Semi-supervised Anomaly Detection on Attributed Graphs

1st Atsutoshi Kumagai
NTT Software Innovation Center
Tokyo, Japan
atsutoshi.kumagai.ht@hco.ntt.co.jp

2nd Tomoharu Iwata
NTT Communication Science Laboratories
Kyoto, Japan
tomoharu.iwata.gy@hco.ntt.co.jp

3rd Yasuhiro Fujiwara
NTT Communication Science Laboratories
Kanagawa, Japan
yasuhiro.fujiwara.kh@hco.ntt.co.jp

***Abstract*—We propose a simple yet effective method for detecting anomalous instances on an attribute graph with label information of a small number of instances. Although standard anomaly detection methods usually assume that instances are independent and identically distributed, in many real-world applications, instances are often explicitly connected, resulting in so-called attributed graphs. The proposed method embeds nodes (instances) on the attributed graph in a latent space by taking into account their attributes as well as the graph structure on the basis of graph convolutional networks (GCNs). To learn node embeddings specialized for anomaly detection, in which there is a class imbalance due to the rarity of anomalies, the parameters of a GCN are trained to minimize the volume of a hypersphere that encloses the node embeddings of normal instances while embedding anomalous ones outside the hypersphere. This enables us to detect anomalies by simply calculating the distances between the node embeddings and hypersphere center. The proposed method can effectively propagate label information on a small amount of nodes to unlabeled ones by taking into account the node's attributes, graph structure, and class imbalance. In experiments with five real-world attributed graph datasets, we demonstrate that the proposed method outperforms various existing anomaly detection methods.**

I. Introduction

Anomaly detection is an important task in machine learning that involves identifying anomalous instances, called anomalies, in a dataset [6], [8]. Anomaly detection methods have been used in a wide variety of applications such as intrusion detection [14], fraud detection [24], and medical care [20].

Although many anomaly detection methods have been proposed such as one-class support vector machines (OSVM) [43], autoencoder (AE) [42], and isolation forest [29], these methods typically assume that instances are independent and identically distributed (i.i.d.). However, in many real-world applications, instances are often explicitly connected, i.e., they have graph structures. For example, in botnet detection on the Internet, each host is connected by its communications [5]. In anomalous user detection on social networking services, users are connected by their social relationships [15]. Such graphs, i.e., graphs in which each node has instance (attributes) information, are called *attributed graphs* or *attributed networks*.

To detect anomalies on attributed graphs, many methods have been proposed [2], [13], [26], [27], [37]. By considering graph structure as well as instance information, these methods often perform better than anomaly detection methods for i.i.d. data. Most existing methods aim to find anomalies on attributed graphs in an unsupervised fashion, i.e., not considering the label (normal and anomalous) information of nodes. However, label information, which is valuable for anomaly detection, may be usable in practice. Semi-supervised learning methods for an attributed graph can use this label information to classify unlabeled instances [18], [23], [39], [44], [51], [53], [56]. Although these methods are effective for standard classification tasks, they do not perform well when there is a class imbalance, i.e., normal instances far outnumber anomalous instances, which is common in anomaly detection tasks due to the rarity of anomalies [52], [59].

In this paper, we propose a novel semi-supervised anomaly detection method for an attribute graph containing a class imbalance. We focus on detecting anomalies on the attributed graph by using the graph structure as well as labeled and unlabeled instance information[1]. In the proposed approach, we embed all nodes on the attribute graph in a latent space to better discriminate between anomalous and normal instances. Specifically, the proposed method is based on graph convolutional networks (GCNs) [23], which can output node embeddings given an attributed graph while considering both graph structure and instance information effectively by stacking graph convolutional layers. In anomaly detection tasks, a data description of the normal class needs to be modeled because the number of anomalies is too small for their description to be modeled [6], [8]. Therefore, the proposed method minimizes the volume of the hypersphere enclosing normal node embeddings to effectively learn a brief data description of the normal class. This data-enclosing hypersphere framework has been successfully used for one-class classification tasks for i.i.d. data [40], [47]. In addition to minimizing the volume of the hypersphere enclosing normal node embeddings, the proposed method simultaneously embeds anomalous instances outside the hypersphere. This enables us to detect anomalies accurately on the basis of the distances between the node embeddings and hypersphere center. To embed anomalies outside the hypersphere, the proposed method uses a differential area under the curve (AUC) as the regularizer. The AUC is a commonly used evaluation measurement for anomaly

[1]Although the term "semi-supervised" sometimes means using normal instances only for training in the anomaly detection literature, we use it to mean using both labeled and unlabeled instances following a previous study [41].

2021 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-3900-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/IJCNN52387.2021.9533507

detection tasks because it can properly evaluate performance with class imbalanced data [19], [25]. Therefore, by using the AUC regularizer, the proposed method can effectively exploit anomalous information for anomaly detection on the attribute graph. Even if a few nodes have label information on the attributed graph, this information can be effectively propagated to other nodes by using both the graph structure and the attributes of all nodes with the GCNs.

Our main contributions are summarized as follows: 1) We propose a simple yet effective semi-supervised anomaly detection method on attributed graphs that unifies the GCNs and data-enclosing hypersphere framework with the AUC regularizer to fit on anomaly detection tasks on attributed graphs. 2) Through experiments using five real-world attributed graph datasets, we demonstrate that the proposed method outperforms various existing methods by a large margin.

II. Related Work

Anomaly detection, which is also called outlier detection or novelty detection, has been widely studied [6], [8]. Many unsupervised anomaly detection methods have been proposed such as OSVM based methods [7], [43], AE based methods [10], [42], [57], and density based methods [31], [61]. Recently, deep one-class classification [40] which is closely related to the proposed method, has showed promising results on i.i.d. data. This method learns compact representations of instances by minimizing a data-enclosing hypersphere in output space, which is an extension of the classical support vector data description (SVDD) [47] to the deep learning. However, this method is not applicable to the attributed graph. The proposed method utilizes this data-enclosing hypersphere approach to learn node embeddings for the normal instances on the attributed graph. Some studies focus on supervised anomaly detection methods that use both anomalous and normal information to obtain anomaly detectors [9], [19], [41], [55]. All these methods assume that instances are i.i.d. and cannot use any graph structure information.

Unsupervised anomaly detection methods on attributed graphs have been proposed [2]. Early methods use graph structure information only such as node degree and egonet features to detect anomalies on the graph [1], [48], [54]. Recent methods such as residual analysis based methods [26], [37], matrix factorization based methods [3], generative adversarial network based methods [11], [12], and graph AE based methods [13], [27] use node instance information as well as graph structure information to improve performance. One method uses the data-enclosing hypersphere approach for modeling community structures [4]. These methods do not use label (anomalous and normal) information. One method assumes only normal information for training [50]. In contrast, the proposed method uses both anomalous and normal label information to improve anomaly detection performance.

Semi-supervised learning for graph structured data has been proposed [44], [58], [60]. These methods propagate label information of a small amount of nodes to unlabeled nodes by using both node instances and a graph structure. By taking advantage of the progress of graph neural networks (GNNs) including GCNs, these methods have achieved state-of-the art results on various semi-supervised node classification tasks [18], [23], [34], [39], [51], [53], [56]. However, these methods do not assume the class imbalance and thus are not appropriate for anomaly detection tasks. Although some methods learn robust node embeddings against anomalies for multi-class classification [28], they cannot use anomalous labels.

Few methods aim to detect anomalies on attributed graphs considering both label (anomalous and normal) information and class imbalance like the proposed method. ImVerde [52] is a semi-supervised learning method based on random walks considering the class imbalance. One method for rare category characterization [59] uses curriculum learning to learn node representations. In addition to learn classifier networks, both methods use the random-walks for learning node embeddings, which have many hyperparameters such as walk lengths, context sizes, and sampling numbers [17], [38], [46]. The proposed method is based on simple GCNs for classifiers and thus will be convenient in practice. In addition, these methods are difficult to apply to one-class classification tasks, which contain only normal training instances. In contrast, the proposed method can be used since it explicitly models representations of the normal class.

III. Proposed Method

A. Task

Let $\mathcal{G} = (\mathcal{V}, \mathbf{A}, \mathbf{X})$ be an undirected attributed graph, where $\mathcal{V}$ represents a set of nodes $\{v_1, \ldots, v_N\}$, $\mathbf{A} \in \mathbb{R}^{N \times N}$ represents a symmetric adjacency matrix where its (n,m)-th element $a_{nm} > 0$ denotes that there is an edge between node v_n and v_m and $a_{nm} = 0$ denotes a no-edge, and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times D}$ represents the set of instances, where $\mathbf{x}_n$ is D-dimensional attribute vector of n-th node v_n. The index sets for anomalous and normal training nodes are represented as $\mathcal{A} = \{n | v_n \text{ is anomalous}\}$ and $\mathcal{N} = \{n | v_n \text{ is normal}\}$, respectively. We assume that the label information is given for a small amount of nodes on the attribute graph, i.e., $|\mathcal{A} \cup \mathcal{N}| \ll N$. In addition, we assume the class imbalance, i.e., $|\mathcal{A}| \ll |\mathcal{N}|$, since anomalies rarely occur. We note that the proposed method is applicable even when there are no anomalous nodes, $|\mathcal{A}| = 0$. Our task is to estimate anomaly scores of unlabeled nodes on the graph, $\mathcal{V} \setminus (\mathcal{A} \cup \mathcal{N})$, so that the anomaly score becomes high (low) when the instance is anomalous (normal), given the attributed graph $\mathcal{G}$ and its label information $\mathcal{A} \cup \mathcal{N}$.

B. Anomaly Scores

We define the anomaly score for each node as $a(v_n) := \|\mathbf{h}_n - \mathbf{c}\|^2$, where $\mathbf{h}_n \in \mathbb{R}^K$ is a K- dimensional learned node embedding for the n-th node, $\mathbf{c} \in \mathbb{R}^K$ is a pre-determined center vector, and $\| \cdot \|$ is the Euclidian norm. This anomaly score takes a small (large) value when node embedding $\mathbf{h}_n$ is close to (far from) center vector $\mathbf{c}$. Therefore, to detect anomalies accurately, we want to learn node embeddings in such a way that node embeddings for normal instances are

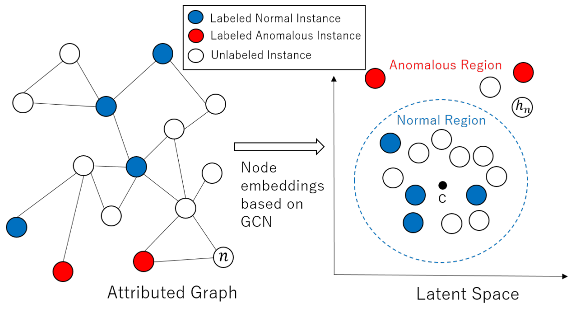


Fig. 1: Overview of the proposed method. The proposed method embeds each node in a latent space so that normal instances are close to a center vector $\mathbf{c}$ and anomalous ones are far away from $\mathbf{c}$.

placed close to center vector $\mathbf{c}$ while those of anomalous instances are far away from $\mathbf{c}$. We will explain how to learn these node embeddings in the next subsection. Figure 1 illustrates the proposed method.

C. Model

The proposed method learns node embeddings specialized for anomaly detection on the attributed graph on the basis of GCNs, which are proposed by Kipf and Welling [23]. The GCNs learn K-dimensional node embedding $\mathbf{h}_n \in \mathbb{R}^K$ for the n-th node by applying multiple layer transformations to attribute vector $\mathbf{x}_n$. Specifically, the $(\ell+1)$-th layer $\mathbf{H}^{(\ell+1)} = [\mathbf{h}_1^{(\ell+1)}, \dots, \mathbf{h}_N^{(\ell+1)}]^\top$ is calculated from the previous ℓ-th layer $\mathbf{H}^{(\ell)}$with the following propagation rule:

$$\mathbf{H}^{(\ell+1)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(\ell)}\mathbf{W}^{(\ell)}), \quad (1)$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix of the graph $\mathcal{G}$ with added self-connections, $\tilde{\mathbf{D}} \in \mathbb{R}^{N \times N}$ is the degree matrix of $\tilde{\mathbf{A}}$, which is a diagonal matrix where its (n,n)-th element is $\sum_{m=1}^N a_{nm} + 1$, $\mathbf{W}^{(\ell)}$ is a layer-specific trainable weight matrix, and $\sigma(\cdot)$ is an activation function such as the ReLU. The initial node embeddings are set to the original attribute vectors, $\mathbf{H}^{(0)} := \mathbf{X}$. Note that the proposed method is applicable even when the graph does not have attributes $\mathbf{X}$ by regarding the identity matrix as $\mathbf{X} = \mathbf{I}$ as described in previous studies [33]. By decomposing equation (1) for each node, we can see that $\mathbf{h}_n^{(\ell+1)}$ is represented as follows:

$$\mathbf{h}_n^{(\ell+1)} = \sigma\Bigg(\frac{1}{d_n+1}\mathbf{W}^{(\ell)\top}\mathbf{h}_n^{(\ell)} + \sum_{m=1}^N \frac{a_{nm}}{\sqrt{(d_n+1)(d_m+1)}}\mathbf{W}^{(\ell)\top}\mathbf{h}_m^{(\ell)}\Bigg), \quad (2)$$

where d_n is the degree of the n-th node, $d_n := \sum_{m=1}^N a_{nm}$. This equation means that node embeddings of the next layer are calculated using node embeddings of its connected nodes and the node itself at the current layer. By applying L layer transformations, information on nodes within the L-hop neighborhood can be used to learn the node embedding. Therefore, the GCNs can learn node embeddings while considering attribute information of all nodes with their graph structure by stacking multiple layers. The L-th layer is regarded as the node embeddings of the proposed method, $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_N]^\top := \mathbf{H}^{(L)}$. The propagation rule of equation (1) can be understood as a first-order approximation of localized spectral filters on graphs. For the details, please refer to the paper [23]. Although we used the GCNs as building blocks of our model since they are simple and thus will be convenient in practice, we can use any graph neural networks to obtain node embeddings, such as GraphSAGE [18] and GAT [49].

We construct the objective function of the proposed method that consists of two terms. We first explain the first term of the objective function. To learn the data description of the normal class, the proposed method minimizes the volume of the hypersphere that encloses the node embeddings of normal instances. Specifically, the first term of objective function with normal instances to be minimized is as follows:

$$\mathcal{L}_{\text{nor}}(\theta) := \frac{1}{|\mathcal{N}|}\sum_{n\in\mathcal{N}} a(v_n) = \frac{1}{|\mathcal{N}|}\sum_{n\in\mathcal{N}} \|\mathbf{h}_n - \mathbf{c}\|^2, \quad (3)$$

where θ is set of trainable weight matrixes of the GCN $\theta = \{\mathbf{W}^{(0)}, \dots, \mathbf{W}^{(L-1)}\}$. Minimizing the mean squared Euclidian distance of the normal node embeddings to hypersphere center $\mathbf{c}$ forces the GCN to learn the compact normal data representations.

The second term of the objective function employs a differential approximation of the AUC to use anomalous instance information effectively:

$$\mathcal{R}_{\text{AUC}}(\theta) := \frac{1}{|\mathcal{A}||\mathcal{N}|}\sum_{n\in\mathcal{A}}\sum_{m\in\mathcal{N}} f(a(v_n) - a(v_m)), \quad (4)$$

where $f(\cdot)$ is sigmoid function; $f(x) = \frac{1}{1+\exp(-x)}$. Since $f(\cdot)$ takes the maximum value one when $a(v_n) \gg a(v_m)$ and the minimal value zero when $a(v_n) \ll a(v_m)$, maximizing (4) encourages the score of anomalous instances to be higher than those of normal ones even though the number of anomalous instances is small. Therefore, this term enables us to use anomalous instance information effectively from class imbalanced data.

The final objective function of the proposed method to be minimized is a weighted sum of $\mathcal{L}_{\text{nor}}(\theta)$ and $\mathcal{R}_{\text{AUC}}(\theta)$:

$$\mathcal{L}(\theta) := \mathcal{L}_{\text{nor}}(\theta) - \lambda\mathcal{R}_{\text{AUC}}(\theta), \quad (5)$$

where $\lambda \geq 0$ is the hyperparameter that controls the influence of the differentiable AUC loss. By omitting the AUC regularizer or $\lambda = 0$, this objective function can be used even though there is no anomalous label information. By adding the AUC regularizer, the proposed method can learn more sophisticated node embeddings that can detect anomalies accurately. Note that, when there are no anomalous instances or $\lambda = 0$, we should use the GCNs without bias terms, use unbounded activation functions such as the ReLU, and avoid using an all-zero vector as center vector $\mathbf{c}$ to prevent a hypersphere collapse, in which any node embedding converge to the center vector $\mathbf{c}$ [40]. When there are anomalies and $\lambda > 0$, the proposed method does not require such constraints for bias terms, activation functions, and the center vector since the AUC regularizer prevents trivial solutions.

The parameters of the GCN can be optimized by minimizing (5) with any gradient-based optimization methods. Even though few nodes have label information, the proposed method can effectively and efficiently propagate this information to other nodes on the basis of the GCN.

D. Estimation

After learning the parameters of the GCN by minimizing equation (5), the anomaly score of unlabeled instance on the graph $a(v_*)$ is obtained as $a(v_*) = \|\mathbf{h}_* - \mathbf{c}\|^2$. Although the proposed method described in this paper is transductive, which means we can only estimate anomaly scores of instances that are already observed in the graph at training time, we can easily extend it to inductive, which means we can estimate anomaly scores for unobserved instances at training time, by applying inductive variants of GNNs such as Planetoid [56] and GraphSage [18].

IV. Experiments

We demonstrated the effectiveness of the proposed method using five real-world attributed graph datasets. To measure anomaly detection performance on the attributed graphs, we used AUC, which is a well used measure for anomaly detection tasks. All experiments were conducted on a Linux server with an Intel Xeon CPU and a NVIDIA GeForce GTX 1080 GPU.

A. Data

We used five popular real-world attributed graph datasets: Cora, Citeseer (Cite), Pubmed (Pub), Amazon-Photo (Photo), and Amazon-Computers (Comp). Cora, Cite, and Pub are public datasets widely used in previous studies [23], [51], [52], [56][2]. All of them are citation networks, each node corresponds to one scientific publication, and the edge represents the citation relationship between two publications. Each publication is represented by a bag-of-words attribute vector. Photo and Comp are also well used public datasets [45][3]. These datasets are segments of the Amazon co-purchase graph [32], where nodes represent goods and the edge indicates that two goods are frequently bought together. Each product review is represented by a bag-of-words attribute vector. For all datasets, the edges are unweighted, i.e., $a_{nm} = 1$ if there is a link between v_n and v_m. Each attribute was linearly rescaled to $[0, 1]$. The statistics of the datasets are summarized in Table I. Although these datasets have several classes, we created a binary class problem for each dataset by regarding the smallest class as anomalous and the remaining classes as normal following the previous studies [25], [52], [59]. The average anomaly rates of Cora, Cite, Pub, Photo, and Comp are 0.06, 0.07, 0.21, 0.04, and 0.02, respectively. For each dataset, we evaluated anomaly detection performance by changing the ratio of labeled instances and all instances within $\{2.5\%, 5\%, 10\%\}$. For each case, we used 10% of all instances for validation and the remaining for testing instances. We randomly generated ten training/validation/test datasets for each case and evaluated the average test AUC over ten sets.

[2]https://github.com/kimiyoung/planetoid

[3]https://pytorch-geometric.readthedocs.io/en/latest/

TABLE I: The statistics of datasets.

Data	Nodes	Edges	Attributes
Cora	2,708	5,278	1,433
Cite	3,327	4,732	3,703
Pub	19,717	44,338	500
Photo	7,487	119,043	745
Comp	13,381	245,778	767

B. Comparison Methods

We evaluated two variants of the proposed method: Ours-AN and Ours-N. Ours-AN is the method explained in Section III, which uses both anomalous and normal label information. Ours-N uses only normal label information. The proposed method was implemented by using PyTorch [36] and PyTorch Geometric [16]. We compared the proposed method with the following nine methods.

OSVM is the one-class support vector machine [43]. The OSVM finds the maximal margin hyperplane which separates the given normal data from the origin in a RKHS. We used the RBF kernel. When the RBF kernel is used, OSVM is equivalent to SVDD [40].

DOC-N is the deep one-class classification (One-class Deep SVDD) [40], which is a recently proposed unsupervised anomaly detection method for i.i.d. data. This method uses the feed-forward neural network to output embeddings and aims to minimize the volume of the hypersphere that encloses the embeddings of normal instances. Although this objective is also used in the proposed method, DOC-N cannot use any graph structure information.

DOC-AN is a supervised extension of DOC-N. We added the differentiable AUC regularizer to the objective function of the OCD-N. We included this method in the comparison methods to evaluate the effectiveness of considering the attributed graph structure in the proposed method.

DSAD is the deep semi-supervised anomaly detection [41], which is an extension of DOC-N for semi-supervised anomaly detection. This method uses anomalous and normal labeled instances and unlabeled instances to learn the anomaly detector for i.i.d. data.

NN is the feed-forward neural network classifier for i.i.d. data. The parameters of NN are trained by minimizing the cross entropy loss.

SLGCN is the semi-supervised learning method based on the GCNs [23]. The parameters of the GCNs are trained by minimizing the cross entropy loss of labeled nodes.

DW is the DeepWalk [38], which is a famous unsupervised node embedding method for graph structured data. This method learns node embeddings on the basis of skip gram models [35], which are applied to the sequences of random-walks. We used negative sampling instead of hierarchical softmax to improve performance the same as [17]. After learning node embeddings, logistic regression was used as classifiers the same as [52], [59].

DOM is the Dominant [13], which is a recently proposed unsupervised anomaly detection method on attributed graphs. This method uses an autoencoder framework to reconstruct the original attributed graph (graph structure and node attributes). The anomaly scores are defined as a weighted sum of the graph structure and node attribute reconstruction errors.

ImVerde is a recently proposed semi-supervised anomaly detection method for class imbalanced attributed graphs [52]. This method learns node embeddings on the basis of a vertex-diminished random-walk model that reduces the transition probability to one node each that it has visited to deal with the class imbalance. We used the authors' implementation[4].

OSVM and DOC-N are anomaly detection methods for i.i.d. data, which learn from normal training instances. Note that DOC-N uses unlabeled instances as well as normal instances for training assuming that there are fewer anomalies than normal instances in the original paper. However, training with only normal instances showed better results in our experiments, so we report the results of DOC-N learned with normal instances in this paper. NN and DOC-AN are supervised learning methods for i.i.d. data, which learn from both anomalous and normal training instances. DSAD learns from anomalous and normal training instances and unlabeled instances. We used DSAD in the transductive setting. i.e., unlabeled training instances are equivalent to testing instances, for a fair comparison. These five methods do not use any graph structure information. DOM uses both graph structure and instance information but not label information. DW uses both the graph structure and anomalous and normal label information. SLGCN and ImVerde use the graph structure, instances, and anomalous and normal label information, which is the same as Ours-AN.

C. Settings

For the proposed method, DOC-N, DOC-AN, DSAD, NN, and SLGCN, the three-layer feed-forward neural networks with 32 hidden nodes and ReLU activation were used. For Ours-N and DOC-N, we removed all bias terms to prevent hypersphere collapses [40]. For DOM, the three-(two-) layer feed-forward neural network with 32 hidden nodes and ReLU activation was used for the encoder (the decoder). For ImVerde, we used the three-layer feed-forward neural network for the classifier the same as the authors' implementation. For Ours-AN, DOC-AN, DSAD, NN, SLGCN, DW, and ImVerde, we selected hyper-parameters by using validation AUC. For Ours-N, OSVM, DOC-N, and DOM, hyper-parameters were selected on the basis of the average anomaly score on validation normal instances since these methods do not use any anomalous label information for training. For OSVM, the kernel parameter was selected from $\{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$. For DW, we used the following typical parameters: the length of a walk was 80, the window size of neighbor in a random walk sequence was 10, and the number of negative samples was 10. For logistic regression of DW, regularization parameter C was chosen from $\{10^{-2}, 10^{-1}, \ldots, 10^{2}\}$. For DOM, the balancing parameter of structure and attribute reconstruction α was set to 0.5, which is the recommended value in the original paper. For ImVerde, we followed the parameters used in the author's implementation. For the proposed method, DOC-N, DOC-AN, DSAD, DW, DOM, and ImVerde, the dimension for node embeddings K was set to 32. For Ours-AN and DOC-AN, regularization parameter for the AUC regularizer λ was chosen from $\{1, 10, \ldots, 10^{4}\}$. For DSAD, the weighting parameter for labeled instances η was selected from $\{10^{-2}, 10^{-1}, \ldots, 10^{4}\}$. For DOC-N and DSAD, we set weight regularization parameter as 10^{-6} the same as in the original papers. In addition, we used the weights from the encoder part of trained AE for initialization and set hypersphere center $\mathbf{c}$ to the mean of the node embeddings for normal instances after performing an initial forward pass, which is a recommended procedure [40], [41]. Following this, for Ours-N, we used the same procedure except for changing the AE as the graph AE [22]. For Ours-AN and DOC-AN, we did not use pre-training weights since both methods worked well without pre-training weights due to the AUC regularizer in our preliminary experiments. We set hypersphere center $\mathbf{c}$ to the mean of the node embeddings for normal instances after performing an initial forward pass as in Ours-N and DOC-N. For all methods, we used the Adam optimizer [21] with a learning rate of 0.001, and the maximum number of epochs was 500, 500, 500, 1000, and 1000 for Cora, Cite, Pub, Photo, and Comp, respectively. We used early-stopping based on the validation data to avoid over-fitting.

D. Results

First, we quantitatively evaluated the anomaly detection performance of the proposed method. Tables II – IV show the average and standard deviation of test AUCs for each dataset with the different ratio of labeled and all instances. Ours-AN showed the best/comparable test AUCs in almost all cases (13 of 15). Overall, methods that use anomalous label information (i.e., Ours-AN, DOC-AN, DSAD, NN, SLGCN, and ImVerde) performed better than other methods, which indicates the usefulness of anomalous label information for anomaly detection tasks. Although DW uses both anomalous and normal label information, it performed poorly. This was most likely because this method takes a two-step approach, i.e., learning classifiers after learning node embeddings, and thus discriminative information disappeared in the process of learning node embeddings. As for methods that do not use anomalous label information (i.e., Ours-N, OSVM, DOC-N, and DOM), Ours-N performed the best in almost all cases (13 of 15). Although DOC-N, DOC-AN, and DSAD aim to minimize the volume of the instances-enclosing hyperspheres like the proposed method, the proposed method outperformed them because it takes the graph structure information into account. As a result, these results indicate the effectiveness of the proposed method in settings in which both anomalous and normal labels or only normal labels are available for training.

[4]https://github.com/jwu4sml/ImVerde

TABLE II: Average and standard deviation of test AUCs [%] when 2.5% of all instances are labeled. Boldface denotes the best and comparable methods according to the paired t-test at a significance level of 5%.

Data	Ours-AN	Ours-N	OSVM	DOC-N	DOC-AN	DSAD	NN	SLGCN	DW	DOM	ImVerde
Cora	**88.8(5.4)**	62.6(9.9)	50.0(0.1)	57.7(3.9)	72.7(6.2)	69.6(6.5)	72.7(6.0)	84.9(6.9)	52.1(4.0)	52.3(0.9)	**85.9(6.1)**
Cite	**65.6(4.7)**	56.0(4.4)	50.6(0.4)	55.3(1.6)	59.9(6.5)	53.8(2.9)	**63.1(5.0)**	60.9(6.0)	49.4(2.4)	53.9(0.6)	60.3(6.5)
Pub	95.6(0.3)	76.5(4.2)	68.9(0.9)	73.7(6.3)	95.1(0.4)	91.3(2.4)	95.3(0.3)	**96.2(0.1)**	52.8(1.3)	50.8(0.4)	94.3(0.5)
Photo	**95.4(1.8)**	55.1(11.)	51.9(0.6)	52.3(1.4)	84.0(5.8)	81.9(5.7)	87.1(3.9)	90.1(2.5)	64.1(5.6)	38.1(0.4)	89.1(1.4)
Comp	**98.8(0.3)**	56.9(5.1)	47.3(0.7)	46.6(1.5)	94.1(2.2)	92.2(2.5)	95.8(1.6)	98.1(0.3)	87.0(4.9)	46.8(1.2)	**98.5(0.7)**
Avg	**88.9(13.)**	61.4(11.)	53.7(7.8)	57.1(10.)	81.2(14.)	77.8(15.)	82.8(14.)	86.0(14.)	61.1(15.)	48.4(5.8)	85.6(14.)

TABLE III: Average and standard deviation of test AUCs [%] when 5% of all instances are labeled. Boldface denotes the best and comparable methods according to the paired t-test at a significance level of 5%.

Data	Ours-AN	Ours-N	OSVM	DOC-N	DOC-AN	DSAD	NN	SLGCN	DW	DOM	ImVerde
Cora	**91.8(5.4)**	67.1(5.8)	50.2(0.1)	58.3(2.8)	74.6(6.2)	72.4(5.7)	77.2(4.1)	89.2(7.6)	56.2(4.5)	52.5(0.9)	**91.1(3.2)**
Cite	**68.3(3.8)**	57.4(3.1)	50.7(0.5)	56.0(0.8)	62.3(3.5)	61.1(4.2)	**66.6(2.2)**	63.9(4.7)	50.7(2.4)	53.9(0.7)	64.5(4.5)
Pub	96.2(0.2)	76.1(4.8)	71.0(1.1)	79.9(3.4)	96.0(0.3)	91.7(1.7)	96.1(0.2)	**96.6(0.1)**	54.1(1.3)	53.0(0.5)	94.9(0.5)
Photo	**97.0(0.7)**	56.2(9.6)	51.9(0.6)	52.3(0.8)	90.1(2.4)	89.8(3.1)	91.8(1.6)	92.3(1.2)	71.0(1.5)	38.1(0.5)	92.2(1.2)
Comp	**99.1(0.3)**	58.2(5.8)	47.2(0,8)	47.0(1.4)	95.6(1.5)	92.8(2.6)	97.1(0.7)	98.3(0.3)	90.5(1.8)	47.2(1.2)	98.6(0.6)
Avg	**90.5(12.)**	63.0(9.2)	54.2(8.7)	58.7(12.)	83.7(14.)	81.6(13.)	85.8(12.)	88.1(13.)	64.5(16.)	48.9(6.0)	88.3(13.)

TABLE IV: Average and standard deviation of test AUCs [%] when 10% of all instances are labeled. Boldface denotes the best and comparable methods according to the paired t-test at a significance level of 5%.

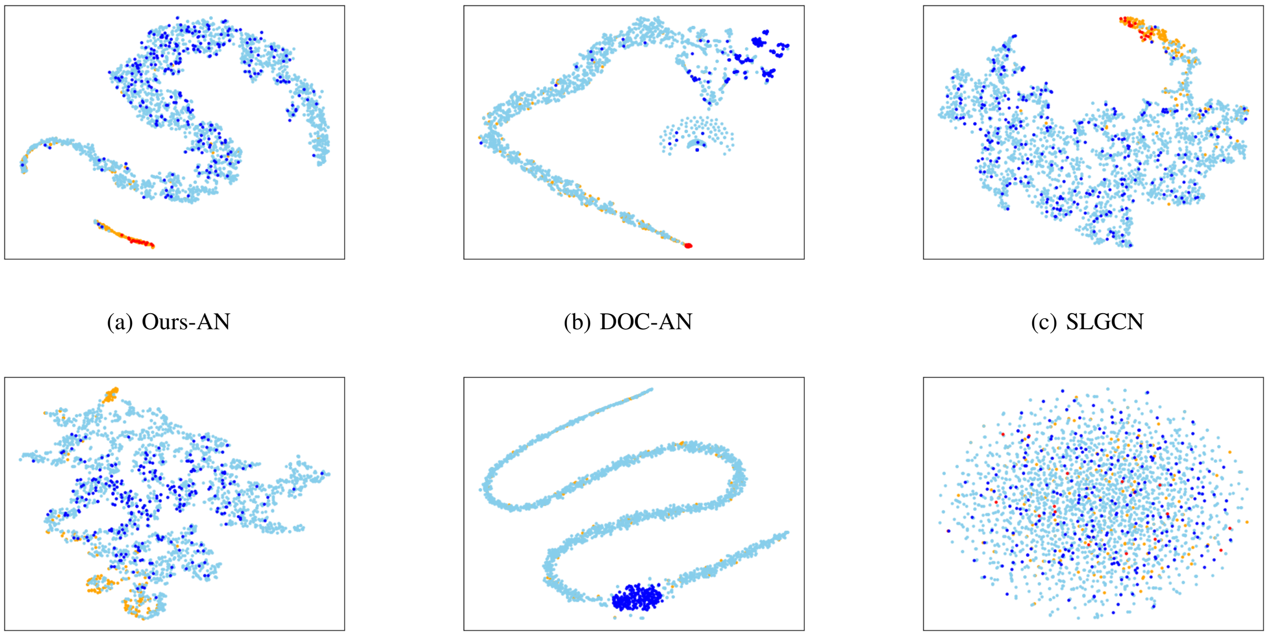
Data	Ours-AN	Ours-N	OSVM	DOC-N	DOC-AN	DSAD	NN	SLGCN	DW	DOM	ImVerde
Cora	**95.4(2.7)**	72.3(7.0)	51.8(1.7)	59.8(4.8)	82.1(3.4)	72.9(3.3)	83.5(3.0)	**94.5(4.3)**	56.3(6.2)	52.6(0.9)	**94.5(2.1)**
Cite	**72.9(4.3)**	60.1(2.1)	51.1(0.9)	56.7(1.7)	67.0(3.7)	62.4(4.4)	68.7(4.0)	68.6(3.3)	51.4(2.4)	53.9(0.8)	68.5(6.9)
Pub	**96.6(0.1)**	73.4(5.7)	73.1(0.8)	76.3(2.8)	**96.7(0.2)**	93.2(1.0)	**96.7(0.1)**	**96.7(0.1)**	54.6(1.1)	53.1(0.4)	95.5(0.4)
Photo	**97.9(0.3)**	53.6(3.9)	51.7(0.9)	53.1(0.6)	92.5(1.4)	89.5(1.9)	93.8(1.0)	93.3(0.8)	72.7(1.3)	38.1(0.7)	92.8(1.0)
Comp	**99.1(0.3)**	58.5(4.6)	47.4(0.8)	47.5(1.0)	96.8(0.7)	93.5(1.9)	97.2(0.5)	98.3(0.3)	91.8(1.1)	46.6(1.9)	**99.1(0.4)**
Avg	**92.4(10.)**	63.6(9.4)	55.0(0.8)	58.7(10.)	87.1(12.)	82.3(13.)	88.0(11.)	90.3(11.)	65.4(15.)	48.8(6.1)	90.1(13.)

Next, we visualized the learned node embeddings to quantitatively evaluate the proposed method. Figure 2 shows the node embeddings on Cora learned by Ours-AN, Ours-N, DOC-N, DOC-AN, DW, and SLGCN when 10% of all instances were labeled. For SLGCN, we used the hidden layer as the node embeddings. We used t-distributed stochastic neighbor embeddings (t-SNE) [30] to reduce the dimensions of the node embeddings from 32 to 2. As for methods that use both anomalous and normal label information for learning node embeddings (i.e., Ours-AN, DOC-AN, and SLGCN), Ours-AN was able to learn better node embeddings than the others in that it separated normal and anomalous node embeddings well. Although DOC-AN separated anomalous and normal training instances well, it did not generalize to testing instances because it does not consider the attributed graph structures. Ours-AN was able to learn useful node embeddings for detecting anomalies by using the attributed graph structures. As for methods that do not use anomalous label information for learning node embeddings, i.e., Ours-N, DOC-N, and DW, Ours-N was able to learn better node embeddings than the others in that anomalous instances was located at the end of region of normal node embeddings. Like DOC-AN, DOC-N did not generalize to testing instances although it embedded normal training instances into the small volume region. DW did not learn good node embeddings because it does not use any label information to learn node embeddings. Overall, these results showed that the proposed method can learn useful node embeddings for anomaly detection tasks on the attribute graph.

Third, we investigated the dependency of the regularization weight for the AUC regularizer λ for Ours-AN and DOC-AN, which use the regularizer. Figure 3 shows the average test AUCs by changing λ when the rate of labeled and all instances was 2.5%. The best λ of Ours-AN differed across datasets. With Photo, large λ, which corresponds to minimize the AUC loss only in (5), performed the best although the small value ($\lambda = 1$) performed better with Cora, Cite, Pub, and Comp. Overall, Ours-AN outperformed DOC-AN in almost all λ with all datasets. This result indicates the robustness of the proposed method against the regularization weight for the AUC regularizer λ.

Fourth, we investigated the dependency of the dimension of embeddings K for the proposed method. Figure 4 shows the average test AUCs by changing K when the rate of labeled and all instances was 2.5%. We compared Ours-AN and Ours-N with the embedding based methods: DOC-AN, DOC-N, DSAD, DW, DOM, and ImVerde. Ours-AN consistently performed well in all K for all datasets. As for methods that do not use anomalies for training, Ours-N performed better than DOC-N with almost all K's. Overall, these results suggest the proposed method is robust against the value of K.

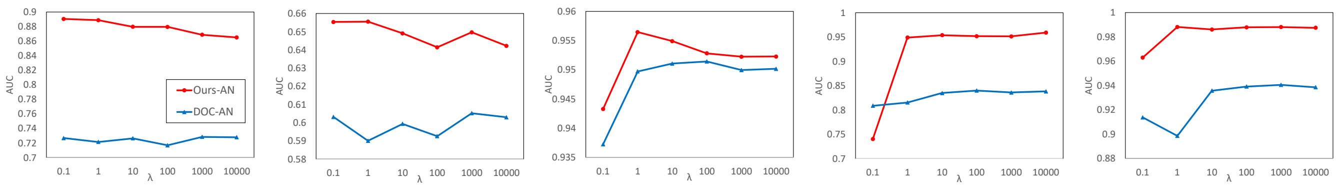
Lastly, we investigated the training times for Ours-AN and Ours-N on Cora when 10% of all instances were labeled and $K = 32$. We also investigated the training time for SLGCN since it is based on the GCNs like the proposed method. The training times of Ours-AN, Ours-N, and SLGCN were 4.92, 3.56, and 2.46 seconds, respectively. Since Ours-AN uses the AUC regularizer, it took more training time than Ours-N. However, the proposed method was able to learn fast enough.



(a) Ours-AN (b) DOC-AN (c) SLGCN

(d) Ours-N (e) DOC-N (f) DW

Fig. 2: Visualization of the learned node embeddings on Cora. Three methods in the first row use both anomalous and normal instances for learning node embeddings. Three methods in the second row do not use anomalous instances for learning node embeddings. Red and blue points represent anomalous and normal training instances, respectively. Orange and sky blue points represent anomalous and normal testing instances, respectively.



(a) Cora (b) Cite (c) Pub (d) Photo (e) Comp

Fig. 3: Average test AUCs of each dataset when λ was changed.

V. Conclusions

In this paper, we proposed a novel semi-supervised anomaly detection method on attribute graphs. The proposed method utilizes graph GCNs to extract node embeddings considering both label information of small nodes and attribute information of all nodes with the graph structure. In experiments using five real-world attributed graph datasets, the proposed method outperformed various existing anomaly detection methods in settings in which both anomalous and normal labels or only normal labels are available for training.

References

[1] L. Akoglu, M. McGlohon, and C. Faloutsos. Oddball: Spotting anomalies in weighted graphs. In *PAKDD*, 2010.

[2] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery*, 29(3):626–688, 2015.

[3] S. Bandyopadhyay, N. Lokesh, and M. N. Murty. Outlier aware network embedding for attributed networks. In *AAAI*, 2019.

[4] S. Bandyopadhyay, S. V. Vivek, and M. Murty. Integrating network embedding and community outlier detection via multiclass graph description. *ECAI*, 2020.

[5] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel. Disclosure: detecting botnet command and control servers through large-scale netflow analysis. In *ACSAC*, 2012.

[6] R. Chalapathy and S. Chawla. Deep learning for anomaly detection: A survey. *arXiv*, 2019.

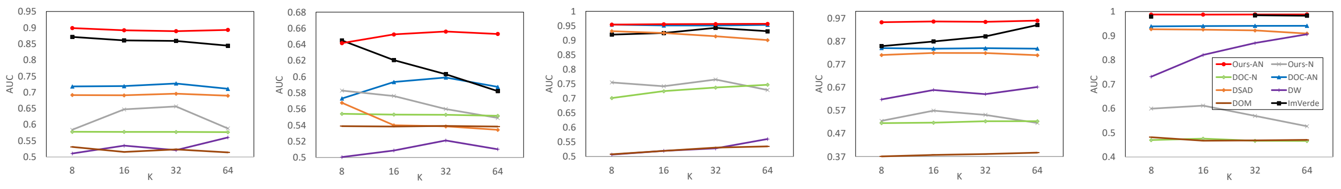
[7] R. Chalapathy, A. K. Menon, and S. Chawla. Anomaly detection using one-class neural networks. *arXiv preprint arXiv:1802.06360*, 2018.

[8] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[10] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga. Outlier detection with autoencoder ensembles. In *SDM*, 2017.

[11] Z. Chen, B. Liu, M. Wang, P. Dai, J. Lv, and L. Bo. Generative adversarial attributed network anomaly detection. In *CIKM*, 2020.



(a) Cora (b) Cite (c) Pub (d) Photo (e) Comp

Fig. 4: Average test AUCs of each dataset when K was changed.

[12] K. Ding, J. Li, N. Agarwal, and H. Liu. Inductive anomaly detection on attributed networks. In *IJCAI*, 2020.

[13] K. Ding, J. Li, R. Bhanushali, and H. Liu. Deep anomaly detection on attributed networks. In *SDM*, 2019.

[14] P. Dokas, L. Ertoz, V. Kumar, A. Lazarevic, J. Srivastava, and P.-N. Tan. Data mining for network intrusion detection. In *Proc. NSF Workshop on Next Generation Data Mining*, pages 21–30, 2002.

[15] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna. Towards detecting compromised accounts on social networks. *IEEE Transactions on Dependable and Secure Computing*, 14(4):447–460, 2015.

[16] M. Fey and J. E. Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

[17] A. Grover and J. Leskovec. node2vec: scalable feature learning for networks. In *KDD*, pages 855–864, 2016.

[18] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[19] T. Iwata and Y. Yamanaka. Supervised anomaly detection based on deep autoregressive density estimators. *arXiv preprint arXiv:1904.06034*, 2019.

[20] F. Keller, E. Muller, and K. Bohm. Hics: high contrast subspaces for density-based outlier ranking. In *ICDE*, 2012.

[21] D. P. Kingma and J. Ba. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] T. N. Kipf and M. Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[23] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.

[24] Y. Kou, C.-T. Lu, S. Sirwongwattana, and Y.-P. Huang. Survey of fraud detection techniques. In *ICNSC*, 2004.

[25] A. Kumagai, T. Iwata, and Y. Fujiwara. Transfer anomaly detection by inferring latent domain representations. In *NeurIPS*, 2019.

[26] J. Li, H. Dani, X. Hu, and H. Liu. Radar: residual analysis for anomaly detection in attributed networks. In *IJCAI*, 2017.

[27] Y. Li, X. Huang, J. Li, M. Du, and N. Zou. Specae: spectral autoencoder for anomaly detection in attributed networks. In *CIKM*, 2019.

[28] J. Liang, P. Jacobs, J. Sun, and S. Parthasarathy. Semi-supervised embedding in attributed networks with outliers. In *SDM*, 2018.

[29] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *ICDM*, 2008.

[30] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[31] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos. Anomaly detection in crowded scenes. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1975–1981. IEEE, 2010.

[32] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel. Image-based recommendations on styles and substitutes. In *SIGIR*, 2015.

[33] N. Mehta, L. Carin, and P. Rai. Stochastic blockmodels meet graph neural networks. In *ICML*, 2019.

[34] M. Mesgaran and A. B. Hamza. Graph fairing convolutional networks for anomaly detection. *arXiv preprint arXiv:2010.10274*, 2020.

[35] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[36] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[37] Z. Peng, M. Luo, J. Li, H. Liu, and Q. Zheng. Anomalous: a joint modeling approach for anomaly detection on attributed networks. In *IJCAI*, 2018.

[38] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: online learning of social representations. In *KDD*, 2014.

[39] Y. Rong, W. Huang, T. Xu, and J. Huang. Dropedge: towards deep graph convolutional networks on node classification. In *ICLR*, 2020.

[40] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. Vandermeulen, A. Binder, E. Müller, and M. Kloft. Deep one-class classification. In *ICML*, 2018.

[41] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft. Deep semi-supervised anomaly detection. In *ICLR*, 2019.

[42] M. Sakurada and T. Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, page 4. ACM, 2014.

[43] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[44] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[45] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

[46] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: large-scale information network embedding. In *WWW*, 2015.

[47] D. M. Tax and R. P. Duin. Support vector data description. *Machine learning*, 54(1):45–66, 2004.

[48] H. Tong and C.-Y. Lin. Non-negative residual matrix factorization with application to graph anomaly detection. In *SDM*, 2011.

[49] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *ICLR*, 2017.

[50] X. Wang, Y. Du, P. Cui, and Y. Yang. Ocgnn: one-class classification with graph neural networks. *arXiv preprint arXiv:2002.09594*, 2020.

[51] F. Wu, T. Zhang, A. H. d. Souza Jr, C. Fifty, T. Yu, and K. Q. Weinberger. Simplifying graph convolutional networks. In *ICML*, 2019.

[52] J. Wu, J. He, and Y. Liu. Imverde: vertex-diminished random walk for learning imbalanced network representation. In *Big Data*, 2018.

[53] C. Xu, Z. Cui, X. Hong, T. Zhang, J. Yang, and W. Liu. Graph inference learning for semi-supervised classification. In *ICLR*, 2020.

[54] X. Xu, N. Yuruk, Z. Feng, and T. A. Schweiger. Scan: a structural clustering algorithm for networks. In *KDD*, 2007.

[55] Y. Yamanaka, T. Iwata, H. Takahashi, M. Yamada, and S. Kanai. Autoencoding binary classifiers for supervised anomaly detections. In *PRICAI*, 2019.

[56] Z. Yang, W. W. Cohen, and R. Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *ICML*, 2016.

[57] C. Zhou and R. C. Paffenroth. Anomaly detection with robust deep autoencoders. In *KDD*, 2017.

[58] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NeurIPS*, 2004.

[59] D. Zhou, J. He, H. Yang, and W. Fan. Sparc: self-paced network representation for few-shot rare category characterization. In *KDD*, 2018.

[60] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.

[61] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. *ICLR*, 2018.