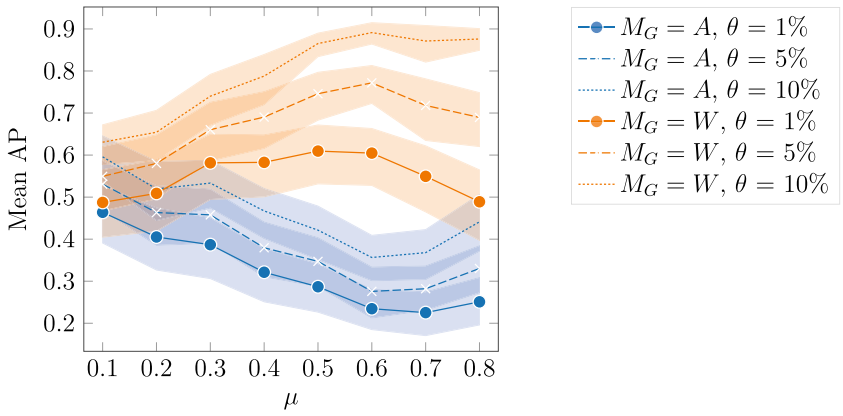

Fig. 6. Mean AP values for different values of μ and θ.

Table 6
Results of experiment III varying different parameters for $n = 500$.

n	AN	θ	M_G	Mean AUC ROC	Mean AP
500	1%	1%	A	0.756 ± 0.16	0.170 ± 0.15
500	1%	1%	W	**0.956 ± 0.04**	**0.339 ± 0.18**
500	1%	5%	A	0.797 ± 0.15	0.233 ± 0.18
500	1%	5%	W	**0.971 ± 0.03**	**0.492 ± 0.21**
500	1%	10%	A	0.847 ± 0.12	0.322 ± 0.21
500	1%	10%	W	**0.990 ± 0.02**	**0.712 ± 0.22**
500	5%	1%	A	0.740 ± 0.11	0.286 ± 0.13
500	5%	1%	W	**0.947 ± 0.02**	**0.550 ± 0.11**
500	5%	5%	A	0.780 ± 0.11	0.356 ± 0.14
500	5%	5%	W	**0.970 ± 0.02**	**0.715 ± 0.12**
500	5%	10%	A	0.836 ± 0.09	0.463 ± 0.16
500	5%	10%	W	**0.985 ± 0.02**	**0.836 ± 0.13**
500	10%	1%	A	0.714 ± 0.12	0.364 ± 0.13
500	10%	1%	W	**0.931 ± 0.02**	**0.631 ± 0.09**
500	10%	5%	A	0.763 ± 0.11	0.444 ± 0.14
500	10%	5%	W	**0.958 ± 0.02**	**0.758 ± 0.09**
500	10%	10%	A	0.815 ± 0.09	0.529 ± 0.13
500	10%	10%	W	**0.975 ± 0.02**	**0.855 ± 0.10**

Table 7
Results of experiment III varying different parameters for $n = 1000$.

n	AN	θ	M_G	Mean AUC ROC	Mean AP
1000	1%	1%	A	0.747 ± 0.13	0.140 ± 0.11
1000	1%	1%	W	**0.951 ± 0.03**	**0.292 ± 0.13**
1000	1%	5%	A	0.792 ± 0.11	0.185 ± 0.14
1000	1%	5%	W	**0.970 ± 0.03**	**0.444 ± 0.18**
1000	1%	10%	A	0.848 ± 0.10	0.257 ± 0.16
1000	1%	10%	W	**0.984 ± 0.02**	**0.614 ± 0.21**
1000	5%	1%	A	0.739 ± 0.10	0.262 ± 0.12
1000	5%	1%	W	**0.944 ± 0.02**	**0.543 ± 0.10**
1000	5%	5%	A	0.782 ± 0.09	0.330 ± 0.13
1000	5%	5%	W	**0.963 ± 0.02**	**0.672 ± 0.13**
1000	5%	10%	A	0.823 ± 0.08	0.406 ± 0.14
1000	5%	10%	W	**0.979 ± 0.02**	**0.793 ± 0.14**
1000	10%	1%	A	0.721 ± 0.10	0.345 ± 0.11
1000	10%	1%	W	**0.927 ± 0.02**	**0.623 ± 0.08**
1000	10%	5%	A	0.757 ± 0.09	0.401 ± 0.12
1000	10%	5%	W	**0.954 ± 0.02**	**0.742 ± 0.10**
1000	10%	10%	A	0.809 ± 0.08	0.489 ± 0.13
1000	10%	10%	W	**0.970 ± 0.02**	**0.825 ± 0.11**

larger values of θ, as the anomalies are more evident. The primary weakness of *SpecF* considering both matrices is apparently in cases where the anomaly is very slight and difficult to be identified.

The anomaly detection strategy using the adjacency matrix A presents worse results as the mixture parameter μ increases. In contrast, the performance improves as the value of μ increases when *SpecF* employs matrix W. This behavior can be explained by the characteristics of the anomalous signal studied in this paper. We analyzed a signal whose anomaly is defined as a vertex whose signal value is different from that of the others in the same community. If the percentage of inter-community edges is low, the average signal values between communities are more distant. Therefore, even if the signal at a node differs from the community where it is located, it will still be closer to the average value of that community than to any other community's average value. When the mixture parameter is higher, the average values of the communities are closer. Moreover, a node whose signal value differs from the signal of its community may be close to the average values of neighboring communities. This characteristic is reflected in the Fourier transform through the expanded adjacency matrix and, then, demonstrated in the applied filter.

The results for experiments I, II and III are clearly superior for all compared scenarios in favor of the usage of an expanded adjacency matrix. Therefore, statistical tests like Wilcoxon signed-rank test [24] attest the superior performance of the expanded adjacency matrix when compared to the standard counterpart, at 99% of significance level.

5.6. Experiment IV

This experiment employs two publicly available databases. The first, SWaT, contains a water treatment testbed collected by a sensor-actuator network with 51 nodes [25]. The SWaT dataset regards data collected of a total of 15 days of the water distribution operation, being 11 of normal operations and 4 of day operations with cyber-attacks. This dataset has an anomaly rate of 5.77%. The second, called WADI, contains a water distribution testbed collected by a sensor-actuator network with 123 nodes [26]. The duration of the attacks was from 2 to 25 min. The WADI dataset regards data collected from a total of 16 days of the water distribution operation, being 14 of normal operation days and 2 with cyber-attacks. The duration of the attacks was from 1.5 to 30 min. This dataset has an anomaly rate of 5.75%.

A comparative analysis is carried out by presenting the results reported in [3]. The authors tested their deep learning-based algorithm, GTA, designed to detect anomalies in multivariate time series data. GTA defines anomalies according to a novel graph learning strategy proposed by the authors referred to as Influence

Table 8
Results of anomaly detection algorithms on SWaT dataset.

Methods	Precision (%)	Recall (%)	F1-score
PCA	24.92	21.63	0.23
KNN	7.83	7.83	0.08
FB	10.17	10.17	0.10
AE	72.63	52.63	0.61
DAGMM	27.46	69.52	0.39
LSTM-VAE	96.24	59.91	0.74
MAD-GAN	98.97	63.74	0.77
GDN	99.35	68.12	0.81
GTA	74.91	96.41	0.84
SpecF	55.54	62.96	0.47

Table 9
Results of anomaly detection algorithms on WADI dataset.

Methods	Precision (%)	Recall (%)	F1-score
PCA	39.53	5.63	0.10
KNN	7.76	7.75	0.08
FB	8.60	8.60	0.09
AE	34.35	34.35	0.34
DAGMM	54.44	26.99	0.36
LSTM-VAE	87.79	14.45	0.25
MAD-GAN	41.44	33.92	0.37
GDN	97.50	40.19	0.57
GTA	74.56	90.50	0.82
SpecF	49.88	49.44	0.35

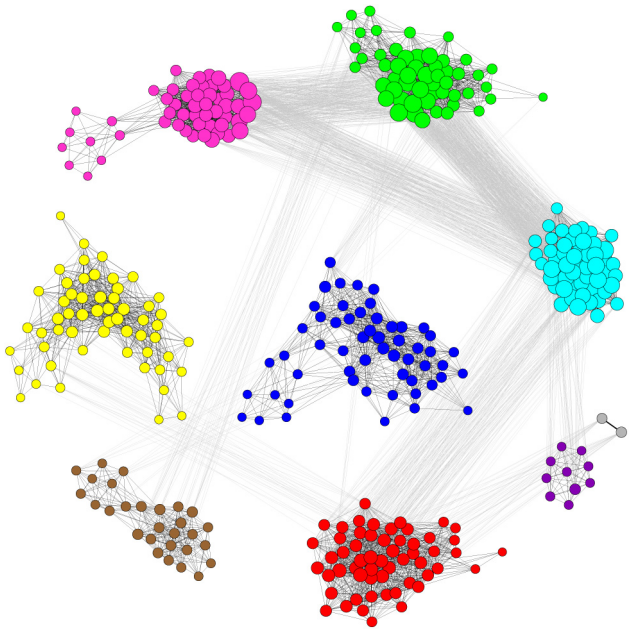

Fig. 7. Graph representing the districts of São José dos Campos, SP, Brazil.

Propagation convolution. They compared GTA with eight state-of-the-art anomaly algorithms in multivariate time series data: Principal Component Analysis (PCA), *k*-Nearest Neighbor (KNN) [27], Feature Bagging (FB) [28], Autoencoders (AE) [29], Long Short Term Memory-based Variational Autoencoder (LSTM-VAE) [30], Multivariate Anomaly Detection with Generative Adversarial Networks (MAD-GAN) [31], Deep Autoencoding Gaussian Mixture Model (DAGMM) [32] and Graph Deviation Network (GDN) [33].

WADI and SWaT were modeled in two graphs with 112 and 51 vertices, respectively, where the vertices correspond to the sensor nodes and the edges between them contain the Pearson correlation between the values measured by the sensors over time. An edge only exists if the pairwise correlation is greater than 0.5.

As *SpecF* is not specifically designed to approach time series data, a methodology to define the anomalous signals according to an analysis of pairwise consecutive 30-second time windows of the time series is followed as explained next.

- Given a pair of consecutive time windows, calculate their distance using the Dynamic Time Warping (DTW) [34] with the implementation proposed by [35].
- For the two time windows, consider the DTW the signal on the node and apply *SpecF*.
- The anomalous nodes returned by *SpecF* are deemed anomalous in the entire analyzed time windows.

Tables 8 and 9 present the Precision, Recall and F1-Score of the results obtained by these algorithms and *SpecF* on SWaT and WADI datasets, respectively. The F1-score measure is given Eq. (12).

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precison+Recall}} \tag{12}$$

According to the F1-score, *SpecF* outperformed the PCA, KNN, FB and DAGMM methods in both datasets. In the WADI dataset, *SpecF* also outperformed the LSTM-VAE method. In contrast, *SpecF* was outperformed by MAD-GAN, GDN and GTA. Nonetheless, different from these methods, *SpecF* does not need any training steps to obtain information on the non-anomalous state of the system. Therefore, we believe that *SpecF* presented a good performance and was able to produce satisfactory results with a low computational cost. Although it was not specifically designed to deal with time series, the results point out that *SpecF* achieved good results, being a good option to identify anomalies in multiple time series that show some correlation.

5.7. Case study - COVID-19 dataset

Here an experiment with real data to validate the performance of *SpecF* was carried out. The dataset used in the experiment contains the number of daily COVID-19 cases in each of the districts of São José dos Campos, an upstate city of São Paulo - Brazil. To model the data, let G be a graph where the nodes represent districts of the city. A pair of nodes v_i and v_j is connected if the distance between the corresponding districts is less than 5 km. In this case, the weight w_{ij} of the edge is proportional to the distance between districts represented by nodes v_i and v_j. Fig. 7 illustrates the graph G, where the nodes were colored according to their communities (meaning that nodes with identical color belong to the same community). The case study graph has 366 nodes and the community structure of the network was identified using a community detection algorithm named Louvain [36]. The Louvain method is a benchmark community detection heuristic that identifies the community structure through the modularity optimization [37].

The data used was collected from March 5, 2020 to March 25, 2021, totaling 385 days (equivalent to 55 weeks). For each week, only the nodes that represented districts that presented at least 2 cases in the previous and current weeks are included. Thus, for each week, the number of nodes and the community structure of the graph are modified due to the removal of some nodes that represent districts that did not have a sufficient number of cases in the analyzed weeks. We define the signal $B^{(d)}$ as the weekly variation in the number of COVID-19 cases, where $b_i^{(d)}$ is the ratio between the sum of the total number of cases registered between days $d-7, d-6, \ldots, d-1$ and days $d, d+1, \ldots, d+6$ in the district represented by node v_i. The proposed anomaly detection strategy was applied to signal $B^{(d)}$ to identify anomalies in the

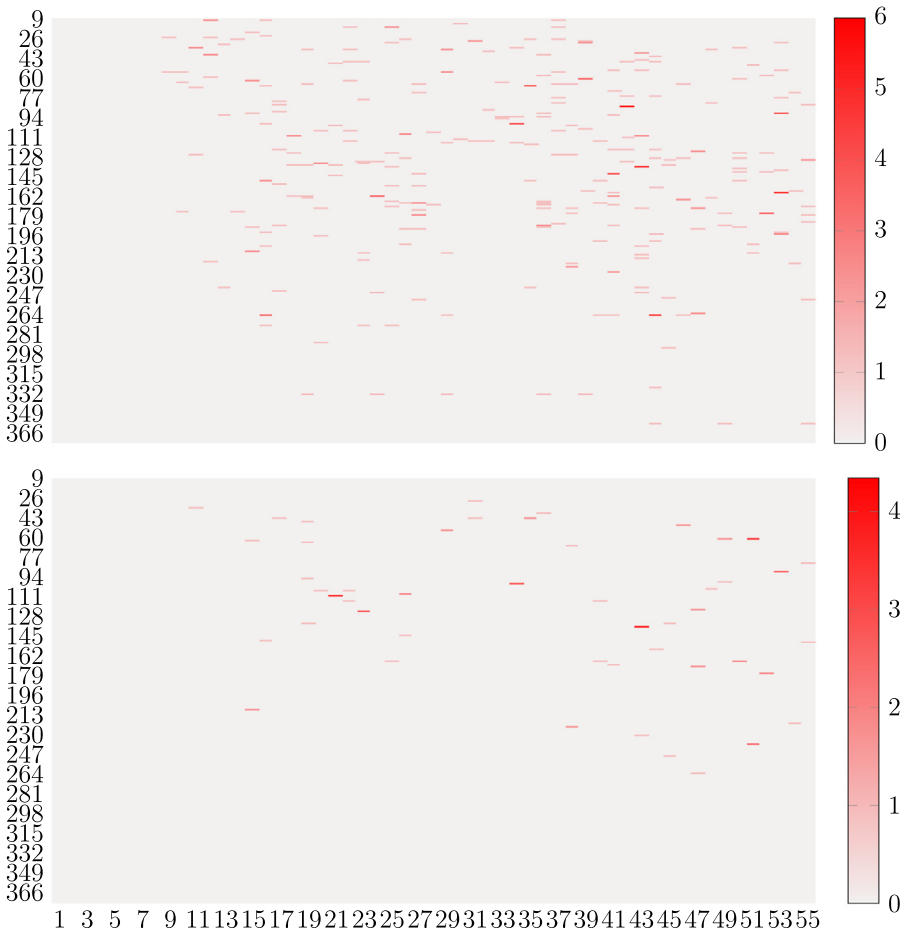

Fig. 8. Weekly variation in the number of cases in each of the districts of São José dos Campos, SP, Brazil.

weekly variation of cases. The hypothesis is that if the number of cases in a given district increases, the number of cases in adjacent districts must increase as well. As a consequence, *SpecF* may identify neighbor districts whose variation in the number of cases differs from a given district in a certain week.

The upper part of Fig. 8 illustrates the weekly variation in the number of cases in each of the districts. The 366 different districts are represented by the y-axis whereas the x-axis represents the weeks. The signal corresponding to week 2, for example, is represented by $B^{(14)}$, since d is reached by multiplying the week number by 7. The intensity of the colors represents the weekly variation in the number of cases. The redder the greater the increase in the weekly number of cases. The modeled graph is subject to *SpecF* to find the $Y^{(d)}$ signal that stores the abnormality of the nodes. The lower part of Fig. 8 illustrates the abnormality in the variation on the number of cases for each district. It is possible to notice that the amount of light dots at the bottom is considerably smaller. This is because *SpecF* filters the signal and highlights the abnormalities, so that only potentially anomalous variations are maintained in the $Y^{(d)}$ signal.

To validate the hypothesis behind the introduced anomaly detection algorithm and attest to the veracity of the result, we compared the values of the nodes identified as anomalous with the values of the other nodes of the same community for every day $d = \{0, 7, 14, \ldots, 378, 385\}$. For example, if $y_i^{(d)}$ is pointed out as an anomaly, it means that the variation in the number of cases between the days $d-7, d-1$ and $d, d+6$ in the district represented by node v_i differs from the standard of the community to which v_i node belongs.

This behavior is best seen in Fig. 9, which illustrates a map with the weekly variation in the number of cases within the districts, with a single anomaly. Shades closer to blue represent a drop in the number of cases, while shades closer to yellow represent an increase in the total of COVID-19 cases. It is possible to observe that most districts experience a reduction in the number of cases, while only one district has an increase. This increase is considered an abnormal incident since the adjacent districts have the number of COVID-19 cases reduced.

A total of 117 values $y_i^{(d)}$ were identified as anomalous. Approximately 95% of these anomalies have the same characteristic: the node identified as anomalous in a time window was the only one that had an increase in the number of cases in its community. This highlights neighborhoods and routes of dissemination of the virus that may require public health interventions. Fig. 10 shows this behavior for a sample of 50 of the 117 anomalies identified. Each box-plot illustrates the variation values of nodes from the same community to which the anomalous node belongs. The variation values of the anomalous nodes are highlighted in blue dots. The anomalous node presents an increase in the number of COVID-19 cases, in contrast to the other nodes of the same community, that showed a reduction in the number of cases.

We have identified that new anomalies usually occur in districts that are adjacent to districts where abnormal variations were observed in the previous two weeks. In 80.3% of the cases, if a node v_i is indicated as anomalous in the signal $y_i^{(d)}$, then there is a node v_j, neighbor of v_i, that is pointed out as anomalous in the signal $y_j^{(d-7)}$ or $y_j^{(d-14)}$, where $d \geq 14$. The results obtained in this case study indicate that our hypothesis was correct. The proposed strategy can identify nodes whose characteristics differ

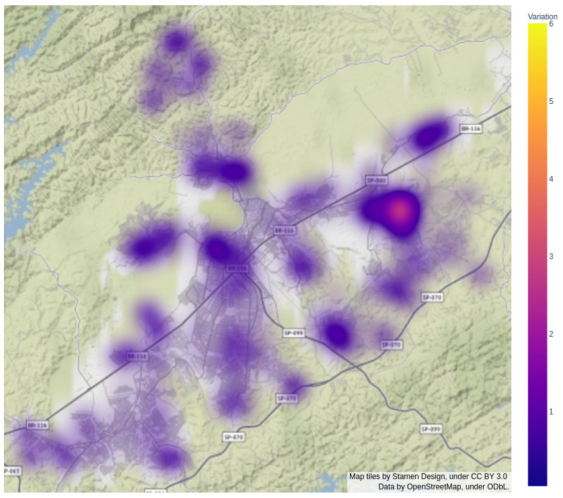

Fig. 9. Heat map with the weekly variation in the number of cases in each of the district of São José dos Campos, SP, Brazil.

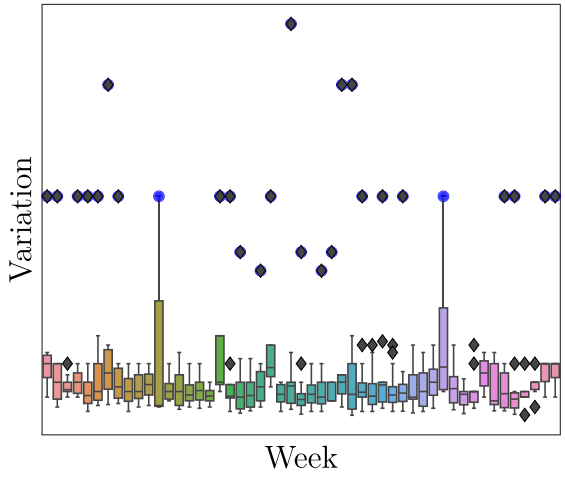

Fig. 10. Box-plots of the weekly variation in the number of COVID-19 cases considering communities with anomalous nodes.

from the others from the same community, even in real-world scenarios. Furthermore, it shows that the anomaly trails through the network, such that an anomalous node hits adjacent nodes that become anomalous a few weeks later. Such analyzes can support anticipated public health decisions for a better management of the pandemics evolution in the city.

6. Conclusions

This paper proposed a method to detect anomalies in signals indexed by graphs, called *SpecF*. *SpecF* uses an extended Laplacian matrix that considers the community structure as the basis for the Fourier transform and a low-pass filter designed to attenuate high frequencies in the signal spectrum. The proposed filter uses as cut-off frequency the eigenvalue whose position is the number of communities in the network. This position can be passed as a parameter or automatically calculated through eigenvalue analysis of the adopted matrix.

Computational experiments attest to the accuracy of *SpecF* in the task of detecting anomalies. Moreover, the automatic calculation of the cut-off frequency using the network community structure makes the method independent of parameters and more robust when dealing with real-world networks. In addition, the experiments show that the proposed method performed better when adopting the expanded adjacency matrix W in the Fourier transform instead of the standard adjacency matrix A.

The results suggest that *SpecF* is an interesting solution to detect anomalies based on signals indexed by graphs. More specifically, the method can successfully identify nodes in the networks whose signal values differ from the expected according to the community where they belong. Therefore, the proposed method effectively finds anomalies in signals whose expected variation is low in nodes from the same community. Moreover, a comparative analysis with state-of-the-art algorithms in two publicly available real world datasets with known anomalies show a good performance of *SpecF*. *SpecF* outperformed the classic algorithms, being competitive with some recent reference methods, even though it has not any learning step and has not been designed for the time series anomaly detection task.

Additionally, a case study was presented to validate the proposed strategy. It consists of data related to the number of COVID-19 cases in the different districts of São José dos Campos, an upstate city of São Paulo, Brazil. By using *SpecF*, it was possible to detect districts whose number of cases soared while neighboring districts showed a reduction in the number of cases in the same week. The number of COVID-19 cases in a district in a given week was evaluated abnormally high or low. The number of COVID-19 cases in some neighbor districts in the former one or two weeks is also anomalous in most of the cases. This gives evidence that the uncontrolled dissemination of the virus occurs following a "path" in the city.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors of this paper are grateful to Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Brazil (Grant Number: 2017/24185-0), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) (Grant: 88887.507037/2020-00) and to the health department of São José dos Campos for providing the COVID-19 data.

References

[1] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q.Z. Sheng, H. Xiong, L. Akoglu, A comprehensive survey on graph anomaly detection with deep learning, IEEE Trans. Knowl. Data Eng. (2021).
[2] X. Dong, D. Thanou, M. Rabbat, P. Frossard, Learning graphs from data: A signal representation perspective, IEEE Signal Process. Mag. 36 (3) (2019) 44–63.
[3] Z. Chen, D. Chen, X. Zhang, Z. Yuan, X. Cheng, Learning graph structures with transformer for multivariate time series anomaly detection in IoT, IEEE Internet Things J. (2021).
[4] R. Francisquini, R. Berton, S.G. Soares, D.S. Pessotti, M.F. Camacho, D. Andrade-Silva, U. Barcick, S.M. Serrano, R. Chammas, M.C.V. Nascimento, A. Zelanis, Community-based network analyses reveal emerging connectivity patterns of protein-protein interactions in murine melanoma secretome, J. Proteom. 232 (2021) 104063.
[5] L. Akoglu, H. Tong, D. Koutra, Graph based anomaly detection and description: a survey, Data Min. Knowl. Discov. 29 (3) (2015) 626–688.
[6] K. Choi, J. Yi, C. Park, S. Yoon, Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines, IEEE Access (2021).
[7] A. Ortega, P. Frossard, J. Kovačević, J.M. Moura, P. Vandergheynst, Graph signal processing: Overview, challenges, and applications, Proc. IEEE 106 (5) (2018) 808–828.
[8] A. Sandryhaila, J.M. Moura, Discrete signal processing on graphs: Graph Fourier transform, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 6167–6170.

[9] D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains, IEEE Signal Process. Mag. 30 (3) (2013) 83–98.
[10] S. Chen, A. Sandryhaila, J.M. Moura, J. Kovacevic, Signal denoising on graphs via graph filtering, in: 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP), IEEE, 2014, pp. 872–876.
[11] Y. Gao, D. Yu, Fault diagnosis of rolling bearing based on Laplacian regularization, Appl. Soft Comput. 111 (2021) 107651.
[12] F.R. Chung, F.C. Graham, Spectral Graph Theory, American Mathematical Soc., 1997.
[13] P. Püschel, J.M. Moura, Algebraic signal processing theory: Foundation and 1-D time, IEEE Trans. Signal Process. 56 (8 I) (2008) 3572–3585.
[14] M. Püschel, J.M. Moura, Algebraic signal processing theory: 1-D space, IEEE Trans. Signal Process. 56 (8 I) (2008) 3586–3599.
[15] A. Sandryhaila, J.M. Moura, Discrete signal processing on graphs: Frequency analysis, IEEE Trans. Signal Process. 62 (12) (2014) 3042–3054.
[16] U. Von Luxburg, A tutorial on spectral clustering, Statist. Comput. 17 (4) (2007) 395–416.
[17] A. Buades, B. Coll, J.-M. Morel, A review of image denoising algorithms, with a new one, Multiscale Model. Simul. 4 (2) (2005) 490–530.
[18] J. Li, H. Izakian, W. Pedrycz, I. Jamal, Clustering-based anomaly detection in multivariate time series data, Appl. Soft Comput. 100 (2021) 106919.
[19] J. Gao, F. Liang, W. Fan, C. Wang, Y. Sun, J. Han, On community outliers and their efficient detection in information networks, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2010, pp. 813–822.
[20] E. Müller, P.I. Sánchez, Y. Mülle, K. Böhm, Ranking outlier nodes in subspaces of attributed graphs, in: 2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW), IEEE, 2013, pp. 216–222.
[21] H.E. Egilmez, A. Ortega, Spectral anomaly detection using graph-based filtering for wireless sensor networks, in: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2014, pp. 1085–1089.
[22] A. Lancichinetti, S. Fortunato, Community detection algorithms: a comparative analysis, Phys. Rev. E 80 (5) (2009) 056117.
[23] T. Saito, M. Rehmsmeier, The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets, PLoS One 10 (3) (2015) e0118432.
[24] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[25] A.P. Mathur, N.O. Tippenhauer, SWaT: A water treatment testbed for research and training on ICS security, in: 2016 International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater), IEEE, 2016, pp. 31–36.
[26] C.M. Ahmed, V.R. Palleti, A.P. Mathur, WADI: a water distribution testbed for research in the design of secure cyber physical systems, in: Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks, 2017, pp. 25–28.
[27] F. Angiulli, C. Pizzuti, Fast outlier detection in high dimensional spaces, in: European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2002, pp. 15–27.
[28] A. Lazarevic, V. Kumar, Feature bagging for outlier detection, in: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 2005, pp. 157–166.
[29] C.C. Aggarwal, Outlier Analysis, Springer, 2013, http://dx.doi.org/10.1007/978-1-4614-6396-2.
[30] D. Park, Y. Hoshi, C.C. Kemp, A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder, IEEE Robot. Autom. Lett. 3 (3) (2018) 1544–1551.
[31] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, S.-K. Ng, MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks, in: International Conference on Artificial Neural Networks, Springer, 2019, pp. 703–716.
[32] B. Zong, Q. Song, M.R. Min, W. Cheng, C. Lumezanu, D. Cho, H. Chen, Deep autoencoding gaussian mixture model for unsupervised anomaly detection, in: International Conference on Learning Representations, 2018, pp. 1–19.
[33] A. Deng, B. Hooi, Graph neural network-based anomaly detection in multivariate time series, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 4027–4035.
[34] H. Sakoe, Dynamic-programming approach to continuous speech recognition, in: 1971 Proc. the International Congress of Acoustics, Budapest, 1971.
[35] W. Meert, K. Hendrickx, T.V. Craenendonck, wannesm/dtaidistance v2.0.0, 2020, http://dx.doi.org/10.5281/zenodo.3981067.
[36] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. Theory Exp. 2008 (10) (2008) P10008.
[37] M.E. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.