Understanding the Influence of Extreme High-degree Nodes in Graph Anomaly Detection

Authors

Institutes

Abstract. Graph Anomaly Detection (GAD) has garnered considerable attention for its potential in identifying anomalies within graph networks. However, a crucial aspect overlooked by prior works pertains to node degree inequality and the existence of nodes with exceptionally high degrees in the graph. This factor is integral to anomaly detection, as it may result in inadequate modeling of anomalous nodes, particularly when anomalies exhibit small degrees. Furthermore, it can introduce noise through neighbor aggregation in Graph Neural Networks (GNNs) when anomalies are connected to these nodes with extremely high degrees. Historically, the limitation of graph anomaly datasets has hindered the exploration of the impact of nodes with extremely high degrees. In this paper, we introduce a novel graph anomaly dataset derived from authentic blockchain data, devoid of injected or synthetic anomalies. The dataset features a top node with a degree exceeding 700 thousand, significantly surpassing that of the second node, illustrating node degree inequality. We conduct various experiments on this dataset, deriving insights to comprehend the implications of nodes with extremely high degrees, thereby propelling advancements in the field of GAD. To facilitate further research and collaboration, we have made both the dataset and codes publicly accessible on Github.

Keywords: Graph Anomaly Detection · Node Degree Inequality · Graph Construction.

1 Introduction

Graph, a data structure with nodes and edges, has been widely used to model real-world scenarios, such as social networks [15], financial trading networks [34] and paper citing networks [14]. Generally, three kinds of tasks are considered on graphs: node classification [33], link prediction [42] and graph classification [43]. Moreover, since graphs can capture structural information and relationships between entities, many anomaly detection methods are also based on graph neural networks (GNNs) [2,9], aiming to identify anomalies that are distinct from the majority in the graph.

Historically, numerous models for graph anomaly detection (GAD) have been put forth. Unsupervised approaches, such as DONE [2], DOMINANT [9], and CONAD [41], have been proposed. Similarly, supervised methods like PCGNN [24],

GAS [23], and BWGNN [34] have been introduced. These models have significantly contributed to the advancement of graph learning and the field of anomaly detection.

A crucially overlooked problem in existing Graph Anomaly Detection (GAD) models is node degree inequality [25,35]. Many graph networks follow a power-law distribution, with a minority of top nodes dominating interactions, while long-tailed nodes occupy a smaller share. This signifies the presence of extreme high-degree nodes in the graph, exerting a pivotal influence on graph anomaly detection tasks. First, node degree inequality introduces bias in modeling embeddings, impacting GNNs that rely on message passing and neighbor aggregation. Extreme high-degree nodes, having more neighbors, can optimize their embeddings more effectively. Conversely, anomalous nodes with limited interactions are more easily detected. This imbalance poses a challenge to anomaly detection by providing insufficient information for modeling anomalies. Additionally, extreme high-degree nodes lead to a tightly connected component in the graph, impacting GNN node aggregation and potentially causing over-smoothing or high aggregation costs. Furthermore, anomalies may aggregate normal nodes' features through edges with extreme high-degree nodes, complicating the representation of abnormal nodes and introducing noise into the learning process.

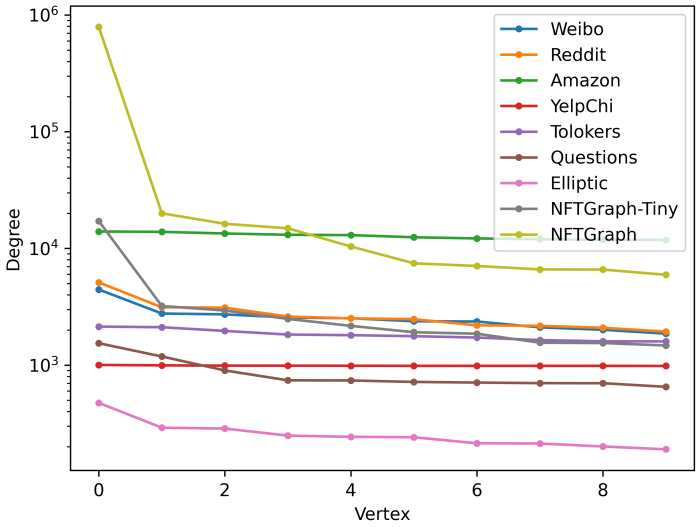

Fig. 1. The distribution of node degrees among the foremost ten nodes in diverse graph anomaly datasets. Notably, our proposed datasets, NFTGraph and NFTGraph-Tiny, manifest pronounced node inequality, particularly evident between the primary and secondary nodes.

Understanding the inequities in node degree and the implications of nodes with extremely high degrees assumes a critical role in the domain of graph

anomaly detection. Regrettably, the constraints posed by scenarios as the primary data sources for historical datasets have hindered the availability of a graph anomaly dataset featuring extreme high-degree nodes. As illustrated in Figure 1, several widely-utilized Graph Anomaly Detection (GAD) datasets, such as Amazon [27] and Yelpchi [29], exhibit subtly distributed node degrees. The utilization of injected or synthetic graph datasets is not feasible since it introduces a potential risk of significant data leakage [17]. These challenges impede the advancement of GAD models in addressing this pivotal concern.

To mitigate the aforementioned limitation, we present a novel graph anomaly dataset named NFTGraph. This dataset is curated based on transactions involving Non-fungible Tokens (NFTs) on the blockchain. Leveraging the transparency of the blockchain, NFTGraph stands as an anomaly dataset derived from genuine transactions, circumventing the use of injected or synthetic anomalies, and featuring an extreme node degree inequality. Illustrated by the yellow and gray lines in Figure 1, NFTGraph and its derived variant, NFTGraph-Tiny, exhibit the most pronounced slope between the first and second nodes, indicating the first node as an extreme high-degree node. This node, in fact, corresponds to the null address (0x00...0000) on the blockchain.

Leveraging the provided graph anomaly dataset, we undertake a series of experiments to scrutinize the influence of extreme high-degree nodes in GAD and derive insightful conclusions. Specifically, in the context of unsupervised anomaly detection, the removal of the first-degree node generally leads to improved performance, suggesting that extreme high-degree nodes likely introduce considerable noise during the aggregation process. However, this improvement is not consistently observed in supervised methods. Additionally, both GNN-based and non-GNN-based models exhibit increased susceptibility to the influence of extreme high-degree nodes when incorporating graph structural information. Conversely, models relying solely on node attributes maintain relatively unchanged or marginally declined performance after the removal of the first-degree node in unsupervised settings, suggesting a potential mitigation of the impact of extreme high-degree nodes by not leveraging graph structure. Notably, the performance of supervised models remains conspicuously affected, underscoring the discernible influence of labels.

In summary, our contributions are as follows:

- We introduce NFTGraph, a graph anomaly dataset derived from blockchain data, free from injected or synthetic anomalies, characterized by extreme high-degree nodes.
- The dataset is open-sourced, presented in an easily accessible format compatible with mainstream graph learning libraries such as `ogb`, `pyg`, and `dgl`.
- A series of experiments are conducted on the proposed NFTGraph to scrutinize and comprehend the impact of extreme high-degree nodes. Meaningful insights are drawn, underscoring the potential for advancing graph anomaly detection.

2 Related Works

2.1 Degree-related GNNs

Historically, several Graph Neural Networks (GNNs) with a focus on degree-related considerations have been introduced to rectify node degree distribution biases. Notable examples include DEMO-Net [39] and SL-DSGCN [35], which implement degree-specific node transformations, and DegFairGNN [26], which employs a learnable function for generating debiasing contexts. Residual2Vec [21] employs random walks as a biased node sampler, while other GNNs addressing degree-related performance differences include Tail-GNN [25] and Rawls-GCN [18]. It is noteworthy, however, that these models have primarily been explored within the context of node or graph classification tasks. To date, there has been a dearth of research exploring the implications of node degree in anomaly detection tasks. These tasks are distinct, as modeling distinct embeddings for anomalous nodes as opposed to normal nodes presents a more challenging endeavor. Furthermore, existing models tend to overlook the nuances associated with unsupervised models in this context.

2.2 Graph Anomaly Datasets

Numerous graph anomaly datasets are widely employed in research endeavors. Notable examples include Weibo [22] and Reddit [22], renowned datasets derived from social networks designed for the identification of suspicious users. Moreover, Amazon [27] and Yelpchi [29] serve as datasets specifically tailored for the detection of fraudulent reviews. Tolokers [28], sourced from real-world collaborative work projects, is utilized for predicting banned workers, while Questions [28] is a question-answering dataset employed for detecting active users. The Elliptic dataset [38], centered around bitcoin transactions, is notable for its focus on illicit transactions. While these classical datasets have significantly contributed to the development of graph anomaly detection models, they lack the distinctive node degree inequality showcased in Figure 1. As a result, they are not adequately qualified to explore the impact of nodes with extremely high degrees in the context of graph anomaly detection models.

3 Dataset Construction

Leveraging blockchain data, we have devised a pipeline that transforms raw data into a graph.

3.1 Raw Data

The blockchain is essentially a ledger, where each transaction is documented. In particular, our attention is directed towards the Ethereum blockchain, recognized as one of the most substantial and influential blockchain networks. Generally,

transactions on Ethereum can be categorized into two primary segments: one pertains to transactions involving fungible tokens (ERC-20 Tokens), such as `Ether` and `USDC`, while the other concerns non-fungible tokens. Furthermore, the latter can be subcategorized into two distinct types, namely, ERC-721 NFTs and ERC-1155 NFTs, based on the contract standard governing their creation. In contrast to ERC-721, ERC-1155 emerges as a novel blockchain contract standard, exuding substantial promise, and poised to potentially surpass ERC-721, courtesy of the cost-efficiency and reduced transaction fees inherent in ERC-1155 [32]. We choose to establish a dataset based on transactions involving ERC-1155 NFTs. The motivation behind this choice lies in the recent introduction of ERC-1155, which is two years newer than ERC-721 and four years newer than ERC-20. The early stage of ERC-1155 increases the likelihood of generating node inequality and extreme high-degree nodes. Over time, the emergence of more high-degree nodes, particularly from entities like cryptocurrency exchanges, may contribute to diminishing the initially observed extreme inequality.

First, we request all ERC-1155 NFT token addresses from the Etherscan website [11] (These token addresses can be found in the GitHub repository [1]). Subsequently, we gather all transaction hashes linked to these NFT token addresses through the official `txlist` API function provided by Etherscan. That is, the token address corresponds to the `from` or `to` fields in a transaction. Lastly, using these transaction hashes as keys and indexes, we proceed to retrieve specific transaction details from the TokenView website [36] through an automated web driver in the Python selenium package. The Python script for this process is available in the GitHub repository [1]). A snapshot of a transaction on the TokenView website can be found in Appendix A.1, and we extract certain fields to compose the format of raw data (Table 1). In total, we obtained 11,009 NFT token addresses, and NFT transactions up to July 31, 2022, with a total number of 8,668,231.

Table 1. Format of raw NFT transaction data. Each row represents a transfer of a certain *Amount* of *Token* worth *Value* between the *From* and *To* addresses at a specific *Timestamp*, spending a transaction fee (*TxFee*), as recorded in the transaction (*TxHash*).

TxHash	*From*	*To*	*Token*	*Timestamp*	*Amount*	*Value*($)	*TxFee*($)
0xb5...b420	0x94...7293	0x6e...b7d3	0xd0...2430	20220730055230	1	78.52	2.23
0xa5...aeea	0x00...0000	0xd8...ac95	0xd0...2430	20220730055230	14	0.0	0.98
0xa2...bdf1	0x5b...1abb	0x4f...6580	0xd0...2430	20220730055138	1	0.0	0.33
...	...	...	...	...	...	...	...

3.2 Graph Construction

Upon acquiring the raw data, our goal is to construct ERC1155-NFTGraph (NFTGraph for short). To emphasize the unconcealment of NFTGraph, we uti-

lize the information from transactions directly without any masking or transformation (Refer to Appendix A.2 for ethics and privacy). Specifically, the *From* and *To* addresses, acting as the sending and receiving parties of a transaction, serve as the source and target nodes in the graph. An edge is established between the source and target nodes if tokens are transferred between them in a transaction. Interactions between nodes within the NFTGraph encompass multiple occurrences, often involving the transfer of tokens from source nodes to target nodes. However, for compatibility with the majority of GNN models, directed multi-edges between any two nodes are merged into a singular edge.

Utilizing the transactions linked to nodes, we emphasize several node features: *Address, OutAmount, OutValue, OutTxFee, InAmount, InValue*, and *InTxFee*. Aside from *Address*, which serves as the unique identifier for each node, the other features represent cumulants from transactions. Specifically, *OutAmount* and *InAmount* represent the cumulative tokens transferred from and to the current node, respectively. Similarly, *OutValue* and *InValue* indicate the cumulative transaction value when the current node is a source and target node separately and *OutTxFee* and *InTxFee* are the cumulative transaction fees paid by the current node as a sender and receiver, respectively.

Formally, the NFTGraph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes, and $\mathcal{E}$ is the set of all edges. $\mathcal{G}$ possesses a node attribute matrix $\mathbf{X} \in \mathcal{R}^{|\mathcal{V}|\times 6}$ and an adjacency matrix $\mathbf{A} \in \mathcal{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. $\mathbf{A}_{uv} = 1$ if there exists an edge e in $\mathcal{E}$ between u and v, where $u, v \in \mathcal{V}$. Otherwise, $\mathbf{A}_{uv} = 0$.

Moreover, we present a variant of NFTGraph achieved through extraction. By extracting approximately 20,000 of the most active nodes while excluding isolated nodes, we form NFTGraph-Tiny, leading to a substantial reduction in the size of the original graph. This is executed with the recognition that certain GNNs may encounter challenges in handling extensive graphs within resource-constrained environments, a scenario often encountered in certain anomaly detection methods.

3.3 Labeling Suspicious Nodes

First, we define *ground-truth fraudulent nodes* that are the account addresses involved in a variety of fraudulent schemes published by previous researchers, encompassing Ponzi schemes [6] and phishing scams [4]. We collect an approximate total of 6,000 ground-truth fraudulent account addresses (Data sources and all ground-truth fraudulent account addresses can be found in the GitHub repository [1]), then we download all the transactions associated with these accounts through the Etherscan's official API [10]. Drawing inspiration from the methodologies delineated in prior works [5], we label nodes that exhibit interactions with the fraudulent nodes exceeding a count of three instances as *suspicious nodes*. The reason behind this approach is based on the idea that these nodes warrant examination due to their extensive participation in transactions with the previously mentioned fraudulent nodes. This recurring pattern is uncommon for legitimate entities, suggesting a potential connection to the same entity controlling these nodes. Suspicious nodes aim to alleviate the notable imbalance,

arising from the limitation of ground-truth fraudulent nodes aligning with the NFTGraph's node set.

4 Properties and Visualization

Table 2. Statistics of NFTGraph and some common graph anomaly datasets.

Dataset	#Nodes	#Edges	Anomaly ratio	Density	Avg. degree	No.1 degree	No.2 degree	No.10 deg. ratio
Weibo	8405	416368	10.3%	0.0059	2151	4447	2769	1.39
Reddit	10984	168016	3.3%	0.0014	2761	5112	3134	1.64
Amazon	11944	8847096	9.5%	0.062	3726	13964	13874	0.18
YelpChi	45954	7739912	14.5%	0.0037	11656	1004	996	0.02
Tolokers	11758	530758	21.8%	0.0038	2984	2140	2113	0.34
Questions	48921	202461	3.0%	8.46E-05	12234	1541	1186	1.36
Elliptic	203769	438124	9.8%	1.06E-05	50944	475	291	1.5
NFTGraph-Tiny	22110	369353	1.14%	0.0007	5542	17156	3212	**10.63**
NFTGraph	1161847	3002840	0.39%	2.22E-06	290464	789782	20000	**131.85**

Table 2 presents statistical metrics for NFTGraph and several other prominent graph anomaly datasets. The original NFTGraph comprises 1,161,847 nodes and 3,002,840 edges, which is a large scale among these datasets. NFTGraph-Tiny, representing the core of the NFT trading network, contains only 22,110 nodes and 369,353 edges. The metric *Density* reflects the node-edge ratio. It is noteworthy that NFTGraph-Tiny exhibits a higher density than NFTGraph, suggesting that the most active nodes have a higher average transaction volume. NFTGraph has an extreme anomaly ratio of 0.39%, the lowest among the datasets, presenting a challenge for detection models. Additionally, the highest degree node (No.1 degree in the table) in NFTGraph has a degree of 789,782, functioning as a super node connecting a large number of nodes in the network. In contrast, other graphs display more comparable node degrees between the No.1 and No.2 degree nodes. The No.10 degree ratio, defined as *(No.1 degree - No.10 degree)/No.10 degree*, is notably over 130 for NFTGraph, indicating significant node degree inequality. NFTGraph-Tiny, extracted from NFTGraph, also exhibits a high degree ratio compared to other graph datasets. Figure 2 provides a visualization of NFTGraph-Tiny. Node sizes in the figure are proportional to their degrees, revealing a prominent super node at the center, identified as the null address on the Ethereum blockchain. (In this context, the term *super node* refers to the node with the highest degree.) Each NFT is either created or burned through interactions with the null address. Different colors represent distinct edge types (Please refer to the github for descriptions of edge types), with the pink color dominating the entire figure, indicating that the null address has the largest number of edges.

Fig. 2. The visualization of NFTGraph-Tiny. It can be observed that there is a super node at the center, which is the null address.

5 Experiments

To comprehend the impact of the extreme high-degree node in anomaly detection, we conduct several experiments in both unsupervised and supervised settings.

5.1 Experimental Settings

Dataset. Since NFTGraph and NFTGraph-Tiny exhibit similar degree distributions, and certain GNNs face challenges with large graphs, we select the proposed NFTGraph-Tiny as the foundational dataset. Furthermore, to assess the impact of the extreme high-degree node, we introduce another variant dataset by eliminating the highest degree node, i.e., the null address 0x00...0000, and the edges connected to it in the graph. These two graphs are denoted as *w/ Super Node* and *w/o Super Node*, respectively, in the tables. From Table 3, without the super node, the first degree decreases from 17,158 to 3,214, bringing it closer to the second degree node.

Task. We design a task to identify whether the node in the graph is suspicious or not. Formally, we aim to train a model $f : f(u) \rightarrow \{0, 1\}, \forall u \in \mathcal{V}$, where 0 represents the normal and 1 represents the suspicious. Specifically, we have 21,858/21,857 normal nodes as positive samples and 252 suspicious nodes as negative samples. It is noteworthy that the imbalance ratio is as high as 99:1.

Table 3. Two datasets derived from the proposed graph: *w/ Super Node* represents the original graph, while *w/o Super Node* depicts the graph after removing the highest degree node and its linked edges.

Dataset	#Nodes	#Edges	#Anomalies	No.1-5 degrees
w/ Super Node	22110	369353	252	[17158,3214,2940,2491,2173]
w/o Super Node	22109	352196	252	[3214,2940,2491,2173,1917]

Metric. Due to the imbalance of anomalous nodes and normal nodes, we select Area Under the Receiver Operating Characteristic Curve (AUROC) as the metric of the task. AUROC offers a comprehensive assessment of model performance, remains insensitive to class imbalance, and retains significance for cross-dataset comparisons.

Baseline. In the broader context, existing Graph Anomaly Detection (GAD) models can be categorized into two primary classifications: unsupervised GAD and supervised GAD. The former consistently relies on attribute and/or structure reconstruction, while the latter encompasses specialized GNNs tailored for supervised anomaly detection. Four unsupervised GAD models, CONAD [41], DOMINANT [9], AnomalyDAE [12], and GCNAE [20] are selected for examination. Additionally, various supervised specialized GNNs, including PCGNN [24], GAS [23], BernNet [16], AMNet [3], GHRN [13], and GATSep [44], are included. Furthermore, within the supervised category, standard GNNs such as GCN [19], SGC [8], GAT [37], GT [31], GIN [40], and GraphSAGE [15], are incorporated to explore the impact. To comprehensively assess the influence of extreme high-degree nodes on graph models, several Anomaly Detection (AD) models not rooted in GNNs are chosen. Specifically, within the unsupervised category, DONE [2], AdONE [2], MLPAE [30], and GAAN [7] are selected, all of which are founded on Multilayer Perceptrons (MLP). In the supervised category, four classical non-GNN models: MLP, KNN, SVM, and Random Forest (RF) are chosen for comparison with graph models.

Data Split, Hyperparameter and Running. Unsupervised anomaly detection models operate in a transductive manner, reconstructing graph attributes and/or structure during training without labels. Anomalous scores are assigned to nodes based on reconstruction loss, and AUROC is evaluated during testing. For supervised models, the dataset is split (40%/20%/40%), and nodes with labels are fed to train the model. GNN aggregation may involve validation and test set nodes, but labels are not leaked. Validation set aids hyperparameter selection, and AUROC is computed on the test set during testing.

To ensure fairness, hyperparameters are tuned. Random search in a predefined space optimizes them in 100 trials. The best unsupervised model hyperparameters maximize AUROC on the entire graph; supervised models optimize AUROC on the validation set. Five trials per model use different seeds, providing mean and standard deviation metrics. Better-performing metrics are bolded in

the table, with underscores for small performance differences between datasets. Details about hyperparameter search space and the best hyperparameters can be found in the github.

5.2 Experimental Results

This subsection addresses three key questions:

(1) What is the overall efficacy of anomaly detection on the graph?

(2) What is the influence of the extreme high-degree node on both **unsupervised and supervised** anomaly detection models?

(3) What is the influence of the extreme high-degree node on both **GNN-based and non-GNN-based** anomaly detection models?

Table 4. Unsupervised Anomaly Detection Models Comparison with and without the highest degree node, the Super Node. The AUROC metric is provided in the table.

Model	w/ Super Node	w/o Super Node
DONE	0.5770±0.0062	**0.6212±0.0111**
AdONE	0.5502±0.0015	**0.6079±0.0039**
MLPAE	0.6536±0.0001	0.6521±0.0001
GAAN	0.6536±0.0003	0.6527±0.0003
CONAD	0.5999±0.0009	**0.6310±0.0110**
DOMINANT	0.6026±0.0138	**0.6251±0.0072**
AnomalyDAE	0.4278±0.0378	**0.4595±0.0309**
GCNAE	0.6023±0.0176	**0.6554±0.0122**

Overall performance. Overall performance provides insight into data quality regarding anomalies and the difficulty of the detection task. Table 4 and Table 5 present the AUROC metrics of unsupervised and supervised anomaly detection models, respectively. The highest performance is attained by a standard GNN, Graph Isomorphism Network (GIN), achieving an AUROC of 0.6825 on the *w/ Super Node* dataset. On the other hand, another standard GNN, GraphSAGE, exhibits the top performance on the *w/o Super Node* dataset, with an AUROC of 0.6644. The results reveal that both unsupervised models and specialized GNNs designed for anomaly detection tasks do not surpass the performance of the standard GNN. Notably, none of the models achieves an AUROC exceeding 0.7. Moreover, the results of unsupervised GAD are comparable to those of supervised GAD in general. This suggests that these models face challenges in handling the proposed anomalous scenario and struggle to discern patterns of anomalies, underscoring the difficulty of detection in our proposed anomaly dataset and the complexity of real-world fraud compared to injected or synthetic anomalies.

Table 5. Supervised Anomaly Detection Models Comparison with and without the highest degree node, the Super Node. The AUROC metric is provided in the table.

Model	w/ Super Node	w/o Super Node
MLP	0.6022±0.0074	**0.6100±0.0016**
KNN	0.5612±0.0233	0.5612±0.0233
SVM	0.5533±0.0358	**0.5700±0.0358**
RF	0.5743±0.0087	**0.6264±0.0087**
GCN	0.6379±0.0016	**0.6544±0.0231**
SGC	0.6408±0.0032	**0.6538±0.0062**
GAT	0.5344±0.0211	**0.6193±0.0244**
GT	**0.5742±0.0206**	0.5465±0.0449
GIN	**0.6825±0.0165**	0.6250±0.0413
GraphSAGE	**0.6795±0.0182**	0.6644±0.0435
PCGNN	0.5470±0.0035	**0.5856±0.0004**
GAS	0.6194±0.0357	**0.6625±0.0054**
BernNet	0.5961±0.0009	**0.6440±0.0247**
AMNet	**0.6132±0.0089**	0.5459±0.0121
GHRN	**0.6540±0.0667**	0.6331±0.0846
GAT-Sep	**0.5814±0.0219**	0.5261±0.0451

Influence on unsupervised and supervised methods. Table 4 indicates a notable improvement in models' performance (over +2%) after removing the first-degree node compared to the original graph, with the exception of MLPAE and GAAN, which exhibit almost the same results across both datasets (fluctuating by less than 0.2% AUROC). In contrast, Table 5 reveals no discernible pattern for supervised models. This may stem from the fact that, for unsupervised models, the absence of labels leads them to rely more on the characteristics of neighbors and the overall graph structure when inferring anomalies. These models reconstruct attribute and/or structure matrices, making it straightforward to identify nodes with higher decision scores as anomalies. However, nodes exhibiting distinct patterns among their neighbors do not necessarily indicate fraudulent behavior. In the case of supervised models, with the availability of labels, they can more directly learn patterns of anomalies as they consider both nodes themselves and their neighbors. Consequently, supervised models are less influenced by the presence of the first-degree node.

Influence on GNN-based and non-GNN-based methods. For unsupervised models, the initial four models abstain from utilizing GNN as their frameworks, in contrast to the subsequent four models, which are GNN-based. Within the first quartet, DONE and AdONE eschew GNN as their underpinnings, opting instead for the utilization of the adjacency matrix for structural reconstruction—a process that can be construed as harnessing the structural information inherent in the graph. In contrast, MLPAE and GAAN forgo structural reconstruction, refraining from using either adjacency matrix reconstruction or GNN propagation, and solely employ the attribute matrix as input.

The data presented in Table 4 indicates that models based on GNN or those relying on structure reconstruction exhibit a marked improvement of over 2% in AUROC after the removal of the first-degree node. This might seem counter-intuitive since the *w/ Super Node* dataset ostensibly contains more information than the *w/o Super Node* dataset. Referring to Table 3, the first node boasts a degree of 17,158, signifying connections to over 77% of the graph's nodes. Consequently, when GNN aggregates messages from 2-hop neighbors, a predominant share of nodes ends up aggregating messages from the same neighbors. This introduces noise to the node message aggregation, particularly challenging for anomaly detection, as it complicates the distinction between normal and anomalous nodes. Notably, MLPAE and GAAN yield disparate results. The alterations in the AUROC metric are negligible, with a slight decline from 0.653 to 0.652 after the exclusion of the first degree. This is likely attributable to the absence of structural information usage. Consequently, non-related information is not propagated.

In the realm of supervised models, owing to the availability of labels, the outcomes diverge. The initial four non-GNN-based models exhibit an augmentation in AUROC metrics when the graph expels the extreme high-degree node. These models exclusively discern anomalies based on node attributes, devoid of reliance on any structural information. The first-degree nodes introduce noise, as high-degree nodes consistently possess large node attribute values. Consequently, by eliminating the super node, the attributes of the remaining nodes are on a smaller scale, facilitating anomaly detection for models. Moreover, in the context of supervised GAD, there is a delicate equilibrium to be struck between label information and noise emanating from other nodes passing through the super node. Determining which of the two factors has gained the upper hand proves challenging. Therefore, the observed patterns with and without the super node vary among GAD models.

6 Conclusion

To understand the influence of nodes with extremely high degrees in graph anomaly detection, we present a novel graph anomaly dataset named NFTGraph, derived from the transaction data of ERC1155-NFT on the Ethereum blockchain. This dataset exhibits a remarkable disparity in node degrees. Our experimental endeavors on this dataset yield valuable insights, probing and comprehending the ramifications for unsupervised and GNN-based anomaly detection models. These findings contribute to the progress of graph anomaly detection.

References

1. Erc1155-nftgraph's data and code github (2023), https://github.com/AnonymousDataCodeHub/.
2. Bandyopadhyay, S., Vivek, S.V., Murty, M.: Outlier resistant unsupervised deep architectures for attributed network embedding. In: inproceedings of the 13th international conference on web search and data mining. pp. 25–33 (2020)

3. Chai, Z., You, S., Yang, Y., Pu, S., Xu, J., Cai, H., Jiang, W.: Can abnormality be detected by graph neural networks. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI), Vienna, Austria. pp. 23–29 (2022)
4. Chen, L., Peng, J., Liu, Y., Li, J., Xie, F., Zheng, Z.: Phishing scams detection in ethereum transaction network. ACM Transactions on Internet Technology (TOIT) **21**(1), 1–16 (2020)
5. Chen, W., Wu, J., Zheng, Z., Chen, C., Zhou, Y.: Market manipulation of bitcoin: Evidence from mining the mt. gox transaction network. In: IEEE conference on computer communications (INFOCOM). pp. 964–972. IEEE (2019)
6. Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P., Zhou, Y.: Detecting ponzi schemes on ethereum: Towards healthier blockchain technology. In: inproceedings of the world wide web conference. pp. 1409–1418 (2018)
7. Chen, Z., Liu, B., Wang, M., Dai, P., Lv, J., Bo, L.: Generative adversarial attributed network anomaly detection. In: Inproceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1989–1992 (2020)
8. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems **29** (2016)
9. Ding, K., Li, J., Bhanushali, R., Liu, H.: Deep anomaly detection on attributed networks. In: Inproceedings of the 2019 SIAM International Conference on Data Mining. pp. 594–602. SIAM (2019)
10. Etherscan api document (2015), https://docs.etherscan.io.
11. Etherscan website (2015), https://etherscan.io.
12. Fan, H., Zhang, F., Li, Z.: Anomalydae: Dual autoencoder for anomaly detection on attributed networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 5685–5689. IEEE (2020)
13. Gao, Y., Wang, X., He, X., Liu, Z., Feng, H., Zhang, Y.: Addressing heterophily in graph anomaly detection: A perspective of graph spectrum. In: Proceedings of the ACM Web Conference 2023. pp. 1528–1538 (2023)
14. Giles, C.L., Bollacker, K.D., Lawrence, S.: Citeseer: An automatic citation indexing system. In: Inproceedings of the third ACM conference on Digital libraries. pp. 89–98 (1998)
15. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)
16. He, M., Wei, Z., Xu, H., et al.: Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. Advances in Neural Information Processing Systems **34**, 14239–14251 (2021)
17. Huang, Y., Wang, L., Zhang, F., Lin, X.: Unsupervised graph outlier detection: Problem revisit, new insight, and superior method. In: 2023 IEEE 39th International Conference on Data Engineering (ICDE). pp. 2565–2578 (2023). https://doi.org/10.1109/ICDE55515.2023.00197
18. Kang, J., Zhu, Y., Xia, Y., Luo, J., Tong, H.: Rawlsgcn: Towards rawlsian difference principle on graph convolutional network. In: Proceedings of the ACM Web Conference 2022. pp. 1214–1225 (2022)
19. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
20. Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016)

21. Kojaku, S., Yoon, J., Constantino, I., Ahn, Y.Y.: Residual2vec: Debiasing graph embedding with random graphs. Advances in Neural Information Processing Systems **34**, 24150–24163 (2021)
22. Kumar, S., Zhang, X., Leskovec, J.: Predicting dynamic embedding trajectory in temporal interaction networks. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 1269–1278 (2019)
23. Li, A., Qin, Z., Liu, R., Yang, Y., Li, D.: Spam review detection with graph convolutional networks. p. 2703–2711. CIKM '19 (2019)
24. Liu, Y., Ao, X., Qin, Z., Chi, J., Feng, J., Yang, H., He, Q.: Pick and choose: a gnn-based imbalanced learning approach for fraud detection. In: Proceedings of the web conference 2021. pp. 3168–3177 (2021)
25. Liu, Z., Nguyen, T.K., Fang, Y.: Tail-gnn: Tail-node graph neural networks. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 1109–1119 (2021)
26. LIU, Z., NGUYEN, T.K., FANG, Y.: On generalized degree fairness in graph neural networks. In: Proceedings of the 37th AAAI Conference on Artificial Intelligence, Washington, USA. pp. 7–14 (2023)
27. McAuley, J.J., Leskovec, J.: From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In: Inproceedings of the 22nd international conference on World Wide Web. pp. 897–908 (2013)
28. Platonov, O., Kuznedelev, D., Diskin, M., Babenko, A., Prokhorenkova, L.: A critical look at the evaluation of gnns under heterophily: are we really making progress? arXiv preprint arXiv:2302.11640 (2023)
29. Rayana, S., Akoglu, L.: Collective opinion spam detection: Bridging review networks and metadata. In: Inproceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. pp. 985–994 (2015)
30. Sakurada, M., Yairi, T.: Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Inproceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis. pp. 4–11 (2014)
31. Shi, Y., Huang, Z., Feng, S., Zhong, H., Wang, W., Sun, Y.: Masked label prediction: Unified message passing model for semi-supervised classification. arXiv preprint arXiv:2009.03509 (2020)
32. Tan, Y., Wu, Z., Liu, J., Wu, J., Zheng, Z., Chen, T.: Bubble or not: Measurements, analyses, and findings on the ethereum erc721 and erc1155 non-fungible token ecosystem. arXiv preprint arXiv:2301.01991 (2023)
33. Tang, J., Liao, R.: Graph neural networks for node classification. Graph Neural Networks: Foundations, Frontiers, and Applications pp. 41–61 (2022)
34. Tang, J., Li, J., Gao, Z., Li, J.: Rethinking graph neural networks for anomaly detection. In: International Conference on Machine Learning. pp. 21076–21089 (2022)
35. Tang, X., Yao, H., Sun, Y., Wang, Y., Tang, J., Aggarwal, C., Mitra, P., Wang, S.: Investigating and mitigating degree-related biases in graph convoltuional networks. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1435–1444 (2020)
36. Tokenview website (2017), https://tokenview.io/.
37. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
38. Weber, M., Domeniconi, G., Chen, J., Weidele, D.K.I., Bellei, C., Robinson, T., Leiserson, C.E.: Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. arXiv preprint arXiv:1908.02591 (2019)

39. Wu, J., He, J., Xu, J.: Demo-net: Degree-specific graph neural networks for node and graph classification. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 406–415 (2019)
40. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
41. Xu, Z., Huang, X., Zhao, Y., Dong, Y., Li, J.: Contrastive attributed network anomaly detection with data augmentation. In: Advances in Knowledge Discovery and Data Mining: 26th Pacific-Asia Conference (PAKDD). pp. 444–457 (2022)
42. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. Advances in neural information processing systems **31** (2018)
43. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: Inproceedings of the AAAI conference on artificial intelligence. vol. 32 (2018)
44. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: Current limitations and effective designs. Advances in neural information processing systems **33**, 7793–7804 (2020)

A Appendix

A.1 Raw data

Figure A.1 is a snapshot of NFT transaction details on the TokenView website. A NFT transaction typically includes several important fields: *TxHash, Timestamp, From, To, Tokens, Transferred Amount, Value* and *TxFee*.

A.2 Ethics and Privacy

Every individual possesses the capacity to effortlessly access the very same raw data as NFTGraph through a node installation, emblematic of the synchronization of blockchain clients. Furthermore, a user-friendly alternative exists in Ethereum-etl, which facilitates the conversion of blockchain data into formats such as CSVs and relational databases. It is notable that `crypto_ethereum` database in Google Cloud `bigquery-public-data` utilizes Ethereum-etl and comprises processed blockchain data, undergoing continuous updates. Hence, it is paramount to underscore that all transaction data is accessible to the public, and our actions do not pose any potential harm in this regard.

Furthermore, the account addresses remain transparent, and if someone seeks information about a specific address on the Etherscan, it is readily accessible. However, it's crucial to recognize that all transaction details are tethered to transaction hashes, and the attributes of users are interlinked with their account addresses, but a direct connection with real-world entities is absent. Some of these addresses may be associated with real-world entities, but solely because those entities openly share them. Consequently, NFTGraph does not exacerbate the disclosure of individuals' privacy.

A.3 Dataset Versatility and Benchmarks

In addition to graph anomaly detection, in order to contribute to other graph learning areas, such as temporal link prediction, dynamic graph neural networks, etc., we also develop some other rich features of this dataset, and conduct a large number of benchmark experiments on the dataset to obtain meaningful insights. Due to page limits, these can be found in the github.

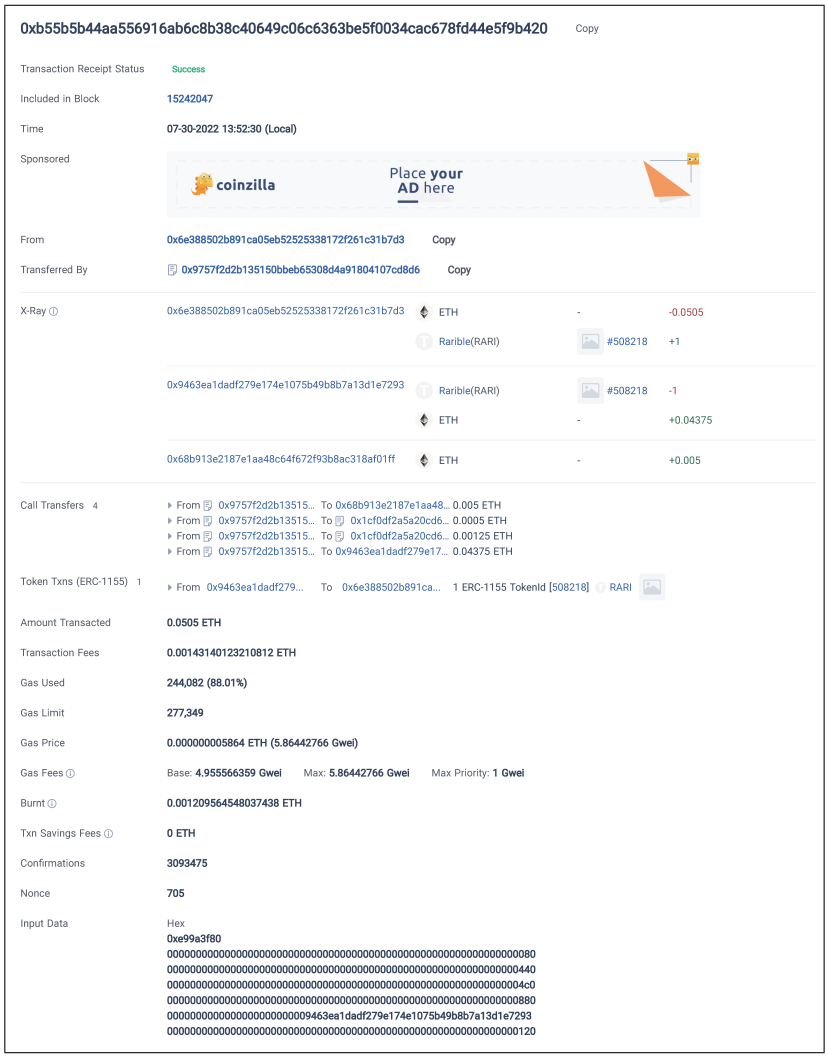

Fig. A.1. A typical NFT transaction on the Ethereum blockchain.