Dual-discriminative Graph Neural Network for Imbalanced Graph-level Anomaly Detection

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph-level anomaly detection aims to distinguish anomalous graphs in a graph dataset from normal graphs. Anomalous graphs represent very few but essential patterns in the real world. The anomalous property of a graph may be referable to its anomalous attributes of particular nodes and anomalous substructures referring to a subset of nodes and edges in the graph. In addition, due to the imbalance nature of anomaly problem, the anomalous information will be diluted by normal graphs with overwhelming quantities. Various anomaly notions in the attributes and/or substructures and the imbalance nature together make detecting anomalous graphs a non-trivial task. In this paper, we propose a dual-discriminative graph neural network for graph-level anomaly detection, namely iGAD. Specifically, an anomalous graph attribute-aware graph convolution and an anomalous graph substructure-aware deep Random Walk Kernel (deep RWK) are welded into a graph neural network to achieve a dual-discriminative ability on anomalous attributes and substructures. The deep RWK in iGAD makes up for the deficiency of graph convolution in distinguishing structural information caused by the simple neighborhood aggregation mechanism. Further, we propose a Point Mutual Information-based loss function to address the imbalance nature of anomaly problem. The loss function enables iGAD to capture the essential correlation between input graphs and their anomalous/normal properties. We evaluate iGAD on four real-world graph datasets. Extensive experiments demonstrate the superiority of iGAD on the graph-level anomaly detection task.

1 Introduction

Graphs have been used as an indispensable tool to describe relational data, such as chemical compounds, transport networks, and social networks. For a graph that models a chemical compound, nodes represent atoms and edges describe the chemical bonds between atoms. Learning graphs to understand the world has been drawing considerable attention in recent years [1, 2, 3, 4]. Given a graph dataset containing a set of graphs, the graph-level anomaly detection task aims to detect graphs having anomalous information. Generally, these anomalous graphs with distinct characteristics or functions are the minority in a graph dataset. For example, among all antibody molecules, only a small part of molecules exhibit the anti-cancer activity on a specific cancer cell line [5]. Graph-level anomaly detection is a research topic of great practical value, say, it can support cancer drug discovery by detecting molecules with anti-cancer activity from other antibody molecules [6]. Also, the research can help to reveal pathogenic brain mechanisms by distinguishing the brain structures of patients with neurological disorders from healthy individuals [7].

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In practice, the attributes of particular nodes are a contributing factor for a graph's anomalous properties. For example, the hydrogen and oxygen atoms at the end side of molecules are the key to discriminating between Alcohol and Alkane. Graph substructures composed of a subset of nodes and edges are another factor for a graph's anomalous properties. For instance, molecular toxicity can be induced by the pharmacophores consisting of some graph substructures, such as triangles and cycle-based graphlets. These graph attributes and substructures that are essential for distinguishing anomalous graphs from normal graphs are regarded as *anomalous attributes* and *anomalous substructures*. Most existing graph mining methods, especially those graph convolution-based models which have achieved state-of-the-art performance [8, 9, 10, 11], focus on the aggregation of the neighbor-structure information. However, the anomalous structure information could be all possible substructures. In addition, the anomalous attributes and substructures will be diluted by the information of normal graphs with overwhelming quantities, which is the *imbalance nature* of the anomaly problem. In summary, various anomaly notions in attributes and/or substructures and the imbalance nature make the graph-level anomaly detection an important yet challenging problem.

In this paper, we propose a dual-discriminative graph neural network for **i**mbalanced **G**raph-level **A**nomaly **D**etection (**iGAD**). Specifically, iGAD investigates a novel anomalous attribute-aware graph convolution, which separates self-representation and neighborhood-representation when representing a node in a graph. In this way, the anomalous attribute can be exposed in the graph convolution without being overshadowed by its neighbors, and the attributes of each node in a graph can evolve individually. To enable the anomalous substructure awareness ability for iGAD, solely relying on graph convolution is tricky. The graph convolution performs message passing depending on the neighborhood structure only and cannot explore all possible substructures in a graph. Even the simple substructures like triangles, graph convolution is unable to count them in a graph [11]. Graph Kernels like the Random Walk Kernel (RWK) [12], as another manner to learn graphs, naturally involve the substructure comparison among graphs. RWK learns graphs by exploring common random walk sequences among all input graphs. To encode anomalous structural information into graph representations, we propose to compare random walk sequences of input graphs against learnable anomalous substructures. Anomalous substructures are modeled as learnable parameters that can be updated through back-propagation. We call this design anomalous substructure-aware deep RWK. In addition, we propose a new loss function to target the imbalance nature of anomaly problem caused by the great difference in the number of normal and anomalous graphs. The newly proposed loss function enables iGAD to capture anomalous properties from an imbalanced graph dataset by fitting the Point Mutual Information (PMI) between graphs and their normal/anomalous properties. Welding three designs mentioned above into a graph neural network, the graph-level anomaly detection model iGAD is achieved. We conclude the contributions of iGAD as follows:

- The anomalous attribute-aware graph convolution highlights the anomalous attributes of particular nodes by keeping the individual evolution of nodes' attributes.
- The anomalous substructure-aware deep RWK learns anomalous substructures, and compares the input graphs' and learned anomalous substructures' random walk sequences.
- The PMI-based loss function addresses the imbalance nature of the anomaly problem and enables iGAD to fit anomalous graphs in the minority effectively.
- Extensive experiments on four real-world graph datasets about chemical compounds demonstrate the superiority of iGAD. The code is available at Anonymous GitHub.

2 Related Work

Anomaly Detection. The task aims to identify samples that deviate from the majority of samples. Early anomaly detection algorithms employ quantitative indicators [13], such as distance and density, to describe the degree of deviation between samples to detect the anomalous ones. In addition, as one of the representatives, isolation forest-based algorithms [14, 15] detect the anomalous by building the binary search tree structure to isolate samples. Due to the small number of anomalous samples and the high cost of labeling, anomaly detection algorithms based on one-class classification [16, 17, 18, 19]

and autoencoder [20, 21] also draw the attention of practitioners. For the anomaly detection task on the graph-structured data, most algorithms dedicate to the anomaly detection task in a single graph [22, 23], such as identifying anomaly nodes [24, 25], edges [26, 27], and communities [28].

Graph-level Anomaly detection. To the best of our knowledge, there is only a few preliminary works for this task. [29] investigated a research question of "should we use graph classification data to evaluate the anomaly detection model?" and this task can be regarded as a special case of graph classification. But graph classification algorithms assume that the number of graphs in each class is balanced. Graph kernels (GKs) [12, 30, 31, 32, 33] used to be the mainstream method for graph classification. Generally, they calculate pairwise kernel values (*i.e.*, similarity) of graphs through well-defined kernel functions to obtain the Gram matrix. Then, an off-the-shelf classifier can be applied to classify graphs. In recent years, researchers pay more attention to Graph Neural Networks (GNNs) [34, 35, 36, 37] since graph classification algorithms based on GKs cannot learn graph representations explicitly and be optimized in an end-to-end fashion. GNNs based on spatial convolution [10, 38, 39] are summarized as message-passing framework (MPNNs) [40, 8, 9, 41, 42]. In general, MPNNs updates the representation of each node by aggregating information coming from their neighbors and applying the readout function or pooling on nodes to obtain graph representations. The spatial graph convolution that mix the representation of nodes and their neighborhoods to represent nodes will smooth node attributes [43]. In addition, [44, 45, 46] proved that spatial graph convolution has limited power in distinguishing graph structures. Performing message passing between subgraphs can enable spatial graph convolution to capture more graph structural information. But such a method consumes exponentially increases memory and computation complexities [11]. All the above algorithms do not consider anomalous attributes and substructures. Moreover, they are likely to under-fit the anomalous graphs since they do not propose any solutions for the issue of imbalanced data distribution.

Imbalanced Learning. Learning data with imbalanced distribution can be divided into the data- and algorithm-level approaches. The former alleviates the imbalanced distribution of the original data by either under-sampling the majority [47] or over-sampling the minority [48]. However, modifying the original distribution of data will bring a lot of drawbacks inevitably, such as over-fitting and discarding important samples. There are different types of approaches address the issue in algorithm-level. The typical approaches include employing post-hoc correction [49], aligning data distribution, [50] and modifying the loss function [51, 52]. Considering the different consequences of false-positive identification (*e.g.*, predicting an inactive chemical compound to be active) and false-negative identification (*e.g.*, predicting a cancer patient as a healthy individual), the situation will be more complicated in the real-world application [53].

3 Preliminaries

Notations. The graph dataset consisting of N graphs is denoted as $\mathcal{G} = \{G_1, ..., G_N\}$. $G_i = \{V_i, E_i\}$ describes a graph in $\mathcal{G}$. V_i and E_i are the node and edge set, respectively. $\mathbf{x}_u \in \mathbb{R}^c$ is the attribute vector of node $u \in V$. The graph structure of G_i can be described by an adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$ ($\mathbf{A}_{uu} = 0$), where $\mathbf{A}_{uv} = 1$ if there is an edge between nodes u and v, otherwise 0. The graph G_i has a label $y_i \in \{0,1\} \in \mathcal{Y}$, where 1 represents *anomalous* and 0 represents *normal*.

Problem Definition. This paper concentrates on the supervised graph-level anomaly detection problem. Given the training set $\mathcal{T} = \{(G_1, y_1), (G_2, y_2), ...\}$, we aim to train a model to predict the normal/anomalous proprieties of unseen graphs. $\mathcal{T}$ contains N_1 *anomalous* graphs and N_0 *normal* graphs ($N_1 \ll N_0$). The imbalanced ratio of $\mathcal{T}$ is defined as δ ($\delta = \frac{N_1}{N_0 + N_1}$).

Spatial Graph Convolution Layer. It updates each node's representation in a graph by aggregating the features coming from its one-hop neighbors iteratively. The formulation is:

$$\mathbf{h}_u^{(k)} = f\left(\mathbf{h}_u^{(k-1)}, f_{Aggregate}\left(\{\mathbf{h}_v^{(k-1)}, v \in \mathcal{N}(u)\}\right)\right), \mathbf{h}_u^{(0)} = \mathbf{x}_u, \tag{1}$$

where $\mathcal{N}(u)$ denotes the one-hop neighborhood of node u. $f_{Aggregate}$ denotes how to the aggregate neighborhood information of node u, and f is the embedding function.

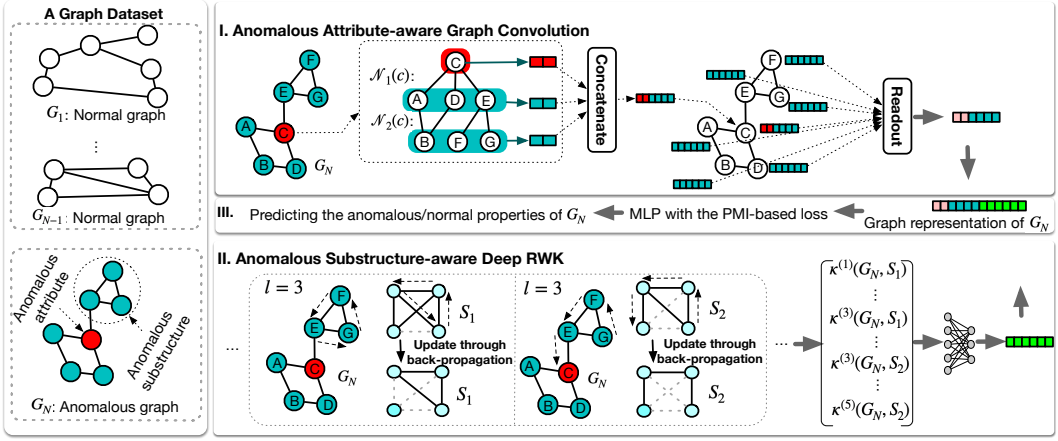

Figure 1: Taking learning the anomalous/normal properties of the graph G_N in a graph dataset as an example, we illustrate the framework of iGAD. The anomalous attribute-aware graph convolution separates the encoding on nodes and their neighbors when it represents nodes in G_N. A readout function is applied to all node representations to represent G_N. The anomalous substructure-aware deep RWK regards two anomalous substructures (*i.e.*, S_1 and S_2) as learnable parameters. It performs random walks (max random walk length $L = 5$) on these learnable anomalous substructures and the input graph G_N simultaneously. Then, deep RWK represents G_N based on kernel values. Finally, an MLP equipped with the PMI-based loss function is employed to predict G_N.

Random-walk Graph Kernels (RWK). Given two graphs $G_1 = \{V_1, E_1\}$ and $G_2 = \{V_2, E_2\}$, the direct product graph is defined as $G_\times = \{V_\times, E_\times\}$, where $V_\times = \{(u, u')|u \in V_1, u' \in V_2\}$ and $E_\times = \{((u, u'), (v, v'))|(u, v) \in E_1, (u', v') \in E_2\}$. Performing the random walks on $G_\times$ acts as doing random walks on G_1 and G_2 simultaneously. The adjacency matrix of direct product graph $\mathbf{A}_\times$ can be obtain by the Kronecker product operation, that is $\mathbf{A}_\times = \mathbf{A}_1 \otimes \mathbf{A}_2$. The number of l-length common walk sequence between G_1 and G_2 is $\kappa^{(l)}(G_1, G_2) = \mathbf{q}_\times^\intercal (A_\times)^l \mathbf{p}_\times$, where $\mathbf{p}_\times$ and $\mathbf{q}_\times$ are the vector about the starting and stopping probabilities, respectively. They should be all ones when there is no prior information. The RWK about G_1 and G_2 can be defined as:

$$\mathcal{K}(G_1, G_2) := \sum_{l=0}^{\infty} \kappa^{(l)}(G_1, G_2). \tag{2}$$

4 iGAD

In this section, we will describe iGAD in detail (see Figure 1 for the illustration). iGAD contains three key designs: (1) anomalous graph attribute-aware graph convolution; (2) anomalous substructure-aware deep RWK; and (3) the PMI-based loss function. The first two designs encode the anomalous information on graph attributes and substructure into graph embeddings, respectively. The loss function empowers iGAD to get rid of the domination of normal graphs with overwhelming quantities.

4.1 iGAD: Anomalous Attribute-aware Graph Convolution

As aforementioned in Section 1, the attributes of particular nodes can make a graph to be anomalous. A graph convolution that has anomalous graph attribute awareness should violate the basic assumption in most graph convolution, that a node has similar properties to its neighborhood in a graph. Following this assumption and mixing the information of nodes with their neighborhoods together will smooth those node attributes that achieve an anomalous graph [43]. And the anomalous attribute information will become more obscure in the anomalous graphs' embeddings. We propose that making each node's attribute can evolve independently during the graph convolution process is critical for capturing anomalous attributes. To achieve that, the graph convolution in iGAD separates the encoding on a node itself from its neighborhood when representing nodes, that is,

$$\mathbf{h}_u = f_{Combine}\left(\mathbf{h}^{(0)}(u), \mathbf{h}(\mathcal{N}_1(u)), ..., \mathbf{h}(\mathcal{N}_K(u))\right), \tag{3}$$

where $\mathbf{h}_u$ is the representation of node $u \in V_i$, $\mathcal{N}_k(u)$ represents the kth-hop neighborhood of the node u, $\mathbf{h}(\mathcal{N}_k(u)$ is the representation of node u's kth-hop neighborhood, and $f_{Combine}$ denotes the concatenation operator. $\mathbf{h}(u)$ and $\mathbf{h}(\mathcal{N}_k(u))$ are formulated as:

$$\mathbf{h}^{(0)}(u) = \sigma(\mathbf{x}_u \mathbf{W}^{(0)}), \tag{4}$$

and

$$\mathbf{h}(\mathcal{N}_k(u)) = \sigma(f_{Aggregate}\{\mathbf{h}^{(0)}(v), v \in \mathcal{N}_k(u)\}\mathbf{W}^{(k)}), \tag{5}$$

respectively, where σ represents the activation function, $f_{Aggregate}$ can be the sum or average aggregator, and $\mathbf{W}^{(k)}, k \in \{0, ..., K\}$ is the trainable matrices. To obtain the entire graph representation $\mathbf{h}_{G_i}^{Conv}$, we apply the permutation invariant Readout function on all nodes in the graph G_i,

$$\mathbf{h}_{G_i}^{Conv} = \text{Readout}\{\mathbf{h}_u | u \in V_i\}, \tag{6}$$

where Readout can be the sum or mean operation.

4.2 iGAD: Anomalous Substructure-aware Deep RWK

The neighborhood aggregation mechanism in graph convolution utilizes the neighbour-structure as the clue for message passing, and cannot figure out all possible substructures in a graph [46, 54]. To make iGAD to be anomalous substructure-aware, we turn to the RWK that can explore all possible substructures by performing random walks in a graph. [55] proposed the pioneering work that represents graphs through doing random walks on input graphs and some substructures simultaneously. This work inspires us to encode anomalous substructures into graph representations by comparing input graphs and anomalous substructures. Hence, we propose an anomalous substructure-aware deep RWK in which the anomalous substructures are modeled by learnable parameters that can be updated through back-propagation. Supposed that $\{S_1, ..., S_M\}$ is the set of M anomalous substructures, and the size of S_m is n'. $\mathbf{P}_m \in \mathbb{R}^{n' \times n'}$ is the adjacency matrix of S_m, where $\mathbf{P}_m(u', v')$ denotes the possibility that an edge exists between nodes u' and v' in S_m. Due to the symmetry of the adjacency matrix, $\mathbf{P}_m$ contains $\frac{n'(n'-1)}{2}$ learnable parameters. Refer to Eq. 2 and suppose that the starting and the stopping possibilities on each node are all one, the number of l-length common random walk sequences between the input graph G_i and the anomalous substructure S_m is:

$$\kappa^{(l)}(G_i, S_m) = \mathbf{1}^{\mathsf{T}}(\mathbf{A}_i \otimes \mathbf{P}_m)^l \mathbf{1} = \sum_i \sum_j \left[(\mathbf{P}_m)^l \left[\mathbf{E}(\mathbf{A}_i)^l\right]\right]_{i,j}, \tag{7}$$

where $\mathbf{1} \in \mathbb{R}^{n'n}$ is a column vector full of 1, and $\mathbf{E} \in \mathbb{R}^{n' \times n}$ is a matrix full of 1. Random walk kernels with the setting of infinite walk length will stuck in the halting issue [56]. Therefore, iGAD only considers the random walk length from 1 to L. The proof of Eq. 7 can be found in **Appendix A.1**. To improve the algorithm efficiency, we can calculate $\mathbf{E}(\mathbf{A}_i)^l$ through the sparse matrix multiplications, that is $\mathbf{E}(\mathbf{A}_i^s)^l = ((\mathbf{E}\mathbf{A}_i^s)\mathbf{A}_i^s)...)\mathbf{A}_i^s)$, where $\mathbf{A}_i^s$ is the sparse representation of $\mathbf{A}_i$. As the size of anomalous substructures (n') is usually small, optimizing $(\mathbf{P}_m)^l$ is not necessary. We put all kernel values about G_i and S_m, $m \in \{1, .., M\}$ into a vector, that is,

$$\mathbf{a}_{G_i} = \left(\kappa^{(1)}(G_i, S_1), ..., \kappa^{(1)}(G_i, S_M); ...; \kappa^{(L)}(G_i, S_1), ..., \kappa^{(L)}(G_i, S_M)\right), \tag{8}$$

where $\mathbf{a}_{G_i} \in \mathbb{R}^{ML}$. The graph representation $\mathbf{h}_{G_i}^{ker}$ based on deep RWK is:

$$\mathbf{h}_{G_i}^{ker} = \text{MLP}(\mathbf{a}_{G_i}), \tag{9}$$

where MLP is the multi-layer perceptrons.

To get the final graph embedding $\mathbf{h}_{G_i}$ for $G_i \in \mathcal{G}$, we concatenate the $\mathbf{h}_{G_i}^{Conv}$ and $\mathbf{h}_{G_i}^{Ker}$ together. Based on $\mathbf{h}_{G_i}$, the anomalous attribute-aware graph convolution and anomalous substructure-aware deep RWK are welded into a graph neural network, and these two parts can be updated through the back-propagation simultaneously.

4.3 iGAD: PMI-based Loss Function

The number of normal graphs is much more than that of anomalous graphs. Under the imbalance nature of anomaly problem, normal graphs will dominate the machine learning model. To address this issue, we propose a new loss function that can model the Pointwise Mutual Information (PMI) between input graphs and their anomalous/normal properties. Specifically, we use $f_{y_i}(G_i;\Theta)$ to denote the proposed model iGAD, where Θ represents model parameters. iGAD models the PMI between input graph G_i and its anomalous/normal properties can be formulated as:

$$f_{y_i}(G_i;\Theta) \sim \log \frac{P_\Theta(G_i, y_i)}{P(G_i)P(y_i)}, \tag{10}$$

where the right part represents the PMI between G_i and y_i, $P(y_i)$ is the proportion of anomalous graphs or normal graphs in the training set $\mathcal{T}$. Compared with the most existing loss, especially the cross-entropy loss that models the conditional distribution (that is $f_{y_i}(G_i;\Theta) \sim P_\Theta(y_i|G_i)$), modeling PMI can make the model give priority to learning the information that reveals the essential correlation between graphs and their anomalous/normal proprieties. Eq. 10 can be reformatted as:

$$\log P_\Theta(y_i|G_i) \sim f_{y_i}(G_i,\Theta) + \log P(y_i). \tag{11}$$

After softmax normalization, we can obtain the distribution of given G_i concerning y_i:

$$P_\Theta(y_i|G_i) = \frac{e^{f_{y_i}(G_i;\Theta)+\log P(y_i)}}{\sum_{y_i} e^{f_{y_i}(G_i;\Theta)+\log P(y_i)}}. \tag{12}$$

Incorporating $P_\Theta(y_i|G_i)$ in to the framework of cross-entropy loss, we can obtain a new loss function $\mathcal{L}$:

$$\mathcal{L} = \frac{1}{N_0+N_1}\sum_{i=1}^{N_0+N_1} -\log P_\Theta(y_i|G_i). \tag{13}$$

Specifically, iGAD predict the graph G_i through:

$$y_i^* = \begin{cases} \underset{y_i}{\operatorname{argmax}} f_{y_i}(G_i;\Theta) + \log P(y_i), & \text{in the training phase} \\ \underset{y_i}{\operatorname{argmax}} f^*_{y_i}(G_i;\Theta), & \text{in the testing phase} \end{cases} \tag{14}$$

The reason that we use $\underset{y_i}{\operatorname{argmax}} f^*_{y_i}(G_i;\Theta)$ to predict a graph in the testing phase is in **Appendix A.2**.

There are other alternative loss functions to solve the imbalanced data distribution of anomaly problem, the typical representative is the cost-sensitive loss function that assigns or learns different weights to the majority and the minority in data [57, 58, 59]. Compared with them, the PMI-based loss function does not introduce any new parameters that need to be carefully tuned. It comes in the form of the most widely used cross-entropy loss function but outputs the prediction results that can realize the max mutual information with the input data.

4.4 Algorithm Description&Time Complexity

We include the pseudocode of iGAD in appendix (see **Algorithm** 1). We employ the mini-batch training techniques [60]. But for illustrating how iGAD predicts a graph more clearly, in the pseudocode, we suppose that the batch size is 1. Given a set of labeled graphs, we initialize anomalous substructures and other parameters (Line 1), and calculate the proportion of normal and anomalous graphs (Line 2) at the beginning. For each graph, iGAD learns its graph representation (Lines 3 to 15). Based on the graph representation, an MLP equipped with the PMI-based loss function predicts the graph (Lines 16 and 17). At the back-propagation stage, anomalous substructures and other parameters will be updated. In the next epoch, deep RWK performs random walk on the input graphs and the updated anomalous substructures.

Lemma 1 *The time complexity of iGAD is $\mathcal{O}(n+|E|)$, where n and $|E|$ is the number of nodes and edges in the largest graph. (We include the proof in **Appendix A.3**).*

Table 1: Statistical information of datasets. We regard the one-hot encoding of node labels (*e.g.,* carbon (C), nitrogen (N), oxygen (O)) as node attributes.

Dataset	**#Graphs** (N)	**#Anomalous Graphs**	δ	**Avg. #Nodes (Avg.** n**)**	**Avg. #edges (Avg.** $\lvert E \rvert$**)**	**# Node Labels** (c)	**Tumor Description**
SW-620	40,532	2,411	5.95%	26.05	28.08	65	Colon
MOLT-4	39,765	2,904	7.90%	26.07	28.13	64	Leukemia
MCF-7	27,770	2,293	8.26%	26.39	28.52	46	Breast
PC-3	27,509	1,568	5.70%	26.35	28.49	45	Prostate

5 Experiments

In this section, we conduct a series of experiments to study the performance of iGAD on the graph-level anomaly detection task. Specifically, we investigate the following questions:

Q1. How efficient is iGAD on the graph-level anomaly detection task?
Q2. How efficient is the anomalous graph attribute-aware graph convolution in iGAD?
Q3. How efficient is iGAD when it utilizes pre-defined anomalous substructures?
Q4. Can the PMI-based loss function improve models' fitting on imbalanced graph data distribution?
Q5. Can the anomalous substructure-aware deep RWK in iGAD learn the anomalous substructures?
Q6. How does the max length of random walk sequences influence the performance of iGAD?

The questions from **Q2** to **Q4** can be regarded as the ablation study on our three main design elements, respectively: (a). regular convolution versus the proposed graph convolution; (b). RWK versus the proposed deep RWK; (c). cross-entropy loss function versus the proposed loss function.

5.1 Experimental Setup

Datasets. SW-610, MOLT-4, PC-3, and MCF-7 are four real-world graph datasets of chemical compounds. These datasets are collected from the PubChem website. PubChem records the anti-cancer activity testing result ("active" or "inactive") of compounds on different types of cancer cell lines. We regard the compound with the outcome "active" as anomalous, and "inactive" as normal. For example, the graph dataset SW-620 is about the anti-cancer activity of a series of compounds on the colon cancer cell line. Compounds that exhibit antibody activity against colon cancer are labeled as anomalous graphs. The statistical information about these four datasets is shown in Table 1.

Baselines&Experimental Settings. To investigate **Q1**, we compare iGAD with the following baseline models: (1) GCN+readout (abbreviated GCN) and GIN [8], these two algorithms employ message-passing graph convolution to learn node representation and apply readout on node representations to obtain graph representations and detect anomalous graphs; (2) DGCNN [9] and SOPool [42]. In these two models, message passing-based graph convolution and graph pooling are employed to complete the graph-level anomaly detection; (3) RWGNN [55] employs random walk sequences to represent graphs and then detect anomalous graphs based on graph representations. All the above baselines use the cross-entropy loss. For the fairness of experiments and to answer **Q2**, we also compare the performance of iGAD with the above baselines in which the loss is substituted by the proposed PMI-based loss function. We denote these baselines with the PMI-based loss function as GCN-GAD, GIN-GAD, DGCNN-GAD, SOPool-GAD, and RWGNN-GAD. To answer **Q3**, we consider a variant iGAD-1 that only utilizes the anomalous graph attribute-aware graph convolution to detect anomalous graphs. To answer **Q4**, we consider the other variant iGAD-2 in which the random walk compares the input graphs against the pre-defined anomalous graph substructures. Specifically, 3-star, triangle, tailed triangle, and 4-cycle graphlets are pre-defined graph anomalous substructures (see Figure 4 in Appendix for these graphlets). The parameter setting and the implementation about all the above models are shown in Appendix B.1.

We use 5-fold cross-validation to conduct experiments. We take the stratified sampling to divide the training and the testing set concerning the division ratio 80% to 20%. Stratified sampling ensures that training and testing sets have a consistent ratio between normal and anomalous graphs.

Table 2: Graph-level anomaly detection performance (%) on four datasets. The number behind $\pm$ denotes the standard deviation (stdev) over different data splits. The best result is highlighted in gray. The description about evaluation metrics *AUC*, *Recall*, *Recall(A)*, and *F-score* is in Appendix B.1.

Data	Metrics	GCN	GCN-GAD	GIN	GIN-GAD	DGCNN	DGCNN-GAD	SOPool	SOPool-GAD	RWGNN	RWGNN-GAD	iGAD-1	iGAD-2	iGAD
SW-620	AUC	74.90±0.74	76.48±1.51	78.61±2.85	76.42±2.80	80.06± 0.42	79.36±1.28	75.51± 5.06	76.89±2.07	73.37±0.36	74.92±1.97	85.40±0.93	85.77±0.72	85.82±0.69
	Recall	51.79±0.47	69.65±1.39	56.62±3.97	70.38±2.18	56.02±1.16	72.65±0.63	56.01±2.29	69.97±1.69	51.43±0.87	68.32±1.76	78.60±0.97	79.08±1.31	79.64±0.83
	Recall(A)	3.78±0.95	67.26±1.88	14.40±8.57	64.11±6.83	12.74±2.41	67.84±1.78	13.28±5.05	62.32±6.23	3.11±1.82	63.82±5.40	74.23±3.66	73.24±2.52	74.27±2.99
	F-score	52.00±0.87	52.44±1.32	58.47±5.43	55.00±2.82	58.73±1.64	56.11±0.89	58.11±2.86	55.63±4.78	51.31±1.62	52.47±3.17	61.82±2.22	63.36±1.58	63.68±1.56
MOLT-4	AUC	72.55±0.52	72.88±0.99	75.86±1.60	75.43±2.82	76.50± 0.60	77.43±0.73	75.11± 0.97	74.50±1.69	71.30±1.23	71.51±0.70	81.59±1.10	82.73±1.12	83.59±1.07
	Recall	51.26±0.54	66.84±0.66	56.87±6.98	69.40±2.47	55.35±1.27	70.14±0.69	55,31±3.74	69.12±1.50	51.39±1.41	65.84±1.27	74.97±1.61	76.33±1.18	76.82±0.82
	Recall(A)	2.74±1.12	67.32±3.04	17.71±9.02	64.36±7.14	11.69±2.87	65.76±1.97	12.23±9.34	63.44±6.55	3.18±3.21	61.24±3.89	68.18±2.72	70.54±4.28	72.10±2.47
	F-score	50.53±1.03	51.25±1.50	55.43±6.52	55.86±2.01	57.33±1.76	56.22±1.32	56.20±3.64	56.02±3.17	50.67±2.64	52.69±2.60	62.22±0.48	63.12±1.31	63.30±1.17
PC-3	AUC	75.36±2.13	75.62±1.22	78.44±1.67	76.95±1.74	79.15± 1.84	78.90±1.70	69.37± 1.53	78.24±1.97	76.27±0.86	76.10±1.92	85.44±1.20	85.83±1.02	86.04±1.14
	Recall	50.72±0.59	69.46±1.57	57.43±4.61	71.49±1.27	52.88±0.28	72.53±1.40	56.78±3.79	71.44±1.53	51.02±1.53	69.90±1.74	78.31±0.76	79.94±0.64	79.59±0.41
	Recall(A)	1.53±1.26	65.29±5.69	16.31±10.47	68.22±6.76	6.05±0.57	70.06±5.19	15.67±8.98	69.60±9.54	2.17±3.26	66.62±3.03	70.45±3.29	75.35±0.74	75.69±1.64
	F-score	49.99±1.63	52.76±1.11	59.07±4.15	54.09±3.04	54.01±0.48	54.46±1.70	57.80±3.74	51.76±5.27	50.44±2.76	52.72±1.54	61.17±2.43	62.97±1.40	63.50±0.73
MCF-7	AUC	72.70±1.05	73.89±0.99	69.54±1.15	75.90±1.44	76.41± 0.81	76.83±0.48	75.64± 2.17	76.69±1.33	70.47±1.26	73.25±1.56	83.02±0.46	83.49±1.03	83.22±0.64
	Recall	50.42±0.28	67.14±1.13	57.08±3.82	69.40±2.32	55.11±1.11	70.74±0.63	56.82±3.57	68.88±1.07	52.09±1.06	66.99±1.91	75.80±0.81	76.81±0.97	76.77±0.47
	Recall(A)	0.96±0.63	61.79±2.14	16.69±9.92	59.48±10.14	11.07±2.33	69.37±3.05	15.34±8.20	63.92±7.30	4.97±5.24	57.73±11.16	67.67±2.54	71.59±2.19	73.38±3.59
	F-score	48.76±0.59	53.37±1.89	58.27±3.20	58.71±2.53	56.99±1.70	55.74±1.54	56.82±3.57	54.82±3.00	51.65±3.46	56.06±2.93	61.56±1.89	63.75±1.18	64.70±2.58

5.2 Experimental Results

A1. iGAD is effective in graph-level anomaly detection. Observed from Table 2 that: 1) Compared with GCN, GIN, DGCNN, SOPool, and RWGNN, iGAD achieves the superior performance on all datasets in terms of all evaluation metrics; 2) iGAD also beats GCN-GAD, GIN-GAD, DGCNN-GAD, SOPool-GAD, and RWGNN-GAD concerning all evaluation metrics. Overall, iGAD exceeds all baselines including the models that are extended with the proposed PMI-based loss function.

A2. The experiment results indicate that the anomalous attribute-aware graph convolution that separates the nodes and their neighborhood when representing nodes is effective in highlighting anomalous attributes. Observed from Table 2, iGAD-1 significantly outperforms all baseline algorithms on all datasets in terms of *AUC*, *Recall*, and *F-score*. Concerning *Recall(A)*, on the dataset MCF-7, iGAD-1 has a little performance degradation compared with DGCNN-GAD. Specifically, DGCNN-GAD achieves $69.37_{\pm 3.05}$ on MCF-7, while iGAD-1 gets $67.67_{\pm 2.54}$. But iGAD-1 beats DGCNN-GAD on other datasets in terms of *Recall(A)*.

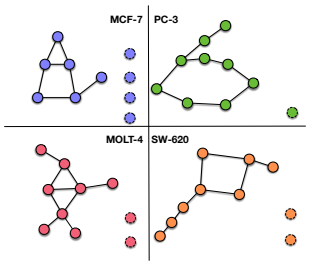

Figure 2: We draw anomalous substructures learned by iGAD under the setting with M=1, n'=10, L=5. We only draw the edge with the learnt weight over 0.5.

A3. As shown in Table 2, iGAD-2 achieves obvious advantages against all baseline models. Moreover, iGAD-2 also beats iGAD-1 on MOLT-4, PC-3, and MCF-7 regarding all evaluation metrics. Not surprisingly, iGAD beats iGAD-2 on all datasets concerning *AUC*, *Recall(A)*, and *F-score*. While on PC-3 and MCF-7, iGAD-2 has a slight lead over iGAD regarding *Recall*. The experiment results demonstrate that in most cases, the performance of iGAD equipped with a deep random kernel is better than using pre-defined anomalous substructures and algorithms that do not consider anomalous substructures. Based on the above analysis, we can conclude that comparing the same number of random walk sequences between the input graph and pre-defined substructure can improve anomalous graph detection to a certain extent, but the algorithm that can learn the anomalous substructures is better.

A4. The PMI-based loss function can capture essential correlations between graphs and anomalous/normal proprieties even if the data distribution is extremely imbalanced. As shown in Table 2, compared with the baseline models with the cross-entropy loss (GCN, GIN, DGCNN, SOPOOL, and RWGNN), the baselines equipped with the newly proposed PMI-based loss function (GCN-GAD, GIN-GAD, DGCNN-GAD, SOPOOL-GAD, and RWGNN-GAD) witness significant experimental result increases regarding *Recall* and *Recall(A)* on all datasets. For example, on the dataset SW-620,

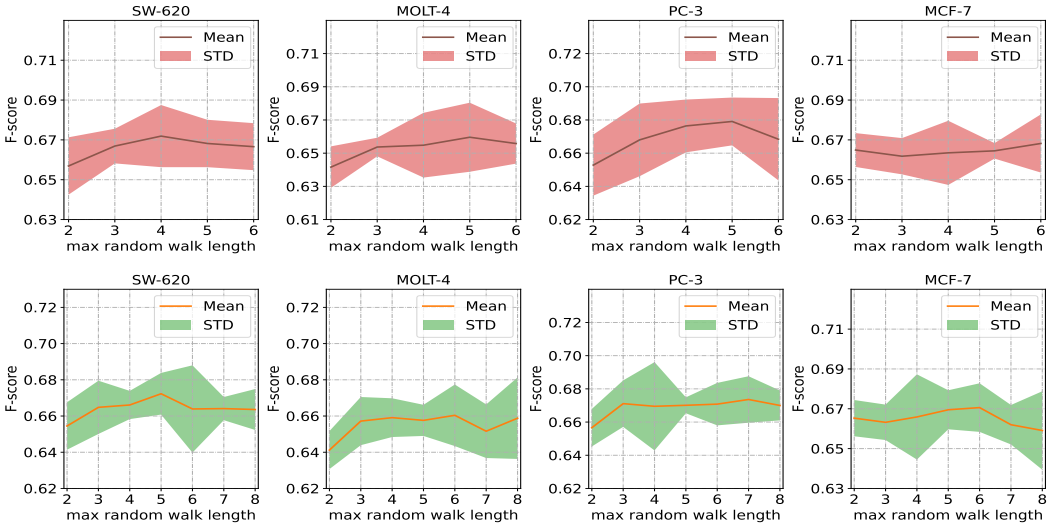

Figure 3: The first row reports the results of iGAD under the setting that M=5 and n'=8. The second row reports the results of iGAD under the setting that M=5 and n'=10.

Recall and *Recall-A* scores of DGCNN on SW-620 are $56.02_{\pm 1.16}$ and $12.74_{\pm 2.41}$, in contrast, $72.65_{\pm 0.63}$ and $67.84_{\pm 1.78}$ are achieved by DGCNN-GAD.

A5. The anomalous substructures in four datasets learned by iGAD are demonstrated in Figure 2. We can observe that anomalous graph substructures on four datasets are composed of the triangle, the tailed triangle, and the 4-cycle graphlets. It explains why iGAD-2 can achieve comparable results with iGAD (as shown in Table 2). We can draw a preliminary conclusion that the graph substructures having the triangle pattern, the tailed triangle pattern, and the 4-cycle graphlets are important for distinguishing anomalous graphs from normal graphs on these four graph datasets.

A6. As shown in Figure 3, we report the performance of iGAD under different max random walk lengths. Under the setting that the anomalous substructures contain 8 nodes, on different datasets, the best *F-score* value can be achieved when the max random walk length ranges from 4 to 6. In addition, under the setting that the anomalous substructures contain 10 nodes, when the max random walk length ranges from 5 to 7, the best *F-score* value on different datasets can be reached. The results verify that appropriately increasing the length of the maximum step can improve the effectiveness of iGAD since longer random walk sequences can capture more structural information.

Overall, the above experiments demonstrate the effectiveness of iGAD in graph-level anomaly detection. In addition, we put more experiments on parameter analysis in Appendix B.2.

6 Conclusion

In this paper, we proposed iGAD, a dual-discriminative graph neural network for imbalanced graph-level anomaly detection. To address various anomaly notions referring to anomalous attributes and/or substructures, and the imbalance nature of anomaly problem, iGAD proposed three key designs. They are anomalous attribute-aware graph convolution, anomalous substructure-aware deep RWK, and the PMI-based loss function. Experiment results on four real-world datasets demonstrate the effectiveness of our proposed model in the graph-level anomaly detection task.

The limitation of iGAD is that it cannot capture the position/location information. In practice, the particular position/location of nodes and/or substructures can be the reason for anomalous properties of a graph. [61] did a preliminary study on enabling GNNs to capture the position/location information of nodes in a graph by considering anchor nodes. We leave exploring the anomalous caused by the position/location of nodes and/or substructures to future work.

References

[1] John Adrian Bondy, Uppaluri Siva Ramachandra Murty, et al. *J. Graph Theory Appl.*, volume 290. Macmillan London, 1976.

[2] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, et al. Convolutional networks on graphs for learning molecular fingerprints. In *NeurIPS*, 2015.

[3] Alex Fout, Jonathon Byrd, Basir Shariat, et al. Protein interface prediction using graph convolutional networks. In *NeurIPS*, 2017.

[4] Shen Wang, Lifang He, Bokai Cao, Chun-Ta Lu, Philip S Yu, and Ann B Ragin. Structural deep brain network mining. In *KDD*, pages 475–484, 2017.

[5] Xifeng Yan, Hong Cheng, Jiawei Han, and Philip S Yu. Mining significant graph patterns by leap search. In *SIGMOD*, pages 433–444, 2008.

[6] H Gerhard Vogel, Wolfgang H Vogel, H Gerhard Vogel, Günter Müller, Jürgen Sandow, and Bernward A Schölkens. *Drug discovery and evaluation: pharmacological assays*, volume 2. Springer, 1997.

[7] Tommaso Lanciano, Francesco Bonchi, and Aristides Gionis. Explainable classification of brain networks via contrast subgraphs. In *KDD*, pages 3308–3318, 2020.

[8] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[9] Muhan Zhang, Zhicheng Cui, Marion Neumann, et al. An end-to-end deep learning architecture for graph classification. In *AAAI*, 2018.

[10] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[11] Muhammet Balcilar, Pierre Héroux, Benoit Gauzere, Pascal Vasseur, Sébastien Adam, and Paul Honeine. Breaking the limits of message passing graph neural networks. In *ICML*, pages 599–608, 2021.

[12] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *J. Mach. Learn. Res.*, 11:1201–1242, 2010.

[13] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *SIGMOD*, pages 93–104, 2000.

[14] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *ICDM*, pages 413–422. IEEE, 2008.

[15] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data*, 6(1):1–39, 2012.

[16] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. In *NeurIPS*, volume 12, 1999.

[17] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, 2001.

[18] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *ICML*, pages 4393–4402, 2018.

[19] Panpan Zheng, Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu. One-class adversarial nets for fraud detection. In *AAAI*, pages 1286–1293, 2019.

[20] Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18, 2015.

[21] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *ICLR*, 2018.

[22] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Min Knowl Discov*, 29(3):626–688, 2015.

[23] 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 Trans. Knowl. Data Eng.*, 2021.

[24] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S Yu. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *CIKM*, pages 315–324, 2020.

[25] Ge Zhang, Jia Wu, Jian Yang, Amin Beheshti, Shan Xue, Chuan Zhou, and Quan Z Sheng. Fraudre: Fraud detection dual-resistant to graph inconsistency and imbalance. In *ICDM*, pages 867–876. IEEE, 2021.

[26] Stephen Ranshous, Steve Harenberg, Kshitij Sharma, and Nagiza F Samatova. A scalable approach for outlier detection in edge streams using sketch-based approximations. In *SDM*, pages 189–197, 2016.

[27] Ge Zhang, Zhao Li, Jiaming Huang, Jia Wu, Chuan Zhou, Jian Yang, and Jianliang Gao. efraudcom: An e-commerce fraud detection system via competitive graph neural networks. *ACM Trans Inf Syst*, 40(3):1–29, 2022.

[28] Fanzhen Liu, Zhao Li, Baokun Wang, Jia Wu, Jian Yang, Jiaming Huang, Yiqing Zhang, Weiqiang Wang, Shan Xue, Surya Nepal, et al. eriskcom: an e-commerce risky community detection platform. *The VLDB Journal*, pages 1–17, 2022.

[29] Lingxiao Zhao and Leman Akoglu. On using classification datasets to evaluate graph outlier detection: Peculiar observations and new insights. *Big Data*, 2021.

[30] Thomas Gärtner, Peter Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*, pages 129–143. Springer, 2003.

[31] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In *AISTATS*, pages 488–495, 2009.

[32] Karsten M Borgwardt and Hans-Peter Kriegel. Shortest-path kernels on graphs. In *ICDM*, pages 8–pp. IEEE, 2005.

[33] Nino Shervashidze and Karsten Borgwardt. Fast subtree kernels on graphs. In *NeurIPS*, volume 22, 2009.

[34] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[35] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, volume 29, 2016.

[36] Muhammet Balcilar, Guillaume Renton, Pierre Héroux, Benoit Gaüzère, Sébastien Adam, and Paul Honeine. Analyzing the expressive power of graph neural networks in a spectral perspective. In *ICLR*, 2021.

[37] Saurabh Sawlani, Lingxiao Zhao, and Leman Akoglu. Fast attributed graph embedding via density of states. In *ICDM*, pages 559–568. IEEE, 2021.

[38] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2017.

[39] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, volume 30, 2017.

[40] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *ICML*, pages 1263–1272, 2017.

[41] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *NeurIPS*, 31, 2018.

[42] Zhengyang Wang and Shuiwang Ji. Second-order pooling for graph neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[43] Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in gnns. In *ICLR*, 2020.

[44] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, pages 4602–4609, 2019.

[45] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. In *NeurIPS*, volume 32, 2019.

[46] Muhan Zhang and Pan Li. Nested graph neural networks. In *NeurIPS*, volume 34, 2021.

[47] Byron C Wallace, Kevin Small, Carla E Brodley, and Thomas A Trikalinos. Class imbalance, redux. In *ICDM*, pages 754–763, 2011.

[48] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *ICML*, pages 872–881, 2019.

[49] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *ICLR*, 2020.

[50] Gang Wu and Edward Y Chang. Kba: Kernel boundary alignment considering imbalanced data distribution. *IEEE Trans. Knowl. Data Eng.*, 17(6):786–795, 2005.

[51] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *CVPR*, pages 2980–2988, 2017.

[52] Buyu Li, Yu Liu, and Xiaogang Wang. Gradient harmonized single-stage detector. In *AAAI*, pages 8577–8584, 2019.

[53] Shirui Pan, Jia Wu, and Xingquan Zhu. Cogboost: Boosting for fast cost-sensitive graph classification. *IEEE Trans. Knowl. Data Eng.*, 27(11):2933–2946, 2015.

[54] Pim de Haan, Taco S Cohen, and Max Welling. Natural graph networks. In *NeurIPS*, volume 33, pages 3636–3646, 2020.

[55] Giannis Nikolentzos and Michalis Vazirgiannis. Random walk graph neural networks. In *NeurIPS*, volume 33, pages 16211–16222, 2020.

[56] Mahito Sugiyama and Karsten Borgwardt. Halting in random walk kernels. In *NeurIPS*, volume 28, 2015.

[57] Hamed Masnadi-Shirazi and Nuno Vasconcelos. Risk minimization, probability elicitation, and cost-sensitive svms. In *ICML*, 2010.

[58] Jiaqi Wang, Wenwei Zhang, Yuhang Zang, Yuhang Cao, Jiangmiao Pang, Tao Gong, Kai Chen, Ziwei Liu, Chen Change Loy, and Dahua Lin. Seesaw loss for long-tailed instance segmentation. In *CVPR*, pages 9695–9704, 2021.

[59] Jacek P Dmochowski, Paul Sajda, and Lucas C Parra. Maximum likelihood in cost-sensitive learning: Model specification, approximations, and upper bounds. *J. Mach. Learn. Res.*, 11(12), 2010.

[60] Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola. Efficient mini-batch training for stochastic optimization. In *KDD*, pages 661–670, 2014.

[61] Jiaxuan You, Rex Ying, and Jure Leskovec. Position-aware graph neural networks. In *ICLR*, pages 7134–7143. PMLR, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] see section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] see Appendix A.1 and A.2.
   (b) Did you include complete proofs of all theoretical results? [Yes] see Appendix A.1 and A.2.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] see section 1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see section 5 and Appendix B.1
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report standard deviation error bars over different data splits.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] see Appendix B.1
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]