Scalable and reliable anomaly detection on Dynamic Graph based on link prediction to identify disinformation

Victor Chomel, Nathanaël Cuvelle-Magar, David Chavalarias

Scalable and reliable anomaly detection on Dynamic Graph based on link prediction to identify disinformation

Nathanaël Cuvelle-Magar*
CNRS - ISCPIF, Ecole polytechnique, Paris, France
nathanael.cuvelle-magar@polytechnique.edu

Victor Chomel*
CNRS - ISCPIF, EHESS, Paris, France
victor.chomel@iscif.fr

David Chavalarias
CNRS - ISCPIF, EHESS, Paris, France
david.chavalarias@iscif.fr

Abstract

As social networks take an ever-prominent role in information access, combating disinformation becomes increasingly important. Given current volumes of data, automated approaches for the detection of disinformation are the only ones able to offer an online solution. We formalize disinformation as an anomaly in the organic evolution of the network, in terms of users, content or coordination. To identify it, we propose a solution based on Temporal Graph Networks (TGN) adapted to the detection of anomalies and completed with a reliability module which permits users to trade precision for recall. Inheriting the performances of TGN, this solution is able to scale up, work on continuous time settings, and handle multimodality (text, image and video). Moreover, when compared to existing models, our approach outperforms state-of-the-art solutions for anomaly detection in dynamic graphs on several classical datasets. Lastly, it has been tested on a Twitter dataset of the French presidential election of 2022, providing useful insights on manipulation during the campaign.

1 Introduction

In recent years, social networks have become a key venue for public debate. Many users have made them their preferred access to information. The information quality in these networks is therefore a matter of concern, with problems ranging from low-quality content to malicious actors following an economic, political or geopolitical agenda.

After the 2016 US presidential election and Brexit, Fake news have been under scrutiny [1] and have been partly held responsible [2] for low information quality online. Given massive volumes of online data, manual moderation of Fake news is infeasible. It is therefore necessary to automate detection so that only uncertain cases need to be checked by humans.

Several frameworks have been created to propose a theoretical approach of disinformation in the last years. Most of them consider disinformation as a deviation from a norm. For instance, the ABC method [3] splits disinformation into Actors, Behaviors and Content, where actors may be malicious, behaviors coordinated and content containing false claims. Disinformation campaigns combine these elements. For example, astroturfing, or false amplification of online organizations or narratives, gathers users, fake or not, to inorganically behave online in order to push the network towards a desired direction.

Labelling interactions as abnormal requires defining a norm. What would be an organic and “normal" evolution of the network without disinformation? We consider the question of anomaly detection as statistical modeling of (ir)regularities over a latent space.

*These authors contributed equally to this work.

Preprint. Preliminary work.

We distinguish two types of anomalies: edge and node anomalies. Edge anomalies consist of adding or removing edges randomly, and may be considered to be “noisy" anomalies. Node anomalies consist of malicious nodes that behave differently, such as creating or removing edges according to a different statistical distribution from that of existing nodes. Edge anomalies must be linked to node anomalies as proposed by Kagan et al. [4].

1.1 Problem statement

Disinformation can take many forms and approaching it with anomaly detection leads to focusing on only some of them. We do not seek to characterize disinformation through contents or fake accounts but with abnormal behaviors. There is no silver bullet solution here, but this approach, is helpful to circumvent limitation of supervised approaches such as their inability to check the veracity of brand new pieces of news [5]. Examples of new behaviors found in our dataset will be presented in the usecase section.

Disinformation detection via anomaly detection in dynamic graphs. Our behavior-based approach is using a dynamic graph, without label on node. This unsupervised setting is much harder than supervised classification but it is preferred as it is similar to real-life usecase. Our hypothesis is that the network has an organic evolution and disinformation is seen as inorganic process deviating it from its trajectory. This process has many components and explanations and we will only looking for one; inauthentic behaviors.

This approach is used on a Twitter dataset about French politics, collected through API in a process described in [6]. It is a temporal retweet graph from the 2022 French presidential election campaign. In this example, we don’t seek fake accounts or false news but only accounts engaged in inauthentic behavior. These accounts can, for instance, be bots or share fake contents, but it is not these characteristics that are used to detect them, as the method focuses on malicious activities.

In this setting, disinformation monitoring in social networks by way of anomaly detection ideally meets several requirements:

1. *Applicable to dynamic graphs in continuous time.* In discrete time settings, one must gather a sufficient amount of data before processing it. In practice, social media events can be time-sensitive, making these approaches of little use in real life.
2. *Scalable.* Social graphs can easily contain more than one million nodes, rendering any $O(n^2)$ solution intractable. One of the viable approaches to detect anomalies is machine learning. Within machine learning approaches, supervised learning may overadapt to anomalies represented in training [7]. Therefore, we turn to self-supervised learning as a more robust and generalizable approach.
3. *Multimodal.* Text, images and videos are essential vectors of disinformation, and thus must be considered in disinformation detection.
4. *Explainable.* A perfect system would be able to return the specific anomalous interaction and assess the extent to which this information is reliable.

Our contribution is a representation learning solution to disinformation detection based on Temporal Graph Networks (TGNs) [8], used as as an edge anomaly detector and enriched with a reliability module. This model meets all aforementioned requirements while being competitive with previous solutions and achieving state-of-the-art on several benchmarks.

2 Related Work

2.1 Graph based approach and Anomaly Detection in the fight against Disinformation

Disinformation has received significant attention in the last years from the psychology [9], sociology [10] and computer science research communities. Most solutions from computer science are based on deep learning [11], where e.g. Monti et al. [12] proposed a geometric deep learning approach that has been followed since. Indeed, Graph Neural Networks have proven to gather content and context in one model and have quickly become state-of-the-art [13].

Fighting disinformation is generally split into three main tasks: fake content detection [14], bot/fake source detection [15], and inauthentic propagation network detection [16]. Several examples illustrate

the importance of GNNs in these three approaches.
Multiple levels of analysis have been explored in the detection of problematic content. Zellers et al. [17] address the problem at the document level and therefore adopts an approach mainly focused on NLP. At a larger scale, Wu et al. [18] propose a new task based on the notion of "cluster of topically related documents", where both problematic documents and clusters must be detected. The authors rely on a heterogeneous GNN to address this new task. Hu et al. [19] propose to check the content against external knowledge via a GNN model.
The performance of GNNs has also been exploited for the task of detecting sources of misinformation, for example by Feng et al. [20] who rely on graph transformers applied to a heterogeneous information network to detect bots on Twitter.
Finally, propagation based approaches like [21] are suited for the use of GNNs, since input the data naturally takes the form of graphs/cascades [22, 23].

Although the interpretation of fake news and fake news spreaders as abnormal elements of the network is recognized [24], approaches based on the anomaly detection paradigm are less developed than more classical supervised classification methods [25]. Existing methods have explored a variety of avenues, [26] focusing on the detection of anomalies in shared images, [27] analyzing the activity patterns of Twitter and Weibo users to identify anomalous behavior and [28] identifying anomalous topics to trace them back to the spammers groups behind them. Nevertheless, none of them has tried to exploit the performance of GNNs to detect anomalous patterns directly on the dynamic social graph.

Given the performances obtained in general by GNNs in the aforementioned methods, it seems relevant to build such an approach of disinformation identification via an anomaly detection task on the social graph. To this end, we need to select an anomaly detection method that meets the requirements of section 1.1, which leads us to a state of the art of anomaly detection methods on dynamic graphs.

2.2 Anomaly Detection in Dynamic Graphs

Approaches in anomaly detection in dynamic graphs can be classified depending on whether or not algorithms make use of learning procedures.
If they do not, methods do not satisfy the above-mentioned needs in disinformation detection. Indeed, as they do not learn the dynamics of the considered dynamic graph, they tend to rely on an a-priori conception of what is an anomaly. For instance, GOutlier [29] characterises anomalies as behaviours that deviate from community patterns that structure the network, CM-Sketch [30] detects outliers based on several (fixed) edge scores, and SpotLight [31] applies graph sketching to identify the appearance of dense sub-blocks in a graph stream, considered abnormal.
Learning approaches solve this difficulty, for example by learning a hypersphere in the latent space of an autoencoder [32]. Various architectures have been proposed to address this detection task. NetWalk [33] relies on random walks and an autoencoder model to produce a network embedding, used to detect anomalies through a clustering of vertex representations. AddGraph [34] and StrGNN [35] are end-to-end GNN anomalous edges detectors that combine GCNs and Gated Recurrent Units (GRUs) to learn structural and temporal patterns of the network. TADDY [36] uses a transformer to jointly model those patterns in a unified setting. OCAN [37] learns the characteristics of benign behaviour via a Long short-term memory-autoencoder and uses a specific Generative Adversarial Network (GAN) model to train a discriminator to detect malicious users.
Nevertheless, despite modeling capabilities of these algorithms, some problems remain, such as (1) the difficulty to treat attributed graphs; (2) the lack of scalability [38]; and (3) reliance on discrete modeling of dynamic graphs.
To solve these issues, we turn to the active field of link prediction in dynamic graphs [39], and more precisely towards deep learning models developed for this task. Indeed, since anomalies can be considered as outliers, it seems relevant to try to detect them via the (low) probability that a link prediction algorithm would assign to them.

2.3 Link prediction in Dynamic Graphs

Link prediction can be divided into two main approaches, depending on whether the dynamic graph representation is discrete or continuous [39]. We will focus here on continuous time representations, more adapted to our problem. As mentioned in the previous section, only deep learning models are compared, leaving aside alternative approaches such as those developed in [40], [41] or [42].

Learning on continuous time dynamic graphs has required the development of specific neural network architectures, of which the main ones are presented below. Dyrep [43], an inductive deep representation learning framework, produces time evolving low-dimensional node embeddings through learned functions. Jodie [44] learns embedding trajectories of nodes through coupled Recurrent Neural Networks (RNNs). TGAT [45] introduces a temporal graph attention layer to aggregate temporal and topological features. Finally, Temporal Graph Networks [8] is a generic framework, adding a memory module to TGAT and of which previous models can be seen as special cases.
The goal being to evaluate the interest of such dynamic link prediction algorithms for the fight against disinformation, we choose to use TGN as a reference model for our anomaly detection method, its framework being the most generic. It can be used to process dynamic graphs of variable sizes in continuous time, while being scalable up to millions of nodes [46], thus bringing an answer to the problems raised previously.

3 Methods

3.1 A tool for Anomaly Detection

3.1.1 Problem formalization

Based on the formalism introduced in Skarding et al. [39], the following definition of dynamic networks is adopted:

Definition: Dynamic graph. A Dynamic Graph is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where: $\mathcal{V} = \{(v, t_s, t_e)\}$ with v a vertex of the graph and t_s, t_e are respectively the start and end timestamps for the existence of the vertex (with $t_s \leq t_e$). $\mathcal{E} = \{(u, v, t_s, t_e)\}$ with $u, v \in \mathcal{V}$ and t_s, t_e are respectively the start and end timestamps for the existence of the edge (with $t_s \leq t_e$). Vertices and edges can be endowed with features.

A continuous representation is chosen, matching the 'graph stream representation' of Skarding et al. [39]. More precisely, in the words of Rossi et al. [8], these dynamic networks are represented as a sequence of time-stamped node-wise events or interaction events. Node-wise events are node creations, feature updates and deletions. Interaction events are temporal edge creations and deletions.

Our goal is to detect, for any interaction (edge creation event), whether it is abnormal or not. As in Liu et al. [36], this anomalous edge detection task is defined as an abnormality scoring problem.

This task is approached in the same setting as Liu et al. [36], i.e. an unsupervised setting, without labeled data, assuming that the train set has no anomalies. For TGN, the training is assessed on the *average precision* obtained for the validation set, which contains anomalies.

3.1.2 Method Description

Our method consists in modifying the Temporal Graph Networks [8] architecture to build an anomaly detection solution with a reliability module. The central idea of TGN is to attach a memory to each node, allowing the recording of its past interactions, in order to produce a vector representation taking into account not only its own features, but also its temporal history. In addition to the memory is an embedding layer, performing aggregation of the neighbourhood information, which can be achieved by different functions: identity, sum, graph attention mechanism [47], etc.

These first two elements, memory and embedding module, form an encoder associating the considered graph to a representation of its nodes in a latent space, that can then be used as an input by a decoder adapted to a specific task. For the link prediction task, this decoding function is performed by a Multi-Layer Perceptron allowing the calculation of an affinity score between nodes giving, after application of a sigmoid, probabilities of interactions.
In the rest of this section, the configuration of the algorithm used in this study is presented. One can consult Rossi et al. [8] for an in-depth description of the other modules that can be used. An overview of the whole architecture can be found in Figure 1.

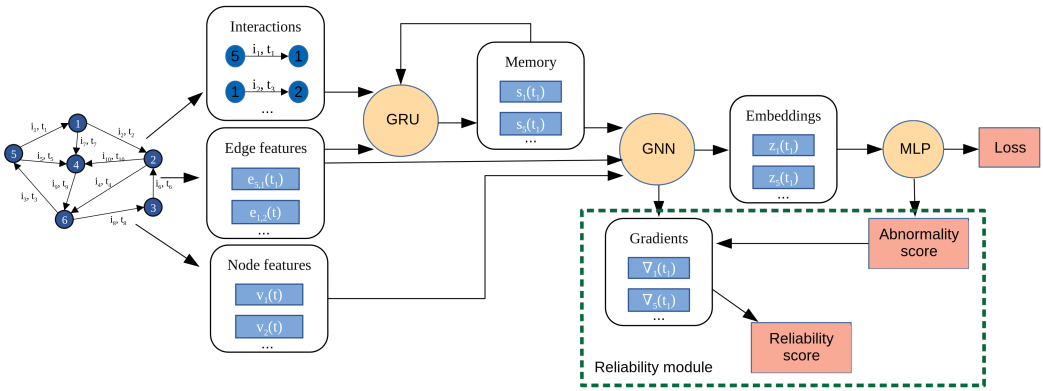


Figure 1: Flow of operations used to train and to compute abnormality and reliability scores

Message treatment. For an interaction involving nodes i and j, messages $m_i(t)$ and $m_j(t)$ will be calculated as the following concatenations:

$$\begin{cases} m_i(t) = s_i(t^-)||s_j(t^-)||e_{ij}(t)||\phi(\Delta t_i) \\ m_j(t) = s_j(t^-)||s_i(t^-)||e_{ij}(t)||\phi(\Delta t_j) \end{cases}$$

where $s_k(t^-)$ is the memory of node $k \in \{i, j\}$ before its update and Δt_k is the time difference between the time of the last update of the node $k \in \{i, j\}$ and the time of the interaction at the origin of the message. $e_{ij}(t)$ denotes the edge features and ϕ a generic time encoding function introduced in Xu et al. [45].
In practice, as interaction data are processed in batches, several links involving the same node may appear before the memory has been updated. Only the most recent message is kept.

Memory. The nodes' memory corresponds to the hidden vector, initially zero, of a recurrent neural network (RNN). Its update mechanism is a GRU unit, shared by the whole network. When a node interacts, the message attached to that interaction is computed and then used as the input vector of the RNN, with the previous node's memory serving as the incoming hidden vector; the outgoing hidden vector becomes the new memory of the node.
The memory of nodes i and j after interaction at time $t \geq 0$ is thus :

$$s_k(t) = \text{GRU}(m_k(t), s_k(t^-)), k \in \{i, j\}\,.$$

Embedding Module. The aggregation function used in this study is the temporal graph attention module, derived from TGAT [45], whose inputs are modified to take into account nodes' memory and temporal features. The embedding of node i at time t is denoted $z_i(t)$.

Final MLP and Loss Function. Prediction of link probability between nodes $i, j \in \{1, ..., n(t)\}$ at time $t \geq 0$ is performed via a two-layer MLP and a sigmoid function:

$$\hat{\mathbb{P}}((i, j), t) = \sigma(\text{MLP}(z_i(t)||z_j(t)).$$

The model is self-supervised, as it learns, for each batch, to predict the probability of next interactions. Negative sampling is used with a rate of one to one, a negative edge being sampled for each considered interaction. In training, link probabilities obtained for batch interactions and for interactions of negative sampling are injected into a BCE loss function.

3.1.3 Use for Anomaly Detection - reliability module

As anomaly detection can be understood as an outlier detection task, it seems natural to consider the probability associated to an edge as a measure of its normality [48]. Conversely, it can be used to construct an anomaly score, which we choose here to define as $a(e, t) = 1 - \hat{\mathbb{P}}(e, t)$, $e \in \mathcal{E}$, while other choices are possible.

Results for anomaly detection can be visualised via a diagram showing the abnormality scores associated with interactions, in order of appearance. Figure 2 is an example of such a diagram,

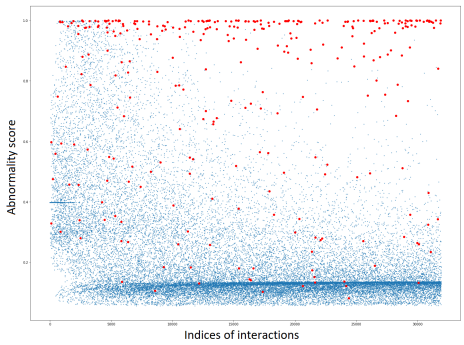


Figure 2: Visualisation of the anomaly scores for the 31933 edges of the Synthetic graph. Anomalies are marked in red. Original interactions are marked in blue. As it can be seen, anomalies were injected in the whole graph, including the training part.

corresponding to a synthetic dynamic graph, denoted **Synthetic** in the following, generated according to modalities detailed in the Appendix A.2.

It can be seen that anomalies are not the only interactions to be associated with high abnormality scores. Although such a situation still allows a high *ROC AUC* to be obtained, it presents practical problems of use in our context. Indeed, as human analysis resources are limited, it is important to ensure a good *precision* of the results, i.e. a low false positive rate.

Thus, to complete the adaptation of TGN to the use we wish to make of it, the last step remaining is to identify an *ad hoc* methodology to mitigate this problem. For this purpose, we adopt an approach based on the norm of the gradient of the abnormality score w.r.t. the model inputs. Indeed, this quantity allows to address both the questions of robustness [49–51] and explainability [52, 53].

As denoted by Yuan et al. [54], gradient-based explanation methods have important limitations. Nevertheless, the development of a brand new explainability method for GNNs on continuous time dynamic graphs goes far beyond the scope of this paper[*]. We have thus decided to use the static method which is the most immediate to adapt to our context and which, moreover, allows to couple explainability and robustness. Moreover, this choice does not alter the appreciated properties of TGN, especially in terms of scalability, since these gradients are computed *online* via PyTorch autograd [57].

By analogy with the abnormality score, we therefore introduce a reliability score constructed from the normalized sum of the gradient norms:

$$r(e,t) = 1 - \frac{\sum\limits_{i \in \text{inputs}(e)} ||\nabla_i a(e,t)||}{\max\limits_{e' \in \mathcal{E}} \left(\sum\limits_{i \in \text{inputs}(e')} ||\nabla_i a(e', t_{e'})|| \right)},$$

where $\nabla_i a(e,t)$ denotes the gradient of a with respect to input i at point (e,t).

See appendix A.1 for a quantitative evaluation of this module.

4 Comparison with Anomaly Detection methods on Dynamic Graphs

Existing anomaly detection solutions didn't fit all stated requirements. Hence, TGN was used as an anomaly detection tool. Once adapted as presented above, it can be compared on some benchmarks. TGN settings and technical details of the experiments are precised in appendix A.3.

[*]To the best of our knowledge, current explainability methods for dynamic GNNs were designed for discrete time settings [55, 56].

4.1 Datasets

In this study, our method is evaluated on six benchmark datasets. In all the datasets, nodes are agents (users or autonomous systems) and edges are interactions between agents (messages, replies, emails or ratings). They are described in appendix A.4.

4.2 Baselines

Four state-of-the-art algorithms for anomaly detection in dynamic graphs are compared to our TGN-based solution:

- **Netwalk** [33], a network representation-based anomaly detection algorithm, relying on the dynamic clustering of nodes embeddings, obtained via random walks and an auto-encoder;
- **AddGraph** [34], an end-to-end anomalous edge detection model, based on a temporal GCN module and an attention-GRU unit;
- **StrGNN** [35], an end-to-end GNN which detects abnormal edges by considering an h-hop enclosing subgraph centered on the considered edge, analysed by combining GCN and GRU;
- **TADDY** [36], an end-to-end transformer model, which learns coupled temporal and spatial information from the dynamic graph, allowing it to generate informative node attributes when these are absent from the raw graph.

In addition to these GNN-based methods, we consider, as in the TADDY paper, the following traditional methods: node2vec [58], Spectral Clustering [59] and DeepWalk [60].

4.3 Experimental Design

The protocol used for this experiment is similar to Liu et al. [36]: the datasets are split into two equal parts to obtain training and testing sets. Different proportions (1%, 5% and 10%) of anomalies are then injected into the testing set. Finally, the performance of the considered algorithm on the task of detecting these anomalies is evaluated via *ROC AUC* metric.

For the generation of anomalies, the TADDY generator[*] has been adapted to a continuous time framework. Once anomalies are generated, timestamps are created based on previous and following interactions. Abnormal edges can thus be injected into the raw data, not having undergone the pre-processing of TADDY (removal of multiple edges, grouping of edges into graph snapshots, etc.). Using the same raw data ensures that all the algorithms have had the same information for their training.

4.4 Results for AD

The results for node2vec, Spectral Clustering, DeepWalk, Netwalk, AddGraph, StrGNN and TADDY are taken directly from the TADDY paper. Results can be found in Table 1. For three datasets, our method outperforms TADDY, which was the previous state-of-the-art solution. It should also be noted that in our approach, TGN can take into account attributes of edges and nodes, which are not meaningful here. Moreover, our approach is scaling up [46], compared to NetWalk, AddGraph and TADDY [38]. The code allowing the reproduction of these results is available online [*].

5 Use Case

As stated in introduction, one of the end goal of detecting disinformation is to identify problematic users. Our use case will focus on this task. Abnormality scores on edges can be combined in many ways to create a score for nodes [4]. Here, a variant of the Sum Edge Label will be used. It consists in summing edge labels for each interaction where the user is the one who retweets. Our solution is trained on a political dataset of retweets in the French political landscape the week before the French presidential election of 2022. Edges features are tweets embedded with BERT and the abnormality and reliability thresholds are set to 0.8 and 0.89 respectively.

[*] https://github.com/yixinliu233/TADDY_pytorch

[*] https://github.com/LoG-sub-16/Anomaly-Detection-on-Dynamic-Graph-to-identify-disinformation

Methods	UCI Messages			Digg			Email-DNC		
	1%	5%	10%	1%	5%	10%	1%	5%	10%
node2vec	0.7371	0.7433	0.6960	0.7364	0.7081	0.6508	0.7391	0.7284	0.7103
Spect. Clust.	0.6324	0.6104	0.5794	0.5949	0.5823	0.5591	0.8096	0.7857	0.7759
DeepWalk	0.7514	0.7391	0.6979	0.7080	0.6881	0.6396	0.7481	0.7303	0.7197
NetWalk	0.7758	0.7647	0.7226	0.7563	0.7176	0.6837	0.8105	0.8371	0.8305
AddGraph	0.8083	0.8090	0.7688	0.8341	*0.8470*	*0.8369*	0.8393	0.8627	0.8773
StrGNN	0.8179	0.8252	0.7959	0.8162	0.8254	0.8272	0.8775	0.9103	0.9080
TADDY	**0.8912**	*0.8398*	*0.8370*	**0.8617**	**0.8545**	**0.8440**	*0.9348*	*0.9257*	*0.9210*
Our method	*0.8846*	**0.8789**	**0.8597**	*0.8575*	0.8358	0.7942	**0.9413**	**0.9352**	**0.9447**

Methods	Bitcoin-Alpha			Bitcoin-OTC			AS-Topology		
	1%	5%	10%	1%	5%	10%	1%	5%	10%
node2vec	0.6910	0.6802	0.6785	0.6951	0.6883	0.6745	0.6821	0.6752	0.6668
Spect. Clust.	0.7401	0.7275	0.7167	0.7624	0.7376	0.7047	0.6685	0.6563	0.6498
DeepWalk	0.6985	0.6874	0.6793	0.7423	0.7356	0.7287	0.6844	0.6793	0.6682
NetWalk	0.8385	0.8357	0.8350	0.7785	0.7694	0.7534	0.8018	0.8066	0.8058
AddGraph	0.8665	0.8403	0.8498	0.8352	0.8455	0.8592	0.8080	0.8004	0.7926
StrGNN	0.8574	0.8667	0.8627	*0.9012*	*0.8775*	*0.8836*	0.8553	0.8352	0.8271
TADDY	**0.9451**	**0.9341**	**0.9423**	**0.9455**	**0.9340**	**0.9425**	*0.8953*	*0.8952*	*0.8934*
Our method	*0.9084*	*0.8849*	*0.8841*	0.8665	0.8553	0.8481	**0.9516**	**0.9490**	**0.9472**

Table 1: AUC results for anomaly detection on benchmark datasets. The best performing method is in bold and the second best in italics.

Users with several interactions labeled as suspicious have been manually inspected. These accounts appear with features that are dramatically different from average users. Most of the users doesn't have a picture and for those who have the large majority doesn't provide information about the user, such as memes for instance. A part of these accounts have Twitter handle with 8 digits in it, meaning that they have been automatically given by the platform. Without being a strong clue, it suggests a lack of importance given to the account. Lastly, almost all labeled accounts are highly active on the platform and some of them among the top active ones. For instance, some accounts produce on average 300 tweets or retweets per day during the past years, suggesting at least partially automated accounts.

Thanks to a manual check, we can confirm that more than half of accounts labeled as suspicious by our solution are engaged in identified inauthentic behaviors. About 10% of the accounts have been deleted or suspended since the data collection six months ago. Most of the accounts belongs to the French AltRight community, defined using graph clustering. Depending on the account, they push anti-vaccine, pro-Russian or anti-immigration narratives. Two kinds of accounts are engaged in behaviors.

The first ones are accounts that seems to be at least partially automated, with a small audience (usually < 100 followers), without profile picture. Most of them retweet massively or are engaged in automated action. Several types of inauthentic behaviors have been observed. For instance, some accounts keep replying to a unique account with pictures. Others comment the same picture under many targeted accounts. Lastly, some accounts repeatedly tweets news links and then the screenshot of the news title.

The second kind of accounts have a larger audience (usually between 1000 and 10000 followers), with a human-like behavior. Some accounts are trying to raise awareness with unauthentic behaviors. For instance, an account uses a strange strategy consisting of various similar interaction with an account. It comments on a post, quotes the same post and tweets mentioning the account with the same message, identical the three times. These accounts also uses a mix of appealing content such as beautiful landscape combined with political content. This deceiving behaviors is meant to create an audience before using it.

This diversity of behaviors are not fully covered by TGN as it does not take into account all interactions but it seems to provide meaningful insight about abnormality of accounts. Moreover, among accounts

labeled as suspicious but not engaged in inauthentic behavior, many accounts seem to be engaged in advertising behaviors, hard to distinguish from truly inauthentic behaviors.

5.1 Explainability

As mentioned before, the explanatory power of gradient norms is limited. In practice, they allow us to understand simple situations to the first order.
For example, one of the two nodes with a score of seven has repeatedly retweeted content from one political community, then from another, and so on. Our solution detected as abnormal all the switching interactions, i.e. the first interaction with one community after several retweets of the other. This analysis of the origin of the anomalies is confirmed with the reliability module. Interactions following these community switches were followed by strong gradients, corresponding to 'echoes' of the above-mentioned anomalies, the network being no longer 'confident' in the representation of the considered node.
The lack of ambiguity in this anomaly made it fairly straightforward to understand and it would be interesting to have an explainability method giving insights in more complex cases. Specifically, consider one of the use case examples. When the abnormal behavior involves, in the eyes of the human analyst, a mixture of tweets, quotes and retweets, we cannot be sure what led TGN to make its prediction. Indeed, it can be seen that the account is abnormal, but we know that this specific type of abnormality could not be detected by an algorithm working only on the retweet graph. This raises the question of what TGN really saw in the retweet patterns, i.e. how the abnormality of the account manifests itself in the retweets behaviour. The limited explainability allowed by gradient norms appears here as one of the main limitations of the model. This corresponds to one of the main avenues of development that we identify for this article and we discuss it in part 6.

6 Discussion

Understanding of the benchmark results. Table 1 shows performances of our method on various datasets. The variability of results can be explained by analyzing the strength of each algorithm.

Thanks to its architecture, TGN has a good expressive power but it is only an advantage when there is enough data for each node to have a truly representative history of its behavior, with continuous time settings. As a result, for datasets with a sufficiently high average degree, such as Email, UCI and, to a lesser extent, Bitcoin-Alpha, our approach works better whereas for low average degree like Digg, it limits its possibilities.

The structure of the algorithm, and in particular the use of a memory module, offers a good model for the behavior of the nodes within a community. For instance, for dataset AS-Topology, even if the average degree appears insufficient, it seems that its clustered structure with a few very active users help our approach to make use of these user activities and to detect cross-community anomalies.

Gradients for interpretability. As stated in the use case, though high gradients can be used to track origins of anomalies in some specific cases, in general, they provide limited explanations. It would thus be interesting to try to adapt more efficient methods developed in the static case [54] to TGN. Nevertheless, it would require to combine these new explainability methods with the respect of the requirements listed in part 1.1.

Disinformation detection. Approaching disinformation as anomalies detection has some limits. For instance, in some fields such as emerging crypto-currencies, fake users and coordinated behaviors can almost represent the norm. In such situations, detecting anomalies would lead to real users.

Working with supervised solutions may lead to strong inductive biases. The strength of our approach is to approach politically sensitive issues with unsupervised methods. Even if this approach is not bias-free, it does not have priors towards shared opinions. The fact that it returns the interaction that creates the anomaly makes it interpretable by end users.

Our use case is an example of how our solution can be used to detect manipulation during a campaign. Starting from anomaly scores and reliability scores, it finds anomalous interactions that, once gathered lead to anomalous users.

A Appendix

A.1 Experimental evaluation of the reliability module

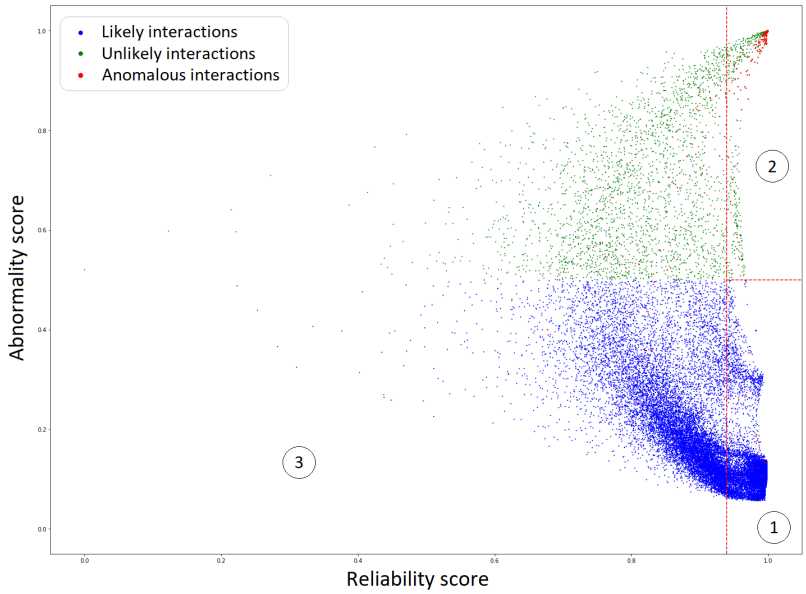


Figure 3: Areas of the abnormality/reliability diagram obtained for the Synthetic dataset. As in 4a-4c, interactions are in blue if their abnormality score is lower than the threshold of 0.5, in green if it is higher and in red if they are anomalies.

For each interaction of the Synthetic dataset, abnormality score is plotted against reliability score for both real and synthetic datasets (see Figures 4a to 4c and 3).

A spike, oriented to the left, is observed which can be interpreted as: the less reliable (in the sense of the gradient norm) the prediction is, the more the model tends to predict a probability of one half, not knowing if the link is plausible. This diagram shows how to separate the reliable predictions from the less reliable ones and, thus, to improve the classification performance of the abnormal edges[*].

Improving results through reliability. Our models with and without reliability module are compared on a task of binary classification of anomalies to check if the reliability module actually helps finding more anomalies. More precisely, thresholds are applied to scores (from 0 to 1) to transform them into binary classes (0 or 1). We compare the performances of thresholding only on the abnormality score (no reliability module) with that of cross-thresholding between abnormality and reliability (by associating the label 0 to the links of insufficient reliability). This is summarized in Figure 3 with Zone 1 considered as normal, Zone 2 as abnormal and Zone 3 as unreliable so not abnormal[*].

The relevance of a reliability threshold is confirmed by Table 2 results. It achieves a better optimum than any threshold of abnormality taken separately. Moreover, the trade-off between *precision* and *recall* is, in the case of double thresholding, to the advantage of *precision*. The *F1-score* and the *AUC ROC* remains of a comparable order of magnitude with those obtained by simple thresholding. It thus seems that the use of the abnormality/reliability diagram is relevant for anomaly detection tasks in real conditions, the global performances of the classifier being then re-oriented towards the robustness of predictions.

[*] Right after an anomaly, the memory of the involved nodes is disturbed, which induces, potentially, high anomaly scores for their subsequent interactions. The reliability score makes it possible to distinguish them from the original anomalies. In short, anomalies are effectively separated from their echoes.

[*] It should be noted that this division could be refined, in particular by using a boundary following the global curvature of the base of the spike.

	Simple threshold		Double threshold	
	Classifier B-F1 ($a = 0.97$)	Classifier B-RA ($a = 0.45$)	Classifier B-F1 ($a = 0.93, r = 0.98$)	Classifier B-RA ($a = 0.41, r = 0.795$)
Precision	*0.551*	0.074	**0.657**	0.101
Recall	0.462	**0.832**	0.509	*0.823*
F1-score	*0.503*	0.137	**0.574**	0.181
AUC ROC	0.729	*0.864*	0.753	**0.875**

Table 2: Comparison of the performance of the best classifiers, in the sense of F1-score and AUC (denoted Classifiers B-F1 and B-RA respectively in the table), obtained by single thresholding and double thresholding. The best results are in bold and the second best in italics.

A.2 Synthetic data generation

The Synthetic dynamic graph is generated by the following procedure:

- three input parameters are first set: the number of users, the time range of exchanges and the proportion of anomalies;
- a random draw, following an exponential law, gives the number of interactions of which each node will be the source;
- the instants of these interactions are then selected in a uniform way on the above-mentioned temporal range;
- similarly, moments for anomalies are drawn uniformly, their number is given by the initially chosen proportion and the total number of interactions;
- with a loop on the instants of the time range, the interactions are then added one after the other. Destinations of interactions are obtained by a uniform draw on the nodes of the graph of the same degree as the source. Conversely, for the anomalies, the destination is chosen among the nodes of different degree than the source.

A.3 TGN settings and experiment details

The configuration of the TGN framework[*] used in our method, corresponding to the TGN-attn variant in the original paper, is the following: nodes memory is used, the memory updater is GRU cell, the embedding consists in attention layer considering the 10 most recent neighbors, message aggregation is done by keeping only the last one and the message function is the identity. The hyperparameters used are the same as in the original paper.

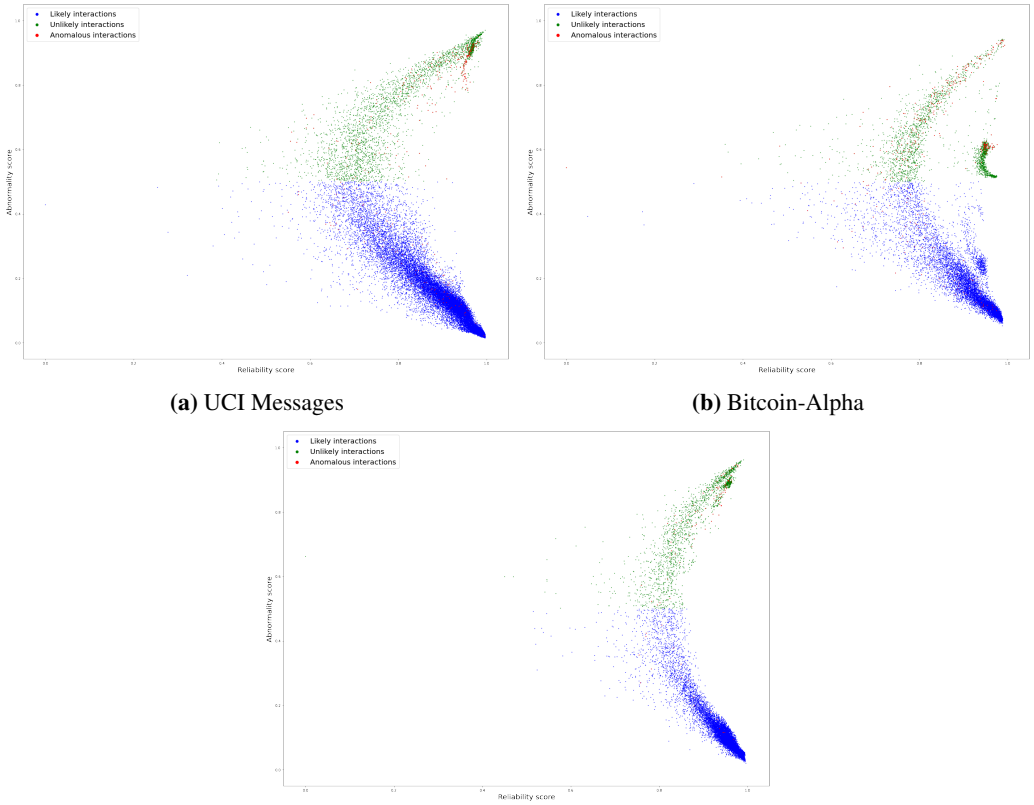
As these datasets have no features for edges and nodes, vectors of size 169, filled with -1, are used as edge features, and vectors of size 172, filled with 1, as node features.

A.4 Datasets description

- **UCI Messages** [61] is a directed social network dataset of online exchanges between students at the University of California, Irvine.
- **Digg** [62] is a reply network collected from the website digg.com.
- **Email-DNC** [63] is a network of emails exchanged by members of the Democratic National Committee, leaked in 2016.
- **Bitcoin-Alpha** [64] is a bitcoin users network collected from www.btc-alpha.com.
- **Bitcoin-OTC** [65] is a dataset network similar to the previous one from www.bitcoin-otc.com.
- **AS-Topology** [66] is a connection network between autonomous systems of the internet.

A.5 Results on datasets

[*] https://github.com/twitter-research/tgn



(a) UCI Messages

(b) Bitcoin-Alpha

(c) Email-DNC

Figure 4: Abnormality/reliability diagrams of some benchmark datasets. Interactions are displayed according to their reliability score on the x-axis and their abnormality score on the y-axis. Edges with an abnormality score higher than 0.5, considered unlikely, are displayed in green. For a score lower than this value, the edges are displayed in blue. Finally, the added anomalies are displayed in red (these diagrams correspond to the 5% of anomalies datasets).

References

[1] Nir Grinberg, Kenneth Joseph, Lisa Friedland, Briony Swire-Thompson, and David Lazer. Fake news on twitter during the 2016 u.s. presidential election. *Science*, 363(6425):374–378, 2019. doi: 10.1126/science.aau2706. URL https://www.science.org/doi/abs/10.1126/science.aau2706. 1

[2] Soroush Vosoughi, Deb Roy, and Sinan Aral. The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018. doi: 10.1126/science.aap9559. URL https://www.science.org/doi/abs/10.1126/science.aap9559. 1

[3] Camille Francois. Actors, behaviors, content: A disinformation abc, 2019. Accessed: 2022-08-12. 1

[4] Dima Kagan, Yuval Elovici, and Michael Fire. Generic anomalous vertices detection utilizing a link prediction algorithm. *Social Network Analysis and Mining*, 8:1–13, 2018. 2, 7

[5] Deepak P. *On Unsupervised Methods for Fake News Detection*, pages 17–40. Springer International Publishing, Cham, 2021. ISBN 978-3-030-62696-9. doi: 10.1007/978-3-030-62696-9_2. URL https://doi.org/10.1007/978-3-030-62696-9_2. 2

[6] Noé Gaumont, Maziyar Panahi, and David Chavalarias. Reconstruction of the socio-semantic dynamics of political activist twitter networks—method and application to the 2017 french presidential election. *PLOS ONE*, 13(9):1–38, 09 2018. doi: 10.1371/journal.pone.0201879. URL https://doi.org/10.1371/journal.pone.0201879. 2

[7] Shuo Yang, Kai Shu, Suhang Wang, Renjie Gu, Fan Wu, and Huan Liu. Unsupervised fake news detection on social media: A generative approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):5644–5651, Jul. 2019. doi: 10.1609/aaai.v33i01.33015644. URL https://ojs.aaai.org/index.php/AAAI/article/view/4508. 2

[8] Emanuele Rossi, Benjamin Paul Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael M. Bronstein. Temporal graph networks for deep learning on dynamic graphs. *ArXiv*, abs/2006.10637, 2020. 2, 4

[9] Gordon Pennycook and David G. Rand. The psychology of fake news. *Trends in Cognitive Sciences*, 25(5):388–402, 2021. ISSN 1364-6613. doi: https://doi.org/10.1016/j.tics.2021.02.007. URL https://www.sciencedirect.com/science/article/pii/S1364661321000516. 2

[10] David M. J. Lazer, Matthew A. Baum, Yochai Benkler, Adam J. Berinsky, Kelly M. Greenhill, Filippo Menczer, Miriam J. Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, Michael Schudson, Steven A. Sloman, Cass R. Sunstein, Emily A. Thorson, Duncan J. Watts, and Jonathan L. Zittrain. The science of fake news. *Science*, 359(6380):1094–1096, 2018. doi: 10.1126/science.aao2998. URL https://www.science.org/doi/abs/10.1126/science.aao2998. 2

[11] Fahim Belal Mahmud, Mahi Md. Sadek Rayhan, Mahdi Hasan Shuvo, Islam Sadia, and Md. Kishor Morol. A comparative analysis of graph neural networks and commonly used machine learning algorithms on fake news detection. *2022 7th International Conference on Data Science and Machine Learning Applications (CDMA)*, pages 97–102, 2022. 2

[12] Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M. Bronstein. Fake news detection on social media using geometric deep learning. *ArXiv*, abs/1902.06673, 2019. 2

[13] M. F. Mridha, Ashfia Jannat Keya, Md. Abdul Hamid, Muhammad Mostafa Monowar, and Md. Saifur Rahman. A comprehensive review on fake news detection with deep learning. *IEEE Access*, 9:156151–156170, 2021. doi: 10.1109/ACCESS.2021.3129329. 2

[14] Xinyi Zhou and Reza Zafarani. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Comput. Surv.*, 53(5), sep 2020. ISSN 0360-0300. doi: 10.1145/3395046. URL https://doi.org/10.1145/3395046. 2

[15] Wajiha Shahid, Yiran Li, Dakota Staples, Gulshan Amin, Saqib Hakak, and Alireza Ghorbani. Are you a cyborg, bot or human? -a survey on detecting fake news spreaders. *IEEE Access*, PP: 1–1, 2022. 2

[16] Iraklis Varlamis, Dimitrios Michail, Foteini Glykou, and Panagiotis Tsantilas. A survey on the use of graph convolutional networks for combating fake news. *Future Internet*, 14(3), 2022. ISSN 1999-5903. doi: 10.3390/fi14030070. URL https://www.mdpi.com/1999-5903/14/3/70. 2

[17] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural fake news. In *NeurIPS*, 2019. 3

[18] Xueqing Wu, Kung-Hsiang Huang, Yi Ren Fung, and Heng Ji. Cross-document misinformation detection based on event graph reasoning. In *NAACL*, 2022. 3

[19] Linmei Hu, Tianchi Yang, Luhao Zhang, Wanjun Zhong, Duyu Tang, Chuan Shi, Nan Duan, and Ming Zhou. Compare to the knowledge: Graph neural fake news detection with external knowledge. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 754–763. Association for Computational Linguistics, August 2021. doi: 10.18653/v1/2021.acl-long.62. URL https://aclanthology.org/2021.acl-long.62. 3

[20] Shangbin Feng, Zhaoxuan Tan, Rui Li, and Minnan Luo. Heterogeneity-aware twitter bot detection with relational graph transformers. In *AAAI*, 2022. 3

[21] Ruichao Yang, Xiting Wang, Yiqiao Jin, Chaozhuo Li, Jianxun Lian, and Xing Xie. Reinforcement subgraph reasoning for fake news detection. *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022. 3

[22] Amila Silva, Yi Han, Ling Luo, Shanika Karunasekera, and Christopher Leckie. Propagation2vec: Embedding partial propagation networks for explainable fake news early detection. *Information Processing Management*, 58(5):102618, 2021. ISSN 0306-4573. doi: https://doi.org/10.1016/j.ipm.2021.102618. URL https://www.sciencedirect.com/science/article/pii/S030645732100114X. 3

[23] Emilio Ferrara. Contagion dynamics of extremist propaganda in social networks. *Information Sciences*, 418-419:1–12, 2017. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2017.07.030. URL https://www.sciencedirect.com/science/article/pii/S0020025517305030. 3

[24] Alessandro Bondielli and Francesco Marcelloni. A survey on fake news and rumour detection techniques. *Inf. Sci.*, 497:38–55, 2019. 3

[25] K Patel Anand, Jay Kumar, and Kunal Anand. Anomaly detection in online social network: A survey. *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 456–459, 2017. 3

[26] Minyoung Huh, Andrew Liu, Andrew Owens, and Alexei A. Efros. Fighting fake news: Image splice detection via learned self-consistency. In *ECCV*, 2018. 3

[27] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. Catchsync: catching synchronized behavior in large directed graphs. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 941–950. ACM, 2014. 3

[28] Qi Dang, Yadong Zhou, Feng Gao, and Qindong Sun. Detecting cooperative and organized spammer groups in micro-blogging community. *Data Mining and Knowledge Discovery*, 31: 573–605, 2016. 3

[29] Charu C. Aggarwal, Yuchen Zhao, and Philip S. Yu. Outlier detection in graph streams. *2011 IEEE 27th International Conference on Data Engineering*, pages 399–409, 2011. 3

[30] Stephen Ranshous, Steve Harenberg, Kshitij Sharma, and Nagiza F. Samatova. A scalable approach for outlier detection in edge streams using sketch-based approximations. In *SDM*, 2016. 3

[31] Dhivya Eswaran, Christos Faloutsos, Sudipto Guha, and Nina Mishra. Spotlight: Detecting anomalies in streaming graphs. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018. 3

[32] Xian Teng, Muheng Yan, Ali Mert Ertugrul, and Y. Lin. Deep into hypersphere: Robust and unsupervised anomaly discovery in dynamic networks. In *IJCAI*, 2018. 3

[33] Wenchao Yu, Wei Cheng, Charu C. Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018. 3, 7

[34] Li Zheng, Zhenpeng Li, Jian Li, Zhao Li, and Jun Gao. Addgraph: Anomaly detection in dynamic graph using attention-based temporal gcn. In *IJCAI*, 2019. 3, 7

[35] Lei Cai, Zhengzhang Chen, Chen Luo, Jiaping Gui, Jingchao Ni, Ding Li, and Haifeng Chen. Structural temporal graph neural networks for anomaly detection in dynamic graphs. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021. 3, 7

[36] Yixin Liu, Shirui Pan, Yu Guang Wang, Fei Xiong, Liang Wang, and Vincent C. S. Lee. Anomaly detection in dynamic graphs via transformer. *ArXiv*, abs/2106.09876, 2021. 3, 4, 7

[37] Panpan Zheng, Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu. One-class adversarial nets for fraud detection. *arXiv preprint arXiv:1803.01798*, 2018. 3

[38] Armin Danesh Pazho, Ghazal Alinezhad Noghre, Arnab A. Purkayastha, Jagannadh Vempati, Otto Martin, and Hamed Tabkhi. A comprehensive survey of graph-based deep learning approaches for anomaly detection in complex distributed systems. *ArXiv*, abs/2206.04149, 2022. 3, 7

[39] Joakim Skarding, Bogdan Gabrys, and Katarzyna Musial. Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey. *IEEE Access*, 9:79143–79168, 2021. 3, 4

[40] Giang Hoang Nguyen, John Boaz Lee, Ryan A. Rossi, Nesreen Ahmed, Eunyee Koh, and Sungchul Kim. Dynamic network embeddings: From random walks to temporal random walks. *2018 IEEE International Conference on Big Data (Big Data)*, pages 1085–1092, 2018. 3

[41] Nikolaos Bastas, Theodoros Semertzidis, Apostolos Axenopoulos, and Petros Daras. evolve2vec: Learning network representations using temporal unfolding. In *MMM*, 2019. 3

[42] Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. Temporal knowledge graph embedding model based on additive time series decomposition, 2019. URL https://arxiv.org/abs/1911.07893. 3

[43] Rakshit S. Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *ICLR*, 2019. 4

[44] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2019. 4

[45] Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *ArXiv*, abs/2002.07962, 2020. 4, 5

[46] Michael Bronstein. Accelerating and scaling temporal graph networks on the graphcore ipu. https://towardsdatascience.com/accelerating-and-scaling-temporal-graph-networks-on-the-graphcore-ipu-c15ac309b765, 2022. Accessed: 2022-07-27. 4, 7

[47] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. 4

[48] Zan Huang and Daniel Dajun Zeng. A link prediction approach to anomalous email detection. *2006 IEEE International Conference on Systems, Man and Cybernetics*, 2:1131–1136, 2006. 5

[49] Philipp Oberdiek, Matthias Rottmann, and Hanno Gottschalk. Classification uncertainty of deep neural networks based on gradient information. In *ANNPR*, 2018. 6

[50] Julia Lust and Alexandru Condurache. Gran: An efficient gradient-norm based detector for adversarial and misclassified examples. In *ESANN*, 2020.

[51] Jan-Philipp Schulze, Philip Sperl, Ana Ruaductoiu, Carla Sagebiel, and Konstantin Bottinger. R2-ad2: Detecting anomalies by analysing the raw gradient. *ArXiv*, abs/2206.10259, 2022. 6

[52] Muriel Gevrey, Ioannis F Dimopoulos, and Sovan Lek. Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecological Modelling*, 160:249–264, 2003. 6

[53] Federico Baldassarre and Hossein Azizpour. Explainability techniques for graph convolutional networks. *ArXiv*, abs/1905.13686, 2019. 6

[54] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2022. 6, 9

[55] Yucai Fan, Yuhang Yao, and Carlee Joe-Wong. Gcn-se: Attention as explainability for node classification in dynamic graphs. *2021 IEEE International Conference on Data Mining (ICDM)*, pages 1060–1065, 2021. 6

[56] Jiaxuan Xie, Yezi Liu, and Yanning Shen. Explaining dynamic graph neural networks via relevance back-propagation. *ArXiv*, abs/2207.11175, 2022. 6

[57] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zach DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 6

[58] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. 7

[59] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007. 7

[60] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014. 7

[61] Tore Opsahl and Pietro Panzarasa. Clustering in weighted networks. *Soc. Networks*, 31:155–163, 2009. 11

[62] Munmun De Choudhury, H. Sundaram, Ajita John, and Dorée D. Seligmann. Social synchrony: Predicting mimicry of user actions in online social media. *2009 International Conference on Computational Science and Engineering*, 4:151–158, 2009. 11

[63] Ryan A. Rossi and Nesreen Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015. 11

[64] Srijan Kumar, Francesca Spezzano, V. S. Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 221–230, 2016. 11

[65] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and V. S. Subrahmanian. Rev2: Fraudulent user prediction in rating platforms. *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018. 11

[66] Beichuan Zhang, Raymond A. Liu, Daniel Massey, and Lixia Zhang. Collecting the internet as-level topology. *Comput. Commun. Rev.*, 35:53–61, 2005. 11