Truncated Affinity Maximization: One-class Homophily Modeling for Graph Anomaly Detection

Hezhe Qiao, Guansong Pang∗
School of Computing and Information Systems, Singapore Management University
hezheqiao.2022@phdcs.smu.edu.sg, gspang@smu.edu.sg

Abstract
We reveal a one-class homophily phenomenon, which is one prevalent property we find empirically in real-world graph anomaly detection (GAD) datasets, i.e., normal nodes tend to have strong connection/affinity with each other, while the homophily in abnormal nodes is significantly weaker than normal nodes. However, this anomaly-discriminative property is ignored by existing GAD methods that are typically built using a conventional anomaly detection objective, such as data reconstruction. In this work, we explore this property to introduce a novel unsupervised anomaly scoring measure for GAD – local node affinity – that assigns a larger anomaly score to nodes that are less affiliated with their neighbors, with the affinity defined as similarity on node attributes/representations. We further propose Truncated Affinity Maximization (TAM) that learns tailored node representations for our anomaly measure by maximizing the local affinity of nodes to their neighbors. Optimizing on the original graph structure can be biased by non-homophily edges (i.e., edges connecting normal and abnormal nodes). Thus, TAM is instead optimized on truncated graphs where non-homophily edges are removed iteratively to mitigate this bias. The learned representations result in significantly stronger local affinity for normal nodes than abnormal nodes. Extensive empirical results on 10 real-world GAD datasets show that TAM substantially outperforms seven competing models, achieving over 10% increase in AUROC/AUPRC compared to the best contenders on challenging datasets. Our code is available at https://github.com/mala-lab/TAM-master/.

1 Introduction
Graph anomaly detection (GAD) aims to identify abnormal nodes that are different from the majority of the nodes in a graph. It has attracted great research interest in recent years due to its broad real-world applications, e.g., detection of abusive reviews or malicious/fraudulent users [9, 16, 32, 37, 57]. Since graph data is non-Euclidean with diverse graph structure and node attributes, it is challenging to effectively model the underlying normal patterns and detect abnormal nodes in different graphs. To address this challenge, graph neural networks (GNNs) have been widely used for GAD. The GNN-based methods are often built using a data reconstruction [8, 11, 31, 59, 65] or self-supervised learning [15, 17, 28, 54, 63] objective. The data reconstruction approaches focus on learning node representations for GAD by minimizing the errors of reconstructing both node attributes and graph structure, while the self-supervised approaches focus on designing a proxy task that is related to anomaly detection, such as prediction of neighbor hops [17] and prediction of the relationship between a node and a subgraph [28], to learn the node representations for anomaly detection.

These approaches, however, ignore one prevalent anomaly-discriminative property we find empirically in real-world GAD datasets, namely one-class homophily, i.e., normal nodes tend to have strong connection/affinity with each other, while the homophily in abnormal nodes is significantly weaker

*Corresponding author: G. Pang

than normal nodes. This phenomenon can be observed in datasets with synthetic/real anomalies, as shown in Fig. 1(a) (see App. A for results on more datasets). The abnormal nodes do not exhibit homophily relations to each other mainly because abnormal behaviors are unbounded and can be drawn from different distributions.

Motivated by the one-class homophily property, we introduce a novel unsupervised anomaly scoring measure for GAD – local node affinity – that assigns a larger anomaly score to nodes that are less affiliated with their neighbors. Since we do not have access to class labels, we define the affinity in terms of similarity on node attributes/representations to capture the homophily relations within the normal class. Nodes having strong local affinity are the nodes that are connected to many nodes of similar attributes, and those nodes are considered more likely to be normal nodes. One challenge of using this anomaly measure is that some abnormal nodes can also be connected to nodes of similar abnormal behaviors. A straightforward solution to this problem is to apply our anomaly measure in a node representation space learned by off-the-shelf popular representation learning objectives like DGI [51], but the representations of normal and abnormal nodes can become similar due to the presence of non-homophily edges (i.e., edges connecting normal and abnormal nodes) that homogenize the normal and abnormal node representations in GNN message passing. To address this issue, we further propose Truncated Affinity Maximization (TAM) that learns tailored node representations for our anomaly scoring measure by maximizing the local node affinity to their neighbors, with the non-homophily edges being truncated to avoid the over-smooth representation issue. The learned representations result in significantly stronger local affinity for normal nodes than abnormal nodes. As shown in Fig. 1(b), it is difficult for using the local node affinity to distinguish normal and abnormal nodes on raw node attributes and DGI-based node representations, whereas the TAM-based node representation space offers well-separable local affinity results between normal and abnormal nodes. In summary, this work makes the following main contributions:

- We, for the first time, empirically reveal the one-class homophily phenomenon that provides an anomaly-discriminative property for GAD. Motivated by this property, we introduce a novel unsupervised anomaly scoring measure, local node affinity (Sec. 3.2).
- We then introduce Truncated Affinity Maximization (TAM) that learns tailored node representations for the proposed anomaly measure. TAM makes full use of the one-class homophily to learn expressive normal representations by maximizing local node affinity on truncated graphs, offering discriminative local affinity scores for accurate GAD.
- We further introduce two novel components (Sec. 3.3), namely Local Affinity Maximization-based graph neural networks (LAMNet for short) and Normal Structure-preserved Graph Truncation (NSGT), to implement TAM. Empirical results on six real-world GAD datasets show that our TAM model substantially outperforms seven competing models.

2 Related Work

Numerous anomaly detection methods have been introduced, including both shallow and deep approaches [5, 37], but most of them are focused on non-graph data. There have been many studies on GAD exclusively. Most of previous work use shallow methods [2], such as Radar [22], AMEN [43],
and ANOMALOUS [41]. Matrix decomposition and residual analysis are commonly used in these methods, whose performance is often bottlenecked due to the lack of representation power to capture the rich semantics of the graph data and to handle high-dimensional node attributes and/or sparse graph structures.

GNN-based GAD methods have shown substantially better detection performance in recent years [32]. Although some methods are focused on a supervised setting, such as CARE-GNN [10], PCGNN [27], Fraudre [58], BWGN [47], and GHRN [12], most of them are unsupervised methods. These unsupervised methods can be generally categorized into two groups: data reconstruction-based and self-supervised-based approach. Below we discuss these most related methods in detail.

Data Reconstruction-based Approach. As one of the most popular methods, graph auto-encoder (GAE) is employed by many researchers to learn the distribution of normal samples for GAD. Ding et al. [8] propose DOMINANT, which reconstructs the graph structure and node attributes by GAE. The anomaly score is defined as the reconstruction error from both node attributes and its structure. It is often difficult for GAE to learn discriminative representations because it can overfit the given graph data. Several mechanisms, including attention, sampling, edge removing, and meta-edge choosing [9, 27, 30, 45], are designed to alleviate this issue during neighborhood aggregation. Some recent variants like ComGA [31] and AnomalyDAE [11] incorporate an attention mechanism to improve the reconstruction. Other methods like SpaceAE [24] and ResGCN [39] aim to differentiate normal and abnormal nodes that do not have significant deviation, e.g., by exploring residual information between nodes. In general, reconstructing the structure of a node based on the similarities to its neighbors is related to but different from local node affinity in TAM (see Sec. 3.2), and GAE and TAM are also learned by a very different objective (see Sec. 3.3). Being GNN-based approaches, both the reconstruction-based approaches and our approach TAM rely on the node’s neighborhood information to obtain the anomaly scores, but we explicitly define an anomaly measure from a new perspective, i.e., local node affinity. This offers a fundamentally different approach for GAD.

Self-supervised Approach. Although data reconstruction methods can also be considered as a self-supervised approach, here we focus on non-reconstruction pre-text tasks for GAD. One such popular method is a proxy classification or contrastive learning task [18, 53, 65]. Liu et al. [28] propose CoLA, which combines contrastive learning and sub-graph extraction to perform self-supervised GAD. Based on CoLA, SL-GAD is proposed to develop generative attribute regression and multi-view contrastive learning [63]. There are some methods that leverage some auxiliary information such degree, symmetric and hop to design the self-supervised task [6, 17, 23, 42, 56, 61]. For example, Huang et al. propose the method HCM-A [17] to utilize hop count prediction for GAD. Although some self-supervised methods construct the classification model on the relationship between the node and the contextual subgraph, this group of methods is not related to local node affinity directly, and its performance heavily depends on how the pre-text task is related to anomaly detection. Similar to the self-supervised approaches, the optimization of TAM also relies on an unsupervised objective. However, the self-supervised approaches require the use of some pre-text tasks like surrogate contrastive learning or classification tasks to learn the feature representations for anomaly detection. By contrast, the optimization of TAM is directly driven by a new, plausible anomaly measure, which enables end-to-end optimization of an explicitly defined anomaly measure.

3 Method

3.1 The Proposed TAM Approach

Problem Statement. We tackle unsupervised anomaly detection on an attributed graph. Specifically, let $G = (V, E, X)$ be an attributed graph, where $V = \{ v_1, \cdots, v_N \}$ denotes its node set, $E \subseteq V \times V$ with $e \in E$ is the edge set, $\delta_{vij} = 1$ represents there is a connection between node $v_i$ and $v_j$, $X \in \mathbb{R}^{N \times M}$ denotes the matrix of node attributes and $x_i \in \mathbb{R}^M$ is the attribute vector of $v_i$, and $A \in \{0, 1\}^{N \times N}$ is the adjacency matrix of $G$ with $A_{ij} = 1$ iff $(v_i, v_j) \in E$, then GAD aims to learn an anomaly scoring function $f : V \rightarrow \mathbb{R}$, such that $f(v) < f(v')$, $\forall v \in V_n, v' \in V_a$, where $V_n$ and $V_a$ denote the set of normal and abnormal nodes, respectively. Per the nature of anomaly, it is typically assumed that $|V_n| \gg |V_a|$. But in unsupervised GAD we do not have access to the class labels of the nodes during training. In this work, our problem is to learn an unsupervised local node affinity-based anomaly scoring function $f$. 


Our Proposed Framework. Motivated by the one-class homophily phenomenon, we introduce a local node affinity-based anomaly score measure. Since it is difficult to capture and quantify the one-class homophily in the raw attribute space and the generic node representation space, we introduce a truncated affinity maximization (TAM) approach to learn a tailored representation space where the local node affinity can well distinguish normal and abnormal nodes based on the one-class homophily property. As shown in Fig. 2, TAM consists of two novel components, namely local affinity maximization-based graph neural networks (LAMNet) and normal structure-preserved graph truncation (NSGT). LAMNet trains a graph neural network using a local affinity maximization objective on an iteratively truncated graph structure yielded by NSGT.

NSGT is designed in a way through which we preserve the homophily edges while eliminating non-homophily edges iteratively. The message passing in LAMNet is then performed using the truncated adjacency matrix rather than the original one. In doing so, we reinforce the strong affinity among nodes with homophily relations to their neighbors (e.g., normal nodes), avoiding the potential bias caused by non-homophily edges (e.g., connections to abnormal nodes). The output node affinity in LAMNet is used to define anomaly score, i.e., the weaker the local affinity is, the more likely the node is an abnormal node. There exists some randomness in our NSGT-based graph truncation. We utilize those randomness for more effective GAD by building a bagging ensemble of TAM.

3.2 Local Node Affinity as Anomaly Measure

As shown in Fig. 1, normal nodes have significantly stronger homophily relations with each other than the abnormal nodes. However, we do not have class label information to calculate the homophily of each node in unsupervised GAD. We instead utilize the local affinity of each node to its neighbors to exploit this one-class homophily property for unsupervised GAD. The local affinity can be defined as an averaged similarity to the neighboring nodes, and the anomaly score $f$ is opposite to the affinity:

$$h(v_i) = \frac{1}{|N(v_i)|} \sum_{v_j \in N(v_i)} \text{sim}(x_i, x_j) ; \quad f(v_i) = -h(v_i), \quad (1)$$

where $N(v_i)$ is the neighbor set of node $v_i$ and $\text{sim}(x_i, x_j) = \frac{x_i^T x_j}{\|x_i\|\|x_j\|}$ measures the similarity of a node pair $(v_i, v_j)$. The local node affinity $h(v_i)$ is a normal score: the larger the affinity is, the stronger homophily the node has w.r.t. its neighboring nodes based on node attributes, and thus, the more likely the node is a normal node.

The measure in Eq. (1) provides a new perspective to quantify the normality/abnormality of nodes, enabling a much simpler anomaly scoring than existing popular measures such as the reconstruction error $f(v_i) = (1 - \alpha)\|\hat{a}_i - \hat{\hat{a}}_i\|_2 + \alpha\|\hat{x}_i - \hat{\hat{x}}_i\|_2$, where $\hat{a}_i$ denotes the neighborhood structure of $v_i$, $\hat{\hat{a}}_i$ and $\hat{x}_i$ are the reconstructed structure and attributes for node $v_i$, and $\alpha$ is a hyperparameter.
Our anomaly measure also provides a new perspective to learn tailored node representations for unsupervised GAD. Instead of minimizing the commonly used data reconstruction errors, based on the one-class homophily, we can learn anomaly-discriminative node representations by maximizing the local node affinity. Our TAM approach is designed to achieve this goal.

### 3.3 TAM: Truncated Affinity Maximization

The local node affinity may not work well in the raw node attribute space since (i) there can be many irrelevant attributes in the original data space and (ii) some abnormal nodes can also be connected to nodes of similar attributes. To tackle this issue, TAM is proposed to learn optimal node representations that maximize the local affinity of nodes that have strong homophily relations with their neighbors in terms of node attributes. Due to the overwhelming presence of normal nodes in a graph, the TAM-based learned node representation space is optimized for normal nodes, enabling stronger local affinity for the normal nodes than the abnormal ones. The TAM-based anomaly scoring using local node affinity can be defined as:

\[
f_{TAM}(v_i; \Theta, A, X) = -\frac{1}{|N(v_i)|} \sum_{v_j \in N(v_i)} \text{sim}(h_i, h_j),
\]

where \(h_i = \psi(v_i; \Theta, A, X)\) is a GNN-based node representation of \(v_i\) learned by a mapping function \(\psi\) parameterized by \(\Theta\) in TAM. Below we introduce how we learn the \(\psi\) function via the two components of TAM, LAMNet and NSGT.

#### Local Affinity Maximization Networks (LAMNet).

LAMNet aims to learn a GNN-based mapping function \(\psi\) that maximizes the affinity of nodes with homophily relations to their neighbors, while keeping the affinity of nodes without homophily edges are weak. Specifically, the projection from the graph nodes onto new representations using \(\ell\) GNN layers can be generally written as

\[
H^{(\ell)} = \text{GNN} \left( A, H^{(\ell-1)}; W^{(\ell)} \right),
\]

where \(H^{(\ell)} \in \mathbb{R}^{N \times h_{(\ell)}}\) and \(H^{(\ell-1)} \in \mathbb{R}^{N \times h_{(\ell-1)}}\) are the \(h_{(\ell)}\)-dimensional and \(h_{(\ell-1)}\)-dimensional representations of node \(v_i\) in the \(\ell\)-th layer and \((\ell-1)\)-th layer, respectively. In the first GNN layer, i.e., when \(\ell = 1\), the input \(H^{(0)}\) is set to the raw attribute matrix \(X\). \(W^{(\ell)}\) are the weight parameters of \((\ell)\)-th layer. For \(\text{GNN}(\cdot)\), multiple types of GNNs can be used [50, 55]. In this work, we employ a GCN (graph convolutional network) [20] due to its high efficiency. Then \(H^{(1)}\) can be obtained via

\[
H^{(1)} = \phi \left( D^{-\frac{1}{2}} AD^{-\frac{1}{2}} H^{(0)} W^{(1)} \right),
\]

where \(D = \text{diag}(D_i) = \sum_j A_{ij}\) is the degree matrix for the graph \(G\), and \(\phi(\cdot)\) is an activation function. Let \(H^{(\ell)} = \{h_1, h_2, \ldots, h_N\}\) be the node representations of the last GCN layer, then the mapping function \(\psi\) is a sequential mapping of graph convolutions as in Eq. (4), with \(\Theta = \{W^1, W^2, \ldots, W^{(\ell)}\}\) be the parameter set in our LAMNet. The following objective can then be used to optimize \(\psi\):

\[
\min_{\Theta} \sum_{v_i \in V} \left( f_{TAM}(v_i; \Theta, A, X) + \lambda \frac{1}{|N(v_i)|} \sum_{v_k \in V \setminus N(v_i)} \text{sim}(h_i, h_k) \right),
\]

where the first term is equivalent to maximizing the local affinity of each node based on the learned node representations, while the second term is a regularization term, and \(\lambda\) is a regularization hyperparameter. The regularization term adds a constraint that the representation of each node should be dissimilar from that of non-adjacent nodes to enforce that the representations of non-local nodes are distinguishable, while maximizing the similarity of the representations of local nodes.

The optimization using Eq. (5) can be largely biased by non-homophily edges. Below we introduce the NSGT component that helps overcome this issue.

#### Normal Structure-preserved Graph Truncation (NSGT).

LAMNet is driven by the one-class homophily property, but its objective and graph convolution operations can be biased by the presence of non-homophily edges, i.e., edges that connect normal and abnormal nodes. The NSGT component
is designed to remove these non-homophily edges, yielding a truncated adjacency matrix \( \tilde{\mathbf{A}} \) with homophily edge-based normal graph structure. LAMNet is then performed using the truncated adjacency matrix \( \tilde{\mathbf{A}} \) rather than the original adjacency matrix \( \mathbf{A} \).

Since homophily edges (i.e., edges connecting normal nodes) connect nodes of similar attributes, the distance between the nodes of homophily edges is often substantially smaller than that of non-homophily edges. But there can also exist homophily edges that connect dissimilar normal nodes. These can be observed in GAD datasets, as shown in Fig. 3(a-b) for datasets with synthetic/real anomalies. Motivated by this, NSGT takes a probabilistic approach and performs the graph truncation as follows: for a given edge \( e_{ij} = 1 \), it is considered as a non-homophily edge and removed (i.e. \( e_{ij} = 0 \)) if and only if the distance between node \( v_i \) and node \( v_j \) is sufficiently large w.r.t. the neighbor sets of both \( v_i \) and \( v_j \), \( N(v_i) \) and \( N(v_j) \). Formally, NSGT truncates the graph by

\[
e_{ij} \leftarrow 0 \text{ iff } d_{ij} > r_i \& d_{ij} > r_j, \forall e_{ij} = 1,
\]

where \( d_{ij} \) is a Euclidean distance between \( v_i \) and \( v_j \) based on node attributes, \( r_i \) is a randomly selected value from the range \([d_{\text{mean}}, d_{i,\text{max}}]\) where \( d_{\text{mean}} = \frac{1}{m} \sum_{(v_i,v_j) \in E} d_{ij} \) is the mean distance of graph, where \( m \) is the number of non-zero elements in the adjacent matrix, and \( d_{i,\text{max}} \) is the maximum distances in \( \{d_{ik}, v_k \in N(v_i)\} \). Similarly, \( r_j \) is randomly sampled from the range \([d_{\text{mean}}, d_{j,\text{max}}]\).

Theoretically, the probability of \( r_i < d_{ij} \) can be defined as

\[
p(r_i < d_{ij}) = \frac{\max(d_{ij} - d_{\text{mean}}, 0)}{d_{i,\text{max}} - d_{\text{mean}}},
\]

Note that this probability is premised on \( d_{i,\text{max}} > d_{\text{mean}} \), and we set \( p(r_i < d_{ij}) = 0 \) if \( d_{i,\text{max}} \leq d_{\text{mean}} \). Then the probability of \( e_{ij} \) being removed during the truncation process is as follows

\[
p(E \setminus e_{ij}) = p(r_i < d_{ij}) p(r_j < d_{ij}),
\]

As shown in Fig. 2(b), if \( e_{ij} \) is a homophily edge, we would have a small \( d_{ij} \), which results in small \( p(r_i < d_{ij}) \) and \( p(r_j < d_{ij}) \), and thus, \( p(E \setminus e_{ij}) \) is also small. By contrast, a non-homophily edge would result in a large \( d_{ij} \), and ultimately a large \( p(E \setminus e_{ij}) \). Therefore, NSGT can help eliminate non-homophily edges with a high probability, while preserving the genuine homophily graph structure.

Considering the large difference in the range \([d_{\text{mean}}, d_{i,\text{max}}]\) for each node and the randomness in truncation, NSGT performs a sequentially iterative truncation rather than a single-pass truncation. In each iteration it randomly removes some non-homophily edges with probability proportional to the distance between their associated nodes, and then it updates the range \([d_{\text{mean}}, d_{i,\text{max}}]\) for nodes that have edges being removed. Thus, for \( K \) iterations on a graph \( G \), it would produce a set of \( K \) sequentially truncated graphs with the corresponding truncated adjacency matrices \( \mathcal{A} = \{\hat{\mathbf{A}}_1, \hat{\mathbf{A}}_2, \cdots, \hat{\mathbf{A}}_K\} \). Note that since the graph is sequentially truncated, we have \( E_{i+1} \subset E_i \), where \( E_i \) is the edge set left after the \( i \)-th iteration. As shown in Fig. 3(c-d), such a sequentially iterative truncation helps largely increase the homophily of the normal nodes, while at the same time effectively removing non-homophily edges gradually.

**Training.** To detect anomalies in different graph truncation scales, we train a LAMNet on each of the \( K \) sequentially truncated adjacency matrices in \( \mathcal{A} \), resulting in \( K \) LAMNets parameterized by \( \{\Theta_1, \Theta_2, \cdots, \Theta_K\} \) for various truncation depths. Particularly, for each LAMNet, instead of using \( \mathbf{A} \), its graph convolutions across all GCN layers are performed on the truncated adjacency matrix \( \hat{\mathbf{A}}_k \) as follows to mitigate the biases caused by the

![Figure 3: (a) and (b) are respectively the Euclidean distance statistics of the homophily (N-N) edges that connect normal nodes and the non-homophily (N-A) edges that connect normal and abnormal nodes on BlogCatalog and Amazon. (c) Homophily of normal nodes vs. (d) the number of non-homophily edges with increasing truncation iterations/depths.](image-url)
non-homophily edges in message passing:

$$H^{(\ell)} = \phi \left( D^{-\frac{1}{2}} \tilde{A} D^{-\frac{1}{2}} H^{(\ell-1)} W^{(\ell-1)} \right).$$  \hspace{1cm} (9)

In doing so, we complete the training of a TAM-based base model, consisting of $K$ LAMNets.

Further, being a probabilistic approach, NSGT has some randomnesses in the graph truncation, and so does the LAMNets. To make use of those randomness, we build an ensemble of TAM models with a size of $T$, as shown in Fig. 2(a). That is, for a graph $G$, we perform NSGT $T$ times independently, resulting in $T$ sets of the truncated adjacency matrix set $\{A_1, A_2, \cdots, A_T\}$, with each $A = \{\tilde{A}_1, \tilde{A}_2, \cdots, \tilde{A}_K\}$. We then train a LAMNet on each of these $T \times K$ adjacency matrices, obtaining an ensemble of $T$ TAM models (i.e., $T \times K$ LAMNets).

Note that in training the LAMNets, the local affinity in the first term in Eq. (5) is computed using the original adjacency matrix $A$. This is because we aim to utilize the primary graph structure for local affinity-based GAD; the graph truncation is designed to mitigate the graph convolution biases only.

**Inference.** During inference, we can obtain one local node affinity-based anomaly score per node from each LAMNet, and we aggregate the local affinity scores from all $T \times K$ LAMNets to compute an overall anomaly score as

$$\text{score} (v_i) = \frac{1}{T \times K} \sum_{t=1}^{T} \sum_{k=1}^{K} f_{TAM}(v_i; \Theta_{t,k}^{*}, A, X), \hspace{1cm} (10)$$

where $\Theta_{t,k}^{*}$ is the learned weight parameters for the LAMNet using the $k$-th truncated adjacency matrix in the $t$-th TAM model. The weaker the local node affinity in the learned representation spaces under various graph truncation scales, the larger the anomaly score the node $v_i$ has.

### 4 Experiments

**Datasets.** We conduct the experiments on six commonly-used publicly-available real-world GAD datasets from diverse online shopping services and social networks, and citation networks, including BlogCatalog [49], ACM [48], Amazon [10], Facebook [56], Reddit, and YelpChi [21]. The first two datasets contain two types of injected anomalies – contextual and structural anomalies [8, 34] – that are nodes with significantly deviated graph structure and node attributes respectively. The other four datasets contain real anomalies. Detailed information about the datasets can be found in App. B.

**Competing Methods and Performance Metrics.** TAM is compared with two state-of-the-art (SOTA) shallow methods – iForest [25] and ANOMALOUS [41] – and five SOTA GNN-based deep methods, including three self-supervised learning based methods – CoLA [28], SL-GAD [63], and HCM-A [17] – and two reconstruction-based methods – DOMINANT [8] and ComGA [31]. iForest works on the raw node attributes, while ANOMALOUS works on the raw node attributes and graph structure. The other methods learn new representation space for GAD.

Following [4, 38, 52, 64], two popular and complementary evaluation metrics for anomaly detection, Area Under the Receiver Operating Characteristic Curve (AUROC) and Area Under the precision-recall curve (AUPRC), are used. Higher AUROC/AUPRC indicates better performance. The reported AUROC and AUPRC results are averaged over 5 runs with different random seeds.

**Implementation Details.** TAM is implemented in Pytorch 1.6.0 with python 3.7 and all the experiments are run on an NVIDIA GeForce RTX 3090 24GB GPU. In TAM, each LAMNet is implemented by a two-layer GCN, and its weight parameters are optimized using Adam [19] optimizer with 500 epochs and a learning rate of $1e^{-5}$ by default. $T = 3$ and $K = 4$ are used for all datasets. Datasets with injected anomalies, such as BlogCatalog and ACM, require strong regularization, so $\lambda = 1$ is used by default; whereas $\lambda = 0$ is used for the four real-world datasets. Hyperparameter analysis w.r.t. $K$ is presented in Sec. 4.2. TAM can perform stably within a range of $T$ and $\lambda$ (see App. C.1 for detail). All the competing methods are based on their publicly-available official source code, and they are trained using their recommended optimization and hyperparameter settings in the original papers.
4.1 Main Results

Effectiveness on Diverse Real-world Datasets. The AUROC and AUPRC results on six real-world GAD datasets are reported in Tab. 1. TAM substantially outperforms all seven competing methods on all datasets except Reddit in both metrics, having maximally 9% AUROC and 12% AUPRC improvement over the best-competing methods on the challenge dataset Amazon; on Reddit, it ranks second and performs similarly well to the best contender CoLA. Further, existing methods perform very unstably across different datasets. For example, DOMINANT performs well on ACM but badly on the other datasets; CoLA works well on Reddit but it fails in the other datasets. Similar observations are found in a recent comprehensive comparative study [26]. By contrast, TAM can perform consistently well on these diverse datasets. This is mainly because i) the proposed one-class homophily is more pervasive than the GAD intuitions used by existing methods in different datasets, and ii) TAM offers an effective anomaly scoring function to utilize this anomaly-discriminative property for accurate GAD.

On Detecting Structural and Contextual Anomalies. We further examine the effectiveness of TAM on detecting two commonly-studied anomaly types, structural and contextual anomalies, with the results on BlogCatalog and ACM reported in Tab. 2, where the three best competing methods on the two datasets in Tab. 1 are used as baselines. Compared to contextual anomalies, it is significantly more challenging to detect structural anomalies, for which TAM outperforms all three methods in both AUROC and AUPRC. As for contextual anomalies, although TAM underperforms SL–GAD and ranks in second in AUROC, it obtains substantially better AUPRC than SL–GAD. Note that compared to AUROC that can be biased by low false positives and indicates overoptimistic performance, AUPRC is a more indicative measure focusing on the performance on the anomaly class exclusively [3, 38]. So, achieving the best AUPRC on all four cases of the two datasets demonstrates the superiority of TAM in precision and recall rates for both types of anomaly. These results also indicate that structural anomalies may have stronger local affinity than contextual anomalies, but both of which often have weaker local affinity than normal nodes.

TAM vs. Raw/Generic Node Representation Space. As discussed in Sec. 1, local node affinity requires a new node representation space that is learned for the affinity without being biased by non-homophily edges. We provide quantitative supporting results in Tab. 3, where TAM is compared with Raw Attribute (RA), and the spaces learned by DGI [51] and GMI [40]. It is clear that the representations learned by TAM significantly outperforms all three competing representation spaces on all six datasets.

Table 1: AUROC and AUPRC results on six real-world GAD datasets with injected/real anomalies. The best performance per row is boldfaced, with the second-best underlined.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Method</th>
<th>BlogCatalog</th>
<th>ACM</th>
<th>Amazon</th>
<th>Facebook</th>
<th>Reddit</th>
<th>YelpChi</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUROC</td>
<td>TAM (Ours)</td>
<td>0.7683±0.013</td>
<td>0.7038±0.024</td>
<td>0.7064±0.021</td>
<td>0.7064±0.021</td>
<td>0.7064±0.021</td>
<td>0.7064±0.021</td>
</tr>
<tr>
<td>AUPRC</td>
<td>TAM (Ours)</td>
<td>0.4120±0.004</td>
<td>0.4120±0.004</td>
<td>0.4120±0.004</td>
<td>0.4120±0.004</td>
<td>0.4120±0.004</td>
<td>0.4120±0.004</td>
</tr>
</tbody>
</table>
Computational Efficiency. The runtime (including both training and inference time) results are shown in Tab. 4. DOMINANT and ComGA are the simplest GNN-based methods, achieving the most efficient methods. Our method needs to perform multiple graph truncation and train multiple LAMNets, so it takes more time than these two reconstruction-based methods, but it runs much faster than the three recent SOTA models, HCM-A, CoLA, and SL-GAD, on most of the datasets. A detailed analysis is provided in App. C.2.

4.2 Ablation Study

Graph Truncation NSGT. Three alternative approaches to our graph truncation NSGT include: i) Raw Graph (RG) that directly performs affinity maximization on the original graph structure without any graph truncation, and ii) Edge Drop (ED) that randomly drops some edges (5% edges by default) [7]. iii) Similarity Cut (SC) (removing 5% least similar edges). We compare NSGT to these three approaches in the TAM framework in Tab. 5. It is clear that NSGT consistently and significantly outperforms both alternative approaches. RawGraph does not work well due to the optimization biases caused non-homophily edges. EdgeDrop is ineffective, which is even less effective than RawGraph, as it often removes homophily edges rather than the opposite due to the overwhelming presence of such edges in a graph. SimilarityCut also significantly underperforms our NSGT on nearly all cases. This is mainly because this variant would fail to take account of local affinity distribution of each node as being captured in NSGT. As a result, it could remove not only non-homophily edges but also homophily edges associated with normal nodes whose local affinity is not as strong as the other normal nodes, which would be the opposite to the objective of the optimization in TAM, leading to less effective detection performance.

Affinity Maximization Network LAMNet. The importance of LAMNet is examined by comparing it to its two variants, including Raw Truncated Affinity (RTA) that directly calculates the local affinity-based anomaly scores after NSGT (i.e., without involving LAMNet at all), and DOM that performs LAMNet but with our affinity maximization objective replaced by the popular graph reconstruction loss used in DOMINANT [8]. As shown by the comparison results reported in Tab. 6, LAMNet consistently and significantly outperforms both RTA and DOM, showing that LAMNet can make much better use of the truncated graphs. In addition, by working on our truncated graphs, DOM can substantially outperform its original version DOMINANT in Tab. 1 on most of the datasets. This indicates that the proposed one-class homophily property may be also exploited to improve existing GAD methods.

Anomaly Scoring. We compare the TAM anomaly scoring to its two variants: i) TAM-T that calculates the node affinity in Eq. (5) on the truncated graph structure rather than the primary graph structure as in TAM, and ii) Degree that directly uses the node degree after our graph truncation as anomaly score. As illustrated by the AUPRC results in Fig. 4(a), TAM consistently and significantly outperforms both variants. Degree obtains fairly good performance, which shows that our graph truncation results in structural changes that are

![Figure 4: (a) TAM vs. Degree and TAM-T. (b) TAM results w.r.t. graph truncation depth K.](image-url)
beneficial to GAD. TAM-T underperforms TAM since the local affinity based on the truncated graph structure is affected by randomness and uncertainty in the truncation, leading to unstable and suboptimal optimization. We also show in Fig. 4 (b) that aggregating the anomaly scores obtained from different truncation scales/depths helps largely improve the detection performance. Similar observations are found in the corresponding AUROC results (see App. C.3).

4.3 Performance on Large-scale Graphs

In order to demonstrate the effectiveness of TAM on the large-scale datasets, we conduct the experiments on the four large-scale datasets with a large set of nodes and edges, Amazon-all and YelpChi-all by treating the different relations as a single relation following [6], T-Finance [47] and OGB-Proteins [14]. The experimental results are shown in Tab. 7. Due to the increased number of nodes and edges, we set $K = 7$ for these datasets. TAM can perform consistently well on these large-scale datasets and outperforms four comparing methods, which provides further evidence about the effectiveness of our proposed one-class homophily and anomaly measure.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Method</th>
<th>Dataset</th>
<th>AUROC</th>
<th>AUPRC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Amazon-all YelpChi-all T-Finance OGB-Proteins</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUROC</td>
<td>DOMINANT</td>
<td>0.6937 0.5390 0.5380 0.7267</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ComGA</td>
<td>0.7154 0.5352 0.5542 0.7134</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CoLA</td>
<td>0.2614 0.4801 0.4829 0.7142</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SL-GAD</td>
<td>0.2728 0.5551 0.4648 0.7371</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAM</td>
<td>0.8476 0.5818 0.6175 0.7449</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUPRC</td>
<td>DOMINANT</td>
<td>0.1015 0.1638 0.0474 0.2217</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ComGA</td>
<td>0.1854 0.1658 0.0481 0.1554</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CoLA</td>
<td>0.0516 0.1361 0.0410 0.1349</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SL-GAD</td>
<td>0.0444 0.1711 0.0386 0.1771</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAM</td>
<td>0.4346 0.1886 0.0547 0.2173</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.4 Handling Camouflage Attributes

Fraudsters may adjust their behaviors to camouflage their malicious activities, which could hamper the performance of GNN-based methods. We evaluate the performance of TAM when there are camouflages in the raw attributes. Particularly, we replace 10%, 20%, 30% randomly sampled original attributes with camouflaged attributes, in which the feature/attribute value of the abnormal nodes is replaced (camouflaged) with the mean feature value of the normal nodes. The results are shown in the Tab. 8, which demonstrates that TAM is robust to a high level of camouflaged features and maintains its superiority over the SOTA models that work on the original features. The reason is that NSGT can still successfully remove some non-homophily edges in the local domain, allowing LAMNet to make full use of the one-class homophily and achieve good performance under the camouflage.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Method</th>
<th>Dataset</th>
<th>0%</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUROC</td>
<td>BlogCatalog</td>
<td>0.8218 0.8045 0.8022 0.7831</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACM</td>
<td>0.8078 0.8272 0.8608 0.8652</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Amazon</td>
<td>0.7906 0.7036 0.6954 0.6838</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Facebook</td>
<td>0.8144 0.8587 0.8804 0.8650</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reddit</td>
<td>0.6023 0.5998 0.5915 0.5876</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>YelpChi</td>
<td>0.5640 0.5447 0.5271 0.5301</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUPRC</td>
<td>BlogCatalog</td>
<td>0.4342 0.4114 0.4025 0.3951</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACM</td>
<td>0.5124 0.4986 0.4614 0.4822</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Amazon</td>
<td>0.4634 0.2902 0.2400 0.2324</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Facebook</td>
<td>0.2233 0.2225 0.2126 0.2015</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reddit</td>
<td>0.3446 0.3444 0.3414 0.3440</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>YelpChi</td>
<td>0.0778 0.0726 0.0701 0.0678</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5 Conclusion and Future Work

This paper reveals an important anomaly-discriminative property, the one-class homophily, in GAD datasets with either injected or real anomalies. We utilize this property to introduce a novel unsupervised GAD measure, local node affinity, and further introduce a truncated affinity maximization (TAM) approach that end-to-end optimizes the proposed anomaly measure on truncated adjacency matrix. Extensive experiments on 10 real-world GAD datasets show the superiority of TAM over seven SOTA detectors. We also show that the one-class homophily can be exploited to enhance the existing GAD methods.

Limitation and Future Work. TAM cannot directly handle primarily isolated nodes in a graph, though those isolated nodes are clearly abnormal. Additionally, like many GNN-based approaches, including GAD methods, TAM also requires a large memory to perform on graphs with a very large node/edge set. The one-class homophily may not hold for some datasets, such as datasets with strong heterophily relations/subgraphs of normal nodes [1, 29, 36, 46, 62, 66]. Our method would require some adaptations to work well on the dataset with strong heterophily or very large graphs, which are also left for future work.
Acknowledgments. We thank the anonymous reviewers for their valuable comments. The work is supported in part by the Singapore Ministry of Education Academic Research Fund Tier 1 grant (21SISSMU031).

References


[6] Bo Chen, Jing Zhang, Xiaokang Zhang, Yuxiao Dong, Jian Song, Peng Zhang, Kaibo Xu, Evgeny Kharlamov, and Jie Tang. Gccad: Graph contrastive learning for anomaly detection. IEEE Transactions on Knowledge and Data Engineering, 2022. 3, 10, 15


[15] Ling Huang, Ye Zhu, Yuefang Gao, Tuo Liu, Chao Chang, Caixing Liu, Yong Tang, and Chang-Dong Wang. Hybrid-order anomaly detection on attributed networks. IEEE Transactions on Knowledge and Data Engineering, 2021. 1


[32] Xiaoxiao Ma, Jia Wu, Shan Xue, Jian Yang, Chuan Zhou, Quan Z Sheng, Hui Xiong, and Leman Akoglu. A comprehensive survey on graph anomaly detection with deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.


[56] Zhiming Xu, Xiao Huang, Yue Zhao, Yushun Dong, and Jundong Li. Contrastive attributed network anomaly detection with data augmentation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 444–457. Springer, 2022. 3, 7, 15


A One-Class Homophily Phenomenon

Fig. 5 shows the one-class homophily phenomenon on the rest of four datasets, in which ACM is a dataset with injected anomalies while the other three datasets contain real anomalies. The results here are consistent with that in the main text.

Given the graph along with the ground truth labels, following [13, 33], the homophily and heterophily are defined as the ratio of the edges connecting the node with the same classes and different classes, respectively:

\[
\begin{align*}
X_{\text{hetero}}(v) &= \frac{1}{|N(v)|} |\{u : u \in N(v), y_u \neq y_v\}| \\
X_{\text{homo}}(v) &= \frac{1}{|N(v)|} |\{u : u \in N(v), y_u = y_v\}|
\end{align*}
\]

where \(y_v\) is the label to denote whether the node \(v\) is an anomaly or not. Note that one-class homophily may not always hold, or it is weak in some datasets. YelpChi which connects the reviews posted by the same user, e.g., YelpCh-RUR is an example of the latter case. As shown in Figure 5, the homophily distribution of normal and abnormal nodes is similar in YelpCh-RUR, whose pattern is different from the large distribution gap in the other three datasets. Since the normal nodes’ homophily is generally stronger than the abnormal nodes, we can still successfully remove the non-homophily edges that connect normal and abnormal nodes with a higher probability than the homophily edges, resulting in a truncated graph with stronger one-class homophily.

B Description of Datasets

We conduct the experiments on two real-world dataset with injected anomalies and four real-world with genuine anomalies in diverse online shopping services, social networks, and citation networks, including BlogCatalog [49], ACM [48], Amazon [10], Facebook [56], Reddit, YelpChi [21], as well as four large-scale graph datasets. The statistical information including the number of nodes, edge, the dimension of the feature, and the anomalies rate of the datasets can be found in Tab. 9.

Particularly, BlogCatalog is a social blog directory where users can follow each other. Each node represents a user, and each link indicates the following relationships between two users. The attributes of nodes are the tags that describe users and their blogs. ACM is a citation graph dataset where the nodes denote the published papers and the edge denotes the citations relationship between the papers. The attributes of each node are the content of the corresponding paper. BlogCatalog and ACM are popular GAD datasets where the anomalies are injected ones, including structural anomalies and contextual anomalies, which are created following the prior work [28]. Amazon is a graph dataset capturing the relations between users and product reviews. Following [10, 60], three different user-user graph datasets are derived from Amazon using different adjacency matrix construction approaches. In this work, we focus on the Amazon-UPU dataset that connects the users who give reviews to at least one same product. The users with less than 20% are treated as anomalies. Facebook [56] is a social network where users build relationships with others and share their same friends. Reddit is a network of forum posts from the social media Reddit, in which the user who has been banned from the platform is annotated as an anomaly. Their post texts were converted to the vector as their attribute. YelpChi includes hotel and restaurant reviews filtered (spam) and recommended (legitimate) by Yelp. Following [35, 44], three different graph datasets derived from Yelp using different connections in user, product review text, and time. In this work, we only use YelpChi-RUR which connects reviews posted by the same user. Note that considering it’s difficult to conduct an evaluation on the isolated nodes in the graph, they were removed before modeling.

Amazon-all and YelpChi-all [10] are two datasets by treating the different relations as a single relation following [6]. T-Finance [47] is a transaction network. The nodes represent the unique account and the edges represent there are records between two accounts. The attributes of nodes are the features related to registration days, logging activities, and interaction frequency. The anomalies are fraud, money laundering and online gambling which are annotated manually. OGB-Protein [14] is a biological network where node represents proteins and edge indicates the meaningful association between proteins.
Figure 5: Homophily distribution of normal nodes and abnormal nodes on the rest of eight datasets
Table 9: Key statistics of the datasets. The real-world datasets with Injected/Real anomalies (I/R).

<table>
<thead>
<tr>
<th>Data set</th>
<th>Type</th>
<th>R/I</th>
<th>Nodes</th>
<th>Edges</th>
<th>Attributes</th>
<th>Anomalies (Rate)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlogCatalog</td>
<td>Social Networks</td>
<td>I</td>
<td>5,196</td>
<td>171,743</td>
<td>8,189</td>
<td>300 (5.77%)</td>
</tr>
<tr>
<td>ACM</td>
<td>Citation Networks</td>
<td>I</td>
<td>16,484</td>
<td>71,980</td>
<td>8,337</td>
<td>597 (3.63%)</td>
</tr>
<tr>
<td>Amazon</td>
<td>Co-review</td>
<td>R</td>
<td>10244</td>
<td>175,608</td>
<td>25</td>
<td>693 (6.66%)</td>
</tr>
<tr>
<td>Facebook</td>
<td>Social Networks</td>
<td>R</td>
<td>4,039</td>
<td>88,234</td>
<td>576</td>
<td>27 (0.67%)</td>
</tr>
<tr>
<td>Reddit</td>
<td>Social Networks</td>
<td>R</td>
<td>10,984</td>
<td>175,608</td>
<td>64</td>
<td>366 (3.33%)</td>
</tr>
<tr>
<td>YelpChi</td>
<td>Co-review</td>
<td>R</td>
<td>24,741</td>
<td>49,315</td>
<td>32</td>
<td>1,217 (4.91%)</td>
</tr>
<tr>
<td>Amazon-all</td>
<td>Co-review</td>
<td>R</td>
<td>11,944</td>
<td>4,398,392</td>
<td>25</td>
<td>821 (6.87%)</td>
</tr>
<tr>
<td>YelpChi-all</td>
<td>Co-review</td>
<td>R</td>
<td>45,941</td>
<td>3,846,979</td>
<td>32</td>
<td>6,674 (14.52%)</td>
</tr>
<tr>
<td>T-Finance</td>
<td>Transaction Record</td>
<td>R</td>
<td>39,357</td>
<td>21,222,543</td>
<td>10</td>
<td>1,803 (4.58%)</td>
</tr>
<tr>
<td>OGB-Protein</td>
<td>Biology Network</td>
<td>I</td>
<td>132,534</td>
<td>39,561,252</td>
<td>8</td>
<td>6000 (4.5%)</td>
</tr>
</tbody>
</table>

Figure 6: AUROC and AUPRC results w.r.t. ensemble parameter $T$

C Additional Experimental Results

C.1 Hyperparameter Analysis

This section analyzes the sensitivity of TAM w.r.t. two key hyperparameters, including the regularization hyperparameter $\lambda$ and the ensemble parameters $T$. The results on $\lambda$ and $T$ are shown in Fig. 6 and Fig. 7, respectively.

**Ensemble Hyperparameter $T$.** As shown in Fig. 6 with increasing $T$, TAM generally performs better and becomes stable around $T \approx 4$. This is mainly because the use of more ensemble models on truncated graphs reduces the impact of the randomness of truncation and increases the probability of weakening the affinity of abnormal nodes to its neighbors, and this effect would diminish when $T$ is sufficiently large. The average of local affinity from multiple truncated graph sets is more conducive to anomaly detection.

**Regularization Hyperparameter $\lambda$.** In order to evaluate the effectiveness of $\lambda$, we adopt different values to adjust the weight of the regularization. From Fig. 7, we can see that for BlogCatalog and Facebook, adding the regularization improves the effectiveness of the model by a large margin. This is mainly because the use of regularization can prevent all nodes from having identical feature representations. For most real-world datasets with genuine anomalies, the regularization does not significantly improve the effectiveness of the model while decreasing the performance as the increasing of $\lambda$. The main reason is that Amazon, Reddit, and YelpChi are real-world datasets with diverse attributes, and the role of regularization term is not reflected during affinity maximization.
C.2 Complexity Analysis

This subsection analyzes the time complexity of TAM. Specifically, the distance calculation takes $O(md_0)$, $m$ is the number of non-zero elements in the adjacent matrix $A$, and $d_0$ is the dimension of attributes for each node. The graph truncation in TAM takes $2N\eta$, where $N$ is the number of nodes and $\eta$ is the average degree in the graph. In LAMNet, we build a GCN for each truncated graph, which takes $O(md_1h)$ using sparse-dense matrix multiplications, where $h$ and $d_1$ denotes summation of all feature maps across different layer and feature dimensions in graph convolution operation, respectively. The construction of a GCN takes $O(md_1h)$. LAMNet also needs to compute all connected pairwise similarities, which takes $O(N^2)$. Thus, the overall complexity of TAM is $O(md_0 + (N^2 + md_1h + 2N\eta)KT)$, where $K$ is the truncation depth and $T$ is the ensemble parameter. The complexity is lower than the time complexity in many existing GNN-based graph anomaly detection methods based on the subgraph sampling and hop counting [17, 63].

C.3 Anomaly Scoring

AUROC Results of TAM and Its Variants. We present the AUPRC results of TAM and its two variants, Degree and TAM-T in the main text, where TAM can consistently and significantly outperform both variants. Similar observation can also be found from the AUROC results in Fig. 8(a). Fig. 8(b) shows the AUROC results of anomaly scoring by aggregating the anomaly scores under all truncation scales/depths. Similar to the AUPRC results in the main text, the anomaly scores obtained from different truncation scales/depths can largely improve the detection performance and the results become stable with increasing graph truncation depth $K$.

Anomaly Scoring Using Multi-scale Truncation vs Single-scale Truncation. Fig. 9 shows the results of TAM that performs anomaly scoring based on a single-scale graph truncation rather than the default multi-scale graph truncation. As shown in Fig. 9, the increase of $K$ improves the performance on large datasets such as the Amazon datasets, but it often downgrades the performance on the datasets such as Reddit and YelpChi. The main reason is that the node attributes in these datasets are more similar than the other datasets, restricting the effect of graph truncation. However, the opposite case can occur on the other datasets. To tackle this issue, we define the overall anomaly score as an average score over the anomaly scores obtained from the LAMNets built using truncated graphs under all truncation depths/scales. This resulting multi-scale anomaly score, as shown in Fig. 8(b), performs much more stably than the single-scale anomaly score.

C.4 Shared-weights LAMNet vs. Non-shared-weights LAMNet

In our experiments, the weight parameters in LAMNets are independent from each other by default, i.e., GNNs in LAMNets are independently trained. In this section, we compare TAM with its variant TAM-S where all LAMNets use a single GNN backbone with shared weight parameter.
The results are shown in Tab. 10. It is clear that TAM performs consistently better than, or comparably well to, TAM-S across the six datasets.

C.5 Simple Similarity-based Non-homophily Edge Removal

Our ablation study in the main text has shown the inferior performance of SimilarityCut method compared to our proposed NSGT, where we directly remove 5% least similar edges. Tab 11 shows the results of removing 10% and 30% least similar edges. It is clear that NSGT outperforms these two other cases of SimilarityCut.

C.6 ROC and Precision-Recall Curves

The ROC curve and Precision-Recall curve of TAM on the six datasets are shown in Fig. 10.
Table 10: AUROC and AUPRC results of TAM using shared-weight LAMNets (TAM-S) vs. non-shared-weight LAMNets (TAM).

<table>
<thead>
<tr>
<th>Metric</th>
<th>Method</th>
<th>Dataset</th>
<th>BlogCatalog</th>
<th>ACM</th>
<th>Amazon</th>
<th>Facebook</th>
<th>Reddit</th>
<th>YelpChi</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUROC</td>
<td>TAM-S</td>
<td>0.8170±0.002</td>
<td>0.8826±0.003</td>
<td>0.7044±0.008</td>
<td>0.9165±0.005</td>
<td>0.6088±0.002</td>
<td>0.5407±0.005</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAM</td>
<td>0.8248±0.003</td>
<td>0.8878±0.004</td>
<td>0.7064±0.010</td>
<td>0.9144±0.008</td>
<td>0.6023±0.004</td>
<td>0.5643±0.007</td>
<td></td>
</tr>
<tr>
<td>AUPRC</td>
<td>TAM-S</td>
<td>0.3908±0.002</td>
<td>0.4960±0.001</td>
<td>0.2597±0.002</td>
<td>0.2087±0.006</td>
<td>0.0459±0.003</td>
<td>0.0691±0.002</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAM</td>
<td>0.4182±0.005</td>
<td>0.5124±0.018</td>
<td>0.2634±0.008</td>
<td>0.2233±0.016</td>
<td>0.0446±0.001</td>
<td>0.0778±0.009</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Performance under different ratio of similarity-based edge removal. \( \theta \) represents the edge-cut ratio based on similarity.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Method</th>
<th>Dataset</th>
<th>( \theta=0.05 )</th>
<th>( \theta=0.1 )</th>
<th>( \theta=0.3 )</th>
<th>TAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUROC</td>
<td>BlogCatalog</td>
<td>0.6650</td>
<td>0.6526</td>
<td>0.6583</td>
<td>0.8210</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACM</td>
<td>0.8668</td>
<td>0.7986</td>
<td>0.6911</td>
<td>0.8878</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Amazon</td>
<td>0.5856</td>
<td>0.5827</td>
<td>0.6106</td>
<td>0.7064</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Facebook</td>
<td>0.6951</td>
<td>0.7293</td>
<td>0.7934</td>
<td>0.9144</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reddit</td>
<td>0.6007</td>
<td>0.5945</td>
<td>0.5758</td>
<td>0.6028</td>
<td></td>
</tr>
<tr>
<td></td>
<td>YelpChi</td>
<td>0.4910</td>
<td>0.4872</td>
<td>0.4754</td>
<td>0.5674</td>
<td></td>
</tr>
<tr>
<td>AUPRC</td>
<td>BlogCatalog</td>
<td>0.1621</td>
<td>0.1829</td>
<td>0.1729</td>
<td>0.4152</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACM</td>
<td>0.5109</td>
<td>0.5068</td>
<td>0.4996</td>
<td>0.5124</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Amazon</td>
<td>0.0924</td>
<td>0.1092</td>
<td>0.2079</td>
<td>0.2634</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Facebook</td>
<td>0.0410</td>
<td>0.1154</td>
<td>0.1374</td>
<td>0.2233</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reddit</td>
<td>0.0467</td>
<td>0.0414</td>
<td>0.0420</td>
<td>0.0446</td>
<td></td>
</tr>
<tr>
<td></td>
<td>YelpChi</td>
<td>0.0598</td>
<td>0.0509</td>
<td>0.0519</td>
<td>0.0771</td>
<td></td>
</tr>
</tbody>
</table>

D Description of algorithms

D.1 Competing Methods

- **iForest** [25] builds multiple trees to isolate the data based on the node’s feature. It has been widely used in outlier detection.

- **ANOMALOUS** [41] proposes a joint network to conduct the selection of attributes in the CUR decomposition and residual analysis. It can avoid the adverse effects brought by noise.

- **DOMINANT** [8] leverages the auto-encoder for graph anomaly detection. It consists of an encoder layer and a decoder layer which construct the feature and structure of the graph. The reconstruction errors from the feature and structural module are combined as the anomaly score.

- **HCM-A** [17] constructs an anomaly indicator by estimating hop count based on both global and local contextual information. It also employs Bayesian learning in predicting the shortest path between node pairs.

- **CoLA** [28] exploits the local information in a contrastive self-supervised framework. They define the positive pair and negative pair for a target node. The anomaly score is defined as the difference value between its negative and positive score.

- **SL-GAD** [63] constructs two modules including generative attribute regression and multi-view contrastive for anomaly detection based on CoLA. The anomaly score is generated from the degree of mismatch between the constructed and original features and the discrimination scores.

- **ComGA** [31] designs a tailor GCN to learn distinguishable node representations by explicitly capturing community structure.

Their implementation is taken directly from their official web pages or the widely-used PyGOD library [26]. The links to the source code pages are as follows:

- iForest: https://github.com/pygod-team/pygod
- ANOMALOUS: https://github.com/pygod-team/pygod
- DOMINANT: https://github.com/kaize0409/GCN_AnomalyDetection_pytorch
Figure 10: The ROC curve and precision-recall curve of our method TAM on all the six datasets used. (a)(b) BlogCatalog; (c)(d) ACM; (e)(f) Amazon; (h)(i) Facebook; (j)(k) Reddit; (l)(m) YelpChi

- HCM-A: https://github.com/TienjinHuang/GraphAnomalyDetection
- CoLA: https://github.com/GRAND-Lab/CoLA:
- SL-GAD: https://github.com/yixinliu233/SL-GAD
- ComGA:https://github.com/XuexiongLuoMQ/ComGA

D.2 Pseudo Codes of TAM.

The training algorithms of TAM are summarized in Algorithm 1 and Algorithm 2. Algorithm 1 describes the process of NSGT. Algorithm 2 describes the training process of TAM.
Algorithm 1 NSGT

**Input:** Attributed Graph, $G = (\mathcal{V}, \mathcal{E})$, Distance Matrix, $\mathbf{M}$

**Output:** A truncated graph structure $\tilde{\mathcal{E}}$.

1: /* Regard the graph as a direct graph */
2: Initialize the directed graph truncation indicator $e_{(v_i, v_j)} = 1$ iff $(v_i, v_j) \in \mathcal{E}$
3: Find the mean $d_{\text{mean}}$ from $\mathbf{M}$ using $d_{\text{mean}} = \frac{1}{m} \sum_{(v_i, v_j) \in \mathcal{E}} d_{ij}$
4: for each $v$ in $\mathcal{V}$ do
5: Find the maximum $d_{v, \text{max}}$ from $\{d(v, v'), (v, v') \in \mathcal{E}\}$
6: if $d_{v, \text{max}} > d_{\text{mean}}$ then
7: Randomly sample $r$ from $[d_{\text{mean}}, d_{v, \text{max}}]$ for node $v$
8: for each $v'$ in $\{v', (v, v') \in \mathcal{E}\}$ do
9: if $d(v, v') > r$ then
10: Plan to cut the edge $v$ to $v'$, i.e., $e_{(v, v')} \leftarrow 0$
11: end if
12: end for
13: end if
14: end for
15: /* The edge will be removed only when it is a non-homophily edge from both directions of the connected nodes */
16: for each $v$ in $\mathcal{V}$ do
17: for each $v'$ in $\{v', (v, v') \in \mathcal{E}\}$ do
18: Cut the edge between $v$ and $v'$, $\tilde{\mathcal{E}} = \mathcal{E} \setminus ((v, v') \cup (v', v))$; iff $e_{(v, v')} = e_{(v', v)} = 0$
19: end for
20: end for
21: return The truncated graph structure $\tilde{\mathcal{E}}$
Algorithm 2 TAM

**Input:** Graph, $G = (V, E, X)$, $N$: Number of nodes, $L$: Number of layers, $E$: Training epochs, $T$: Ensemble parameter, $K$: Truncation depth.

**Output:** Anomaly scores of all nodes $s$.

1: Compute the Euclidean distance $M = \{d_{ij}\}$ for each connected node pair $(v_i, v_j) \in E$

2: Randomly initialize $\text{GNN} (h^{(0)}_1, h^{(0)}_2, \ldots, h^{(0)}_N) \leftarrow X$, $\mathcal{E}(1, 0) \leftarrow E$

3: for $k = 1, \ldots, K$ do

4: for $t = 1, \ldots, T$ do

5: /* Graph truncation and update the graph structure */

6: $\mathcal{E}(t, k) = \text{NSGT}(V, \mathcal{E}(t, k-1), M)$ /* LAMNet */

7: for epoch = 1, \ldots, $E$ do

8: for each $v$ in $V$ do

9: $h_{v, t} = \phi(h_{v, t-1}; \Theta_{t, k})$

10: $h_{v, t} = \text{ReLU} (\text{AGG}(\{h_{v', t} : (v, v') \in \mathcal{E}(t, k)\}))$

11: end for

12: Calculate $f_{\text{TAM}}(v_i; \Theta_{t, k}, A, X)$ by Eq. (2).

13: end for

14: /* Affinity maximization */

15: Minimize $\sum_{v_i \in V} \left( f_{\text{TAM}}(v_i; \Theta_{t, k}, A, X) + \lambda \frac{1}{|V \setminus X(v_i)|} \sum_{v_k \in V \setminus N(v_i)} \text{sim} (h_i, h_k) \right)$

16: Update $\Theta_{t, k}$ by using stochastic gradient descent

17: end for

18: end for

19: /* Aggregated anomaly score over $T$ sets of multi-scale graph truncation depths */

20: return Anomaly Score by $s(v) = \frac{1}{T \times K} \sum_{k=1}^{K} \sum_{t=1}^{T} f_{\text{TAM}}(v_i; \Theta^*_{t, k}, A, X)$