MAD: Multi-Scale Anomaly Detection in Link Streams

Esteban Bautista, Laurent Brisson, Cécile Bothorel and Grégory Smits
{esteban.bautista-ruiz,laurent.brisson,cecile.bothorel,gregory.smits}@imt-atlantique.fr
IMT Atlantique - Lab-STICC - CNRS UMR 6285
Brest, France

ABSTRACT

Given an arbitrary group of computers, how to identify abnormal changes in their communication pattern? How to assess if the absence of some communications is normal or due to a failure? How to distinguish local from global events when communication data is extremely sparse and volatile? Existing approaches for anomaly detection in interaction streams, focusing on edge, nodes and graphs, lack the flexibility to monitor arbitrary communication topologies. Moreover, they rely on structural features that are not adapted to highly sparse settings. In this work, we introduce MAD, a novel Multi-scale Anomaly Detection algorithm that (i) allows to query for the normality/abnormality state of an arbitrary group of observed/non-observed communications at a given time; and (ii) handles the highly sparse and uncertain nature of interaction data through a scoring method that is based on a novel probabilistic and multi-scale analysis of sub-graphs. In particular, MAD is (a) *flexible*: it can assess if any time-stamped subgraph is anomalous, making edge, node and graph anomalies particular instances; (b) *interpretable*: its multi-scale analysis allows to characterize the scope and nature of the anomalies; (c) *efficient*: given historical data of length N and M observed/non-observed communications to analyze, MAD produces an anomaly score in $O(NM)$; and (d) *effective*: it significantly outperforms state-of-the-art alternatives tailored for edge, node and graph anomalies.

KEYWORDS

Anomaly detection, Link streams, Multi-scale analysis, Anomalous subgraphs

ACM Reference Format:
Esteban Bautista, Laurent Brisson, Cécile Bothorel and Grégory Smits. 2018. MAD: Multi-Scale Anomaly Detection in Link Streams. In *ACM WSDM 2024: The 17th ACM International Conference on Web Search and Data Mining, March 04–08, 2024, Mérida, Yucatán, Mexico.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

A link stream is a set of triplets (t, u, v) modeling that u and v interacted at time t. Triplets in a link stream may represent that computer u sent a packet to computer v at time t or that bank account u made a transaction to account v at time t. Detection of likely/unlikely interactions that suddenly disappear/appear is an important step towards identifying various events crucial interest, such as financial frauds, network attacks, or infrastructure failures. For example, consider the case illustrated in Figure 1, depicting interactions between servers and users requesting their services. It is rather normal that the servers frequently exchange traffic between them and also with some regular users. Figure 1 represents such users and servers that frequently interact in blue, while it represents users that connect less frequently in grey. If at a given time the communication pattern depicted in the left panel is observed, the situation can be labeled as normal given that the observed interactions only concern entities that usually interact together. Yet, if the observation corresponds to the one depicted in the right panel, such change in the communication pattern may be an indication of a failure or an attack. Another example can be a bank account that suddenly starts to make transactions to several unexpected accounts. Such behavior may be indicative of a fraud. (hackers have taken control of the machines) (engineers are troubleshooting)

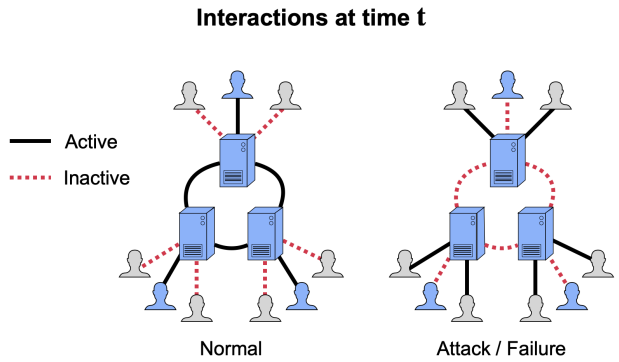

Figure 1: Examples of normal and abnormal communication patterns. A group of servers usually exchange between them and with external users. Historical data may suggest that interactions between blue-colored nodes are highly likely. The sudden halt of likely traffic and the emergence of unlikely one may be an indication of an attack (hackers have taken control of the machines) or a failure (engineers are troubleshooting).

To spot the aforementioned events, numerous link stream-based anomaly detection algorithms have been proposed in recent years. In a nutshell, such algorithms can be seen as black boxes that receive two inputs, a *query* and a *context*, and answer to the *question*: how abnormal is the query given the context? The essential differences between proposed algorithms are: (i) which types of queries they accept; (ii) how they define anomalies; and (iii) how they exploit the context. For example, numerous algorithms accept time-stamped

ACM WSDM 2024, March 04–08, 2024, Mérida, Yucatán, Mexico

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

edges as queries, yet they differ in the criterion used to label a query as abnormal: some do it when the query implies a sudden change in edge counts [1], node embeddings [2], or walk statistics [3], while others do it when the query cannot be well predicted from the past [4, 5]. Algorithms addressing coarser resolution queries, like time-stamped nodes [6, 7] or entire graph snapshots [8–11], have also been proposed. Such algorithms also vary in their anomaly definitions and use of context. Namely, nodes may be deemed abnormal if they suddenly change their centrality [6] or communication counts [7], while graphs may be considered abnormal if they have sudden densifications [9], spectrum changes [10], or community re-configurations [11], to list some examples.

In spite of numerous successes, existing anomaly detection algorithms remain not fully satisfactory. In particular, the fact that they only accept time-stamped subgraphs of a specific form as queries (either edges, nodes, or entire graphs) makes them too rigid for several real-world use cases. In many situations it is desirable to question an algorithm if an arbitrary communication topology behaves abnormally: take for instance the example of Figure 1, where one wants to track the communications between a specific group of servers and users; or take the case we aim to monitor the transactions between a specific group of bank accounts believed to belong to a criminal organization. To handle these different cases, an ultimate solution is to dispose of an algorithm capable to respond to queries consisting of arbitrary time-stamped subgraphs. Some algorithms for anomalous subgraph detection have been proposed [12–14], yet such approaches automatically search for subgraphs that meet some criteria, like being a dense community, thus preventing users from querying the algorithms with arbitrary subgraphs. It is also worth noticing that most algorithms targeting coarse-grain queries rely on anomaly definitions that do not satisfactorily account for the highly uncertain and sparse nature of temporal interactions. For example, many works rely on anomaly definitions that are based on changes to node centralities, walk statistics, spectral properties or community structures. These definitions implicitly assume dense link streams that slowly evolve, which may be unrealistic in numerous situations.

The aim of this work is to address the limitations listed above. We introduce MAD, a novel anomaly detection algorithm that (i) accepts arbitrary time-stamped subgraphs as queries; and (ii) labels queries as abnormal if they have/lack many unlikely/likely interactions, thus allowing to handle the data uncertainty and sparsity. MAD is based on a novel multi-scale probabilistic analysis for subgraphs that essentially permits to map a query sub-graph into a set of random variables from which it is possible to identify the scale(s) at which a queried sub-graph cannot be well explained from its past activity. Our main contributions are as follows:

- *Flexibility*: MAD can be used to determine if any arbitrary time-stamped subgraph is anomalous, thus making edge, node and graph anomaly detection particular instances of our approach.
- *Interpretability*: the developed multi-scale analysis allows to identify the scale(s) and the nature of the event(s) making a query abnormal.
- *Efficiency*: given a query of size M and a historical context of duration N, MAD answers in $O(NM)$.
- *Effectiveness*: MAD significantly outperforms state-of-the-art alternatives for edge, node and graph anomaly detection in the tasks of identifying communications that abnormally appear, disappear or get redirected.

2 NOTATIONS AND RELATED WORKS

2.1 Notations

Let V be a set of vertices, T refer to the non-negative integers, $\mathcal{E} = V \times V$ denote a relation space, and $\phi \subseteq \mathcal{E}$ be an arbitrary set of relations of size $|\phi| = M$. A discrete-time link stream is denoted by the set $L \subseteq T \times \mathcal{E}$. The restriction of L to a time interval $[t_1, t_2]$ and set of relations ϕ is expressed as $L(t_1 : t_2, \phi) = \{(t, u, v) \in L : t_1 \leq t \leq t_2, (u, v) \in \phi\}$. The case $t_1 = t_2$ corresponds to a slice, or snapshot, of L. Strictly speaking, the interactions of a slice remain time-stamped, yet in some situations it is useful to strip the time reference from them so that they can be considered as the edges of a graph that is independent of time. Therefore, given the restriction sets $t_1 = t_2 = t$ and ϕ, we let $L(t : t, \phi)$ refer to the slice of L in which interactions remain time-stamped, while we let $L(t, \phi)$ denote the the case in which the time-stamps are striped-out. Relations are considered directed, hence $(u, v) \neq (v, u)$. Moreover, for the sake of notation lightness, the relations emerging from node u are denoted by $\mathcal{E}_u = \{(u, v) : v \in V\}$.

In this work, we extensively use indicator functions to characterize subsets of a set: a binary function indicating which elements from the set belong or not to the subset. The concept plays an important role in this work given that algorithms working directly with a set $A \subseteq B$ cannot make decisions based on the elements of B not included in A, as they ignore them; while algorithms working with the indicator function know such information. Formally, the indicator function of $A \subseteq B$ is denoted by $\mathbb{1}_A^B : B \to \{0, 1\}$, where $\mathbb{1}_A^B(x) = 1$ if $x \in A$ and zero otherwise. Thus, the indicator function of a link stream is given by $\mathbb{1}_L^{T \times \mathcal{E}}(t, u, v) = 1$ if $(t, u, v) \in L$ and zero otherwise. Also, with the aim of notation lightness, we denote the sum of a function $f : B \to \mathbb{R}$ over a sub-domain $C \subseteq B$ by $f(C) = \sum_{c \in C} f(c)$. Hence, $\mathbb{1}_A^B(C) = \sum_{c \in C} \mathbb{1}_A^B(c)$ for $C \subseteq B$. Lastly, $\mathcal{Q}$ refers to a query and $\mathcal{H}$ to a historical context. The nature of $\mathcal{Q}$ and $\mathcal{H}$ depend on the considered algorithm, as detailed next.

2.2 Related Works

The ultimate algorithm for link stream-based anomaly detection is one that receives two arbitrary sub-link streams as inputs, constituting a query $\mathcal{Q}$ and a context $\mathcal{H}$, and that responds to the question: can the query be explained from the provided context? See Figure 2 for an illustration. Naturally, no algorithm is able to explore all the possible ways in which an arbitrary query may be explained from an arbitrary context. Thus, proposed algorithms in the literature essentially narrow the search by (i) focusing on queries and contexts that adhere to a specific form; and (ii) establishing a specific criterion, or anomaly definition, that the query must possess in order to be considered as explained by the context. As a result, there is a rich variety of approaches that focus on different combinations of inputs and anomaly definitions. In the following, we briefly review algorithms proposed in the literature, structured according to types of queries they handle. A summary is presented in Table 1.

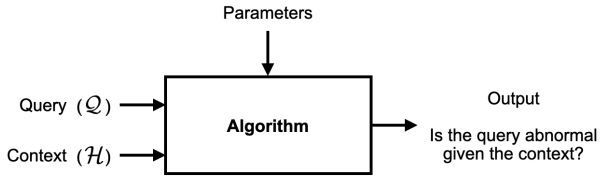

Figure 2: Unified view of anomaly detection. Algorithms aim to assess the abnormality of a *query* in a given *context*.

Edge anomaly. Algorithms in this category respond queries consisting of time-stamped edges. The difference between algorithms lie in the criteria employed consider a query abnormal. Namely, MIDAS [1] uses the historical time series of the query edge to predict its future activity. It labels the query as abnormal if it appears in a period predicted to be of low activity. F-FADE [2] uses historical interactions to compute a node embedding that explains edge frequencies. It considers a query as abnormal if its observed frequency is unlikely according to the embedding. SEDANSPOT [3] uses past interactions to estimate a graph of stable community activity. It labels the query as abnormal if its inclusion to such graph breaks random walks statistics. AER-AD [5] uses past interactions to train a custom-made recurrent neural network for link prediction. It labels a query as abnormal if it is not predicted well by the model. PIKACHU [4] extracts temporal random walks from historical interactions to train a custom-made encoder for link prediction. It labels a query as abnormal if it is not predicted well.

Node anomaly. These algorithms focus on queries consisting of time-stamped nodes, which can be represented by their set of incident edges or by the indicator function of such set. It is worth recalling that the difference between working with a set or with its indicator function is that algorithms accepting the latter handle both active and inactive interactions, while the former only the active ones. DYNANOM [6] uses a short term context to monitor the evolution of PageRank scores. It labels a query as abnormal if its PageRank score drastically changed. BADSN [7] uses the historical interactions of the query node to model the probability of observing it with a given degree. It labels the query as abnormal if the observed degree is unlikely according to the model. DSEDN [15] uses past interactions to train an auto-encoder that embeds nodes in a way that stable structures over time form clusters. It labels a query as abnormal if it is an outlier in the embedding space. GEABS [16] leverages historical interactions to fit a custom-made generative model that jointly accounts for community structure and node popularity. It labels a query as abnormal if its community membership is unstable according to the model. DEGOD [17] uses past interactions to compute the degree distribution of nodes over time. It labels a query as abnormal if it causes the current degree distribution to not match the past.

Graph anomaly. These algorithms accept entire time-stamped slices of a link stream as queries. In particular, ANOMRANK [8] uses a local context to track the evolution of PageRank scores of vertices. It labels a query as abnormal if its first derivatives indicate a drastic change. SPOTLIGHT [9] measures the density of random partitions of historical data to compute a set of reference vectors. It labels a query as abnormal if its corresponding vector is an outlier with respect to the computed references. LAD [10] uses long-term and short-term contexts to predict the spectral shape of the query slice. It labels the query as abnormal if its spectrum is far from the expected values. CADENCE [11] uses past interactions to identify community structures that are stable over time. It labels a query as abnormal if it implies a community reconfiguration. ODGS [18] uses historical data to fit a community-oriented generative model. A query is abnormal if it contains many inter-community edges.

Algorithm	**Query** ($\mathcal{Q}$)	**Context** ($\mathcal{H}$)
MIDAS [1]	$L(t,(u,v))$	$L(0\colon t-1,(u,v))$
F-FADE [2]	$L(t,(u,v))$	$L(0\colon t-1,\mathcal{E})$
SEDANSPOT [3]	$L(t,(u,v))$	$L(0\colon t-1,\mathcal{E})$
DEGOD [17]	$L(t,\mathcal{E}_u)$	$L(0\colon t,\mathcal{E})$
DYNANOM [6]	$\mathbb{1}^{\mathcal{E}_u}_{L(t,\mathcal{E}_u)}$	$L(t-1:t,\mathcal{E})$
DSEDN [15]	$\mathbb{1}^{\mathcal{E}_u}_{L(t,\mathcal{E}_u)}$	$L(0\colon t,\mathcal{E})$
ANOMRANK [8]	$\mathbb{1}^{\mathcal{E}}_{L(t,\mathcal{E})}$	$L(t-2:t,\mathcal{E})$
SPOTLIGHT [9]	$L(t,\mathcal{E})$	$L(0:t-1,\mathcal{E})$
LAD [10]	$\mathbb{1}^{\mathcal{E}}_{L(t,\mathcal{E})}$	$L(t-k:t,\mathcal{E})$

Table 1: Summary of the most representative works for anomaly detection in link streams.

Problem statement. As it can be seen from the list above, proposed algorithms essentially fix a time t and a group of relations ϕ of a specific form: $\phi = (u,v)$, $\phi = \mathcal{E}_u$, or $\phi = \mathcal{E}$ and address $L(t,\phi)$ as queries. While this allows to tackle many abnormal events, the fact that ϕ must possess a specific form (either edges, nodes, or graphs) drastically limits the flexibility and effectiveness of algorithms in many application scenarios, like the one illustrated in Figure 1 where one may be interested in monitoring an arbitrary group of communications. This calls for an algorithm that allows to set ϕ as an arbitrary subgraph. Some algorithms have been proposed to address the case in which ϕ is a community clique [12–14]. Yet, such algorithms automatically search for the communities, thus they prevent users to query them. Moreover, we stress that the community criterion assumes dense link streams slices, which is unrealistic in most real-world interaction data. Thus, the aim of this work is to address these two situations: to propose an algorithm that allows to set ϕ as an arbitrary subgraph and that does it by taking into account the highly uncertain and sparse nature of link streams.

3 PROPOSED ALGORITHM

In this section, we introduce MAD, our proposed algorithm for anomaly detection. In addition to being able to respond to arbitrary time-stamps as queries, we aim that MAD also takes into account that (i) real-world interactions are highly dynamic and uncertain; and (ii) anomalous events can be of different scales. Indeed, in interaction data, it is normal that persons, computers, or bank accounts that register an interaction at a time t do not register another one at time $t+1$, even though it may not be surprising if it occurs, thus making the data highly dynamic and uncertain. Moreover, notice that an intrusion by a hacker may be at the scale of only one or a

few communications, while an infrastructure failure may involve most of them. To account for these situations, develop MAD in two steps. Firstly, in Section 3.1, we develop its scoring function based on novel multi-scale analysis of subgraphs. This multi-scale analysis assumes the existence of a probabilistic model explaining the uncertainty seen in the data, and builds upon it to define a set of multi-scale random variables that allow to spot observations that deviate from the past in a way that cannot be explained from the usual uncertainty. Then, in section 3.2, we focus on the task of estimating such probabilistic model from the historical context. We do it by assuming that normal interactions should be locally stationary, for which design a stationary test that allow us to determine the time interval in which the data is locally stationary and from which the model can be estimated. In sum, MAD is a multi-scale anomaly detection algorithm that accepts $\mathcal{Q} = \mathbb{1}^{\phi}_{L(t,\phi)}$ and $\mathcal{H} = L(t-N\colon t-1, \phi)$ as inputs, where ϕ is an arbitrary subgraph and N is the context duration, and returns an anomaly score denoted by $score(\mathcal{Q})$.

3.1 A multi-scale analysis of sub-graphs

Our goal is to assess if the binary state (active or inactive) of a group of relations $\phi \subseteq \mathcal{E}$ at time t is abnormal. For simplicity, we denote the set of active relations is by $\hat{\phi} := L(t, \phi)$ and its binary state function by $\mathbb{1}^{\phi}_{\hat{\phi}} := \mathbb{1}^{\phi}_{L(t,\phi)}$. As mentioned above, an important property to take into account concerns the uncertainty relative to each interaction involved in ϕ. Thus, we interpret that ϕ is the result of a random processes. In particular, we assume the existence of a function $P : \phi \rightarrow [0, 1]$ so that the state of each $e_i \in \phi$ is seen as the result of running a Bernoulli trial with probability $P(e_i)$: if the trial is successful then $\mathbb{1}^{\phi}_{\hat{\phi}}(e_i) = 1$ and zero otherwise. In complex networks terminology, this is equivalent to interpret that $\mathbb{1}^{\phi}_{\hat{\phi}}$ is generated by an extended Erdös-Rényi model where the probabilities of edges are individually tuned. Then, we consider an observation $\mathbb{1}^{\phi}_{\hat{\phi}}$ as abnormal if it is unlikely to have been generated by such process. Throughout the rest of this subsection, we assume that P is known and that it accurately models normality. In practice, we must estimate P from $\mathcal{H}$. Section 3.2 addresses such problem.

Given our assumed probabilistic model, a simple and straightforward way to spot unlikely realizations at the subgraph level consists in computing the exact probability of observing ϕ, which is given by:

$$Pr(\mathbb{1}^{\phi}_{\hat{\phi}}) = \Pi_{e_i \in \hat{\phi}} P(e_i) \times \Pi_{e_j \in \phi \setminus \hat{\phi}} (1 - P(e_j)). \tag{1}$$

However, while simple, this approach is unsatisfactory for anomaly detection for the following two reasons: (i) small-scale anomalies have little impact in (1); and (ii) expected observations are not ranked as the most probable (hence normal) by (1). Indeed, notice that a few abnormal edges may not drive the value of (1) sufficiently low to be considered a clear anomaly. Moreover, consider a case where $\phi = \{e_1, e_2, e_3\}$ and $P(e_i) = 1/3$ for all e_i. According to (1), the most probable observation for this setting is the empty subgraph, i.e. when $\hat{\phi} = \emptyset$. This is undesirable as the empty subgraph is not the one expected to appear from such process: it is expected one success out of those three Bernoulli trials. Thus, this raises the question of how to find meaningful anomaly scores that allow to spot either small or large scale anomalies.

Interestingly, a potential alternative consists in using random variables that measure properties of the analyzed subgraphs, like their number of active relations $|\hat{\phi}|$. The advantage of using random variables is that we can characterize the the values they take when they are computed on subgraphs generated by the underlying process. Thus, when rare values are observed, the underlying subgraph can be considered anomalous. The challenge with this approach mainly lies in how to define meaningful random variables that measure the necessary properties to spot all targeted anomalies. For instance, $|\hat{\phi}|$ is a useful random variable that allows to readily spot densification or sparsification events by using its expected value as a reference. However, $|\hat{\phi}|$ alone is not enough to detect all anomalies: an event where k likely relations are inactive and k unlikely ones are active would be normal by only using the criterion of $|\hat{\phi}|$. In the following, we address this challenge by introducing a multi-scale analysis of subgraphs. This multi-scale analysis defines a group of $M = |\phi|$ random variables that quantify and compare the activity of the query subgraph at multi resolution scales. We show that it is possible to characterize the distribution of these random variables for normal queries, allowing us to spot the scale and group of relations that make a given observation anomalous.

Let us begin the development of our multi-scale analysis of $\mathbb{1}^{\phi}_{\hat{\phi}}$ by making two assumptions about its domain. Firstly, we assume $M = |\phi|$ to be a power of two. If ϕ lacks relations for this to hold, then we assume that virtual elements e_i of probability $P(e_i) = 0$ are added into ϕ until the assumption holds. We stress that the inclusion of these virtual relations is of pure mathematical convenience and they do not hinder our analysis as those elements are always switched-off in $\mathbb{1}^{\phi}_{\hat{\phi}}$, which is in agreement with their null probability. Secondly, we assume that that the elements of ϕ are indexed in decreasing order of their probability. This is, we assume that $P(e_1) \geq \cdots \geq P(e_i) \geq P(e_{i+1}) \geq \cdots \geq P(e_M)$ for all $e_i \in \phi$. Based on these assumptions, the first step of our multi-scale analysis consists in recursively partitioning ϕ at different resolution scales. To do it, we set an initial set $\mathcal{E}^{(0)}_0 = \phi$ that we split in halves according to the probability of its elements: the top-half likely relations are assigned to a set $\mathcal{E}^{(1)}_0 = \{e_1, \ldots, e_{\frac{M}{2}}\}$ and the bottom-half ones to a set $\mathcal{E}^{(1)}_1 = \{e_{\frac{M}{2}+1}, \ldots, e_M\}$. This recursive partitioning is applied until singletons are obtained $\mathcal{E}^{(\log_2(M))}_i = e_i$. For a visual reference, see the binary tree structure displayed in Figure 3, where the root node is $\mathcal{E}^{(0)}_0$ and the children nodes represent the partitioned sets. Algebraically, the partitioning rule is:

$$\mathcal{E}^{(\ell)}_k = \mathcal{E}^{(\ell+1)}_{2k} \cup \mathcal{E}^{(\ell+1)}_{2k+1}, \tag{2}$$

where

$$\mathcal{E}^{(\ell)}_k = \left\{ e_i \in \phi : \frac{kM}{2^\ell} + 1 \leq i \leq \frac{(k+1)M}{2^\ell} \right\} \tag{3}$$

Thus, the procedure above partitions ϕ into disjoint subgraphs at different resolutions, as indicated by the super-script ℓ. Particularly, 2^ℓ partitions arise at level ℓ and they satisfy the following crucial property: no relation contained in $\mathcal{E}^{(\ell)}_{k+1}$ is more probable than the

relations contained in $\mathcal{E}_k^{(\ell)}$. In the second step of our analysis, we leverage this property by defining random variables that compare the activity of $\mathcal{E}_k^{(\ell)}$ with that of $\mathcal{E}_{k+1}^{(\ell)}$. This is a natural approach to spot anomalies at multiple scales, as we know that, by construction, $\mathcal{E}_k^{(\ell)}$ should be more active than $\mathcal{E}_{k+1}^{(\ell)}$. In precise terms, we define the following set of random variables:

$$s = \frac{1}{\sqrt{M}} \mathbb{1}_{\hat{\phi}}^{\phi}(\phi), \tag{4}$$

and

$$w_k^{(\ell)} = \frac{\sqrt{2^\ell}}{\sqrt{M}} \left[\mathbb{1}_{\hat{\phi}}^{\phi}\left(\mathcal{E}_{2k}^{(\ell+1)}\right) - \mathbb{1}_{\hat{\phi}}^{\phi}\left(\mathcal{E}_{2k+1}^{(\ell+1)}\right) \right]. \tag{5}$$

for all k and ℓ. In total, (4) and (5) define M random variables as there are 2^ℓ sets associated to ℓ and this one runs from 0 to $\log_2(M) - 1$. Therefore, by doing this analysis we do not change the size of the problem: we transition from analyzing the state of M relations in $\mathbb{1}_{\hat{\phi}}^{\phi}$ to M random variables. Moreover, it is worth noticing that the random variables can be computed in $O(M)$ using the binary tree shown in Figure 3: by setting $\mathbb{1}_{\hat{\phi}}^{\phi}$ in the leaves, successive parents compare the activity of relations appearing in their left and right branches, producing the desired random variables.

Concerning the analysis of the random variables, notice that s corresponds to a normalized version of $|\hat{\phi}|$, which, as mentioned previously, is relevant to detect densification or sparsification events. The variables $w_k^{(\ell)}$, on the other hand, allow to spot the anomalies not captured by s. They do it by comparing the activity between $\mathcal{E}_k^{(\ell)}$ and $\mathcal{E}_{k+1}^{(\ell)}$, where the former has relations that are more probable to appear than the latter. This way, a group of likely relations in $\mathcal{E}_k^{(\ell)}$ suddenly disappearing and a group of less likely ones in $\mathcal{E}_{k+1}^{(\ell)}$ suddenly appearing have a strong impact in $w_k^{(\ell)}$. A natural question that may arise is why (5) only compares activity between such specific choices of subsets of relations, given that there are many more ways in which two groups, one with elements more probable than the other, can be chosen and used to define similar random variables. Our next result demonstrates that the family defined by (4) and (5) already contains all the necessary details to discern anomalies, as it does not involve any information loss about $\mathbb{1}_{\hat{\phi}}^{\phi}$.

Proposition 1. *Let $\mathbb{1}_{\hat{\phi}}^{\phi}$ and $\{s, w_k^{(\ell)}\}$ denote a binary state function and its associated set of random variables as defined in (4) and (5), respectively. It holds that:*

$$\mathbb{1}_{\hat{\phi}}^{\phi} = \frac{1}{\sqrt{M}} s \mathbb{1}_{\phi}^{\phi} + \sum_{\ell,k} \frac{\sqrt{2^\ell}}{\sqrt{M}} w_k^{(\ell)} \left[\mathbb{1}_{\mathcal{E}_{2k}^{(\ell+1)}}^{\phi} - \mathbb{1}_{\mathcal{E}_{2k+1}^{(\ell+1)}}^{\phi} \right]. \tag{6}$$

The interesting connection between this multi-scale analysis and the assumed random process is that the first and second theoretical moments of s and $w_k^{(\ell)}$ can be expressed in terms of P. This is a crucial property for anomaly detection as it allows to characterize the ranges of values that s and $w_k^{(\ell)}$ normally take when they are computed on realizations generated by P. Our next result states this connection.

Proposition 2. *Let $\mathbb{E}[\cdot]$ and $\sigma^2[\cdot]$ denote the expectation and variance operators, respectively. If the functions $\mathbb{1}_{\hat{\phi}}^{\phi}$ are drawn from the generative model defined above, it holds that:*

(a) $\mathbb{E}[s] = \frac{1}{\sqrt{M}} P(\phi)$,

(b) $\mathbb{E}[w_k^{(\ell)}] = \frac{\sqrt{2^\ell}}{\sqrt{M}} \left[P\left(\mathcal{E}_{2k}^{(\ell+1)}\right) - P\left(\mathcal{E}_{2k+1}^{(\ell+1)}\right) \right]$,

(c) $\sigma^2[s] = \frac{1}{M} \sum_{e_i \in \phi} P(e_i)[1 - P(e_i)]$,

(d) $\sigma^2[w_k^{(\ell)}] = \frac{2^\ell}{M} \sum_{e_i \in \mathcal{E}_k^{(\ell)}} P(e_i)[1 - P(e_i)]$.

From the Chebyshev inequality, we know that the probability that a random process produces observations of a random variable that are λ standard deviations away from its expectation cannot be larger than $1/\lambda^2$. Hence, our suspicion about an observation should increase quadratically in the number of standard deviations that its random variables values are away from the mean. Based on this property, we can define an anomaly score for each random variable given as the inverse of its Chebyshev bound. If we let x_i denote the i-th random variable, then its anomaly score is given as follows:

$$\text{score}(x_i) = (x_i - \mathbb{E}[x_i])^2 / \sigma^2[x_i]. \tag{7}$$

If we aim to favor interpretability, we can return the M anomaly scores above as the output of the algorithm, allowing an user to identify which parts of the query subgraph are at the origin of an anomaly. For simplicity, we return a single anomaly score summarizing the the total anomaly level of the query. Yet, we stress that, for a more refined study, it is possible to recompute our multi-scale analysis on the queries that our approach identifies as abnormal. We produce anomaly score for the entire query as follows:

$$\text{score}(\mathbb{1}_{\hat{\phi}}^{\phi}) = \frac{(s - \mathbb{E}[s])^2}{\sigma^2[s]} + \sum_{\ell,k} \frac{\left(w_k^{(\ell)} - \mathbb{E}[w_k^{(\ell)}]\right)^2}{\sigma^2[w_k^{(\ell)}]}. \tag{8}$$

Figure 3 provides a comprehensive illustration of our multi-scale approach to anomaly detection. In short, MAD takes as input an history of past interactions $\mathcal{H}$ and query Q. It constructs a model P from $\mathcal{H}$ (see Section 3.2) and uses it to set the ordering of the tree. Then, it uses the tree decompose the query into a set of random variables. MAD also uses the model to estimate the theoretical moments of the multi-scale random variables. Then, it measures how many standard deviations away from the mean the query is in order to set an anomaly score. While MAD sets equal importance to the different random variables involved in the computation of the anomaly score, making both large-scale or small-scale anomalies equally relevant, we stress that it is possible to favor anomalies at any desired scale by giving more weight to the random variables associated to such level.

3.2 Estimation of the Model Probabilities

The question of estimating the model P assumed above from historical data $\mathcal{H}$ is now addressed. This is a key issue as the normality assessment of the query depends on it. Thus, we must compute P so that it captures what we intend by normality. In this work, it is assumed that normal interactions should be locally stationary.

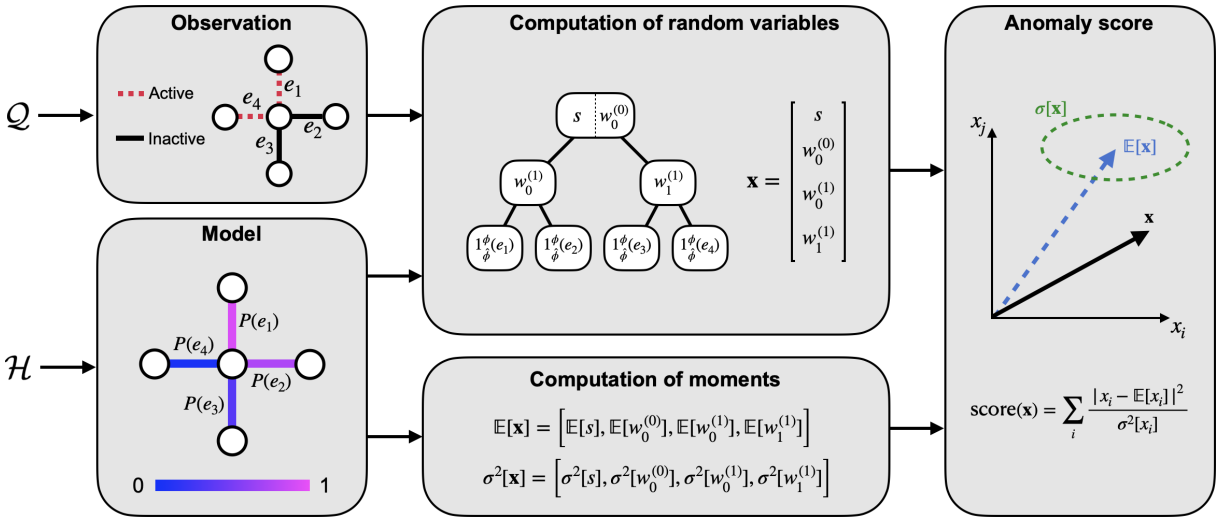

Figure 3: Schematic representation of the proposed multi-scale approach for anomaly detection.

Stationarity means that the underlying random process generating the data remains stable over time. Thus, we assume that there is a single model that produced interactions in the recent past and that, in order to consider the interactions at time t as normal, they should also be generated by such model. Hence, our challenge is to spot the model that produced interactions in the recent past and use it to define normality at time t. Notice that if we identify a window in which the interactions are stationary, then we can straightforwardly estimate P through a simple time averaging. This is because stationarity means that all the observed states of relation e_i over time are samples of the same Bernoulli experiment of probability $P(e_i)$. Hence, $P(e_i)$ can be estimated from its time samples as:

$$P = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}^{\phi}_{L(t-k,\phi)} \tag{9}$$

where K is the length of the stationary window. The challenge of estimating P therefore lies in identifying a sub-window of length K from the context of length N in which all the slices are stationary. Notably, we can leverage our multi-scale analysis to design a simple stationarity test that addresses this challenge.

The idea of our stationarity test is an hypothesis testing one: we hypothesize that the window is stationary and then we try to reject the hypothesis using our multi-scale analysis. Assuming stationary means that all the K slices within the window are realizations of one same P, which can be estimated as in (9). Then, if we compute one of our random variables across all the slices within the window, we must obtain K values that are distributed as predicted by Proposition 2, given that they are realizations of the same P. One can assess is these K values are indeed distributed in such way by comparing their sample moments with the theoretical ones predicted by Proposition 2. The stationarity hypothesis is therefore rejected if these distributions differ from each other. Based on this stationarity test, we can automatically explore the N-length context to identify the sub-window of size K in which the stationarity assumption best holds. This is done by simply growing a window backwardly, starting from $t-1$, for all possible values of K. Then, we run our stationarity test for each window and retain the one in which the distributions best match. The match of distributions is quantified by setting a fitness score given by the sum of squared differences between the sample and theoretical variance for each random variable.[1]

	Synthetic	Hospital	Emails	Traffic
Triplets	1.24M	32.4K	30.7K	382K
Nodes	925	75	1646	1622
Max. time	5K	17.4K	44.5K	7.2K
Empty slices	0	7.9K	32K	0
Activity peak	308	20	76	94

Table 2: Datasets statistics.

4 NUMERICAL EXPERIMENTS

This section evaluates the performance of MAD through experimentation that aim to address the following questions. **Q1. Accuracy**: how accurately can MAD detect anomalous events consisting of likely/unlikely interactions that suddenly disappear/appear compared to state-of-the-art alternatives? **Q2. Flexibility**: can MAD handle equally well queries of varying form? **Q3. Interpretability**: can MAD allow to characterize the signature of abnormal events? The implementation of MAD and code to reproduce the experiments is available in https://anonymous.4open.science/r/MAD_Sub-AE3E

Datasets. One synthetic and three real world datasets are used (Table 2). The synthetic one is composed of a graph sequence with stable community structure but where some edges appear more frequently than others. It is done by fixing a model P and generating a sequence of realizations according to the procedure detailed in Section 3.1. The model P consists of a heterogeneous stochastic

[1] Only the variance is employed as the fact that P is estimated using a sample mean implies that the expectations from Proposition 2 equal the sample ones.

block model: edges within and between communities have different probabilities. See the supplementary for a detailed description of how the model is set. Real datasets are: *Hospital* [19] containing temporal interactions between patients and health-care workers in a hospital ward. Slices represent interactions within a 20-second resolution interval. *Emails* [20]: the directed network of emails in the 2016 Democratic National Committee email leak. Slices represent emails within a one-minute interval. *Traffic* [21]: two-hours of TCP traffic between the Lawrence Berkeley Laboratory and the rest of the world. Slices represent traffic within a one-second interval. In general, these datasets are very dynamic and sparse.

Anomaly injection. There are no known anomalies within the selected datasets. Therefore, the whole datasets are considerd normal and different abnormal events are added: (i) sudden densifications; (ii) sudden sparsifications; and (iii) sudden rewirings. We inject abnormal events according to the type of queries to be assessed (see the supplementary for a full description):

- *Edge anomalies.* A relation (u, v) is randomly selected and attacked at various times. Densification attacks make (u, v) active at times where it is very infrequent, while sparsification attacks suppress (u, v) at times where it appears frequently. For each dataset, we attack 50 relations.
- *Node anomalies.* A randomly selected node is attacked at various times with densifications/sparsifications or rewirings. The former attack injects/suppresses communications emerging from the attacked node, while the latter redirects its communications towards other nodes. We make sure that created edges due to densifications or rewirings always point towards nodes that the attacked node has already communicated with in the past. For each dataset, 10 nodes are attached over time and each attack is bounded to 3 edges.
- *Graph anomalies.* Anomalies here concern densification/-sparsification or rewiring events applied to link stream slices chosen at random. For each dataset, 1% of its active slices are attacked. Attacks are bounded to 5 edges.

Baselines. Six state-of-the-art algorithms form the baselines. Two for edge anomalies: MIDAS [1] and F-FADE [2]. Two for node anomalies: DynAnom [6] and F-FADE-N [2], the variant of F-FADE proposed by their authors to address node anomalies. Two for graph anomalies: AnomRank [8] and LAD [10].

4.1 Accuracy of MAD

This subsection aims to address Q1 and Q2 by assessing the accuracy of MAD and baselines, in AUC score, in the tasks of edge, node and graph anomaly detection. For all experiments we tried numerous hyper-parameters configurations and retained the best ones. See the supplementary for our choices of hyper-parameters.

Edge detection. To assess the accuracy of MAD and edge-anomaly baselines, the algorithms are questionned as follows: for each relation (u, v) that was attacked by our injection method, we ask algorithms to produce an anomaly score for (u, v) at all possible timestamps. Algorithms must then return high scores for timestamps at which (u, v) was attacked. Two versions of each dataset are analyzed, one with injected densifications and one with injected sparsifications.

		MIDAS	F-FADE	MAD
Densification	Synthetic	0.49	0.53	**0.58**
	Hospital	0.50	0.80	**0.82**
	Emails	0.73	**0.98**	0.76
	Traffic	0.52	0.56	**0.76**
Sparsification	Synthetic	-	-	**0.80**
	Hospital	-	-	**0.85**
	Emails	-	-	**0.84**
	Traffic	-	-	**0.89**

Table 3: Edge anomaly detection performance in AUC.

		F-FADE-N*	DynAnom	MAD
Densification & Sparsification	Synthetic	0.52	0.56	**0.88**
	Hospital	0.82	0.51	**0.92**
	Emails	0.82	0.54	**0.84**
	Traffic	**0.77**	0.51	0.74
Rewiring	Synthetic	0.53	0.52	**0.83**
	Hospital	0.57	0.54	**0.99**
	Emails	0.59	0.51	**0.99**
	Traffic	0.53	0.54	**0.99**

Table 4: Node anomaly detection performance in AUC. *F-FADE-N is evaluated only on the subset of scores that it is able to produce.

Results are reported in Table 3. As it can be seen, MAD systematically performs well in the detection of both densification and sparsification events, while MIDAS and F-FADE are inconsistent in the detection of densifications and they cannot handle sparsification anomalies. Such inconsistent behavior may be due to the fact that (i) MIDAS considers global aggregates and hence is agnostic to short intervals of low activity; and (ii) F-FADE requires a stable embedding to produce accurate frequency estimations and is only able to attain it for the datasets that have many empty slices. Moreover, we stress that MIDAS and F-FADE cannot respond to queries consisting of inactive relations, making them unable to spot sparsifications. Notice that MAD solves these two issues by being able to spot the two anomaly types and in a consistent manner. Additionally, MAD does it by just considering a context based on the past activity of the query edge while F-FADE needs to use all the past link stream interactions.

Node detection. MAD and baselines are also evaluated in a node detection setting trough a similar experimental setup: algorithms are asked to determine the abnormality of each node u that was attacked for all possible timestamps. F-FADE-N is not able to produce an answer for queries with no communications in them, hence its accuracy is assessed on the subset of scores that it is able to produce. Two versions of each dataset are analyzed, one with injected densifications/sparsifications and one with rewiring events.

Table 4 clearly shows that MAD performs very well in the detection of both sparsification/densification and rewiring events. In particular, MAD is able to detect the rewiring events in the real datasets with almost perfect accuracy. Such a performance of MAD is due to the fact that rewiring events in those very sparse datasets essentially replace their few likely edges with only unlikely ones, making the attacked queries extremely inconsistent with the recent past. It can be observed that F-FADE-N performs well in detecting densifications/sparsifications, even though it only produces an output at times in which the queried nodes have communications in them. Thus, a massive event that completely shuts-down a node would be missed by F-FADE-N. DynAnom systematically performs very poorly. This poor performance should not come as a surprise: DynAnom bases its anomaly scores on the stability of the PageRank of nodes, which clearly is not a meaningful feature for such sparse and fastly evolving link streams.

Graph detection. The accuracy of MAD and baselines are evaluated in a graph detection setting by feeding the algorithms with the slices of the attacked datasets. Two versions of each dataset are analyzed, one with injected densifications/sparsifications and one with rewiring events.

Results are reported in Table 5. As it can be seen, MAD performs very well in the detection of both events regardless of the dataset, while LAD performs inconsistently and AnomRank very poorly. As in the node case, attacks make likely edges disappear and unlikely ones appear. Therefore, MAD sees those collective events as hard to explain from the context, explaining its good accuracy. LAD is inconsistent as it depends on the stability of eigenvalue distributions, which is not guaranteed when edges are fully replaced between snapshots. AnomRank relies on PageRank, thus it suffers from the same issues of DynAnom.

4.2 Interpretability of MAD

In this subsection, Q3 is addressed by studying how the different attacks influence the individual anomaly scores produced by Equation (7). This study is conducted by (i) taking our model P used to generate synthetic data; (ii) generating a normal graph using the model; (iii) applying different types of attacks on this graph; (iv) performing the multi-scale analysis to each of the resulting graphs; and (v) computing the anomaly scores of each random variable.

Figure 4 displays the distribution of anomaly scores for the different types of attacks. The random variables are ordered so that the left-most ones in the plot are the ones associated to the coarsest scales, i.e. s and $w_0^{(0)}$, and the right-most ones are the ones associated to the finest scales. As it can be seen, the normal graph produces low anomaly scores for most random variables: only few fine-scale ones have large scores, which is due to the inherent uncertainty associated to a graph generated at random. When this graph is subject to a densification attack, it can be seen that a large number of random variables immediately activate producing large scores. Since the likely edges remain present in the graph and the majority of unlikely ones remain inactive in the attacked graph, the large scores mostly appear at fine scales as most of the activity in the graph remains well explained by the model. Yet, notice that s immediately activates pointing the densification. Notice that a sparsification attack suppresses most likely edges and this immediately

		AnomRank	LAD	MAD
Densification & Sparsification	Synthetic	0.49	0.50	**0.76**
	Hospital	0.51	0.80	**0.95**
	Emails	0.54	0.92	**0.98**
	Traffic	0.43	0.46	**0.77**
Rewiring	Synthetic	0.59	0.52	**0.63**
	Hospital	0.58	0.77	**0.94**
	Emails	0.53	**0.87**	**0.87**
	Traffic	0.44	0.55	**0.85**

Table 5: Graph anomaly detection performance in AUC.

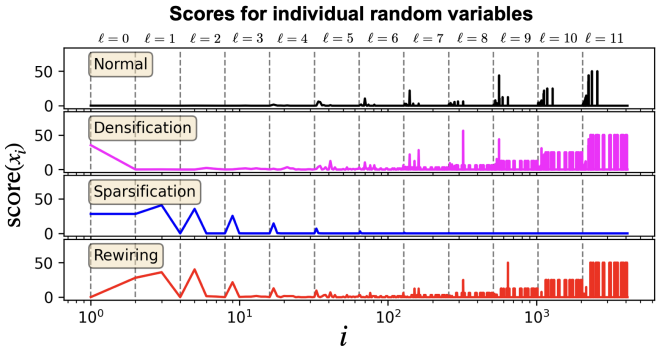

Figure 4: Distribution of anomaly scores across random variables. Different attacks produce different signatures.

triggers the scores associated to coarse resolutions, particularly s and $w_0^{(0)}$. s because the activity of this graph does not match the expected one, and $w_0^{(0)}$ because the attack mostly affect the left-side of the tree, making the scores mount at the top levels. Since a rewiring attack is essentially a combination of a sparsification and a densification, one can remark that the anomaly scores of this event combine the signature of the previous two. Thus, in sum, the different attacks produce different signatures in our anomaly scores, paving the way to study the signature of more complex and real events as further research.

5 CONCLUSION

In this work we introduced MAD, a multi-scale anomaly detection algorithm for link streams that allows to evaluate if any arbitrary time-stamped subgraph is abnormal. Through a numerical evaluation, we demonstrated that MAD performs significantly better than state-of-the-art alternatives, even when the data at hand is very uncertain and sparse, in the tasks of detecting edges, nodes or graphs that were subject to densification, sparsification and redirection attacks. This flexibility and good accuracy of MAD stems from its scoring mechanism, which builds on a novel probabilistic and multi-scale analysis of sub-graphs that allows to decompose them into a set of random variables that capture anomalies at various resolution scales. This makes MAD not only accurate but also inherently interpretable and theoretically sound. The next step concerns

the combination of MAD with an anomaly explanation mechanism to assist final users in the analysis of the found anomalies.

REFERENCES

[1] S. Bhatia, B. Hooi, M. Yoon, K. Shin, and C. Faloutsos, "Midas: Microcluster-based detector of anomalies in edge streams," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 3242–3249, 2020.

[2] Y.-Y. Chang, P. Li, R. Sosic, M. Afifi, M. Schweighauser, and J. Leskovec, "F-fade: Frequency factorization for anomaly detection in edge streams," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 589–597, 2021.

[3] D. Eswaran and C. Faloutsos, "Sedanspot: Detecting anomalies in edge streams," in *2018 IEEE International conference on data mining (ICDM)*, pp. 953–958, IEEE, 2018.

[4] R. Paudel and H. H. Huang, "Pikachu: Temporal walk based dynamic graph embedding for network anomaly detection," in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–7, IEEE, 2022.

[5] L. Fang, K. Feng, J. Gui, S. Feng, and A. Hu, "Anonymous edge representation for inductive anomaly detection in dynamic bipartite graph," *Proceedings of the VLDB Endowment*, vol. 16, no. 5, pp. 1154–1167, 2023.

[6] X. Guo, B. Zhou, and S. Skiena, "Subset node anomaly tracking over large dynamic graphs," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 475–485, 2022.

[7] N. A. Heard, D. J. Weston, K. Platanioti, and D. J. Hand, "Bayesian anomaly detection methods for social networks," 2010.

[8] M. Yoon, B. Hooi, K. Shin, and C. Faloutsos, "Fast and accurate anomaly detection in dynamic graphs with a two-pronged approach," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 647–657, 2019.

[9] D. Eswaran, C. Faloutsos, S. Guha, and N. Mishra, "Spotlight: Detecting anomalies in streaming graphs," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1378–1386, 2018.

[10] S. Huang, Y. Hitti, G. Rabusseau, and R. Rabbany, "Laplacian change point detection for dynamic graphs," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 349–358, 2020.

[11] M. McNeil, C. Mattsson, F. W. Takes, and P. Bogdanov, "Cadence: Community-aware detection of dynamic network states," in *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 1–9, SIAM, 2023.

[12] S. Fernandes, H. Fanaee-T, J. Gama, L. Tišljarić, and T. Šmuc, "Wintended: Windowed tensor decomposition for densification event detection in time-evolving networks," *Machine Learning*, vol. 112, no. 2, pp. 459–481, 2023.

[13] Y. Jiang and G. Liu, "Two-stage anomaly detection algorithm via dynamic community evolution in temporal graph," *Applied Intelligence*, vol. 52, no. 11, pp. 12222–12240, 2022.

[14] Z. Tasnádi and N. Gaskó, "A new type of anomaly detection problem in dynamic graphs: An ant colony optimization approach," in *International Conference on Bioinspired Optimization Methods and Their Applications*, pp. 46–53, Springer, 2022.

[15] M. Bansal and D. Sharma, "Density-based structural embedding for anomaly detection in dynamic networks," *Neurocomputing*, vol. 500, pp. 724–740, 2022.

[16] P. Jiao, T. Li, Y. Xie, Y. Wang, W. Wang, D. He, and H. Wu, "Generative evolutionary anomaly detection in dynamic networks," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[17] A. Wilmet, T. Viard, M. Latapy, and R. Lamarche-Perrin, "Degree-based outliers detection within ip traffic modelled as a link stream," in *2018 Network Traffic Measurement and Analysis Conference (TMA)*, pp. 1–8, IEEE, 2018.

[18] C. C. Aggarwal, Y. Zhao, and S. Y. Philip, "Outlier detection in graph streams," in *2011 IEEE 27th international conference on data engineering*, pp. 399–409, IEEE, 2011.

[19] P. Vanhems, A. Barrat, C. Cattuto, J.-F. Pinton, N. Khanafer, C. Régis, B.-a. Kim, B. Comte, and N. Voirin, "Estimating potential infection transmission routes in hospital wards using wearable proximity sensors," *PloS one*, vol. 8, no. 9, p. e73970, 2013.

[20] J. Kunegis, "KONECT – The Koblenz Network Collection," in *Proc. Int. Conf. on World Wide Web Companion*, pp. 1343–1350, 2013.

[21] V. Paxson and S. Floyd, "Wide area traffic: the failure of poisson modeling," *IEEE/ACM Transactions on networking*, vol. 3, no. 3, pp. 226–244, 1995.