# Community-Augmented Local-Link Intensity: a score for anomaly detection in graphs

Theodor-Adrian Badea
*Department of Automatic Control and Computers*
*Univeristy Politehnica of Bucharest*
Bucharest, Romania
theodor.badea@upb.ro

Bogdan Dumitrescu
*Department of Automatic Control and Computers*
*Univeristy Politehnica of Bucharest*
Bucharest, Romania
bogdan.dumitrescu@upb.ro

***Abstract*—Graphs can model various systems, activities, or interactions. As a consequence, the ability to detect anomalies in graph-structured data is needed in many scenarios. This paper introduces *Community-Augmented Local-Link Intensity* (CALLI), an algorithm for quantifying the abnormality extent of nodes in a weighted digraph. Experiments are performed on both synthetically generated and real graphs modeling financial transactions. Throughout the tests, CALLI is directly used as the sole decision-making anomaly score, but also as a feature in a conventional outlier detection approach.**

***Index Terms*—anomaly detection, outlier detection, graphs, graph-structured data**

## I. INTRODUCTION

One of the earliest definitions of anomalous observations dates back to 1969, in [1] an outlier being characterized as an observation that *"appears to deviate markedly from other members of the sample in which it occurs"*. Generally, the terms "anomaly" and "outlier" are used interchangeably, as a consequence of having similar meanings and of being hard to formally define individually. There is a subtle and contextual line between these terms. In most cases, an observation is called an anomaly when it is confirmed as illegitimate, whereas an outlier can be produced by the same legitimate process as the other observations, but it diverges from the normal distribution.

Naturally, graphs are an appropriate means of representing both structural information and relationships between instances. Suitable applications arise in a multitude of scenarios where observations exhibit considerable relationships: packets flowing on the network, social platforms, search engines, biochemistry, etc. Perhaps the most widespread and intuitive use case for graph-structured data is in finance. Fraud is a major challenge in this area, therefore overcoming it plays an important role. Anomalous behavior, e.g. money laundering or illicit payments, might be identified as unusual patterns in financial transactions modeled under the shape of graphs. In such networks, an account can be represented as a node (vertex), and a weighted and directed edge between two vertices models a transaction between these accounts.

This work was supported by a grant of the Ministry of Research, Innovation and Digitization, CNCS - UEFISCDI, project number PN-III-P4-PCE-2021-0154, within PNCDI III.

### A. Content and contributions

We feel appropriate to expand the context of anomaly detection for graph-structured data provided so far, therefore we further include in the next subsection a view of the problem we address in this paper. Then, we shall proceed in Section II with a succinct depiction of related work from domain literature. Our main contribution is presented in Section III. It consists of introducing a new algorithm for measuring the abnormality of nodes in graphs: *Community-Augmented Local-Link Intensity* (CALLI). The fourth section is dedicated to experimental results. We test the proposed algorithm both in a direct manner, but also in a conventional approach. For the first round of experiments, we rely on CALLI as a sole anomaly score and test its capability of straightforwardly identifying anomalous vertices. Then, we include CALLI in the so-called conventional anomaly detection workflow in graphs: used as a feature, in combination with other several promising graph features. We follow the approach from [2], hence we choose Isolation Forest [3] as the anomaly detection algorithm into which we feed the constructed set of features.

### B. Problem statement

The input around which each of our experiments is focused is a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, constructed from a list of transactions, either synthetically generated or real. Vertices constituting the set $\mathcal{V}$ represent bank accounts and transactions between each two accounts form the set of directed edges $\mathcal{E}$; the amounts transferred during the transactions are the weights of edges. The structure of the transactions lists used for conducting the experiments in this paper is broadly described in [2]. The resulting graphs are simple, i.e. there can be only one edge connecting two vertices (direction considered) and there is no edge having the same vertex as both origin and destination (self-loop). Our purpose is to find tools that classify the nodes of the graph into two categories: normal and abnormal. The notion of anomaly in this context can be better understood from the data description.

We use the data described in [2]. A first graph is constructed from actual bank accounts and transactions between them provided by Libra Internet Bank. A preprocessing step was carried out. After it, the graph contains $385100$ nodes and $597165$ edges, with an average in/out degree of $1.55$. A total

TABLE I
GRAPHS CHARACTERISTICS

| graph | nodes | edges | alerts | $W_a$ | reports | $W_r$ |
|---|---|---|---|---|---|---|
| Libra | 385100 | 597165 | 600 | 1034 | 15 | 22 |
| G7 | 50000 | 377789 | 248 | 1910 | - | - |
| G12 | 50000 | 629242 | 249 | 1910 | - | - |
| G17 | 50000 | 879091 | 250 | 1910 | - | - |
| G22 | 50000 | 1126992 | 248 | 1910 | - | - |

of 517 transactions were labeled as potentially fraudulent by specialized bank personnel and called alerts. Moreover, 11 of these transactions are assigned with reports, meaning that they have been reported to state authorities for further investigations. There are 600 vertices involved in suspicious transactions, with a cumulated number of alerts $W_a = 1034$. The 15 vertices that are part of reported transactions, amount to $W_r = 22$ reports.

There are also synthetically generated graphs, also thoroughly presented in [2], that have a general ”normal” structure in which anomalies suitable for fraud (particularly money laundering) detection have been injected: cliques, stars, and rings. There are four such graphs, each with 50000 nodes and different number of edges, thus the average degrees range from $\approx 7$ to $\approx 22$. Anomalous transactions sum up to a total of 1910, with 248-250 distinct vertices involved in such transactions. To keep the terminology of the Libra graph, we name alerts the anomalies. Table I-B presents the main characteristics of the Libra and synthetic graphs.

The goal is to identify anomalies (nodes labeled as suspicious) in the graphs presented above. In our experiments, we tackle this task in an unsupervised manner: anomaly detection is performed and then we compare the results with the ground-truth labels (assigned by bank personnel in the real graph and injected anomalies in the synthetically generated graphs).

## II. RELATED WORK

OddBall [4] is a popular technique for tackling the graph anomaly detection problem. It leverages the concept of *egonet*, which is defined as the single step neighborhood around a node, including the node itself, its direct neighbors, and all the edges among this group of vertices. The approach is to extract egonet-based features and find patterns followed by most of the egonets in the graph, thus spotting anomalous egonets and, implicitly, nodes. Examples of features include total weight of the egonet edges, number of triangles, etc. The patterns to be analyzed are formed as power laws and then outlierness scores are defined based on deviations, thus also providing a way of sorting nodes according to their abnormality.

In [2], the reduced egonet (egored) is defined as the egonet from which the nodes that have only connections with the center are removed. Using egored features is shown to enhance the detection ability with respect to approaches based on egonet only, at least in graphs resulting from bank transactions. Also, the bibliography of [2] contains many useful entries on this topic.

A method for detecting anomalous graph substructures (subgraphs) is described in [5]. The score is built by iteratively finding the best substructure which yields the largest compression when every occurrence is replaced by a super-node. Anomalies are identified according to the compression: the better the compression, the lower the anomaly score.

The problem of anomaly detection in graphs (particularly signals indexed by graphs) is presented in [6] from the perspective of graph signal processing and graph spectral theory. The approach incorporates the community structure into an extended adjacency matrix, which can be interpreted as a change of the original graph through inclusion of new edges. Then, vertices whose signal values (which, for example, can represent account balances) are different than expected for the community they belong are flagged as potential anomalies.

A technique which further exploits the community structure is shown in [7]. It addresses the problem of finding outliers in bipartite graphs in two steps. First, communities are formed and relevance scores are assigned to the nodes. The second step of the approach consists in computing normality scores based on the previously assigned relevance scores.

AutoPart [8] is another community-oriented method for finding outliers in graphs. Nodes are clustered into communities through a process of organizing the adjacency matrix into blocks according to densities, then nodes with multiple cross-community connections are considered anomalous.

CADA [9] is an algorithm that assigns abnormality scores to nodes, considering the extent to which a node belongs to a multitude of communities without strongly belonging to one of them.

A statistical approach is proposed in [10], a multitude of scores being built through means of constructing a ”normal” and a $p$-*value* measurement. We mention three of these node scores, resulted from the geometric average of weights: GAW, GAW10, GAW20.

## III. PROPOSED APPROACH

This section is committed to presenting the main contribution of this paper. We propose *Community-Augmented Local Link Intensity* (CALLI): an algorithm for constructing anomaly scores for vertices, in an attempt to quantify the outlierness by analyzing the intensity of the connections between a node and its neighbors and the extent to which a node is intensely connected to adjacent communities compared to its own. Mainly, our method falls in the category of direct approaches. We are concerned with measuring the abnormality in a straightforward manner and generate features which may be directly assimilated as scores. However, throughout the experiments, we shall also regard the output of the proposed algorithm as an engineered graph feature.

The development of the algorithm is inspired by work described in [11] and [9]. We also acknowledge the influence of the statistical scores from [10] mentioned above, in the sense that it hinted towards working with the geometric average of weights as a way of measuring the interactions between vertices.

## A. Prerequisite: community detection

A prerequisite of the CALLI approach is unfolding the community structure of the graph. Communities are sets of nodes with high inwardly connection densities, whereas vertices belonging to different communities being sparsely linked. The procedure partitions the graph into densely interconnected sub-structures and uncovers sets of nodes with considerable relations. In the development of the proposed algorithm, we took into consideration exclusively the computation time required for performing this essential pre-processing stage. Therefore we used the *Louvain* community detection method, which has the advantage of having limitations only in terms of storage memory, rather than computation time [12]. However, we feel that any other technique for finding the community structure of the graph can be adopted.

## B. Algorithm description

Let $x$ be a node of the weighted digraph $\mathcal{G} = (\mathcal{V},\ \mathcal{E})$ and let $\mathcal{E}\left(x,\ y\right)$ denote the weight of the directed edge with origin $x$ and destination $y$. We define:

$$\mathcal{P}\left(x\right) = \{y \in \mathcal{V} \mid (y,\ x) \in \mathcal{E}\} \tag{1}$$

and

$$\mathcal{S}\left(x\right) = \{y \in \mathcal{V} \mid (x,\ y) \in \mathcal{E}\} \tag{2}$$

the sets of predecessors and successors of the node $x$. In plain terms, $\mathcal{P}\left(x\right)$ is the set of nodes which have outgoing edges towards $x$, whereas $\mathcal{S}\left(x\right)$ is the set of nodes with incoming edges from $x$.

Then, we proceed to computing the geometric average of weights corresponding to the edges between vertices in $\mathcal{P}\left(x\right)$ and $x$, and between $x$ and vertices in $\mathcal{S}\left(x\right)$, designated by $\mathcal{P}_{gaw}\left(x\right)$ and $\mathcal{S}_{gaw}\left(x\right)$, respectively:

$$\mathcal{P}_{gaw}\left(x\right) = \left(\prod_{y \in \mathcal{P}(x)} \mathcal{E}\left(y,\ x\right)\right)^{\frac{1}{|\mathcal{P}(x)|}}, \tag{3}$$

$$\mathcal{S}_{gaw}\left(x\right) = \left(\prod_{y \in \mathcal{S}(x)} \mathcal{E}\left(x,\ y\right)\right)^{\frac{1}{|\mathcal{S}(x)|}}, \tag{4}$$

where $|\cdot|$ denotes the cardinality.

Based on $\mathcal{P}_{gaw}\left(x\right)$ and $\mathcal{S}_{gaw}\left(x\right)$, we further introduce a fundamental measure of *intensity* of the link between the node $x$ and an adjacent vertex $y$. If $y \in \mathcal{P}\left(x\right)$, we define

$$i_y\left(x\right) = \max\left(\mathcal{P}_{gaw}\left(x\right),\ \mathcal{S}_{gaw}\left(y\right)\right). \tag{5}$$

If $y \in \mathcal{S}\left(x\right)$:

$$i_y\left(x\right) = \max\left(\mathcal{P}_{gaw}\left(y\right),\ \mathcal{S}_{gaw}\left(x\right)\right). \tag{6}$$

The measure of intensity for $x$ w.r.t. a neighboring node $y$ attempts to quantify the strength of the connection between the two nodes in relation to other incoming or outgoing edges. Considering the case of $y \in \mathcal{P}\left(x\right)$, the intensity $i_y\left(x\right)$ is the most powerful link by viewing either $y$ as a predecessor of $x$ or $x$ as a successor of $y$. Contrarily, for $y \in \mathcal{S}\left(x\right)$, the intensity $i_y\left(x\right)$ determined from the perspective of either regarding $x$ as being a predecessor of $y$ or the perspective of $y$ being a successor of $x$. In other words, we quantify how strong is the connection between $x$ and $y$ based on the most dominant of the two nodes in relation with the edge orientation between them and in relation to the other edges adjacent to both of these vertices.

The next step of the proposed algorithm elevates the intensities $i_y\left(x\right)$ to community level. Let us consider the set $\mathcal{C}\left(x\right) = \{c_y \mid \exists (y,\ x) \in \mathcal{E}\} \cup \{c_y \mid \exists (x,\ y) \in \mathcal{E}\}$, where $c_y$ is the identifier of the community to which the node $y$ belongs. We can straightforwardly regard $\mathcal{C}\left(x\right)$ as the list of communities to which neighboring vertices of $x$ belong. For each neighboring community identified by $c \in \mathcal{C}\left(x\right)$ of $x$, we compute an augmented community connection intensity $com\_i_c\left(x\right)$ formulated as follows:

$$com\_i_c\left(x\right) = \sum_{\substack{y \,\in\, \mathcal{P}(x) \,\cup\, \mathcal{S}(x) \\ y \,\in\, c}} i_y\left(x\right). \tag{7}$$

By means of the community intensities, we augment the intensities $i_y\left(x\right)$ to the community structure of the graph and quantify the strength of the relation between $x$ and its adjacent communities, including its own.

Having computed the augmented intensities of $x$ w.r.t. the adjacent communities, we introduce the following measure of density:

$$com\_d\left(x\right) = \frac{\sum_{c \,\in\, \mathcal{C}(x)} com\_i_c\left(x\right)}{com\_i_{c_x}} \tag{8}$$

The density $com\_d\left(x\right)$ characterizes the neighborhood of $x$ embedding at the same time both the strengths of the connections between $x$ and its predecessors and successors, and the strengths of the connections between $x$ and the community structure of the graph. One can notice that $com\_d\left(x\right)$ is not determined by the number of edges directly, but by intensities of the links. It quantifies the extent to which a node is strongly connected to adjacent communities relative to its own. The density value is larger for a vertex "more intensely" linked to the neighboring communities than to the community it belongs to.

Lastly, the CALLI score of a vertex $x$ is defined as the ratio between the total density across all the neighboring vertices of $x$ and the density of $x$:

$$CALLI\left(x\right) = \frac{\sum_{y \,\in\, \mathcal{P}(x) \,\cup\, \mathcal{S}(x)} com\_d\left(y\right)}{com\_d\left(x\right)}. \tag{9}$$

## IV. Experiments

Throughout this section, we present the experimental results. As mentioned, we test the proposed algorithm both as an anomaly score to directly identify suspicious nodes and as a feature fed into the Isolation Forest [3] algorithm (alone and in combination with other several features). In Isolation Forest, we use 200 trees for fitting the model, the other parameters being the default ones, as shown in the PyOD implementation [13]. Graphomaly library [14] has been used for extracting additional node features. The following are used for the synthetic graphs: in/out degree, in/out amount, egonet edge density, egored in/out relative degree, egored in/out relative amount, and egored edge density. For the Libra bank graph, we add some more features to the previous list: average in/out amount and egored average in/out amount. All of them are thoroughly described in [2]. The method based on these egonet and reduced egonet features, fed into the Isolation Forest anomaly detector, is called EGO. In addition to these features, we likewise test CALLI in comparison with three metrics (anomaly scores) from [10]: GAW, GAW10, GAW20; these scores are added and the resulting method is called here GAW.

We also combine CALLI with these methods. In conjunction with EGO, we append the CALLI score to the EGO features and run Isolation Forest for anomaly detection. The resulting method is called EGO & CALLI. In conjunction with GAW, we first scale the CALLI score then add it to the GAW scores. This operation is necessary for making the addition relevant. The scaling procedure is performed in accordance with the following formula:

$$s = t_{min} + \frac{n - r_{min}}{r_{max} - r_{min}} \left( t_{max} - t_{min} \right), \qquad (10)$$

where $s$ is the result (scaled CALLI score), $n$ is the initial value (original CALLI score), $r_{min}$ and $r_{max}$ are the minimum and maximum of the reference scaling interval (minimum and maximum of the CALLI scores), respectively, and $t_{min}$ and $t_{max}$ are the minimum and maximum of the target interval (minimum and maximum of the cumulated GAW scores), respectively. We call GAW & CALLI this combined score.

Some programs that are relevant for CALLI and the testing environment can be found at http://asydil.upb.ro/software/.

Tables II–VII present the results obtained with the above methods. TPR $0.1\%$ is the true positive rate for the first $0.1\%$ of the nodes with highest anomaly scores. Here, the TPR is computed taking into account the weights of the nodes; for example, in the Libra graph, the total number of alerts is 1034; if a node appears in three transactions marked as alerts, then its (relative) weight is $3/1034$. TPR $0.2\%$, $0.5\%$ and $1\%$ are defined similarly. We are interested in these rates because in financial applications only a small number of transactions or accounts can be checked thoroughly by a human expert, hence we desire good accuracy of the top positives given by the machine learning algorithm. AUC $1\%$ is the area under curve in the first $1\%$ of the TPR curve; this is an integral measure, characterizing the whole interval. AN $0.1\%$ is the number of anomalous nodes in the top $0.1\%$ highest anomaly scores. This time the nodes are counted individually; they are no longer weighted. So, for the Libra graph, there are 600 distinct nodes involved in alerts. AN $0.2\%$, $0.5\%$ and $1\%$ are defined similarly. An identical reasoning applies for Table VII (results for reports in the Libra graph).

We notice that TPR for CALLI on the synthetic datasets tends to increase with the density of the graph; this is rather surprising, since cliques, rings and stars are more difficult to find in a denser graph; the other methods show an opposite behavior. CALLI clearly outperforms GAW, and for the most dense two of the syntethic graphs, $\mathcal{G}_{17}$ and $\mathcal{G}_{22}$, the TPR $0.1\%$ and $0.2\%$ approach the EGO results. In some cases, the number of nodes correctly identified as anomalies is the same, but the TPR is different. For example, in Table V, CALLI, EGO, and EGO & CALLI display the same AN $0.1\%$; this is justified by the fact the uncovered anomalous nodes are different, and they may have different weights; the methods with larger TPR find nodes with larger weights, when AN is the same. When testing CALLI in conjunction with GAW on synthetic graphs, in each experiment CALLI improves the results compared with GAW; however, GAW does not seem to be a suitable method for these datasets. For this reason, CALLI alone yields better results than GAW & CALLI. When combined with EGO and fed into the Isolation Forest algorithm, our proposed score has mixed results. In Table II, it can be seen that TPR $0.1\%$ and $0.2\%$ are slightly higher for EGO alone, but AN $0.1\%$ and AN $0.2\%$ are the same, hinting that other anomalous nodes are detected. The situation changes for TPR $0.5\%$ and $1\%$, as well as for AN $0.5\%$ and $1\%$, where EGO & CALLI is better than EGO. Similar behavior can be observed in Tables III–V. The combination of CALLI and EGO proves most beneficial in Table IV, managing to obtain the best results among all the tested techniques.

Table VI shows results for CALLI closer to Tables II and III, obtained for the least dense synthetic graphs. The Libra graph is a sparse graph; recall that it has a density of only $1.55$. We can see an improvement for GAW, outperforming CALLI for TPR $0.2\%$, $0.5\%$, $1\%$ and for all AN thresholds. Remarkably, TPR $0.1\%$ is larger for CALLI, but AN $0.1\%$ shows a difference of 16 anomalous nodes in favour of GAW, implying that CALLI detects anomalies with larger weights. When combining CALLI and GAW, each TPR is better than individually, most notably for $0.1\%$ and $0.2\%$. The situation is almost similar for AN, only $1\%$ displaying the same number of anomalous nodes detected. EGO has the best TPR for $0.2\%$, $0.5\%$; only TPR $0.1\%$ is slightly improved by combining it with CALLI. For reports, GAW, CALLI, and their combination produces identical TPR $0.1\%$ and AN $0.1\%$. The same for EGO and EGO & CALLI. GAW & CALLI has the largest TPR $0.2\%$ and AN $0.2\%$.

Note that in almost all cases, the methods find the nodes with higher weights first. For example, in Table VI, CALLI finds 153 anomalous nodes in the first $1\%$ results, which corresponds to a rate of $153/600 = 0.255$; the corresponding weighted TPR is $0.315$, which means that the found 153 nodes have higher weights than the average.

## Conclusion and further work

Our proposed algorithm proved capable of detecting anomalous nodes both in synthetic and real graphs, with an apparent affinity for dense graphs. Furthermore, used as a feature and fed into Isolation Forest or directly as a score in conjunction with other scores, CALLI showed able to at least offer a new perspective and uncover different anomalous vertices if

TABLE II
SYNTHETIC GRAPH $\mathcal{G}_7$ RESULTS

| Method | TPR 0.1% | TPR 0.2% | TPR 0.5% | TPR 1% | AUC 1% | AN 0.1% | AN 0.2% | AN 0.5% | AN 1% |
|---|---|---|---|---|---|---|---|---|---|
| CALLI | 0.0921 | 0.1309 | 0.2314 | 0.3335 | 0.2123 | 14 | 22 | 38 | 58 |
| GAW | 0.0094 | 0.0094 | 0.0361 | 0.0387 | 0.0267 | 2 | 2 | 6 | 7 |
| EGO | **0.3346** | **0.6220** | 0.9759 | 0.9911 | **0.8214** | **50** | **100** | 211 | 237 |
| EGO & CALLI | 0.3262 | 0.6204 | **0.9775** | **0.9932** | 0.8213 | **50** | **100** | **214** | **238** |
| GAW & CALLI | 0.0471 | 0.0759 | 0.1236 | 0.2047 | 0.1237 | 8 | 12 | 21 | 34 |

TABLE III
SYNTHETIC GRAPH $\mathcal{G}_{12}$ RESULTS

| Method | TPR 0.1% | TPR 0.2% | TPR 0.5% | TPR 1% | AUC 1% | AN 0.1% | AN 0.2% | AN 0.5% | AN 1% |
|---|---|---|---|---|---|---|---|---|---|
| CALLI | 0.0853 | 0.1586 | 0.3209 | 0.5105 | 0.3023 | 13 | 26 | 55 | 90 |
| GAW | 0.0000 | 0.0073 | 0.0251 | 0.0487 | 0.0259 | 0 | 1 | 5 | 9 |
| EGO | **0.3298** | 0.5995 | **0.9597** | 0.9749 | **0.8056** | **50** | **100** | **198** | 216 |
| EGO & CALLI | 0.3230 | **0.6005** | 0.9550 | **0.9759** | 0.8036 | **50** | **100** | 196 | **218** |
| GAW & CALLI | 0.0497 | 0.1063 | 0.1901 | 0.3026 | 0.1785 | 8 | 17 | 31 | 52 |

TABLE IV
SYNTHETIC GRAPH $\mathcal{G}_{17}$ RESULTS

| Method | TPR 0.1% | TPR 0.2% | TPR 0.5% | TPR 1% | AUC 1% | AN 0.1% | AN 0.2% | AN 0.5% | AN 1% |
|---|---|---|---|---|---|---|---|---|---|
| CALLI | 0.2686 | 0.5047 | 0.6984 | 0.7267 | 0.6012 | 45 | 87 | 129 | 141 |
| GAW | 0.0058 | 0.0293 | 0.0445 | 0.0581 | 0.0412 | 1 | 5 | 9 | 12 |
| EGO | 0.3115 | 0.5984 | 0.9414 | **0.9670** | 0.7927 | **50** | **100** | 188 | 206 |
| EGO & CALLI | **0.3147** | **0.5995** | **0.9435** | **0.9670** | **0.7950** | **50** | **100** | **190** | **207** |
| GAW & CALLI | 0.1497 | 0.2288 | 0.3236 | 0.3885 | 0.2858 | 25 | 39 | 56 | 71 |

TABLE V
SYNTHETIC GRAPH $\mathcal{G}_{22}$ RESULTS

| Method | TPR 0.1% | TPR 0.2% | TPR 0.5% | TPR 1% | AUC 1% | AN 0.1% | AN 0.2% | AN 0.5% | AN 1% |
|---|---|---|---|---|---|---|---|---|---|
| CALLI | 0.3136 | 0.5639 | 0.6597 | 0.6822 | 0.5882 | **50** | 97 | 123 | 131 |
| GAW | 0.0487 | 0.0644 | 0.1466 | 0.1707 | 0.1215 | 8 | 11 | 25 | 31 |
| EGO | **0.3319** | **0.6047** | 0.9251 | 0.9565 | 0.7888 | **50** | **100** | 180 | 195 |
| EGO & CALLI | 0.3288 | 0.6000 | **0.9351** | **0.9613** | **0.7896** | **50** | **100** | **184** | **198** |
| GAW & CALLI | 0.2241 | 0.3288 | 0.4581 | 0.5670 | 0.4238 | 37 | 55 | 76 | 98 |

TABLE VI
LIBRA GRAPH RESULTS FOR ALERTS

| Method | TPR 0.1% | TPR 0.2% | TPR 0.5% | TPR 1% | AUC 1% | AN 0.1% | AN 0.2% | AN 0.5% | AN 1% |
|---|---|---|---|---|---|---|---|---|---|
| CALLI | 0.1025 | 0.1509 | 0.2418 | 0.3153 | 0.2199 | 38 | 63 | 112 | 153 |
| GAW | 0.0977 | 0.1857 | 0.5019 | 0.7515 | 0.4500 | 54 | 102 | 247 | **414** |
| EGO | 0.3791 | **0.5000** | **0.6518** | 0.7476 | **0.5985** | 151 | **221** | **312** | 379 |
| EGO & CALLI | **0.3946** | 0.4894 | 0.6441 | 0.7389 | 0.5900 | **152** | 215 | 307 | 374 |
| GAW & CALLI | 0.2379 | 0.4584 | 0.5106 | **0.7524** | 0.5244 | 101 | 202 | 250 | **414** |

TABLE VII
LIBRA GRAPH RESULTS FOR REPORTS

| Method | TPR 0.1% | TPR 0.2% | TPR 0.5% | TPR 1% | AUC 1% | AN 0.1% | AN 0.2% | AN 0.5% | AN 1% |
|---|---|---|---|---|---|---|---|---|---|
| CALLI | 0.1364 | 0.1364 | 0.2273 | 0.3636 | 0.2041 | 3 | 3 | 4 | 6 |
| GAW | 0.1364 | 0.2727 | 0.5000 | 0.5909 | 0.4191 | 3 | 4 | 7 | 9 |
| EGO | **0.2273** | 0.4091 | **0.6818** | **0.9091** | **0.6060** | **4** | 6 | **9** | **13** |
| EGO & CALLI | **0.2273** | 0.4091 | **0.6818** | 0.7727 | 0.5877 | **4** | 6 | **9** | 11 |
| GAW & CALLI | 0.1364 | **0.5000** | 0.5000 | 0.5909 | 0.4672 | 3 | **7** | 7 | 9 |

not exhibiting an improvement in terms of TPR. Also, we stress that CALLI relies on a single score, whereas GAW is constructed by a summation of three scores and EGO approach used in our experiments assembles $10$ and $14$ features for synthetic and Libra graphs, respectively.

A direction for further work is certainly integrating egonet and reduced egonet concepts into the algorithm in the sense that these could even replace the current graph community structure foundation of CALLI.

## REFERENCES

[1] Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, feb 1969.

[2] Bogdan Dumitrescu, Andra Băltoiu, and Ştefania Budulan. Anomaly detection in graphs of bank transactions for anti money laundering applications. *IEEE Access*, 10:47699–47714, 2022.

[3] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, page 413–422, USA, 2008. IEEE Computer Society.

[4] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. oddball: Spotting anomalies in weighted graphs. In Mohammed J. Zaki, Jeffrey Xu Yu, B. Ravindran, and Vikram Pudi, editors, *Advances in Knowledge Discovery and Data Mining*, pages 410–421, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[5] Meng-Chieh Lee, Hung T. Nguyen, Dimitris Berberidis, Vincent S. Tseng, and Leman Akoglu. Gawd: Graph anomaly detection in weighted directed graph databases. In *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '21, page 143–150, New York, NY, USA, 2022. Association for Computing Machinery.

[6] Rodrigo Francisquini, Ana Carolina Lorena, and Mariá C.V. Nascimento. Community-based anomaly detection using spectral graph filtering. *Applied Soft Computing*, 118:108489, 2022.

[7] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *In ICDM*, pages 418–425, 2005.

[8] Deepayan Chakrabarti. Autopart: Parameter-free graph partitioning and outlier detection. In *PKDD*, 2004.

[9] Thomas J. Helling, Jan C. Scholtes, and Frank W. Takes. A community-aware approach for identifying node anomalies in complex networks. In *COMPLEX NETWORKS*, 2018.

[10] Andrew Elliott, Mihai Cucuringu, Milton Martinez Luaces, Paul Reidy, and Gesine Reinert. Anomaly detection in networks with application to financial transaction networks, 2019.

[11] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, may 2000.

[12] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.

[13] Y. Zhao, Z. Nasrullah, and Z. Li. PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research*, 20(96):1–7, 2019.

[14] Paul Irofti, Ștefania Budulan, Bogdan Dumitrescu, and Andra Băltoiu. Graphomaly, 2022. https://pypi.org/project/graphomaly/.