Decoupling Representation Learning and Classification for GNN-based Anomaly Detection

Yanling Wang[1,2], Jing Zhang[1,2,*], Shasha Guo[1,2], Hongzhi Yin[3], Cuiping Li[1,2], Hong Chen[1,2]
[1]Key Laboratory of Data Engineering and Knowledge Engineering of Ministry of Education, Renmin University of China
[2]School of Information, Renmin University of China
[3]School of Information Technology and Electrical Engineering, The University of Queensland
{wangyanling, zhang-jing, guoshashaxing, licuiping, chong}@ruc.edu.cn, h.yin1@uq.edu.au

ABSTRACT

GNN-based anomaly detection has recently attracted considerable attention. Existing attempts have thus far focused on jointly learning the node representations and the classifier for detecting the anomalies. Inspired by the recent advances of self-supervised learning (SSL) on graphs, we explore another possibility of decoupling the node representation learning and the classification for anomaly detection. We conduct a preliminary study to show that decoupled training using existing graph SSL schemes to represent nodes can obtain performance gains over joint training, but it may deteriorate when the behavior patterns and the label semantics become highly inconsistent. To be less biased by the inconsistency, we propose a simple yet effective graph SSL scheme, called Deep Cluster Infomax (DCI) for node representation learning, which captures the intrinsic graph properties in more concentrated feature spaces by clustering the entire graph into multiple parts. We conduct extensive experiments on four real-world datasets for anomaly detection. The results demonstrate that decoupled training equipped with a proper SSL scheme can outperform joint training in AUC. Compared with existing graph SSL schemes, DCI can help decoupled training gain more improvements.

CCS CONCEPTS

• **Information systems** → **Collaborative and social computing systems and tools**; • **Computing methodologies** → **Knowledge representation and reasoning**;

KEYWORDS

anomaly detection, graph neural network, decoupled training, self-supervised learning

ACM Reference Format:
Yanling Wang[1,2], Jing Zhang[1,2,*], Shasha Guo[1,2], Hongzhi Yin[3], Cuiping Li[1,2], Hong Chen[1,2]. 2021. Decoupling Representation Learning and Classification for GNN-based Anomaly Detection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462944

∗ Corresponding author.

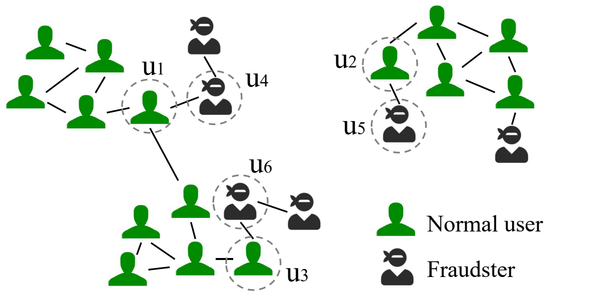


Figure 1: An illustration of the inconsistency between the behavior patterns and the label semantics. u_1, u_2, and u_3 are normal users but perform different behaviors. Similarly, u_4, u_5, and u_6 are fraudsters but perform different behaviors. In contrast, (u_1, u_4), (u_2, u_5) and (u_3, u_6) have opposite labels, but their behaviors are similar.

1 INTRODUCTION

Anomaly detection, which aims to discover the rare occurrences in datasets [2], has numerous high-impact applications in various domains, such as detecting opinion deception and review spams [32], credit card fraud [3], calling card and telecommunications fraud [7], and misinformation [37]. The most promising developments have been on discovering and incorporating the graph structural patterns [2, 16, 23, 35], as graphs effectively describe the correlations among inter-dependent users or objects that participate in the fraudulent activities [2].

Recently, driven by the advances of graph neural networks (GNNs) [5, 8, 11, 21, 45, 54], many attempts adopt GNNs for anomaly detection [9, 28, 29, 47, 60]. The main idea of GNN-based anomaly detection is to leverage the power of GNNs to learn expressive node representations with the goal of identifying abnormal instances in the embedding space. In this paper, we focus on detecting abnormal users in the real-life graph structured data.

Despite the great success, existing GNN-based models jointly learn the node representations and the classifier for detecting the anomalies on the graph. Conceivably, such a joint training scheme performs well if the behavior patterns expressed by node embeddings are discriminative to reflect the label semantics. But the cases in reality may be far from satisfactory. Figure 1 illustrates an example of user-rating-product graph, where the green (black) ones denote the normal users (fraudsters). For highlighting the structures of users, we fold the products by connecting two users if they rate the same product. In the graph, users with similar behavior patterns are presented closer to each other, but their label semantics are not always consistent with the behavior patterns. For example, distant users locating in different communities of the graph, such as the normal users (u_1, u_2, u_3) or the fraudsters (u_4, u_5, u_6), behave differently from each other, even if they share the same label. On the contrary, the users that are close to each other, such as (u_1, u_4), (u_2, u_5) or (u_3, u_6), behave similarly but have opposite labels. Such **inconsistency between the behavior patterns and the label semantics will result in hard instances (the mentioned users in the above example)** which put the GNN encoder in a dilemma: to learn the intrinsic graph properties or to capture the label semantics. Although GraphConsis [29] and CARE-GNN [9] also explore the inconsistency issue, they take effort on designing promising GNN encoders. This paper explores another way to alleviate the inconsistency's impact by raising a question: Shall we decouple the representation learning and the classification for anomaly detection?

In order to answer the question, we conduct a preliminary study to compare between the joint training and the decoupled training. Inspired by the recent progress in self-supervised learning (SSL) [10, 11, 20, 33], after decoupling, we adopt the graph SSL schemes for unsupervised representation learning. The preliminary experiment starts with a representative graph SSL scheme, Deep Graph Infomax (DGI) [46], for encoding the global information into node representations to represent the individual behavior patterns as well as the normal pattern occupied by the majority. We conduct the experiment on multiple anomaly detection tasks where the learning difficulty gradually varies from hard to easy (with hard instances removed gradually during the classification). We make the observation that **the joint training performs well on the easier tasks, while the decoupled training is less biased by the hard instances**. Motivated by this observation, we conduct extensive experiments on other datasets (cf. Table 2). However, decoupled training with DGI offers limited performance gains. Such a result drives us to explore the deeper reason. Since we make the conjecture that inconsistency between the behavior patterns and the label semantics impacts the anomaly detection, we explore how the decoupled training with DGI performs under different inconsistency levels. Silhouette coefficient [36] is a popular measure for quantifying the cohesion within the same class and separation across classes, so we use the additive inverse of silhouette coefficient to quantify the inconsistency level. The preliminary experimental results show that **the decoupled training with DGI gains slight or even negative improvements over the joint training when the inconsistency is quite high**. The above study implies that inconsistency is an important but often neglected factor for learning high-quality node representations.

Inspired by the above insights, we propose a new graph SSL scheme, Deep Cluster Infomax (DCI), which inherits the strength of DGI. In real life, normal users usually occupy the majority, so we can represent the whole graph to approximate the distribution of the normal users. From this perspective, DGI is a proper choice, as it encodes the graph-level information into each node representation to help identify the fraudsters from the normal users. However, when users behave diversely, it would be difficult to represent a unique normal pattern. To overcome this limitation, we introduce a clustering step to discover more concentrated feature spaces. Then we encode normal patterns within clusters instead of encoding a unique normal pattern in the whole graph. Consequently, DCI reduces the impact from the inconsistency caused by diverse behaviors across clusters.

This work makes the following observations and contributions:

- We conduct a study to compare between the joint training and the decoupled training on GNN-based anomaly detection, which is helpful for understanding the merits and limitations of decoupled training in practice.
- Our study reveals an intriguing phenomenon—inconsistency between the behavior patterns and the label semantics highly impacts the performance of graph representation learning—that has rarely been discussed before.
- We suggest that decoupled training equipped with a proper SSL objective can be an alternative way for effective anomaly detection. And we develop a graph SSL scheme called DCI .
- We conduct extensive experiments on four real-world datasets. The results demonstrate the advantages of decoupled training with DCI.

2 RELATED WORK

Our work is closely related to graph neural network, graph-based anomaly detection and self-supervised graph learning.

2.1 Graph Neural Networks

GNNs have made prominent progress in graph representation learning. The core idea behind GNNs is to update node representations by aggregating messages from the local neighborhoods. The state-of-the-art GNN models include GCN [21], GraphSAGE [11], GAT [45], GIN [54], etc. These models differ from each other in the way to aggregate the neighborhood information. For example, GCN [21] propagates messages based on the graph Laplacian matrix in a transductive manner. GraphSAGE [11] proposes an inductive learning framework, in which the aggregation function such as mean, max or LSTM generates node embeddings by aggregating messages from a node's local neighborhood. GIN [54] adopts the sum-like aggregation function, which is proved to be as powerful as the Weisfeiler-Lehman graph isomorphism test [25]. Graph attentive networks, with the pioneer work GAT [45], are studied to assign different weights to different neighbors via attention mechanisms. These models have been widely used in various real-world applications, such as recommendation systems [14, 51, 56], computational biology [39, 40] as well as anomaly detection.

2.2 Graph-based Anomaly Detection

Early researches detect anomalies via dense block identification [16, 38, 41], iterative learning [23, 27, 30, 48, 49, 59] or belief propagation on graphs [1, 35]. Nevertheless, these early attempts usually rely on the human-defined rules or features, which is not easy to generalize to various datasets. Motivated by the success of GNNs, modern algorithms tend to summarize the anomalous patterns automatically using GNNs. Examples include GAS [26], FdGars [50], GraphConsis [29] and CARE-GNN [9] for review fraud detection, GeniePath [28] and SemiGNN [47] for financial fraud detection, FANG [31] for fake news detection, ASA [53] for mobile fraud detection, and MTAD-GAT [61] for time-series anomaly detection. These models extend the existing vanilla GCN [21], the graph attention network [45, 52] or the heterogeneous GNNs [55] to tackle the problem of anomaly detection. Existing attempts have focused on proposing a promising GNN encoder for node representation learning guided by the labels. In these models, the representation learning and the classification are usually trained jointly. This work explores another possibility of decoupling these two parts and proposing a proper graph SSL scheme to capture the desired patterns for anomaly detection.

2.3 Self-supervised Graph Learning

Self-supervised learning (pre-training) is a common and effective scheme in the area of computer vision [6, 22, 58]. Among the SSL schemes, contrastive learning (CL), raises a recent surge of interest in visual representation learning [6, 13]. On a parallel note, CL-based SSL schemes have also been investigated on graph data. The early attempts of unsupervised graph learning such as GAE [20], GraphSage [11], node2vec [10], deepwalk [33] and LINE [44], which try to reconstruct the adjacency information of nodes, can be viewed as a kind of "local contrast" between a node and its neighbors to preserve the local homophily. The later proposed GCC [34], designing subgraph-level instance discrimination to capture transferable local structural patterns, can be viewed as a "local contrast" between multi-views of nodes (sampled ego-networks). Motivated by DIM [15], DGI [46] and InfoGraph [43] have been proposed to contrast between the node and the global graph, which can be viewed as a "local-global contrast" to capture the global structure information. In addition to graph structures, GPT-GNN [17] learns node attributes by a generative SSL scheme. You et al. [57] and Hassani et al. [12] explore different types of graph data augmentations upon DGI-based SSL scheme.

Our work differs from the above methods in two aspects. First, we explore the effectiveness of decoupled training on anomaly detection, and reveal that inconsistency is a key factor that impacts the quality of representation learning. Although Kang et al. [18] have also investigated the effectiveness of decoupled training in visual representation learning, they target at solving the class imbalance issue. To the best of our knowledge, we are the first to connect the problem of inconsistency with decoupled training. Second, we propose DCI to contrast the local and the semi-global representations.

3 PROBLEM DEFINITION

In this section, we first formalize the problem of graph-based anomaly detection and then define the joint training and the decoupled training, two training schemes for solving the problem.

Let $G = (V, A, X)$ be a graph, where V denotes the set of nodes and A denotes the adjacency matrix of nodes. X is the initial feature matrix with $\boldsymbol{x}_i \in \mathbb{R}^{d_0}$ denoting the d_0-dimensional initial feature vector of v_i. The nodes and links can be instantiated differently according to the concrete applications. For example, in business websites, for detecting the fraudsters who provide unreliable ratings on products, the graph is a bipartite graph with users, products as nodes and the user-rating-product relationships as links. Although the nodes usually include users and objects, we only care about detecting the abnormal users.

In this work, we represent nodes and the graph only based on the graph structures, as the side information of nodes is not always available, and the pure structure-dependent methods can generalize well across various applications. Without side information to initialize the node features, we perform eigen-decomposition on the normalized adjacency matrix s.t. $D^{-1/2}AD^{-1/2} = U\Lambda U^\top$ where D is the degree matrix, and use the top eigenvectors in U as the initial node features. The eigenvectors roughly capture the users' behavior patterns, since they preserve the adjacency information. Practically, other adjacency-based methods such as LINE [44] and node2vec [10] can be used to initialize the node features.

The objective is to predict the abnormal nodes in a graph. However, since the labels are often arduously expensive to obtain, the input graph G is usually partially labeled. Thus we formulate the anomaly detection on a graph as follows:

Problem 1. ***Anomaly detection on a Graph.*** *Given a partially labeled $G = \left(V, A, X, Y^L\right)$, where Y^L is the set of the partial labels on nodes and each $y_i \in Y$ is a binary value which takes value 1 if the corresponding node v_i is abnormal and 0 otherwise, the objective of anomaly detection is to learn a predictive function:*

$$\mathcal{F} : G = \left(V, A, X, Y^L\right) \rightarrow Y, \tag{1}$$

where $Y = Y^L \cup Y^U$ with Y^U as the unobserved labels of the nodes in G, which are to be inferred in the learning process. We use n to denote the number of nodes.

Graph neural networks have recently been attempted to solve the above defined problem, as introduced in Section 2.2. In most of these models, the predictive function $\mathcal{F}$ is divided into a GNN encoder $g : V \rightarrow \mathbb{R}^d$ and a binary classifier $f : \mathbb{R}^d \rightarrow \{0, 1\}$, where g is to encode the structure patterns into the node representations, and f is usually performed on top of the node representations output by g to distinguish the normal and abnormal nodes. Formally,

$$H = g(G, \theta), \hat{Y} = f(H, \phi), \tag{2}$$

where H, a $n \times d$ matrix, denotes the node embeddings and $\hat{Y}$, a $|Y| \times 1$ matrix, denotes the predicted node labels. θ and ϕ are the parameters to be learned. Most of the existing GNN models learn θ and ϕ in a joint manner, which is defined as:

Definition 3.1. **Joint training** estimates the node labels via $\hat{Y} = f(g(G, \theta), \phi)$ and then trains the parameters θ and ϕ through a single supervised loss function $\mathcal{L}_{SL}$.

In contrast to the commonly adopted joint training, we propose to use the decoupled training, a new training manner for anomaly detection as follows:

Definition 3.2. **Decoupled training** decouples the representation learning and the classification. The first step estimates node embeddings by $H = g(G, \theta)$ and trains θ by an additional self-supervised loss function $\mathcal{L}_{SSL}$ which is independent from the observed node labels Y^L. Then based on the learned θ, the second step estimates the node labels by $\hat{Y} = f(g(G,\theta), \phi)$, and trains ϕ as well as tunes θ by a supervised loss function $\mathcal{L}_{SL}$.

In the decoupled training, representation learning, which encourages the encoder g to encapsulate desired features for the anomaly detection target via self-supervised learning on the unlabeled graph data, can be regarded as a pre-training process. Classification, which encourages g and f to have strong semantic discrimination ability via supervised learning on the labeled graph data, can be regarded as a tuning process.

Given the above defined problem of anomaly detection and two training schemes, the questions to be answered include:

- Which training scheme, the joint training or the decoupled training, is more promising for anomaly detection?
- How to determine a proper $\mathcal{L}_{SSL}$ for representation learning in decoupled training?

4 METHODOLOGY

In this section, we systematically study the performance of decoupled training in anomaly detection. More specifically, we compare the decoupled training with the joint training given anomaly detection tasks with varying learning difficulties. We dig deeper into data inconsistency, the possible reason that causes the hard instances to learn, in order to explore the desired graph SSL for decoupled training. Finally, based on the observations, we propose Deep Cluster Infomax, a novel graph SSL scheme for anomaly detection.

4.1 Preliminary

We introduce the instantiated graph encoder and the basic self-supervised learning scheme to perform our study.

Graph Encoder g and Classifier f. We adopt GIN [54], a state-of-the-art graph neural network, to instantiate the GNN encoder g in the defined problem. GIN calculates the representation for each node via a sum-like neighborhood aggregation function, i.e.,

$$\boldsymbol{h}_i^{(l)} = \mathrm{MLP}^{(l)}\left(\left(1+\epsilon^{(l)}\right)\cdot \boldsymbol{h}_i^{(l-1)} + \sum_{v_j\in N(v_i)} \boldsymbol{h}_j^{(l-1)}\right), \quad (3)$$

where $\boldsymbol{h}_i^{(l)} \in \mathbb{R}^d$ is the embedding of node v_i at the l-th layer, and $\boldsymbol{h}_i^{(0)} = \boldsymbol{x}_i$. $N(v_i)$ is the set of neighboring nodes of node v_i. MLP denotes the multi-layer perceptron. $\epsilon^{(l)}$ is either a learnable parameter or a fixed scalar. We stack L layers to obtain the final node representation $\boldsymbol{h}_i^{(L)}$. Compared with other general GNNs [21, 45], the simple sum-like aggregator in GIN is able to encapsulate the neighborhood homophily as well as the structural homophily, which are both vital for representing the behavior patterns. Compared with the specific GNNs for anomaly detection [9, 28, 29, 47], GIN can generalize to more datasets (Table 2 illustrates that GIN outperforms other GNNs on most datasets).

We instantiate f based on a linear mapping followed by a non-linear activation function, i.e.,

$$p_i = \sigma(W^\top \boldsymbol{h}_i^{(L)} + b), \quad (4)$$

where σ is the sigmoid activation function, W and b are the parameters to be learned, and p_i is the predicted suspicious score (i.e., abnormal score) of node v_i.

Basic $\mathcal{L}_{SL}$. We adopt cross-entropy (CE) loss, the most popular one in supervised learning, to instantiate the supervised loss function $\mathcal{L}_{SL}$ in both the joint training and the decoupled training, i.e.,

$$\mathcal{L}_{CE} = -\frac{1}{|Y^L|}\sum_{i=1}^{|Y^L|}\left(y_i \cdot \log p_i + (1-y_i)\cdot \log(1-p_i)\right), \quad (5)$$

Basic $\mathcal{L}_{SSL}$. We adopt Deep Graph Infomax (DGI) [46], a state-of-the-art graph SSL scheme, to instantiate the SSL loss function $\mathcal{L}_{SSL}$ in the decoupled training, i.e.,

$$\mathcal{L}_{DGI} = -\frac{1}{2n}\sum_{i=1}^{n}\left(\mathbb{E}_G \log \mathcal{D}(\boldsymbol{h}_i^{(L)}, \boldsymbol{s}) + \mathbb{E}_{\tilde{G}} \log(1-\mathcal{D}(\tilde{\boldsymbol{h}}_i^{(L)}, \boldsymbol{s}))\right), \quad (6)$$

where $\mathcal{D}$ is a discriminator that outputs the affinity score of each local-global (i.e., node-graph) pair. The graph $\tilde{G}$, generated by a row-wise shuffling of the initial feature matrix X, provides the node representation $\tilde{\boldsymbol{h}}_i^{(L)}$ that can be paired with the graph representation $\boldsymbol{s}$ as a negative sample.

For computing the graph representation $\boldsymbol{s}$, we follow DGI to average all the nodes' representations and then apply a sigmoid activation function on the pooled result, i.e.

$$\boldsymbol{s} = \sigma\left(\frac{1}{n}\sum_{i=1}^{n} h_i^{(L)}\right). \quad (7)$$

Choosing DGI is inspired by the definition of "anomaly":

Definition 4.1. Anomalies are the instances which stand out as being dissimilar to all others [4].

The above definition of anomaly implies that understanding how the majority act (i.e., the normal pattern) is vital for anomaly detection. Since the normal instances usually occupy the majority of the data, we can represent the whole graph to approximate the distribution of the normal instances. In view of this, DGI is a proper choice that enables node representations to capture the global information of the entire graph. Henceforth, we abbreviate $\boldsymbol{h}_i^{(L)}$ to $\boldsymbol{h}_i$ for simplicity.

4.2 Why Decoupled Training?

Figure 1 shows an example of graph data, where the hard instances increase the difficulty of anomaly detection. In this section, we study the necessity of decoupled training by exploring how the joint training and the decoupled training perform when the learning difficulty is varied. To answer this question, we define the learning difficulty, design the experimental protocol and present the observed results.

Learning Difficulty. It is non-trivial to determine the learning difficulty of a dataset by the raw features and labels of the instances in it. Instead, we first jointly train g and f using all the labeled users and obtain the predicted suspicious scores. Specifically, we conduct the joint training for 100 epochs, and use the averaged predicted suspicious score over the 100 epochs for each user. Then we sort the averaged predictive probabilities that quantify how anomalous each user is in descending order. The top ρ (%) normal users and the bottom ρ (%) fraudsters are viewed as the hard instances to be predicted. Adjusting the value of ρ and removing the corresponding hard instances during classification leads to various learning difficulties. Note that we encode node representations based on the whole graph structure, while conduct prediction on the datasets of various learning difficulties.

Experimental Protocol. Using GIN in Eq.(3) as the graph encoder, the 1-layer MLP in Eq.(4) as the classifier, we perform joint training and decoupled training on different tasks controlled via ρ. Specifically, the first scheme jointly trains g and f by a unique cross-entropy loss in Eq.(5), while the second scheme first pre-trains g by the DGI loss in Eq.(6) and then tunes g and f together by the cross-entropy loss. We run 50 epochs for the pre-training model and run 100 epochs to train the classifier. Finally, we report the averaged best AUC score over 10 folds.

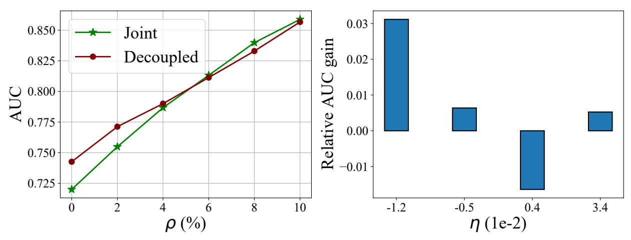
We construct six datasets from Reddit by varying its learning difficulty level ρ (%) from 0 to 10 with interval 2, where Reddit is a benchmark consisting of posts made by users on subreddits using the banned users from the website Reddit as the ground-truth anomalies [24].

Decoupled Training Helps Address Hard Instances. The results are summarized in Figure 2(a), from which we make an important observation: compared with the joint training, **the decoupled training is less biased by the hard instances**. The AUC performance of joint training increases from 0.720 to 0.859 with varying ρ from 0 to 10, which indicates the hard instances impact the learning process a lot. On the easier tasks (corresponding to a larger ρ), the decoupled training has comparable performance with the joint training. But when the dataset becomes difficult to learn (corresponding to a smaller ρ), the performance gain derived by the decoupled training tends to increase (derives **3.1%** relative AUC gain when ρ=0). Such an intriguing observation implies that decoupling representation learning and classification for anomaly detection can lower the negative impact caused by hard instances. Motivated by the above observation, we conduct extensive experiments on other datasets (cf. Table 2), but we note that the power of decoupled training with DGI could be limited (derives **less than 1.5%** relative AUC gains over the joint training on the Wiki, Alpha and Amazon). Such a result drives us to explore the deeper reason and the possible limitation of DGI.

4.3 Is Decoupled Training Stably Better?

Definition 4.2. **Inconsistency:** the behavior patterns and the label semantics disagree with each other.

Various limitations of the data distributions such as label imbalance [18] and scarce label [42] may increase the learning difficulty of a dataset, but we make the conjecture that the inconsistency between the behavior patterns and the label semantics also leads to the learning difficulty. To verify our conjecture, we explore how the decoupled training with DGI performs under different inconsistency levels. To answer the question, we formulate the inconsistency, design the experimental protocol and present the observed results.



(a) AUC over learning difficulty (b) Relative AUC gain over inconsistency

Figure 2: Preliminary experiments. (a) Performance of different training schemes over various learning difficulties; (b) Relative performance gains obtained by decoupled training over different inconsistency levels.

Inconsistency. A dataset is more consistent if the behavior patterns of the same-labeled users are more similar, and those between the users with opposite labels are more different. Intuitively, silhouette coefficient [36], a clustering metric which measures the cohesion within the same class and separation across classes, can help formulate the inconsistency. We use the additive inverse of silhouette coefficient to represent inconsistency η. Since there are only two classes (i.e., normal and abnormal), η can be defined as:

$$\begin{aligned}
\eta &= -\frac{1}{|V|}\sum_{i=1}^{|V|}\frac{b_i - a_i}{\max\{b_i, a_i\}}, \qquad (8)\\
a_i &= \frac{1}{|V_i|}\sum_{v_j \in V_i}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2, V_i = \{v_j : y_j = y_i\},\\
b_i &= \frac{1}{|\bar{V}_i|}\sum_{v_j \in \bar{V}_i}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2, \bar{V}_i = \{v_j : y_j \neq y_i\},
\end{aligned}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the initial representations of users v_i and v_j respectively. a_i is the average distance of v_i to all the other users of the same label, and b_i is the average distance of v_i to all the other users of the opposite label. When a_i is smaller and b_i is bigger, v_i is more consistent to the users of the same label. More users satisfy the property will result in a larger silhouette coefficient and a smaller inconsistency.

Experimental Protocol. We perform joint training and decoupled training on different datasets controlled via η. The evaluation process also follows the 10-fold evaluation.

We construct four separate datasets from Reddit with different η. Specifically, we perform METIS [19], a graph partition algorithm, to partition the the original graph into four sub-graphs with η (1e-2) $\in$ {-1.2, -0.5, 0.4, 3.4}. With METIS, we can construct multiple datasets with different inconsistency levels as well as preserving the original graph structures within each dataset as much as possible.

DGI Helps Less on the Highly Inconsistent Data. The relative performance gains obtained by decoupled training with DGI are shown in Figure 2(b), from which we observe that compared with joint training, **decoupled training with DGI may not always improve, and even brings negative influence when the data is highly inconsistent**. The figure shows decoupled training with

DGI obtains very slight or even negative AUC gains on three of the four datasets. Only on the less inconsistent dataset (corresponding to η=-1.2), the decoupled training attains 3.1% relative AUC gain over the joint training. In other words, it is challenging for DGI to summarize a proper normal pattern in the highly inconsistent dataset. Thus, a more effective graph SSL scheme is demanded addressing such a problem.

4.4 The Proposed SSL Scheme DCI

We propose a new self-supervised scheme, called Deep Cluster Infomax (DCI), for anomaly detection. Similar to DGI, DCI also encodes the normal pattern of the majority following Definition 4.1. However, when users behave quite diversely, a unique normal pattern is difficult to be represented. Fortunately, we observe that users can be naturally partitioned into different clusters, and the behavior patterns within the same cluster are often much more concentrated than those in the whole graph. As a result, the inconsistency presented by the same labels but diverse behavior patterns in the whole graph (e.g., (u_1, u_2, u_3) or (u_4, u_5, u_6) in Figure 1) will be reduced within a small cluster. Meanwhile, the inconsistency presented by the opposite labels but close behavior patterns (e.g., (u_1, u_4), (u_2, u_5) or (u_3, u_6) in Figure 1) will also be reduced, as the distance between these users is amplified when the context is restricted into a small cluster. In view of this, we perform the cluster-level summary instead of the graph-level summary.

The first step of DCI is to partition the given graph G into K clusters $[C_1, C_2, \cdots, C_K]$, where each cluster C_k contains n_k nodes. The node set in C_k is denoted as V_k. Note that for the user-interacting-object graphs, the cluster C_k contains both users and objects. We apply the classic K-Means algorithm to cluster all nodes based on the node features X. Since X is instantiated by the top eigenvectors of the normalized adjacency matrix, it preserves the neighborhood proximity. In this way, the users who behave similarly tend to be clustered into the same cluster. A self-defined method is designed to help determine the number of clusters K, and the details are explained in Section 5.5.

After clustering, we compute the cluster-level representation $\boldsymbol{s}_k$ for each cluster to summarize how the majority in C_k act, i.e.,

$$\boldsymbol{s}_k = \sigma\left(\frac{1}{n_k}\sum_{v_i \in V_k} \boldsymbol{h}_i\right). \tag{9}$$

For each cluster C_k, we encode this semi-global representation $\boldsymbol{s}_k$ into the node representations via the following loss function:

$$\mathcal{L}_{DCI}^{k} = -\frac{1}{2n_k}\sum_{v_i \in V_k}\left(\mathbb{E}_{C_k}\log\mathcal{D}(\boldsymbol{h}_i, \boldsymbol{s}_k) + \mathbb{E}_{\tilde{C}_k}\log(1-\mathcal{D}(\tilde{\boldsymbol{h}}_i, \boldsymbol{s}_k))\right), \tag{10}$$

where $\mathcal{D}$ is a discriminator that outputs the affinity score of each local-semi-global (i.e., node-cluster) pair. Similar to DGI, node representation $\tilde{\boldsymbol{h}}_i$ is paired with the cluster representation $\boldsymbol{s}_k$ as a negative sample.

The final loss function of DCI is the average of the losses of the K clusters, i.e.,

$$\mathcal{L}_{DCI} = \frac{1}{K}\sum_{k=1}^{K}\mathcal{L}_{DCI}^{k}. \tag{11}$$

In practice, we re-cluster the nodes based on the node embeddings after every $\bar{t}$ training epochs. Algorithm 1 gives the pseudocode for DCI.

```
Algorithm 1: Deep Cluster Infomax
   Input  : Graph $G = (V, A, X)$, Number of clusters $K$,
            Number of training epochs $t$, Number of
            re-clustering epochs $\bar{t}$.
   Output : Optimized GNN encoder $g$
 1 Initialize clusters $[C_1, C_2, \cdots, C_K]$ = K-Means($X$);
 2 Initialize the parameters $\theta$ and $\omega$ for the encoder $g$ and
     the discriminator $\mathcal{D}$ ;
 3 for $epoch \leftarrow 1$ to $t$ do
 4     $H = g(G, \theta)$;
 5     $\mathcal{L}_{DCI} = \frac{1}{K}\sum_{k=1}^{K}\mathcal{L}_{DCI}^{k}(H, C_k, \omega)$;
 6     $\theta, \omega \leftarrow$Adam($\mathcal{L}_{DCI}$);
 7     if $t$ mod $\bar{t}$ == 0 then
 8         $[C_1, C_2, \cdots, C_K]$ = K-Means($g(G, \theta)$)
   Return : encoder $g$
```

4.5 Discussions and Summaries

We discuss why the decoupled training (DCI) can work compared with other choices.

Compared with joint training supervised by the labeled data, the decoupled training additionally trains the GNN encoder self-supervised by the intrinsic graph structures beforehand, which can be less biased by the data inconsistency.

In terms of the decoupled training for anomaly detection, existing graph SSL schemes offer limited benefits. For example, GAE [20] and GraphSAGE [11] reconstruct the adjacency matrix following the neighborhood proximity assumption, which can be hurt when too many interactions exist between the normal users and the fraudsters. GCC [34] which contrasts multi-views of nodes (sampled ego-networks) to capture transferable structural patterns across graphs, over-emphasizes the structural homophily. DGI [46] contrasts the whole graph with the node in it to encode the graph-level information into each node's representation. The graph-level embedding reveals the normal pattern occupied by the majority (i.e., the normal nodes), so it helps imply how far a node deviates from what is normal. DCI can be viewed as cluster-based DGI. Compared with DGI, the semi-global context encoded by DCI is less diverse, consequently alleviating the impact induced by the inconsistency between the behavior patterns and the label semantics.

5 EXPERIMENTAL EVALUATION

In this section, we present the results of different models to identify fraudulent users on real-world datasets. Particularly, we mainly answer the following research questions:

- **RQ1**: How does the decoupled training perform compared with the joint training?
- **RQ2**: How does DCI perform compared with other state-of-the-art graph SSL schemes?

Table 1: Statistics of the datasets.

Graph	#Users(% normal, abnormal)	#Objects	#Edges
Reddit	10,000 (96.34%, 3.66%)	984	78,516
Wiki	8,227 (97.36%, 2.64%)	1,000	18,257
Alpha	3,286 (61.21%, 38.79%)	3,754	24,186
Amazon	27,197 (91.73%, 8.27%)	5,830	52,156

- **RQ3**: How does the decoupled training perform compared with the multi-task learning?
- **RQ4**: How to determine the number of clusters for DCI?

5.1 Experimental settings

Datasets. We evaluate on four real-world user-object graphs. Detail statistics about these datasets are listed in Table 1.

- **Reddit** [24] is a user-subreddit graph, which consists of one month of posts made by users on subreddits. This dataset contains ground-truth labels of banned users from Reddit.
- **Wiki** [24] is an editor-page graph, which describes one month of edits on Wikipedia pages. This dataset contains public ground-truth labels of banned users.
- **Alpha** [23] is a user-user trust graph of Bitcoin users trading on the platform Alpha. This graph is made bipartite by splitting each user into a "rater" with all its outgoing edges and an "object" with all incoming edges. 214 users in this dataset are labeled.
- **Amazon** [23] is a user-product graph, where the edges describe users' rating behaviors. 278 users in this dataset are labeled.

Specifically, the graphs used in our experiments are unweighted, where the edge represents a user has ever interacted with an object. Amazon is extracted from a large user-product graph [23], which contains 256,059 users, 74, 258 products and 560,804 interactions. We use METIS [19] to partition the original Amazon dataset into 20 sub-graphs, and merge two sub-graphs to simulate a graph which has higher inconsistency. METIS partitions the graph according to the interconnections between nodes, so it preserves the original graph structure within the sampled dataset as much as possible.

Evaluation Protocols. In practice, the training data of anomaly detection is usually class-imbalance, that is, the instance-rich class (i.e. the class of normal users) dominates during the training procedure. As a result, all the predicted suspicious scores tend to be small, which makes it difficult to set a proper threshold for classifying the fraudsters and the normal users. So we adopt the widely used metric AUC to consider all the possible thresholds for classification. AUC measures the probability that a randomly sampled fraudster has a higher suspicious score than a randomly sampled normal user. For a fair comparison, we conduct 10-fold evaluation on the Reddit, Wiki and Alpha. Specially, we conduct 5-fold evaluation on the Amazon, since this dataset has limited labeled fraudsters.

Baselines. We compare with two categories of baselines. First, ***Joint training algorithms*** working in an end-to-end manner are compared to show the effectiveness of decoupled training. Among them, **CARE-GNN** [9], which applies reinforcement learning to filter noisy neighbors, is a state-of-the-art GNN model for anomaly detection. Other GNN-based anomaly detection models such as GraphConsis [29] and Semi-GNN [47] have been shown to be less useful than CARE-GNN, thus they are ignored in the experiments. **GAT** [45] and **GeniePath** [28] are both graph attentive networks. GAT calculates attentions for one-hop neighbors, and GeniePath extends them to multi-hop neighbors. GeniePath has shown effectiveness on malicious account detection in Alipay. **GIN** [54] proposes a powerful feature aggregation function to effectively preserve the structure homophily.

Second, ***SSL schemes for decoupled training*** are compared to show the superiority of DCI. Among them, **Graph Auto-encoder (GAE)** [20] and **Random walk-based objective (RW)** [10, 11] reconstruct the one-hop or the multi-hop adjacency information (obtained by local random walks) between nodes. **Graph Contrastive Coding (GCC)** [34] and **Deep Graph Infomax (DGI)** [46] perform contrastive learning between node-node pairs or graph-node pairs. The four SSL baselines are popular and have shown effectiveness on various real-world applications. GAE and RW target on preserving the local adjacency between nodes. GCC and DGI allow to discover the structural similarities in a global environment – for example, distant nodes with similar structural roles. Distinct from the above baselines, DCI preserves the structural similarities in a semi-global context. It is worth noting that we do not compare with GPT-GNN [17], because our model only involves the structure information, while GPT-GNN requires the input of node attributes.

Parameter Settings. We use input feature dimension (64), node representation dimension (128), number of GNN layers (2), learning rate (0.01) and optimizer (Adam) for all models. ϵ in Eq.3 is set to be 0. The source code of CARE-GNN fixes the number of GNN layer to be 1, so we use this default setting. The number of re-clustering epochs $\bar{t}$ is set to be 20. For the SSL pre-training, we fix the number of training epochs to be 50. Specially, we adopt the early stopping strategy in the decoupled training with RW, since we found too many training epochs can harm this model's performance. To better analyze different SSL schemes, we unify their backbones as the GIN's encoder. For the classification, we record the best testing result after 100 epochs on each fold, then report the averaged best AUC score over different folds.

Code Implementation. For CARE-GNN, GIN, DGI and GCC, we use the source code provided by their authors. For GAT and GeniePath, we use the open-source implementation[1]. We modified these codes to make them adapt to our tasks. We implement DCI with Pytorch. The source code of DCI is available on Github [2].

5.2 Overall Evaluation

According to the results shown in Table 2, we summarize the following conclusions: **(1) Decoupled training contributes to the anomaly detection.** All the decoupled models use GIN's encoder as their backbones. Compared with GIN (joint training), most of the decoupled models perform better. On the Amazon, decoupled training with DCI obtains 6.4% relative performance gain over the joint training. **(2) DCI is an effective graph SSL scheme for decoupled training.** Compared with other SSL schemes, DCI helps decoupled training attain more performance gains on most datasets.

[1]https://github.com/shawnwang-tech/GeniePath-pytorch
[2]https://github.com/wyl7/DCI-pytorch

Table 2: Overall evaluation on four real-world datasets.

		Reddit	Wiki	Alpha	Amazon
Joint	CARE-GNN	0.700	0.702	0.802	0.729
	GAT	0.738	0.681	0.848	0.696
	GeniePath	0.720	0.689	0.849	0.738
	GIN	0.720	0.727	0.884	0.761
Decoupled	GAE	0.730	0.714	0.884	0.806
	RW	0.728	0.740	**0.908**	0.782
	GCC	0.669	0.695	0.865	0.733
	DGI	0.743	0.737	0.884	0.771
	DCI (ours)	**0.746**	**0.762**	0.907	**0.810**
Inconsistency η (1e-2)		-0.676	0.841	-	-

Note: All the decoupled models use GIN's encoder as the backbone.

Table 3: Comparison between the joint training (GIN) and the decoupled training (DGI and DCI) on datasets which are increasingly easier.

ρ (%)		1	2	3	4	5
Reddit	GIN (joint)	0.734	0.755	0.774	0.787	0.803
	DGI (decoupled)	0.755	0.771	0.777	0.790	0.798
	DCI (decoupled)	0.756	0.778	0.783	0.790	0.800
Wiki	GIN (joint)	0.745	0.757	0.781	0.803	0.808
	DGI (decoupled)	0.759	0.769	0.790	0.797	0.816
	DCI (decoupled)	0.777	0.788	0.817	0.822	0.834
Alpha	GIN (joint)	0.886	0.905	0.923	0.933	0.940
	DGI (decoupled)	0.893	0.907	0.928	0.948	0.955
	DCI (decoupled)	0.907	0.906	0.936	0.939	0.947
Amazon	GIN (joint)	0.764	0.781	0.783	0.803	0.799
	DGI (decoupled)	0.780	0.797	0.796	0.817	0.793
	DCI (decoupled)	0.805	0.838	0.835	0.845	0.844

Note: All the models use GIN's encoder as the backbone.

Table 4: Evaluation of the multi-task learning.

		Reddit	Wiki	Alpha	Amazon
Joint	GIN	0.720	0.727	0.884	0.761
Multi-task	GAE	0.726	0.705	0.904	0.766
	DGI	0.647	0.664	0.891	0.806
	DCI	0.675	0.670	0.893	0.803

Note: All the multi-task models use GIN's encoder as the backbone.

As an extension of DGI, DCI demonstrates the necessity of clustering. In other words, given a more concentrated feature space, it would be easier to figure out the differences between the normal instances and the anomalies. **(3) Adjacency information is useful for anomaly detection.** We observe that GAE and RW also perform well across different datasets. On the Alpha, RW attains the best performance. However, the effectiveness of GCC is limited. In the traditional studies [1, 23], user behaviors (i.e., who review/edit/rate what) are regarded as important information for anomaly detection. The user behaviors are described by the edges in a graph. Such information is preserved by GAE and RW, but is ignored by GCC which emphasizes the structural similarity between nodes. In view of this, decoupled training with adjacency-based self-supervision deserves further study. DCI conducts pre-clustering according to the node features decomposed from the normalized adjacency matrix, so it implicitly captures the adjacency information. **(4) A simple GNN encoder can also perform well.** CARE-GNN and GeniePath are two promising algorithms for the graph-based anomaly detection. They design complicated graph convolutions to solve the challenges of anomaly detection. However, we find that these models need more training epochs to converge, and GIN outperforms them on most datasets. We suggest that decoupled training composed of the GIN's encoder and a proper SSL objective can be an alternative way for effective anomaly detection. **(5) Decoupled training with DCI shows promising performance on the more inconsistent dataset.** We can compute the inconsistency levels for the Reddit and Wiki, as users in the two datasets are fully labeled. DCI shows only 0.4% relative AUC gain over DGI on the Reddit, but derives 3.4% relative AUC gain over DGI on the Wiki. Wiki is the more inconsistent dataset, on which DCI shows more performance gains over DGI.

5.3 Study of DCI over Learning Difficulty

In this section, we take insight into: how decoupled training with DCI helps address the hard instances. We follow the experimental protocols in Section 4.2 to vary the learning difficulty via controlling ρ (%) $\in \{1, 2, 3, 4, 5\}$ and report the performance of joint training (GIN) and decoupled training (DGI and DCI).

As summarized in Table 3, on the datasets consisting of more hard instances (corresponding to a smaller ρ), the decoupled training brings more performance gains over the joint training. Compared with DGI, DCI contributes more to the decoupled training. When the dataset becomes easier to learn (corresponding to a larger ρ), the decoupled training only obtains comparable performance with the joint training. That is to say, besides the inconsistency, the effectiveness of anomaly detection could be impacted by other factors, such as the label imbalance and scarce label. These problems deserve further study.

5.4 Comparison with the Multi-task Learning

We compare with the multi-task learning, which also utilizes the self-supervision. The final loss of multi-task learning can be formulated as: $\mathcal{L}_{MTL} = \alpha \cdot \mathcal{L}_{SL} + (1 - \alpha) \cdot \mathcal{L}_{SSL}$, where α is the balancing term between the cross-entropy loss and the self-supervised loss. We instantiate $\mathcal{L}_{SSL}$ with the objective of GAE/DGI/DCI. α is searched from 0.1 to 0.9 with interval 0.1. In our experiments, the objectives of RW and GCC are optimized under the mini-batch setting, while the cross-entropy loss is optimized under the full-batch setting. Thus we do not consider the objectives of RW and GCC in this section.

As shown in Table 4, the multi-task learning is able to outperform the joint training, but does not always bring the positive improvements. Even though the multi-task learning outperforms the joint training on some occasions, the decoupled training still

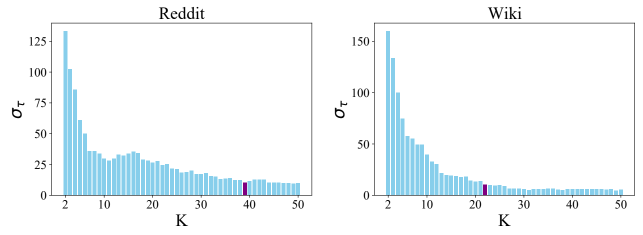


Figure 3: Study of the clustering number in DCI (best seen in color).

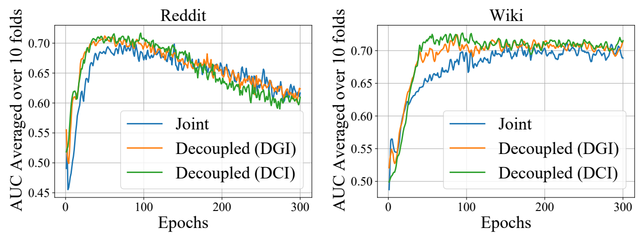


Figure 4: Convergence study (best seen in color).

shows advantages over the multi-task learning. We also observe that using $\mathcal{L}_{DGI}$ or $\mathcal{L}_{DCI}$ for multi-task learning results in very poor performance on the Reddit and Wiki, while $\mathcal{L}_{GAE}$ is less vulnerable on the two datasets. Compared with GAE which preserves the local adjacency, DGI and DCI allow to learn more implicit and expressive structural patterns. Thus using $\mathcal{L}_{DGI}$ or $\mathcal{L}_{DCI}$ could amplify the inconsistency between the structural patterns and the label semantics.

5.5 Study of the Clustering Number

As the clustering plays a pivotal role in DCI, we investigate how to determine the number of clusters. Due to the various behavior patterns of users, it is not suitable to choose the clustering number K only according to the results on a validation set. Thus we suggest to narrow the search space of K before the learning process. We adopt inertia that measures how internally coherent clusters are to help narrow the search space. Inertia is computed by $\sum_{i=1}^{|V|} \min_{\boldsymbol{\mu}_j \in C} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$, where C is the set of disjoint clusters and $\boldsymbol{\mu}_j$ is the embedding of the j-th cluster center.

Specifically, we calculate the inertia under different clustering numbers $K \in \{2, 3, \ldots, K^*\}$ (K^* is the maximal clustering number and is set to be 50), and compute τ_K which denotes the inertia gap between K and $K+1$. Then given r consecutive inertia gaps $\{\tau_K, \tau_{K+1}, \cdots, \tau_{K+r-1}\}$ (r is set to be 15), we calculate the standard deviation σ_τ. The optimal value of K tends to locate at the interval, where σ_τ begins to be stable. Taking the Reddit and Wiki as the examples, we show σ_τ under different clustering numbers in Figure 3. On each dataset, the purple bar corresponds to the clustering number that leads to the best performance. We can see that the optimal value of K matches the above principle. Note that the optimal value of K could be somewhat different under different environments (e.g., different versions of PyTorch). In practice, we suggest to use the above principle to set an initial clustering number beforehand, then determine the final clustering number by searching around the initial one according to the results on a validation set. In our experiments, the datasets are not very large, so the results on a validation set could be less instructive. In light of this, we directly report the performance under the optimal value of K, reflecting the potential power of DCI. In the future work, we will explore a more effective way to find the optimal clustering number.

5.6 Convergence Study

Taking the Reddit and Wiki as the examples, we explore the convergence of DCI. Here we mainly compare with GIN (joint) and DGI (decoupled). These models use the GIN's encoder as the backbone. Different from the evaluation setting used in the above experiments, here we report the testing AUC averaged over 10 folds at each training epoch.

From Figure 4, we observe that: on both datasets, the models using decoupled training tend to converge faster than the models using joint training. Benefiting from the SSL step in the decoupled training, we can obtain a better parameter initialization for the following classification to speed up the convergence. Besides, we observe that the convergence trends of different models are similar. That is because they all adopt the cross-entropy loss for the classifier optimization. Therefore, to further improve the effectiveness of anomaly detection, other objectives are also worth studying.

6 CONCLUSION

This work piloted studies on performance of decoupled training for anomaly detection, and made intriguing findings. At the heart of these findings is the inconsistency between the structural patterns and the label semantics, which is identified to be a vital factor that impacts the representation learning. It provides a new perspective for understanding the SSL. This work further developed a new graph SSL scheme DCI by injecting a clustering step to reduce the data inconsistency. We believe that decoupled training composed of the GIN's encoder and a proper SSL objective can be an alternative way for effective anomaly detection. The findings and DCI develped here could be inspiring for the future research on representation learning, even not restricted to anomaly detection.

ACKNOWLEDGMENTS

This work is supported by Beijing Natural Science Foundation (Grant No. 4212022), National Natural Science Foundation of China (Grant No. 62076245), National Key Research & Develop Plan (Grant No. 2018YFB1004401), CCF-Tencent Open Fund and ARC Discovery Project (Grant No. DP190101985). We would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion Fraud Detection in Online Reviews by Network Effects. In *ICWSM*.
[2] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery* 29, 3 (2015), 626–688.
[3] Richard J Bolton, David J Hand, et al. 2001. Unsupervised profiling methods for fraud detection. *Credit scoring and credit control VII* (2001), 235–255.
[4] Raghavendra Chalapathy and Sanjay Chawla. 2019. Deep Learning for Anomaly Detection: A Survey. *CoRR* (2019).

[5] Hongxu Chen, Hongzhi Yin, Tong Chen, Quoc Viet Hung Nguyen, Wen-Chih Peng, and Xue Li. 2019. Exploiting Centrality Information with Graph Convolutions for Network Representation Learning. In *ICDE*. 590–601.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *ICML*. 1597–1607.

[7] Corinna Cortes, Daryl Pregibon, and Chris Volinsky. 2001. Communities of interest. In *IDA*. 105–114.

[8] Vachik S. Dave, Baichuan Zhang, Pin-Yu Chen, and Mohammad Al Hasan. 2019. Neural-Brane: Neural Bayesian Personalized Ranking for Attributed Network Embedding. *Data Sci. Eng.* 4, 2 (2019), 119–131.

[9] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. 2020. Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters. In *CIKM*. 315–324.

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *KDD*. 855–864.

[11] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NeurIPS*. 1024–1034.

[12] Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive multi-view representation learning on graphs. In *ICML*. 4116–4126.

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *CVPR*. 9729–9738.

[14] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*. 639–648.

[15] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. In *ICLR*.

[16] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *KDD*. 895–904.

[17] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. GPT-GNN: Generative Pre-Training of Graph Neural Networks. In *KDD*. 1857–1867.

[18] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2019. Decoupling Representation and Classifier for Long-Tailed Recognition. In *ICLR*.

[19] George Karypis and Vipin Kumar. 1995. Multilevel Graph Partitioning Schemes. In *ICPP*. 113–122.

[20] Thomas N Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. *NIPS Workshop on Bayesian Deep Learning*.

[21] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[22] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. 2019. Revisiting self-supervised visual representation learning. In *CVPR*. 1920–1929.

[23] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and V. S. Subrahmanian. 2018. REV2: Fraudulent User Prediction in Rating Platforms. In *WSDM*. 333–341.

[24] Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting Dynamic Embedding Trajectory in Temporal Interaction Networks. In *KDD*. 1269–1278.

[25] AA Leman and B Weisfeiler. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya* 2, 9 (1968), 12–16.

[26] Ao Li, Zhou Qin, Runshi Liu, Yiqun Yang, and Dong Li. 2019. Spam Review Detection with Graph Convolutional Networks. In *CIKM*. 2703–2711.

[27] Rong-Hua Li, Jeffrey Xu Yu, Xin Huang, and Hong Cheng. 2012. Robust Reputation-Based Ranking on Bipartite Rating Networks. In *SDM*. 612–623.

[28] Ziqi Liu, Chaochao Chen, Longfei Li, Jun Zhou, Xiaolong Li, Le Song, and Yuan Qi. 2019. GeniePath: Graph Neural Networks with Adaptive Receptive Paths. In *AAAI*. 4424–4431.

[29] Zhiwei Liu, Yingtong Dou, Philip S. Yu, Yutong Deng, and Hao Peng. 2020. Alleviating the Inconsistency Problem of Applying Graph Neural Network to Fraud Detection. In *SIGIR*. 1569–1572.

[30] Abhinav Mishra and Arnab Bhattacharya. 2011. Finding the bias and prestige of nodes in networks based on trust scores. In *WWW*. 567–576.

[31] Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. FANG: Leveraging Social Context for Fake News Detection Using Graph Representation. In *CIKM*. 1165–1174.

[32] Myle Ott, Claire Cardie, and Jeff Hancock. 2012. Estimating the prevalence of deception in online review communities. In *WWW*. 201–210.

[33] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *KDD*. 701–710.

[34] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In *KDD*. 1150–1160.

[35] Shebuti Rayana and Leman Akoglu. 2015. Collective Opinion Spam Detection: Bridging Review Networks and Metadata. In *KDD*. 985–994.

[36] Peter J. Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* 20 (1987), 53–65.

[37] Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. Csi: A hybrid deep model for fake news detection. In *CIKM*. 797–806.

[38] Neil Shah, Alex Beutel, Brian Gallagher, and Christos Faloutsos. 2014. Spotting Suspicious Link Behavior with fBox: An Adversarial Perspective. In *ICDM*. 959–964.

[39] Chence Shi, Minkai Xu, Hongyu Guo, Ming Zhang, and Jian Tang. 2020. A Graph to Graphs Framework for Retrosynthesis Prediction. In *ICML*. 8818–8827.

[40] Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. 2020. GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation. In *ICLR*.

[41] Kijung Shin, Bryan Hooi, and Christos Faloutsos. 2016. M-Zoom: Fast Dense-Block Detection in Tensors with Quality Guarantees. In *ECML PKDD*, Vol. 9851. 264–280.

[42] Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. 2020. When does self-supervision improve few-shot learning?. In *ECCV*. 645–666.

[43] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2020. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *ICLR*.

[44] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. 1067–1077.

[45] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[46] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. 2019. Deep Graph Infomax. In *ICLR*.

[47] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. 2019. A Semi-supervised Graph Attentive Network for Financial Fraud Detection. (2019), 598–607.

[48] Guan Wang, Sihong Xie, Bing Liu, and Philip S. Yu. 2011. Review Graph Based Online Store Review Spammer Detection. In *ICDM*. 1242–1247.

[49] Guan Wang, Sihong Xie, Bing Liu, and Philip S. Yu. 2012. Identify Online Store Review Spammers via Social Review Graph. *ACM Trans. Intell. Syst. Technol.* 3, 4 (2012), 61:1–61:21.

[50] Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, and Jian Xion. 2019. FdGars: Fraudster Detection via Graph Convolutional Networks in Online App Review System. In *WWW*. 310–316.

[51] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*. 165–174.

[52] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S. Yu. 2019. Heterogeneous Graph Attention Network. In *WWW*. 2022–2032.

[53] Rui Wen, Jianyu Wang, Chunming Wu, and Jian Xiong. 2020. ASA: Adversary Situation Awareness via Heterogeneous Graph Convolutional Networks. In *WWW*. 674–678.

[54] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *ICLR*.

[55] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. 2020. Heterogeneous Network Representation Learning: Survey, Benchmark, Evaluation, and Beyond. *CoRR* abs/2004.00216 (2020).

[56] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*. 974–983.

[57] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *NeurIPS* (2020).

[58] Feng Zhang, Jidong Zhai, Bingsheng He, Shuhao Zhang, and Wenguang Chen. 2016. Understanding co-running behaviors on integrated CPU/GPU architectures. *IEEE Transactions on Parallel and Distributed Systems* 28, 3 (2016), 905–918.

[59] Feng Zhang, Jidong Zhai, Xipeng Shen, Dalin Wang, Zheng Chen, Onur Mutlu, Wenguang Chen, and Xiaoyong Du. 2021. TADOC: Text analytics directly on compression. *The VLDB Journal* 30, 2 (2021), 163–188.

[60] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Hung Nguyen, Zi Huang, and Lizhen Cui. 2020. GCN-Based User Representation Learning for Unifying Robust Recommendation and Fraudster Detection. In *SIGIR*.

[61] Hang Zhao, Yujing Wang, Juanyong Duan, Congrui Huang, Defu Cao, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. 2020. Multivariate Time-series Anomaly Detection via Graph Attention Network. In *ICDM*. 841–850.