Few-shot Network Anomaly Detection via Cross-network Meta-learning

Kaize Ding†, Qinghai Zhou‡, Hanghang Tong‡, Huan Liu†
†Arizona State University, {kaize.ding, huan.liu}@asu.edu
‡University of Illinois at Urbana-Champaign, {qinghai2, htong}@illinois.edu

ABSTRACT
Network anomaly detection aims to find network elements (e.g., nodes, edges, subgraphs) with significantly different behaviors from the vast majority. It has a profound impact in a variety of applications ranging from finance, healthcare to social network analysis. Due to the unbearable labeling cost, existing methods are predominantly developed in an unsupervised manner. Nonetheless, the anomalies they identify may turn out to be data noises or uninteresting data instances due to the lack of prior knowledge on the anomalies of interest. Hence, it is critical to investigate and develop few-shot learning for network anomaly detection. In real-world scenarios, few labeled anomalies are also easy to be accessed on similar networks from the same domain as of the target network, while most of the existing works omit to leverage them and merely focus on a single network. Taking advantage of this potential, in this work, we tackle the problem of few-shot network anomaly detection by (1) proposing a new family of graph neural networks – Graph Deviation Networks (GDN) that can leverage a small number of labeled anomalies for enforcing statistically significant deviations between abnormal and normal nodes on a network; (2) equipping the proposed GDN with a new cross-network meta-learning algorithm to realize few-shot network anomaly detection by transferring meta-knowledge from multiple auxiliary networks. Extensive evaluations demonstrate the efficacy of the proposed approach on few-shot or even one-shot network anomaly detection.

ACM Reference Format:

1 INTRODUCTION
Network-structured data, ranging from social networks [45] to team collaboration networks [53], from citation networks [32] to molecular graphs [44], has been widely used in modeling a myriad of real-world systems. Nonetheless, real-world networks are commonly contaminated with a small portion of nodes, namely, anomalies, whose patterns significantly deviate from the vast majority of nodes [5, 7, 50]. For instance, in a citation network that represents citation relations between papers, there are some research papers with a few spurious references (i.e., edges) which do not comply with the content of the papers [2]; In a social network that represents friendship of users, there may exist camouflaged users who randomly follow different users, rendering properties like homophily not applicable to this type of relationships [9]. As the existence of even few abnormal instances could cause extremely detrimental effects, the problem of network anomaly detection has received much attention in industry and academy alike.

Due to the fact that labeling anomalies is highly labor-intensive and takes specialized domain-knowledge, existing methods are predominantly developed in an unsupervised manner. As a prevailing paradigm, people try to measure the abnormality of nodes with the reconstruction errors of autoencoder-based models [6, 18] or the residuals of matrix factorization-based methods [2, 17, 34]. However, the anomalies they identify may turn out to be data noises or uninteresting data instances due to the lack of prior knowledge on the anomalies of interest. A potential solution to this problem is to leverage limited or few labeled anomalies as the prior knowledge to learn anomaly-informed models, since it is relatively low-cost in real-world scenarios – a small set of labeled anomalies could be either from a deployed detection system or be provided by user feedback. In the meantime, such valuable knowledge is usually scattered among other networks within the same domain of the target one, which could be further exploited for distilling supervised signal. For example, LinkedIn and Indeed have similar social

Figure 1: Since anomalies usually have distinct patterns, (a) existing methods may easily fail to distinguish them from normal nodes in the latent representation space with only few labeled anomalies, (b) while they can be well separated in an anomaly score space by enforcing statistically significant deviations between abnormal and normal nodes.

The first two authors contributed equally to this work.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

WWW ’21, April 19–23, 2021, Ljubljana, Slovenia
© 2021 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
https://doi.org/10.1145/3442381.3449922

†In this paper, we primarily focus on detecting abnormal nodes.
networks that represent user friendship in the job-search domain; ACM and DBLP can be treated as citation networks that share similar citation relations in the computer science domain. According to previous studies [33, 51, 52], because of the similarity of topological structure and nodal attributes, it is feasible to transfer valuable knowledge from source network(s) to the target network so that the performance on the target one is elevated. As such, in this work we propose to investigate the novel problem of few-shot network anomaly detection under the cross-network setting.

Nonetheless, solving this under-explored problem remains non-trivial, mainly owing to the following reasons: (1) From the micro (intra-network) view, since we only have limited knowledge of anomalies, it is hard to precisely characterize the abnormal patterns. If we directly adopt existing semi-supervised [37] or PU [40] learning techniques, those methods often fail short in achieving satisfactory results as they might still require a relatively large percentage of positive examples [23]. To handle such incomplete supervision challenge [47] as illustrated in Figure 1(a), instead of focusing on abnormal nodes, how to leverage labeled anomalies as few as possible to learn a high-level abstraction of normal patterns is necessary to be explored; (2) From the macro (inter-network) view, though networks in the same domain might share similar characteristics in general, anomalies exist in different networks may be from very different manifolds. Previous studies on cross-network learning [29, 41] mostly focus on transferring the knowledge only from a single network, which may cause unstable results and the risk of negative transfer. As learning from multiple networks could provide more comprehensive knowledge about the characteristics of anomalies, a cross-network learning algorithm that is capable of adapting the knowledge is highly desirable.

To address the aforementioned challenges, in this work we first design a new GNN architecture, namely Graph Deviation Networks (GDN), to enable network anomaly detection with limited labeled data. Specifically, given an arbitrary network, GDN first uses a GNN-backboned anomaly score learner to assign each node with an anomaly score, and then defines the mean of the anomaly scores based on a prior probability to serve as a reference score for guiding the subsequent anomaly score learning. By leveraging a deviation loss [23], GDN is able to enforce statistically significant deviations of the anomaly scores of anomalies from that of normal nodes in the anomaly score space (as shown in Figure 1(b)). To further transfer this ability from multiple networks to the target one, we propose a cross-network meta-learning algorithm to learn a well-generalized initialization of GDN from multiple few-shot network anomaly detection tasks. The seamlessly integrated framework Meta-GDN is capable of extracting comprehensive meta-knowledge for detecting anomalies across multiple networks, which largely alleviates the limitations of transferring from a single network. Subsequently, the initialization can be easily adapted to a target network via fine-tuning with few or even one labeled anomaly, improving the anomaly detection performance on the target network to a large extent. To summarize, our main contributions is three-fold:

- **Problem**: To the best of knowledge, we are the first to investigate the novel problem of few-shot network anomaly detection. Remarkably, we propose to solve this problem by transferring the knowledge across multiple networks.

- **Algorithms**: We propose a principled framework Meta-GDN, which integrates a new family of graph neural networks (i.e., GDN) and cross-network meta-learning to detect anomalies with few labeled instances.

- **Evaluations**: We perform extensive experiments to corroborate the effectiveness of our approach. The experimental results demonstrate the superior performance of Meta-GNN over the state-of-the-art methods on network anomaly detection.

## 2 RELATED WORK

In this section, we review the related work in terms of (1) network anomaly detection; and (2) graph neural networks.

### 2.1 Network Anomaly Detection

Network anomaly detection methods have a specific focus on the network structured data. Previous research mostly study the problem of anomaly detection on plain networks. As network structure is the only available information modality in a plain network, this category of anomaly detection methods try to exploit the network structure information to spot anomalies from different perspectives [1, 43]. For instance, SCAN [43] is one of the first methods that target to find structural anomalies in networks. In recent days, attributed networks have been widely used to model a wide range of complex systems due to their superior capacity for handling data heterogeneity. In addition to the observed node-to-node interactions, attributed networks also encode a rich set of features for each node. Therefore, anomaly detection on attributed networks has drawn increasing research attention in the community, and various methods have been proposed [22, 26]. Among them, ConOut [22] identifies the local context for each node and performs anomaly ranking within the local context. More recently, researchers also propose to solve the problem of network anomaly detection using graph neural networks due to its strong modeling power. DOMINANT [6] achieves superior performance over other shallow methods by building a deep autoencoder architecture on top of the graph convolutional networks. Semi-GNN [37] is a semi-supervised graph neural model which adopts hierarchical attention to model the multi-view graph for fraud detection. GAS [16] is a GCN-based large-scale anti-scam method for detecting spam advertisements. Zhao et al. propose a novel loss function to train GNNs for anomaly-detectable node representations [48]. Apart from the aforementioned methods, our approach focus on detecting anomalies on a target network with few labels by learning from multiple auxiliary networks.

### 2.2 Graph Neural Networks

Graph neural networks [4, 12, 14, 35] have achieved groundbreaking success in transforming the information of a graph into low-dimensional latent representations. Originally inspired by graph spectral theory, spectral-based graph convolutional networks (GCNs) have emerged and demonstrated their efficacy by designing different graph convolutional layers. Among them, The model proposed by Kipf et al. [14] has become the most prevailing one by using a linear filter. In addition to spectral-based graph convolution models, spatial-based graph neural networks that follow neighborhoods aggregation schemes also have been extensively
investigated. Instead of training individual embeddings for each node, those methods learn a set of aggregator functions to aggregate features from a node’s local neighborhood. GraphSAGE [12] learns an embedding function that can be generalized to unseen nodes, which enables inductive representation learning on network-structured data. Similarly, Graph Attention Networks (GATs) [35] proposes to learn hidden representations by introducing a self-attention strategy when aggregating neighborhood information of a node. Furthermore, Graph Isomorphism Network (GIN) [42] extends the idea of parameterizing universal multiset functions with neural networks, and is proven to be as theoretically powerful as the Weisfeiler-Lehman (WL) graph isomorphism test. To go beyond a single graph and transfer the knowledge across multiple ones, more recently, researchers have explored to integrate GNNs with meta-learning techniques [33, 51, 54]. For instance, PA-GNN [33] transfers the robustness from cleaned graphs to the target graph via meta-optimization. Meta-NA [51] is a graph alignment model that learns a unified metric space across multiple graphs, where one can easily link entities across different graphs. However, those efforts cannot be applied to our problem and we are the first to study the problem of few-shot cross-network anomaly detection.

3 PROBLEM DEFINITION

In this section, we formally define the problem of few-shot cross-network anomaly detection. Throughout the paper, we use bold uppercase letters for matrices (e.g., A), bold lowercase letters for vectors (e.g., x), lowercase letters for scalars (e.g., s) and calligraphic fonts to denote sets (e.g., \( \mathcal{V} \)). Notably, in this work we focus on attributed network for a more general purpose. Given an attributed network \( G = (\mathcal{V}, \mathcal{E}, \mathcal{X}) \) where \( \mathcal{V} \) is the set of nodes, \( \mathcal{E} \) denotes the set of edges, \( \mathcal{X} \) the set of attributes; \( \mathcal{A} \) is the attribute vector for node \( v_i \). More concretely, we represent the attributed network as \( G = (\mathcal{A}, \mathcal{X}) \) where \( \mathcal{A} = \{0, 1\}^{n \times d} \) is an adjacency matrix representing the network structure. Specifically, \( A_{i,j} = 1 \) indicates that there is an edge between node \( u_i \) and node \( v_j \); otherwise, \( A_{i,j} = 0 \).

Generally speaking, few-shot cross-network anomaly detection aims to maximally improve the detection performance on the target network through transferring very limited supervised knowledge of ground-truth anomalies from the auxiliary network(s). In addition to the target network \( G^\prime \), in this work we assume there exist \( p \) auxiliary networks \( \mathcal{G}^A = \{G_1^A, G_2^A, \ldots, G_p^A\} \) sharing the same or similar domain with \( G \). For an attributed network, the set of labeled abnormal nodes is denoted as \( \mathcal{V}_L \) and the set of unlabeled nodes is represented as \( \mathcal{V}_U \). Note that \( \mathcal{V} = \mathcal{V}_L \cup \mathcal{V}_U \) and in our problem \( |\mathcal{V}_L| \ll |\mathcal{V}_U| \) since only few-shot labeled data is given. As network anomaly detection is commonly formulated as a ranking problem [1], we formally define the few-shot cross-network anomaly detection problem as follows:

**Problem 1. Few-shot Cross-network Anomaly Detection**

**Given:** \( p \) auxiliary networks, i.e., \( \mathcal{G}^A = \{G_1^A = (A_1^A, X_1^A), G_2^A = (A_2^A, X_2^A), \ldots, G_p^A = (A_p^A, X_p^A)\} \) and a target network \( G^\prime = (A^\prime, X^\prime) \), each of which contains a set of few-shot labeled anomalies (i.e., \( \mathcal{V}_L^1, \mathcal{V}_L^2, \ldots, \mathcal{V}_L^p \) and \( \mathcal{V}_U^1 \)).

**Goal:** to learn an anomaly detection model, which is capable of leveraging the knowledge of ground-truth anomalies from the multiple auxiliary networks, i.e., \( \mathcal{G}^A = \{G_1^A, G_2^A, \ldots, G_p^A\} \), to detect abnormal nodes in the target network \( G^\prime \). Ideally, anomalies that are detected should have higher ranking scores than that of the normal nodes.

4 PROPOSED APPROACH

In this section, we introduce the details of the proposed framework – Meta-GDN for few-shot network anomaly detection. Specifically, Meta-GDN addresses the discussed challenges with the following two key contributions: (1) Graph Deviation Networks (GDN), a new family of graph neural networks that enable anomaly detection on an arbitrary individual network with limited labeled data; and (2) a cross-network meta-learning algorithm, which empowers GDN to transfer meta-knowledge across multiple auxiliary networks to enable few-shot anomaly detection on the target network. An overview of the proposed Meta-GDN is provided in Figure 2.

4.1 Graph Deviation Networks

To enable anomaly detection on an arbitrary network with few-shot labeled data, we first propose a new family of graph neural networks, called Graph Deviation Network (GDN). In essence, GDN is
composed of three key building blocks, including (1) a network encoder for learning node representations; (2) an abnormality valuator for estimating the anomaly score for each node; and (3) a deviation loss for optimizing the model with few-shot labeled anomalies. The details are as follows:

**Network Encoder.** In order to learn expressive nodes representations from an input network, we first build the network encoder module. Specifically, it is built with multiple GNN layers that encode each node to a low-dimensional latent representation. In general, GNNs follow the neighborhood message-passing mechanism, and compute the node representations by aggregating features from local neighborhoods in an iterative manner. Formally, a generic GNN layer computes the node representations using two key functions:

\[
\begin{align*}
    h_i^l &= \text{AGGREGATE}^l(h_{j}^{l-1} | \forall j \in N_i \cup \{ i \}), \\
    h_i^l &= \text{TRANSFORM}^l(h_{i}^{l-1}, f_{\theta_i}^l),
\end{align*}
\]

where \( h_i^l \) is the latent representation of node \( v_i \) at the \( l \)-th layer and \( N_i \) is the set of first-order neighboring nodes of node \( v_i \). Notably, \( \text{AGGREGATE}() \) is an aggregation function that aggregates messages from neighboring nodes and \( \text{TRANSFORM}() \) computes the new representation of a node according to its previous-layer representation and the aggregated messages from neighbors.

To capture the long-range node dependencies in the network, we stack multiple GNN layers in the network encoder. Thus, the network encoder can be represented by:

\[
\begin{align*}
    H^1 &= \text{GNN}^1(A, X), \\
    \vdots \\
    Z &= \text{GNN}^L(A, H^{L-1}),
\end{align*}
\]

where \( Z \) is the learned node representations from the network encoder. For simplicity, we use a parameterized function \( f_{\theta_i}() \) to denote the network encoder with \( L \) GNN layers throughout the paper. It is worth noting that the network encoder is compatible with arbitrary GNN-based architecture [12, 14, 35, 39], and here we employ Simple Graph Convolution (SGC) [39] in our implementation.

**Abnormality Valuator.** Afterwards, the learned node representations from the network encoder will be passed to the abnormality valuator \( f_{\theta}() \) for further estimating the abnormality of each node. Specifically, the abnormality valuator is built with two feed-forward layers that transform the intermediate node representations to scalar anomaly scores:

\[
\begin{align*}
    o_i &= \text{ReLU}(W_i z_i + b_i), \\
    s_i &= u_i^T o_i + b_s,
\end{align*}
\]

where \( z_i \) is the anomaly score of node \( v_i \) and \( o_i \) is the intermediate output. \( W_i \) and \( u_i \) are the learnable weight matrix and weight vector, respectively. \( b_i \) and \( b_s \) are corresponding bias terms.

To be more concrete, the whole GDN model \( f_{\theta}(\cdot) \) can be formally represented as:

\[
f_{\theta}(A, X) = f_{\theta}(f_{\theta}(A, X)),
\]

which directly maps the input network to scalar anomaly scores, and can be trained in an end-to-end fashion.

**Deviation Loss.** In essence, the objective of GDN is to distinguish normal and abnormal nodes according to the computed anomaly scores with few-shot labels. Here we propose to adopt the deviation loss [23] to enforce the model to assign large anomaly scores to those nodes whose characteristics significantly deviate from normal nodes. To guide the model learning, we first define a reference score (i.e., \( \mu_v \)) as the mean value of the anomaly scores of a set of randomly selected normal nodes. It serves as the reference to quantify how much the scores of anomalies deviate from those of normal nodes.

According to previous studies [15, 23], Gaussian distribution is commonly a robust choice to fit the abnormality scores for a wide range of datasets. Based on this assumption, we first sample a set of \( k \) anomaly scores from the Gaussian prior distribution, i.e., \( \mathcal{R} = \{ r_1, r_2, \ldots, r_k \} \sim \mathcal{N}(\mu, \sigma^2) \), each of which denotes the abnormality of a random normal node. The reference score is computed as the mean value of all the sampled scores:

\[
\mu_v = \frac{1}{k} \sum_{i=1}^{k} r_i,
\]

With the reference score \( \mu_v \), the deviation between the anomaly score of node \( v_i \) and the reference score can be defined in the form of standard score:

\[
\text{dev}(v_i) = \frac{s_i - \mu_v}{\sigma},
\]

where \( \sigma_v \) is the standard deviation of the set of sampled anomaly scores \( \mathcal{R} = \{ r_1, \ldots, r_k \} \). Then the final objective function can be derived from the contrastive loss [11] by replacing the distance function with the deviation in Eq. (6):

\[
\mathcal{L} = (1 - y_i) \cdot \text{dev}(v_i) + y_i \cdot \max(0, m - \text{dev}(v_i)),
\]

where \( y_i \) is the ground-truth label of input node \( v_i \). If node \( v_i \) is an abnormal node, \( y_i = 1 \), otherwise, \( y_i = 0 \). Note that \( m \) is a confidence margin which defines a radius around the deviation.

By minimizing the above loss function, GDN will push the anomaly scores of normal nodes as close as possible to \( \mu_v \) while enforcing a large positive deviation of at least \( m \) between \( \mu_v \) and the anomaly scores of abnormal nodes. This way GDN is able to learn a high-level abstraction of normal patterns with substantially less labeled anomalies, and empowers the node representation learning to discriminate normal nodes from the rare anomalies. Accordingly, a large anomaly score will be assigned to a node if its pattern significantly deviates from the learned abstraction of normal patterns.

Our preliminary results show that GDN is not sensitive to the choices of \( \mu \) and \( \sigma \) as long as \( \sigma \) is not too large. Specifically, we set \( \mu = 0 \) and \( \sigma = 1 \) in our experiments, which helps GDN to achieve stable detection performance on different datasets. It is also worth mentioning that, as we cannot access the labels of normal nodes, we simply consider the unlabeled node in \( V^U \) as normal. Note that this way the remaining unlabeled anomalies and all the normal nodes will be treated as normal, thus contamination is introduced to the training set (i.e., the ratio of unlabeled anomalies to the total unlabeled training data \( V^U \)). Remarkably, GDN performs very well by using this simple strategy and is robust to different contamination levels. The effect of different contamination levels to model performance is evaluated in Sec. 5.4.

### 4.2 Cross-network Meta-learning

Having the proposed Graph Deviation Networks (GDN), we are able to effectively detect anomalies on an arbitrary network with limited
labeled data. When auxiliary networks from the same domain of the target network are available, how to transfer such valuable knowledge is the key to enable few-shot anomaly detection on the target network. Despite its feasibility, the performance would be rather limited if we directly borrow the idea of existing cross-network learning methods. The main reason is that those methods merely focus on transferring the knowledge from only a single network [29, 41], which may cause negative transfer due to the divergent characteristics of anomalies on different networks. To this end, we turn to exploit multiple auxiliary networks to distill comprehensive knowledge of anomalies.

As an effective paradigm for extracting and transferring knowledge, meta-learning has recently received increasing research attention because of the broad applications in a variety of high-impact domains [8, 19, 20, 27, 36, 38]. In essence, the goal of meta-learning is to train a model on a variety of learning tasks, such that the learned model is capable of effectively adapting to new tasks with very few or even one labeled data [13]. In particular, Finn et al. [10] propose a model-agnostic meta-learning algorithm to explicitly learn the model parameters such that the model can achieve good generalization to a new task through a small number of gradient steps with limited labeled data. Inspired by this work, we propose to learn a meta-learner (i.e., Meta-GDN) as the initialization of GDN from multiple auxiliary networks, which possesses the generalization ability to effectively identify anomalous nodes on a new target network. Specifically, Meta-GDN extracts meta-knowledge of ground-truth anomalies from different few-shot network anomaly detection tasks on auxiliary networks during the training phase, and will be further fine-tuned for the new task on the target network, such that the model can make fast and effective adaptation.

We define each learning task as performing few-shot anomaly detection on an individual network, whose objective is to enforce large anomaly scores to be assigned to anomalies as defined in Eq. (7). Let \( T_i \) denote the few-shot network anomaly detection task constructed from network \( G^i \), then we have \( P \) learning tasks in each epoch. We consider a GDN model represented by a parameterized function \( f_\theta \) with parameters \( \theta \). Given \( P \) tasks, the optimization algorithm first adapts the initial model parameters \( \theta \) to \( \theta' \) for each learning task \( T_i \) independently. Specifically, the updated parameter \( \theta' \) is computed using \( L_{T_i} \) on a batch of training data sampled from \( V_{i}^V \) and \( V_{i}^V \) in \( G^i \). Formally, the parameter update with one gradient step can be expressed as:

\[
\theta' = \theta - \alpha \nabla_\theta L_{T_i}(f_\theta).
\]

where \( \alpha \) controls the meta-learning rate. Note that Eq. (8) only includes one-step gradient update, while it is straightforward to extend to multiple gradient updates [10].

The model parameters are trained by optimizing for the best performance of \( f_\theta \) with respect to \( \theta \) across all learning tasks. More concretely, the meta-objective function is defined as follows:

\[
\min_\theta \sum_{i=1}^P L_{T_i}(f_\theta) = \min_\theta \sum_{i=1}^P L_{T_i}(f_\theta - \alpha \nabla_\theta L_{T_i}(f_\theta)).
\]

By optimizing the objective of GDN, the updated model parameter can preserve the capability of detecting anomalies on each network. Since the meta-optimization is performed over parameters \( \theta \) with the objective computed using the updated parameters

### Algorithm 1: The learning algorithm of Meta-GDN

**Input:** (1) \( P \) auxiliary networks, i.e., \( G^1 = (A^1, X^1), G^2 = (A^2, X^2), \ldots, G^P = (A^P, X^P) \); (2) a target network \( G^T = (A^T, X^T) \); (3) sets of few-shot labeled anomalies and unlabeled nodes for each network (i.e., \( \{V^L_1, V^U_1\}, \ldots, \{V^L_P, V^U_P\} \) and \( \{V^L_T, V^U_T\} \)); (4) training epochs \( E \), batch size \( b \), and meta-learning hyper-parameters \( \alpha, \beta \).

**Output:** Anomaly scores of nodes in \( V^U_T \).

1. Initialize parameters \( \theta \);
2. while \( e < E \) do
3. for each network \( G^i \) (task \( T_i \)) do
4. Randomly sample \( \frac{b}{2} \) nodes from \( V^U_i \) and \( \frac{b}{2} \) from \( V^U_i \) to comprise the batch \( B_i \);
5. Evaluate \( \nabla_\theta \mathcal{L}(f_\theta) \) using \( B_i \) and \( \mathcal{L}(\cdot) \) in Eq. (7);
6. Compute adapted parameters \( \theta' \) with gradient descent using Eq. (8), \( \theta' \leftarrow \theta - \beta \nabla_\theta \mathcal{L}_{T_i}(f_\theta) \);
7. Sample a new batch \( B_i' \) for the meta-update;
8. end for
9. Update \( \theta \leftarrow \theta - \beta \nabla_\theta \sum_{i=1}^P \mathcal{L}_{T_i}(f_\theta) \) using \( \{B_i'\} \) and \( \mathcal{L}(\cdot) \) according to Eq. (7);
10. end while
11. Fine-tune \( \theta \) on target network \( G^T \) with \( \{\mathcal{V}^L_T, \mathcal{V}^U_T\} \);
12. Compute anomaly scores for nodes in \( \mathcal{V}^U_T \);

(i.e., \( \theta' \)) for all tasks, correspondingly, the model parameters are optimized such that one or a small number of gradient steps on the target task (network) will produce great effectiveness.

Formally, we leverage stochastic gradient descent (SGD) to update the model parameters \( \theta \) across all tasks, such that the model parameters \( \theta \) are updated as follows:

\[
\theta \leftarrow \theta - \beta \nabla_\theta \sum_{i=1}^P \mathcal{L}_{T_i}(f_\theta),
\]

where \( \beta \) is the meta step size. The full algorithm is summarized in Algorithm 1. Specifically, for each batch, we randomly sample the same number of nodes from unlabeled data (i.e., \( \mathcal{V}^U_T \)) and labeled anomalies (i.e., \( \mathcal{V}^L_T \)) to represent normal and abnormal nodes, respectively (Step-4).

## 5 EXPERIMENTS

In this section, we perform empirical evaluations to demonstrate the effectiveness of the proposed framework. Specifically, we aim to answer the following research questions:

- **RQ1.** How effective is the proposed approach Meta-GDN for detecting anomalies on the target network with few or even one labeled instance?
- **RQ2.** How much will the performance of Meta-GDN change by providing different numbers of auxiliary networks or different anomaly contamination levels?
- **RQ3.** How does each component of Meta-GDN (i.e., graph deviation networks or cross-network meta-learning) contribute to the final detection performance?
Table 1: Statistics of evaluation datasets. $r_1$ denotes the ratio of labeled anomalies to the total anomalies and $r_2$ is the ratio of labeled anomalies to the total number of nodes.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Yelp</th>
<th>PubMed</th>
<th>Reddit</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes (avg.)</td>
<td>4,872</td>
<td>3,675</td>
<td>15,860</td>
</tr>
<tr>
<td># edges (avg.)</td>
<td>43,728</td>
<td>8,895</td>
<td>136,781</td>
</tr>
<tr>
<td># features</td>
<td>10,000</td>
<td>500</td>
<td>602</td>
</tr>
<tr>
<td># anomalies (avg.)</td>
<td>223</td>
<td>201</td>
<td>796</td>
</tr>
<tr>
<td>$r_1$ (avg.)</td>
<td>4.48%</td>
<td>4.97%</td>
<td>1.26%</td>
</tr>
<tr>
<td>$r_2$ (avg.)</td>
<td>0.21%</td>
<td>0.27%</td>
<td>0.063%</td>
</tr>
</tbody>
</table>

5.1 Experimental Setup

Evaluation Datasets. In the experiment, we adopt three real-world datasets, which are publicly available and have been widely used in previous research [12, 14, 24, 28]. Table 1 summarizes the statistics of each dataset. The detailed description is as follows:

- **Yelp** [24] is collected from Yelp.com and contains reviews for restaurants in several states of the U.S., where the restaurants are organized by ZIP codes. The reviewers are classified into two classes, abnormal (reviewers with only filtered reviews) and normal (reviewers with no filtered reviews) according to the Yelp anti-fraud filtering algorithm. We select restaurants in the same location according to ZIP codes to construct each network, where nodes represent reviewers and there is a link between two reviewers if they have reviewed the same restaurant. We apply the bag-of-words model [46] on top of the textual contents to obtain the attributes of each node.

- **PubMed** [28] is a citation network where nodes represent scientific articles related to diabetes and edges are citations relations. Node attribute is represented by a TF/IDF weighted word vector from a dictionary which consists of 500 unique words. We randomly partition the large network into non-overlapping sub-networks of similar size.

- **Reddit** [12] is collected from an online discussion forum where nodes represent threads and an edge exits between two threads if they are commented by the same user. The node attributes are constructed using averaged word embedding vectors of the threads. Similarly, we extract non-overlapping sub-networks from the original large network for our experiments.

Note that except the **Yelp** dataset, we are not able to access ground-truth anomalies for **PubMed** and **Reddit**. Thus we refer to two anomaly injection methods [7, 31] to inject a combined set of anomalies (i.e., structural anomalies and contextual anomalies) by perturbing the topological structure and node attributes of the original network, respectively. To inject structural anomalies, we adopt the approach used by [7] to generate a set of small cliques since small clique is a typical abnormal substructure in which a small set of nodes are much more closely linked to each other than average [30]. Accordingly, we randomly select $c$ nodes (i.e., clique size) in the network and then make these nodes fully linked to each other. By repeating this process $K$ times (i.e., $K$ cliques), we can obtain $K \times c$ structural anomalies. In our experiment, we set the clique size $c$ to 15. In addition, we leverage the method introduced by [31] to generate contextual anomalies. Specifically, we first randomly select a node $i$ and then randomly sample another 50 nodes from the network. We choose the node $j$ whose attributes have the largest Euclidean distance from node $i$ among the 50 nodes. The attributes of node $i$ (i.e., $x_i$) will then be replaced with the attributes of node $j$ (i.e., $x_j$). Note that we inject structural and contextual anomalies with the same quantity and the total number of injected anomalies is around 5% of the network size.

Comparison Methods. We compare our proposed Meta-GDN framework and its base model GDN with two categories of anomaly detection methods, including (1) feature-based methods (i.e., LOF, Autoencoder and DeepSAD) where only the node attributes are considered, and (2) network-based methods (i.e., SCAN, ConOut, Radar, DOMINANT, and SemiGNN) where both topological information and node attributes are involved. Details of these compared baseline methods are as follows:

- **LOF** [3] is a feature-based approach which detects outliers at the contextual level.

- **Autoencoder** [49] is a feature-based unsupervised deep autoencoder model which introduces an anomaly regularizing penalty based upon L1 or L2 norms.

- **DeepSAD** [25] is a state-of-the-art deep learning approach for general semi-supervised anomaly detection. In our experiment, we leverage the node attribute as the input feature.

- **SCAN** [43] is an efficient algorithm for detecting network anomalies based on a structural similarity measure.

- **ConOut** [26] identifies network anomalies according to the corresponding subgraph and the relevant subset of attributes in the local context.

- **Radar** [17] is an unsupervised method that detects anomalies on attributed network by characterizing the residuals of attribute information and its coherence with network structure.

- **DOMINANT** [6] is a GCN-based autoencoder framework which computes anomaly scores using the reconstruction errors from both network structure and node attributes.

- **SemiGNN** [37] is a semi-supervised GNN model, which leverages the hierarchical attention mechanism to better correlate different neighbors and different views.

Evaluation Metrics. In this paper, we use the following metrics to have a comprehensive evaluation of the performance of different anomaly detection methods:

- **AUC-ROC** is widely used in previous anomaly detection research [6, 17]. Area under curve (AUC) is interpreted as the probability that a randomly chosen anomaly receives a higher score than a randomly chosen normal object.

- **AUC-PR** is the area under the curve of precision against recall at different thresholds, and it only evaluates the performance on the positive class (i.e., abnormal objects). AUC-PR is computed as the average precision as defined in [21] and is used as the evaluation metric in [23].

- **Precision@K** is defined as the proportion of true anomalies in a ranked list of $K$ objects. We obtain the ranking list in descending order according to the anomaly scores that are computed from a specific anomaly detection algorithm.

Implementation Details. Regarding the proposed GDN model, we use Simple Graph Convolution [39] to build the network encoder with degree $K=2$ (two layers). As shown in Eq. (3), the abnormality...
Table 2: Performance comparison results (10-shot) w.r.t. AUC-ROC and AUC-PR on three datasets.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Yelp AUC-ROC</th>
<th>Yelp AUC-PR</th>
<th>PubMed AUC-ROC</th>
<th>PubMed AUC-PR</th>
<th>Reddit AUC-ROC</th>
<th>Reddit AUC-PR</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOF</td>
<td>0.375 ± 0.011</td>
<td>0.042 ± 0.004</td>
<td>0.575 ± 0.007</td>
<td>0.187 ± 0.016</td>
<td>0.518 ± 0.015</td>
<td>0.071 ± 0.006</td>
</tr>
<tr>
<td>Autoencoder</td>
<td>0.365 ± 0.013</td>
<td>0.041 ± 0.008</td>
<td>0.584 ± 0.018</td>
<td>0.236 ± 0.005</td>
<td>0.722 ± 0.012</td>
<td>0.347 ± 0.007</td>
</tr>
<tr>
<td>DeepSAD</td>
<td>0.460 ± 0.008</td>
<td>0.062 ± 0.005</td>
<td>0.528 ± 0.008</td>
<td>0.115 ± 0.004</td>
<td>0.503 ± 0.010</td>
<td>0.066 ± 0.005</td>
</tr>
<tr>
<td>SCAN</td>
<td>0.397 ± 0.011</td>
<td>0.046 ± 0.005</td>
<td>0.421 ± 0.016</td>
<td>0.048 ± 0.005</td>
<td>0.298 ± 0.009</td>
<td>0.048 ± 0.002</td>
</tr>
<tr>
<td>ConOut</td>
<td>0.402 ± 0.015</td>
<td>0.041 ± 0.005</td>
<td>0.511 ± 0.019</td>
<td>0.093 ± 0.012</td>
<td>0.551 ± 0.008</td>
<td>0.085 ± 0.007</td>
</tr>
<tr>
<td>Radar</td>
<td>0.415 ± 0.012</td>
<td>0.045 ± 0.007</td>
<td>0.573 ± 0.013</td>
<td>0.244 ± 0.011</td>
<td>0.721 ± 0.008</td>
<td>0.281 ± 0.007</td>
</tr>
<tr>
<td>DOMINANT</td>
<td>0.578 ± 0.018</td>
<td>0.109 ± 0.003</td>
<td>0.636 ± 0.021</td>
<td>0.337 ± 0.013</td>
<td>0.735 ± 0.013</td>
<td>0.357 ± 0.009</td>
</tr>
<tr>
<td>SemiGNN</td>
<td>0.497 ± 0.004</td>
<td>0.058 ± 0.003</td>
<td>0.523 ± 0.008</td>
<td>0.065 ± 0.006</td>
<td>0.610 ± 0.007</td>
<td>0.134 ± 0.003</td>
</tr>
<tr>
<td>GDN (ours)</td>
<td>0.678 ± 0.015</td>
<td>0.132 ± 0.009</td>
<td>0.736 ± 0.012</td>
<td>0.438 ± 0.012</td>
<td>0.811 ± 0.015</td>
<td>0.379 ± 0.011</td>
</tr>
<tr>
<td>Meta-GDN (ours)</td>
<td>0.724 ± 0.012</td>
<td>0.175 ± 0.011</td>
<td>0.761 ± 0.014</td>
<td>0.485 ± 0.010</td>
<td>0.842 ± 0.011</td>
<td>0.395 ± 0.009</td>
</tr>
</tbody>
</table>

Figure 3: Performance comparison results (10-shot) w.r.t. Precision@K on three datasets. Figure best viewed in color.

5.2 Effectiveness Results (RQ1)

Overall Comparison. In the experiments, we evaluate the performance of the proposed framework Meta-GDN along with its base model GDN by comparing with the included baseline methods. We first present the evaluation results (10-shot) w.r.t. AUC-ROC and AUC-PR in Table 2 and the results w.r.t. Precision@K are visualized in Figure 3. Accordingly, we have the following observations, including: (1) in terms of AUC-ROC and AUC-PR, our approach Meta-GDN outperforms all the other compared methods by a significant margin. Meanwhile, the results w.r.t. Precision@K again demonstrate that Meta-GDN can better rank abnormal nodes on higher positions than other methods by estimating accurate anomaly scores; (2) unsupervised methods (e.g., DOMINANT, Radar) are not able to leverage supervised knowledge of labeled anomalies and therefore have limited performance. Semi-supervised methods (e.g., DeepSAD, SemiGNN) also fail to deliver satisfactory results. The possible explanation is that DeepSAD cannot model network information and SemiGNN requires a relatively large number of labeled data and multi-view data, which make them less effective in our evaluation; and (3) compared to the base model GDN, Meta-GDN is capable of extracting comprehensive meta-knowledge across multiple auxiliary networks by virtue of the cross-network meta-learning algorithm, which further enhances the detection performance on the target network.

Few-shot Evaluation. In order to verify the effectiveness of Meta-GDN in few-shot as well as one-shot network anomaly detection, we evaluate the performance of Meta-GDN with different numbers...
of labeled anomalies on the target network (i.e., 1-shot, 3-shot, 5-shot and 10-shot). Note that we respectively set the batch size $b$ to 2, 4, 8, and 16 to ensure that there is no duplication of labeled anomalies exist in a sampled training batch. Also, we keep the number of labeled anomalies on auxiliary networks as 10. Table 3 summarizes the AUC-ROC/AUC-PR performance of Meta-GDN under different few-shot settings. By comparing the results in Table 2 and Table 3, we can see that even with only one labeled anomaly on the target network (i.e., 1-shot), Meta-GDN can still achieve good performance and significantly outperforms all the baseline methods. In the meantime, we can clearly observe that the performance of Meta-GDN increases with the growth of the number of labeled anomalies, which demonstrates that Meta-GDN can be better fine-tuned on the target network with more labeled examples.

Table 3: Few-shot performance evaluation of Meta-GDN w.r.t. AUC-ROC and AUC-PR.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Yelp AUC-ROC</th>
<th>Yelp AUC-PR</th>
<th>PubMed AUC-ROC</th>
<th>PubMed AUC-PR</th>
<th>Reddit AUC-ROC</th>
<th>Reddit AUC-PR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-shot</td>
<td>0.702 ± 0.008</td>
<td>0.159 ± 0.015</td>
<td>0.742 ± 0.012</td>
<td>0.462 ± 0.013</td>
<td>0.821 ± 0.013</td>
<td>0.380 ± 0.011</td>
</tr>
<tr>
<td>3-shot</td>
<td>0.709 ± 0.006</td>
<td>0.164 ± 0.010</td>
<td>0.748 ± 0.008</td>
<td>0.468 ± 0.008</td>
<td>0.828 ± 0.012</td>
<td>0.386 ± 0.007</td>
</tr>
<tr>
<td>5-shot</td>
<td>0.717 ± 0.013</td>
<td>0.169 ± 0.007</td>
<td>0.753 ± 0.011</td>
<td>0.474 ± 0.005</td>
<td>0.834 ± 0.009</td>
<td>0.389 ± 0.008</td>
</tr>
<tr>
<td>10-shot</td>
<td>0.724 ± 0.012</td>
<td>0.175 ± 0.011</td>
<td>0.761 ± 0.014</td>
<td>0.485 ± 0.010</td>
<td>0.842 ± 0.011</td>
<td>0.395 ± 0.009</td>
</tr>
</tbody>
</table>

5.3 Sensitivity & Robustness Analysis (RQ2)

In this section, we further analyze the sensitivity and robustness of the proposed framework Meta-GDN. By providing different numbers of auxiliary networks during training, the model sensitivity results w.r.t. AUC-ROC are presented in Figure 4(a). Specifically, we can clearly find that (1) as the number of auxiliary networks increases, Meta-GDN achieves constantly stronger performance on all the three datasets. It shows that more auxiliary networks can provide better meta-knowledge during the training process, which is consistent with our intuition; (2) Meta-GDN can still achieve relatively good performance when training with a small number of auxiliary networks (e.g., $p = 2$), which demonstrates the strong capability of its base model GDN. For example, on Yelp dataset, the performance barely drops 0.033 if we change the number of auxiliary networks from $p = 6$ to $p = 2$.

Figure 4: (a) Sensitivity analysis of Meta-GDN w.r.t. different number of auxiliary networks; (b) Model robustness study w.r.t. AUC-ROC with different contamination levels.

5.4 Ablation Study (RQ3)

As discussed in Sec. 4.1, we treat all the sampled nodes from unlabeled data as normal for computing the deviation loss. This simple strategy introduces anomaly contamination in the unlabeled training data. Due to the fact that $r_c$ is a small number in practice, our approach can work very well in a wide range of real-world datasets. To further investigate the robustness of Meta-GDN w.r.t. different contamination levels $r_c$ (i.e., the proportion of anomalies in the unlabeled training data), we report the evaluation results of Meta-GDN, GDN and the semi-supervised baseline method SemiGNN in Figure 4(b). As shown in the figure, though the performance of all the methods decreases with increasing contamination levels, both Meta-GDN and GDN are remarkably robust and can consistently outperform SemiGNN to a large extent.

Figure 5: (a) AUC-ROC results of Meta-GDN and its variants; (b) Precision@100 results of Meta-GDN and its variants.

Moreover, we conduct an ablation study to better examine the contribution of each key component in the proposed framework. In addition to Meta-GDN and its base model GDN, we include another variant GDN$^-$ that excludes the network encoder and cross-network meta-learning in Meta-GDN. We present the results of AUC-ROC and Precision@100 in Figure 5(a) and Figure 5(b), respectively. The corresponding observations are two-fold: (1) by incorporating GNN-based network encoder, GDN largely outperforms GDN$^-$ in anomaly detection on the target network. For example, GDN achieves 8.1% performance improvement over GDN$^-$ on PubMed in terms of precision@100. The main reason is that the GNN-based network encoder is able to extract topological information of nodes and to learn highly expressive node representations; and (2) the complete framework Meta-GDN performs consistently better than the base model GDN on all the three datasets. For instance, Meta-GDN improves AUC-ROC by 5.75% over GDN on
Yelp dataset, which verifies the effectiveness of the proposed cross-network meta-learning algorithm for extracting and transferring meta-knowledge across multiple auxiliary networks.

6 CONCLUSION

In this paper, we make the first investigation on the problem of few-shot cross-network anomaly detection. To tackle this problem, we first design a novel GNN architecture, GDN, which is capable of leveraging limited labeled anomalies to enforce statistically significant deviations between abnormal and normal nodes on an individual network. To further utilize the knowledge from auxiliary networks, we propose a cross-network meta-learning approach, Meta-GDN, which is able to extract comprehensive meta-knowledge from multiple auxiliary networks in the same domain of the target network. Through extensive experimental evaluations, we demonstrate the superiority of Meta-GDN over the state-of-the-art methods.

ACKNOWLEDGEMENT

This work is partially supported by NSF (2029044, 1947135 and 1939725) and ONR (N00014-21-1-4002).

REFERENCES

[16] Ao Li, Zhou Qin, Runshu Liu, Yiqun Yang, and Dong Li. 2019. Spam review detection with graph convolutional networks. In CIKM.

Few-shot Network Anomaly Detection via Cross-network Meta-learning

WWW ’21, April 19–23, 2021, Ljubljana, Slovenia