Supplementary of Zero-shot Generalist Graph Anomaly Detection with Unified Neighborhood Prompts

Chaoxi Niu[1†], **Hezhe Qiao**[2†], **Changlu Chen**[3], **Ling Chen**[1], **Guansong Pang**[2*]
[1] AAII, University of Technology Sydney, Sydney, Australia
[2] School of Computing and Information Systems, Singapore Management University, Singapore
[3]Faculty of Data Science, City University of Macau, Macau, China
chaoxi.niu@student.uts.edu.au, hezheqiao.2022@phdcs.smu.edu.sg,
clchen@cityu.edu.mo, ling.chen@uts.edu.au, gspang@smu.edu.sg

1 Graph Similarity

In addition to the visualization results presented in Figure 1 in the main paper, we further provide the distributional similarity of various graphs in this section. Specifically, for dimension-aligned graphs across different distributions and domains, we measure their distributional similarity to analyze their diverse semantics.

Given a graph $\mathcal{G}^{(i)} = (A^{(i)}, \tilde{X}^{(i)})$, the coordinate-wise mean $\boldsymbol{\mu}^{(i)} = [\mu_1^{(i)}, \ldots, \mu_{d'}^{(i)}]$ and variance $\boldsymbol{\sigma}^{(i)} = [\sigma_1^{(i)}, \ldots, \sigma_{d'}^{(i)}]$ of $\tilde{X}^{(i)}$ are calculated and concatenated to form the distributional vector of $\mathcal{G}^{(i)}$, *i.e.*, $\mathbf{d}_i = [\boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{(i)}]$. Then, the distribution similarity between $\mathcal{G}^{(i)}$ and $\mathcal{G}^{(j)}$ is measured via the cosine similarity,

$$s_{ij} = \text{sim}(\mathbf{d}_i, \mathbf{d}_j)\,. \tag{S.1}$$

The distributional similarities between graphs from different domains or distributions are shown in Figure 1(a).. From the figure, we can see that the distributional similarities are typically small, demonstrating the diverse semantics of node features across graphs. Noth that, for Amazon & Amazon-all and YelpChi & YelpChi-all, their distribution similarity is one, which can be attributed to the fact that they are from the same distributions respectively but with different numbers of nodes and structures.

To reduce the semantic gap among graphs for generalist GAD, we propose to calibrate the distributions of all graphs into the same frame with coordinate-wise normalization. The distributional similarity with normalization is illustrated in Figure 1(b). It is clear that the node attributes share the same distribution after the normalization. In this way, the generalist model can better capture the shared GAD patterns and generalize to different target graphs, as demonstrated in our experimental results.

2 Details on Pre-training of Neighborhood Aggregation Networks

We pre-train the neighborhood aggregation network g via graph contrastive learning [Zhu *et al.*, 2020] for subsequent graph prompt learning so that the generic normality and abnormality can be captured in the prompts.

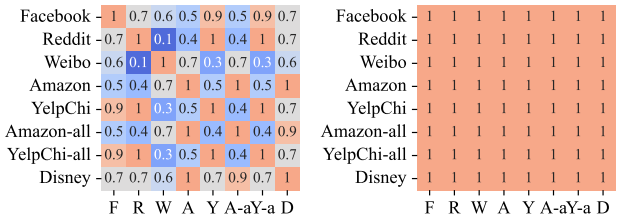


Figure 1: **(a)** Distributional similarity between different graphs without coordinate-wise normalization. **(b)** Distributional similarity between different graphs with the coordinate-wise normalization.

Specifically, given the training graph $\mathcal{G} = (A, X)$, to construct contrastive views for graph contrastive learning, two widely used graph augmentations are employed, *i.e.*, edge removal and attribute masking [Zhu *et al.*, 2020]. The edge removal randomly drops a certain portion of existing edges in $\mathcal{G}$ and the attribute masking randomly masks a fraction of dimensions with zeros in node attributes, *i.e.*,

$$\hat{A} = A \circ R\,, \quad \hat{X} = [\mathbf{x}_1 \circ \mathbf{m}, \ldots, \mathbf{x}_N \circ \mathbf{m}]^T\,, \tag{S.2}$$

where $R \in \{0,1\}^{N\times N}$ is the edge masking matrix whose entry is drawn from a Bernoulli distribution controlled by the edge removal probability, $\mathbf{m} \in \{0,1\}^d$ is the attribute masking vector whose entry is independently drawn from a Bernoulli distribution with the attribute masking ratio, and $\circ$ denotes the Hadamard product.

By applying the graph augmentations to the original graph, the corrupted graph $\hat{\mathcal{G}} = (\hat{A}, \hat{X})$ forms the contrastive view for the original graph $\mathcal{G} = (A, X)$. Then, $\hat{\mathcal{G}}$ and $\mathcal{G}$ are fed to the shared model g followed by the non-linear projection to obtain the corresponding node embeddings, *i.e.*, $\hat{Z}'$ and Z'. For graph contrastive learning, the embeddings of the same node in different views are pulled closer while the embeddings of other nodes are pushed apart. The pairwise objective

*Corresponding author: G. Pang (gspang@smu.edu.sg). † indicates equal contribution.

for each node pair $(\hat{\mathbf{z}}'_i, \mathbf{z}'_i)$ can be formulated as:

$$\ell(\hat{\mathbf{z}}'_i, \mathbf{z}'_i) = -\log \frac{e^{\text{sim}(\hat{\mathbf{z}}'_i, \mathbf{z}'_i)/\tau}}{e^{\text{sim}(\hat{\mathbf{z}}'_i, \mathbf{z}'_i)/\tau} + \sum_{j\neq i}^{N} e^{\text{sim}(\hat{\mathbf{z}}'_i, \mathbf{z}'_j)/\tau} + \sum_{j\neq i}^{N} e^{\text{sim}(\hat{\mathbf{z}}'_i, \hat{\mathbf{z}}'_j)/\tau}}, \tag{S.3}$$

where $\text{sim}(\cdot)$ represents the cosine similarity and τ is a temperature hyperparameter. Therefore, the overall objective can be defined as follows:

$$\mathcal{L}_{\text{contrast}} = \frac{1}{2N} \sum_{i=1}^{N} (\ell(\hat{\mathbf{z}}'_i, \mathbf{z}'_i) + \ell(\mathbf{z}'_i, \hat{\mathbf{z}}'_i)) . \tag{S.4}$$

With the objective Eq.(S.4), the model g is optimized to learn transferable discriminative representations of nodes.

3 Algorithms

The training and inference processes of UNPrompt are summarized in Algorithms 1 and Algorithm 2, respectively.

Algorithm 1 Training of UNPrompt

Input: Training graph $\mathcal{G}_{\text{train}} = (A, X)$; training epoch E
Output: Neighborhood aggregation network g, graph prompts $P = [\mathbf{p}_1, \ldots, \mathbf{p}_K]$, and transformation h.
Perform feature unification of X.
Pre-train g on $\mathcal{G}_{\text{train}}$ with graph contrastive learning in Eq.(S.4).
Keep model g frozen.
for $e = 1, \ldots, E$ **do**
    Obtain modified node attribute with prompts via Eq.(6).
    Obtain the neighborhood aggregated representation $\tilde{Z}$ via Eq.(3).
    Obtain the node representations Z via Eq.(4).
    Transform $\tilde{Z}$ and Z with h via Eq.(7).
    Optimize P and h by minimizing Eq.(8).
end for

Algorithm 2 Inference of UNPrompt

Input: Testing graphs $\mathcal{T}_{\text{test}} = \{\mathcal{G}^{(1)}_{\text{test}}, \ldots, \mathcal{G}^{(n)}_{\text{test}}\}$, neighborhood aggregation network g, graph prompts $P = [\mathbf{p}_1, \ldots, \mathbf{p}_K]$, and transformation h.
Output: Normal score of testing nodes.
for $\mathcal{G}^{(i)}_{\text{test}} = (A^{(i)}, X^{(i)}) \in \mathcal{T}_{\text{test}}$ **do**
    Perform feature unification of $X^{(i)}$.
    Obtain modified node attribute with prompts via Eq.(6).
    Obtain the neighborhood aggregated representation $\tilde{Z}^{(i)}$ via Eq.(3).
    Obtain the node representations $Z^{(i)}$ via Eq.(4).
    Transform $\tilde{Z}^{(i)}$ and $Z^{(i)}$ with h via Eq.(7).
    Obtain the normal score of nodes via Eq.(5).
end for

4 Unsupervised GAD with UNPrompt

To demonstrate the wide applicability of the proposed method UNPrompt, we further perform unsupervised GAD with UNPrompt which focuses on detecting anomalous nodes within one graph and does not have access to any node labels during training. Specifically, we adopt the same pipeline in the generalist GAD setting, *i.e.*, graph contrastive pertaining and neighborhood prompt learning. Since we focus on anomaly detection on each graph separately, the node attribute unification is discarded for unsupervised GAD. However, the absence of node labels poses a challenge to learning meaningful neighborhood prompts for anomaly detection. To overcome this issue, we propose to utilize the pseudo-labeling technique to guide the prompt learning. Specifically, the normal score of each node is calculated by the neighborhood-based latent attribute predictability after the graph contrastive learning process:

$$s_i = \text{sim}(\mathbf{z}_i, \tilde{\mathbf{z}}_i) , \tag{S.5}$$

where $\mathbf{z}_i$ is the node representation learned by graph contrastive learning and $\tilde{\mathbf{z}}_i$ is the corresponding aggregated neighborhood representation. Higher s_i of node v_i typically indicates a higher probability of v_i being a normal node. Therefore, more emphasis should be put on high-score nodes when learning neighborhood prompts. To achieve this, the normal score s_i is transformed into the loss weight $w_i = \text{Sigmoid}(\alpha(s_i - t))$ where t is a threshold and α is the scaling parameter. In this way, w_i would approach 1 if $s_i > t$ and 0 otherwise. Overall, the objective for unsupervised GAD using UNPrompt can be formulated as follows:

$$\mathcal{L} = \sum_{i}^{N} (-w_i \text{sim}(\mathbf{z}_i, \tilde{\mathbf{z}}_i) + \lambda \sum_{j,j\neq i}^{N} \text{sim}(\mathbf{z}_i, \tilde{\mathbf{z}}_j)) , \tag{S.6}$$

where the second term is a regularization term employed to prevent the node embeddings from being collapsed into the same and λ is a trade-off hyperparameter.

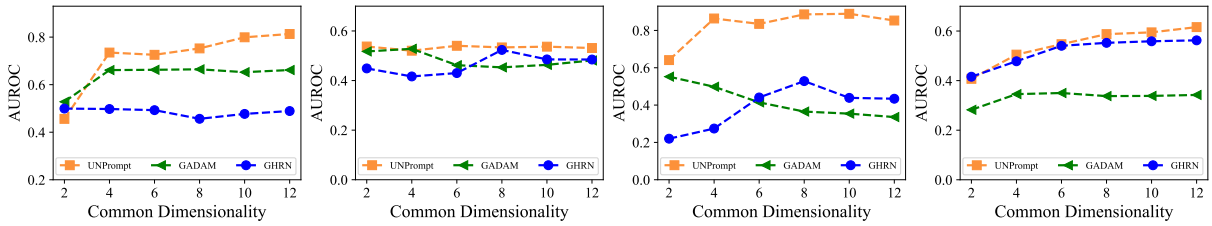
Note that we only focus on maximizing the latent attribute predictability of high-score nodes without minimizing the predictability of low-score nodes in the above objective. These low-score nodes could also be normal nodes with high probability as the score from Eq.(S.5) is only obtained from the pre-trained model, resulting in the score not being fully reliable. If the predictability is also minimized for these nodes, conflicts would be induced for neighborhood prompt learning, limiting the performance of unsupervised GAD. After optimization, the latent attribute predictability is also directly used as the normal score for the final unsupervised GAD.

4.1 Experimental Setup.

Six datasets from different distributions and domains are used, *i.e.*, Amazon, Facebook, Reddit, YelpChi, Amazon-all, and YelpChi-all. Following [Qiao and Pang, 2023], eight SotA unsupervised baselines are used for comparison, *i.e.*, iForest [Liu *et al.*, 2012], ANOMALOUS [Peng *et al.*, 2018], CoLA [Liu *et al.*, 2021a], SL-GAD [Zheng *et al.*, 2021], HCM-A [Huang *et al.*, 2022], DOMINANT [Ding *et al.*, 2019], ComGA [Luo *et al.*, 2022] and TAM [Qiao and Pang, 2023]. For each method, we report the average performance with standard deviations after 5 independent runs with different random seeds. The implementation details of UNPrompt remain the same as in the generalist GAD setting. More experimental details on unsupervised GAD are in Sec. 7.2.

Metric	Method	Dataset						
		Amazon	Facebook	Reddit	YelpChi	Amazon-all	YelpChi-all	Avg.
AUROC	iForest (TKDD'12)	$0.5621_{\pm 0.008}$	$0.5382_{\pm 0.015}$	$0.4363_{\pm 0.020}$	$0.4120_{\pm 0.040}$	$0.1914_{\pm 0.002}$	$0.3617_{\pm 0.001}$	0.4169
	ANOMALOUS (IJCAI'18)	$0.4457_{\pm 0.003}$	$0.9021_{\pm 0.005}$	$0.5387_{\pm 0.012}$	$0.4956_{\pm 0.003}$	$0.3230_{\pm 0.021}$	$0.3474_{\pm 0.018}$	0.5087
	DOMINANT (SIAM'19)	$0.5996_{\pm 0.004}$	$0.5677_{\pm 0.002}$	$0.5555_{\pm 0.011}$	$0.4133_{\pm 0.010}$	$0.6937_{\pm 0.028}$	$0.5390_{\pm 0.014}$	0.5615
	CoLA (TNNLS'21)	$0.5898_{\pm 0.008}$	$0.8434_{\pm 0.011}$	$\underline{0.6028}_{\pm 0.007}$	$0.4636_{\pm 0.001}$	$0.2614_{\pm 0.021}$	$0.4801_{\pm 0.016}$	0.5402
	SL-GAD (TKDE'21)	$0.5937_{\pm 0.011}$	$0.7936_{\pm 0.005}$	$0.5677_{\pm 0.005}$	$0.3312_{\pm 0.035}$	$0.2728_{\pm 0.012}$	$0.5551_{\pm 0.015}$	0.5190
	HCM-A (ECML-PKDD'22)	$0.3956_{\pm 0.014}$	$0.7387_{\pm 0.032}$	$0.4593_{\pm 0.011}$	$0.4593_{\pm 0.005}$	$0.4191_{\pm 0.011}$	$0.5691_{\pm 0.018}$	0.5069
	ComGA (WSDM'22)	$0.5895_{\pm 0.008}$	$0.6055_{\pm 0.000}$	$0.5453_{\pm 0.003}$	$0.4391_{\pm 0.000}$	$0.7154_{\pm 0.014}$	$0.5352_{\pm 0.006}$	0.5716
	TAM (NeurIPS'23)	$\underline{0.7064}_{\pm 0.010}$	$\underline{0.9144}_{\pm 0.008}$	$0.6023_{\pm 0.004}$	$\underline{0.5643}_{\pm 0.007}$	$\underline{0.8476}_{\pm 0.028}$	$\underline{0.5818}_{\pm 0.033}$	$\underline{0.7028}$
	UNPrompt (Ours)	$\mathbf{0.7335}_{\pm 0.020}$	$\mathbf{0.9379}_{\pm 0.006}$	$\mathbf{0.6067}_{\pm 0.006}$	$\mathbf{0.6223}_{\pm 0.007}$	$\mathbf{0.8516}_{\pm 0.004}$	$\mathbf{0.6084}_{\pm 0.001}$	**0.7267**
AUPRC	iForest (TKDD'12)	$0.1371_{\pm 0.002}$	$0.0316_{\pm 0.003}$	$0.0269_{\pm 0.001}$	$0.0409_{\pm 0.000}$	$0.0399_{\pm 0.001}$	$0.1092_{\pm 0.001}$	0.0643
	ANOMALOUS (IJCAI'18)	$0.0558_{\pm 0.001}$	$0.1898_{\pm 0.004}$	$0.0375_{\pm 0.004}$	$0.0519_{\pm 0.002}$	$0.0321_{\pm 0.001}$	$0.0361_{\pm 0.005}$	0.0672
	DOMINANT (SIAM'19)	$0.1424_{\pm 0.002}$	$0.0314_{\pm 0.041}$	$0.0356_{\pm 0.002}$	$0.0395_{\pm 0.020}$	$0.1015_{\pm 0.018}$	$0.1638_{\pm 0.007}$	0.0857
	CoLA (TNNLS'21)	$0.0677_{\pm 0.001}$	$0.2106_{\pm 0.017}$	$\underline{0.0449}_{\pm 0.002}$	$0.0448_{\pm 0.002}$	$0.0516_{\pm 0.001}$	$0.1361_{\pm 0.015}$	0.0926
	SL-GAD (TKDE'21)	$0.0634_{\pm 0.005}$	$0.1316_{\pm 0.020}$	$0.0406_{\pm 0.004}$	$0.0350_{\pm 0.000}$	$0.0444_{\pm 0.001}$	$0.1711_{\pm 0.011}$	0.0810
	HCM-A (ECML-PKDD'22)	$0.0527_{\pm 0.015}$	$0.0713_{\pm 0.004}$	$0.0287_{\pm 0.005}$	$0.0287_{\pm 0.012}$	$0.0565_{\pm 0.003}$	$0.1154_{\pm 0.004}$	0.0589
	ComGA (WSDM'22)	$0.1153_{\pm 0.005}$	$0.0354_{\pm 0.001}$	$0.0374_{\pm 0.001}$	$0.0423_{\pm 0.000}$	$0.1854_{\pm 0.003}$	$0.1658_{\pm 0.003}$	0.0969
	TAM (NeurIPS'23)	$\underline{0.2634}_{\pm 0.008}$	$\underline{0.2233}_{\pm 0.016}$	$0.0446_{\pm 0.001}$	$\underline{0.0778}_{\pm 0.009}$	$\underline{0.4346}_{\pm 0.021}$	$\underline{0.1886}_{\pm 0.017}$	$\underline{0.2054}$
	UNPrompt (Ours)	$\mathbf{0.2688}_{\pm 0.060}$	$\mathbf{0.2622}_{\pm 0.028}$	$\mathbf{0.0450}_{\pm 0.001}$	$\mathbf{0.0895}_{\pm 0.004}$	$\mathbf{0.6094}_{\pm 0.014}$	$\mathbf{0.2068}_{\pm 0.004}$	**0.2470**

Table 1: AUROC and AUPRC results of unsupervised GAD methods on six real-world GAD datasets. The best performance per column within each metric is boldfaced, with the second-best underlined. "Avg" denotes the average performance of each method.



(a) Amazon (b) Reddit (c) Weibo (d) YelpChi

Figure 2: AUROC results of UNPrompt and other baselines with different common dimensionalities.

4.2 Main Results.

The AUROC and AUPRC results of all methods are presented in Table 1. Despite being a generalist GAD method, UNPrompt works very well as a specialized GAD model too. UNPrompt substantially outperforms all the competing methods on all datasets in terms of both AUROC and AUPRC. Particularly, the average performance of UNPrompt surpasses the best-competing method TAM by over 2% in both metrics. Moreover, UNPrompt outperforms the best-competing method by 2%-6% in AUROC on most of the datasets. The superior performance shows that the latent node attribute predictability can be a generalized GAD measure that holds for different graphs, and this property can be effectively learned by the proposed neighborhood prompting method.

5 More Experimental Results

5.1 Generalist Performance with different common dimensionalities

For the results reported in the main paper, the common dimensionality is set to eight. In this subsection, we further evaluate the generalist anomaly detection with different common dimensionalities. Specifically, the dimensionality varies in $[2, 4, 6, 8, 10, 12]$ and the results are reported in Figure 2. Without loss of generality, we conduct the analysis on Amazon, Reddit, Weibo and YelpChi and employ two baselines (GADAM and GHRN) for comparisons which are unsupervised and supervised methods, respectively.

From the figure, we can see that small dimensionality leads to poor generalist anomaly detection performance of UNPrompt. This is attributed to the fact that much attribute information would be discarded with a small dimensionality. By increasing the common dimensionality, more attribute information is retained, generally resulting in much better detection performance. However, this trend does not hold for the baselines as the baselines achieve inconsistent performance gain or loss on different datasets with the increase of the common dimensionality. This is largely attributed to the fact that these one-model-for-one-dataset methods fail to capture the generalized anomaly measure.

5.2 Incorporating coordinate-wise normalization into baselines

We further conduct experiments by incorporating the proposed coordinate-wise normalization into the baselines to evaluate whether the normalization could facilitate the baselines. Without loss of generality, three unsupervised methods (AnomalyDAE, CoLA and TAM) and three supervised methods (GCN, BWGNN and GHRN) are used and the results are reported in Table 2.

From the table, we can see that the proposed coordinate-wise normalization does not improve the baselines consis-

Metric	Method	Dataset							Avg.
		Amazon	Reddit	Weibo	YelpChi	Aamzon-all	YelpChi-all	Disney	
AUROC	Unsupervised Methods								
	AnomalyDAE	$0.5818_{\pm 0.039}$	$0.5016_{\pm 0.032}$	$\underline{0.7785}_{\pm 0.058}$	$0.4837_{\pm 0.094}$	$0.7228_{\pm 0.023}$	$0.5002_{\pm 0.018}$	$0.4853_{\pm 0.003}$	$\underline{0.5791}$
	+ CN	$0.4359_{\pm 0.053}$	$0.4858_{\pm 0.063}$	$0.4526_{\pm 0.074}$	$\mathbf{0.5992}_{\pm 0.028}$	$0.2833_{\pm 0.039}$	$0.5080_{\pm 0.013}$	$0.5042_{\pm 0.065}$	0.4670
	CoLA	$0.4580_{\pm 0.054}$	$0.4623_{\pm 0.005}$	$0.3924_{\pm 0.041}$	$0.4907_{\pm 0.017}$	$0.4091_{\pm 0.052}$	$0.4879_{\pm 0.010}$	$0.4696_{\pm 0.065}$	0.4529
	+ CN	$0.4729_{\pm 0.019}$	$0.5299_{\pm 0.008}$	$0.3401_{\pm 0.026}$	$0.3640_{\pm 0.006}$	$0.5424_{\pm 0.019}$	$0.4882_{\pm 0.008}$	$0.5593_{\pm 0.079}$	0.4710
	TAM	$0.4720_{\pm 0.005}$	$\mathbf{0.5725}_{\pm 0.004}$	$0.4867_{\pm 0.028}$	$0.5035_{\pm 0.014}$	$0.7543_{\pm 0.002}$	$0.4216_{\pm 0.002}$	$0.4773_{\pm 0.003}$	0.5268
	+ CN	$0.4509_{\pm 0.015}$	$0.5526_{\pm 0.006}$	$0.4723_{\pm 0.007}$	$0.5189_{\pm 0.006}$	$\underline{0.7580}_{\pm 0.004}$	$0.4057_{\pm 0.002}$	$0.2431_{\pm 0.029}$	0.4859
	Supervised Methods								
	GCN	$\underline{0.5988}_{\pm 0.016}$	$\underline{0.5645}_{\pm 0.000}$	$0.2232_{\pm 0.074}$	$0.5366_{\pm 0.019}$	$0.7195_{\pm 0.002}$	$\underline{0.5486}_{\pm 0.001}$	$0.5000_{\pm 0.000}$	0.5273
	+ CN	$0.5694_{\pm 0.014}$	$0.5349_{\pm 0.008}$	$0.0632_{\pm 0.005}$	$0.3954_{\pm 0.002}$	$0.6798_{\pm 0.009}$	$0.5550_{\pm 0.005}$	$0.5507_{\pm 0.015}$	0.4783
	BWGNN	$0.4769_{\pm 0.020}$	$0.5208_{\pm 0.016}$	$0.4815_{\pm 0.108}$	$0.5538_{\pm 0.027}$	$0.3648_{\pm 0.050}$	$0.5282_{\pm 0.015}$	$0.6073_{\pm 0.026}$	0.5048
	+ CN	$0.4745_{\pm 0.048}$	$0.4942_{\pm 0.011}$	$0.2538_{\pm 0.038}$	$0.4727_{\pm 0.016}$	$0.6307_{\pm 0.077}$	$0.5221_{\pm 0.025}$	$0.6042_{\pm 0.039}$	0.4932
	GHRN	$0.4560_{\pm 0.033}$	$0.5253_{\pm 0.006}$	$0.5318_{\pm 0.038}$	$0.5524_{\pm 0.020}$	$0.3382_{\pm 0.085}$	$0.5125_{\pm 0.016}$	$0.5336_{\pm 0.030}$	0.4928
	+ CN	$0.4308_{\pm 0.024}$	$0.5061_{\pm 0.026}$	$0.2621_{\pm 0.043}$	$0.4781_{\pm 0.018}$	$0.5712_{\pm 0.046}$	$0.5200_{\pm 0.009}$	$\underline{0.6220}_{\pm 0.032}$	0.4843
	UNPrompt (Ours)	$\mathbf{0.7525}_{\pm 0.016}$	$0.5337_{\pm 0.002}$	$\mathbf{0.8860}_{\pm 0.007}$	$\underline{0.5875}_{\pm 0.016}$	$\mathbf{0.7962}_{\pm 0.022}$	$\mathbf{0.5558}_{\pm 0.012}$	$\mathbf{0.6412}_{\pm 0.030}$	**0.6790**
AUPRC	Unsupervised Methods								
	AnomalyDAE	$0.0833_{\pm 0.015}$	$0.0327_{\pm 0.004}$	$\underline{0.6064}_{\pm 0.031}$	$0.0624_{\pm 0.017}$	$0.1921_{\pm 0.026}$	$0.1484_{\pm 0.009}$	$0.0566_{\pm 0.000}$	$\underline{0.1688}$
	+ CN	$0.0596_{\pm 0.009}$	$0.0333_{\pm 0.007}$	$0.1910_{\pm 0.049}$	$\mathbf{0.0874}_{\pm 0.011}$	$0.0495_{\pm 0.006}$	$0.1527_{\pm 0.007}$	$\underline{0.1232}_{\pm 0.023}$	0.0995
	CoLA	$0.0669_{\pm 0.002}$	$0.0391_{\pm 0.004}$	$0.1189_{\pm 0.014}$	$0.0511_{\pm 0.000}$	$0.0861_{\pm 0.019}$	$0.1466_{\pm 0.003}$	$0.0701_{\pm 0.023}$	0.0827
	+ CN	$0.0669_{\pm 0.002}$	$0.0360_{\pm 0.002}$	$0.1618_{\pm 0.027}$	$0.0370_{\pm 0.000}$	$0.0934_{\pm 0.017}$	$0.1446_{\pm 0.005}$	$0.0870_{\pm 0.025}$	0.0895
	TAM	$0.0666_{\pm 0.001}$	$\underline{0.0413}_{\pm 0.001}$	$0.1240_{\pm 0.014}$	$0.0524_{\pm 0.002}$	$0.1736_{\pm 0.004}$	$0.1240_{\pm 0.001}$	$0.0628_{\pm 0.001}$	0.0921
	+ CN	$0.0606_{\pm 0.003}$	$0.0394_{\pm 0.001}$	$0.1044_{\pm 0.005}$	$0.0542_{\pm 0.001}$	$\mathbf{0.2482}_{\pm 0.013}$	$0.1213_{\pm 0.001}$	$0.0366_{\pm 0.003}$	0.0950
	Supervised Methods								
	GCN	$\underline{0.0891}_{\pm 0.007}$	$\mathbf{0.0439}_{\pm 0.000}$	$0.1109_{\pm 0.020}$	$0.0648_{\pm 0.009}$	$0.1536_{\pm 0.002}$	$\underline{0.1735}_{\pm 0.000}$	$0.0484_{\pm 0.000}$	0.0977
	+ CN	$0.0770_{\pm 0.003}$	$0.0355_{\pm 0.001}$	$0.0548_{\pm 0.000}$	$0.0401_{\pm 0.000}$	$0.1383_{\pm 0.006}$	$0.1789_{\pm 0.002}$	$0.0968_{\pm 0.020}$	0.0888
	BWGNN	$0.0652_{\pm 0.002}$	$0.0389_{\pm 0.003}$	$0.2241_{\pm 0.046}$	$0.0708_{\pm 0.018}$	$0.0586_{\pm 0.003}$	$0.1605_{\pm 0.005}$	$0.0624_{\pm 0.003}$	0.0972
	+ CN	$0.0684_{\pm 0.014}$	$0.0320_{\pm 0.001}$	$0.2576_{\pm 0.031}$	$0.0516_{\pm 0.004}$	$0.1557_{\pm 0.115}$	$0.1585_{\pm 0.010}$	$0.0975_{\pm 0.016}$	0.1173
	GHRN	$0.0633_{\pm 0.003}$	$0.0407_{\pm 0.002}$	$0.1965_{\pm 0.059}$	$0.0661_{\pm 0.010}$	$0.0569_{\pm 0.006}$	$0.1505_{\pm 0.005}$	$0.0519_{\pm 0.003}$	0.0894
	+ CN	$0.0586_{\pm 0.004}$	$0.0330_{\pm 0.002}$	$0.2663_{\pm 0.038}$	$0.0525_{\pm 0.004}$	$0.0898_{\pm 0.015}$	$0.1570_{\pm 0.007}$	$0.1051_{\pm 0.021}$	0.1089
	UNPrompt (Ours)	$\mathbf{0.1602}_{\pm 0.013}$	$0.0351_{\pm 0.000}$	$\mathbf{0.6406}_{\pm 0.026}$	$\underline{0.0712}_{\pm 0.008}$	$\underline{0.2430}_{\pm 0.028}$	$\mathbf{0.1810}_{\pm 0.012}$	$\mathbf{0.1236}_{\pm 0.031}$	**0.2078**

Table 2: AUROC and AUPRC results of several baselines with coordinate-wise normalization (CN).

tently but downgrades most of the baselines. This can be attributed to two reasons. First, while the proposed coordinate-wise normalization unifies the semantics of different graphs into a common space, the discrimination between normal and abnormal patterns would also be compressed. This requires the generalist anomaly detector to capture the fine-grained differences between normal and abnormal patterns. Second, these baselines are not designed to capture generalized abnormality and normality across graphs, failing to capture and discriminate the generalized nuance. By contrast, we reveal that the predictability of latent node attributes can serve as a generalized anomaly measure and learn highly generalized normal and abnormal patterns via latent node attribute prediction. In this way, the graph-agnostic anomaly measure can be well generalized across graphs.

5.3 Generalist performance with different training graphs

In the main paper, we report the generalist performance of UNPrompt by using Facebook as the training graph. To further demonstrate the generalizability of UNPrompt, we conduct additional experiments by using Amazon as the training graph and testing the learned generalist model on the rest graphs. Note that Facebook and Amazon are from different domains, which are the social network and the co-review network respectively.

The AUROC and AUPRC results of all methods are reported in Table 3. Similar to the observations when taking Facebook as the training graph, UNPrompt achieves the best average performance in terms of both AUROC and AUPRC when training on Amazon, demonstrating the generalizability and effectiveness of UNPrompt with different training graphs. Note that the training graph Amazon and the target graph Amazon-all come from the same distribution but have different numbers of nodes and graph structures. Intuitively, all the methods should achieve promising performance on Amazon-all. However, only a few methods achieve this goal, including BWGNN, GHRN, XGBGraph, and our method. The failures of other baselines can be attributed to the more complex graph structure of Amazon-all hinders the generalizability of these methods. Moreover, compared to BWGNN, GHRN and XGBGraph, our method performs more stably across different datasets. This demonstrates the importance of capturing intrinsic normal and abnormal patterns for graph anomaly detection.

5.4 Sensitivity of Unsuperivised GAD w.r.t the threshold

To evaluate the sensitivity of the unsupervised GAD performance of our method w.r.t the threshold, we vary the threshold in the range of $[5\%, 50\%]$ at a step size of 5%. Without loss of generality, three datasets are employed, including Facebook, Reddit, and Amazon. The results of these datasets are reported in the following table.

From the table, we can see that the unsupervised performance of UNPrompt can perform well and stably with a sufficiently large threshold (e.g., no less than 30%), but it may drop significantly with a small threshold, e.g., thresholds like 5%-15% that are close to the ground-truth anomaly rate. This is because more abnormal nodes would have a substantially higher chance of being mistakenly treated as normal nodes with such a small threshold, which would in turn mislead

Metric	Method	Dataset							Avg.
		Facebook	Reddit	Weibo	YelpChi	Aamzon-all	YelpChi-all	Disney	
AUROC	Unsupervised Methods								
	AnomalyDAE	$0.6123_{\pm 0.141}$	$\mathbf{0.5799}_{\pm 0.035}$	$\underline{0.7884}_{\pm 0.031}$	$0.4788_{\pm 0.046}$	$0.6233_{\pm 0.070}$	$0.4912_{\pm 0.009}$	$0.4938_{\pm 0.005}$	0.5811
	CoLA	$0.5427_{\pm 0.109}$	$0.4962_{\pm 0.025}$	$0.3987_{\pm 0.017}$	$0.3358_{\pm 0.012}$	$0.4751_{\pm 0.014}$	$0.4937_{\pm 0.003}$	$0.5455_{\pm 0.031}$	0.4697
	HCM-A	$0.5044_{\pm 0.047}$	$0.4993_{\pm 0.057}$	$0.4937_{\pm 0.056}$	$0.5000_{\pm 0.000}$	$0.4785_{\pm 0.016}$	$0.4958_{\pm 0.003}$	$0.2051_{\pm 0.034}$	0.4538
	TAM	$0.5496_{\pm 0.038}$	$\underline{0.5764}_{\pm 0.003}$	$0.4876_{\pm 0.029}$	$0.5091_{\pm 0.014}$	$0.7525_{\pm 0.002}$	$0.4268_{\pm 0.002}$	$0.4850_{\pm 0.004}$	0.5410
	GADAM	$0.6024_{\pm 0.033}$	$0.4720_{\pm 0.062}$	$0.4324_{\pm 0.047}$	$0.4299_{\pm 0.023}$	$0.5199_{\pm 0.072}$	$0.5289_{\pm 0.017}$	$0.3966_{\pm 0.021}$	0.4832
	Supervised Methods								
	GCN	$\underline{0.6892}_{\pm 0.004}$	$0.5658_{\pm 0.000}$	$0.2355_{\pm 0.019}$	$0.5277_{\pm 0.002}$	$0.7503_{\pm 0.002}$	$0.5565_{\pm 0.000}$	$0.5000_{\pm 0.000}$	0.5464
	GAT	$0.3886_{\pm 0.118}$	$0.4997_{\pm 0.012}$	$0.3897_{\pm 0.134}$	$0.5051_{\pm 0.019}$	$0.5007_{\pm 0.006}$	$0.4977_{\pm 0.006}$	$0.5840_{\pm 0.111}$	0.4808
	BWGNN	$0.5441_{\pm 0.020}$	$0.4026_{\pm 0.028}$	$0.4214_{\pm 0.039}$	$0.4908_{\pm 0.013}$	$\underline{0.9684}_{\pm 0.005}$	$0.5841_{\pm 0.062}$	$0.4196_{\pm 0.047}$	0.5473
	GHRN	$0.5242_{\pm 0.013}$	$0.4096_{\pm 0.021}$	$0.4783_{\pm 0.021}$	$0.5036_{\pm 0.016}$	$0.9601_{\pm 0.018}$	$\mathbf{0.6045}_{\pm 0.022}$	$0.4000_{\pm 0.098}$	0.5543
	XGBGraph	$0.4869_{\pm 0.069}$	$0.4869_{\pm 0.069}$	$0.7843_{\pm 0.090}$	$0.4773_{\pm 0.022}$	$\mathbf{0.9815}_{\pm 0.000}$	$\underline{0.5869}_{\pm 0.014}$	$0.4376_{\pm 0.044}$	$\underline{0.6059}$
	GraphPrompt	$0.3093_{\pm 0.000}$	$0.4511_{\pm 0.000}$	$0.2032_{\pm 0.000}$	$\mathbf{0.7093}_{\pm 0.000}$	$0.6331_{\pm 0.000}$	$0.4994_{\pm 0.000}$	$\mathbf{0.7128}_{\pm 0.000}$	0.5026
	Our	$\mathbf{0.7917}_{\pm 0.021}$	$0.5356_{\pm 0.005}$	$\mathbf{0.8192}_{\pm 0.015}$	$\underline{0.5362}_{\pm 0.007}$	$0.9289_{\pm 0.007}$	$0.5448_{\pm 0.009}$	$\underline{0.6959}_{\pm 0.042}$	**0.6932**
AUPRC	Unsupervised Methods								
	AnomalyDAE	$\underline{0.0675}_{\pm 0.028}$	$0.0413_{\pm 0.005}$	$\mathbf{0.6172}_{\pm 0.015}$	$0.0647_{\pm 0.016}$	$0.1025_{\pm 0.026}$	$0.1479_{\pm 0.006}$	$0.0583_{\pm 0.001}$	0.1571
	CoLA	$0.0468_{\pm 0.026}$	$0.0327_{\pm 0.002}$	$0.0956_{\pm 0.005}$	$0.0361_{\pm 0.001}$	$0.0678_{\pm 0.005}$	$0.1474_{\pm 0.001}$	$0.0717_{\pm 0.015}$	0.0712
	HCM-A	$0.0249_{\pm 0.003}$	$0.0374_{\pm 0.008}$	$0.0979_{\pm 0.011}$	$0.0511_{\pm 0.000}$	$0.0727_{\pm 0.006}$	$0.1453_{\pm 0.000}$	$0.0452_{\pm 0.020}$	0.0678
	TAM	$0.0243_{\pm 0.002}$	$\underline{0.0417}_{\pm 0.001}$	$0.1266_{\pm 0.015}$	$0.0532_{\pm 0.002}$	$0.1771_{\pm 0.002}$	$0.1271_{\pm 0.001}$	$0.0682_{\pm 0.002}$	0.0883
	GADAM	$0.0461_{\pm 0.014}$	$0.0299_{\pm 0.004}$	$0.0917_{\pm 0.007}$	$0.0428_{\pm 0.002}$	$0.0773_{\pm 0.024}$	$0.1602_{\pm 0.010}$	$0.0732_{\pm 0.004}$	0.0745
	Supervised Methods								
	GCN	$0.0437_{\pm 0.001}$	$\mathbf{0.0449}_{\pm 0.000}$	$0.2527_{\pm 0.026}$	$\underline{0.0763}_{\pm 0.001}$	$0.1738_{\pm 0.002}$	$0.1759_{\pm 0.000}$	$0.0484_{\pm 0.000}$	0.1165
	GAT	$0.0445_{\pm 0.039}$	$0.0327_{\pm 0.001}$	$0.0892_{\pm 0.016}$	$0.0595_{\pm 0.003}$	$0.0697_{\pm 0.001}$	$0.1478_{\pm 0.003}$	$0.0760_{\pm 0.035}$	0.0742
	BWGNN	$0.0289_{\pm 0.003}$	$0.0263_{\pm 0.002}$	$0.2735_{\pm 0.026}$	$0.0543_{\pm 0.004}$	$\underline{0.8406}_{\pm 0.012}$	$0.1975_{\pm 0.031}$	$0.0494_{\pm 0.004}$	0.2101
	GHRN	$0.0254_{\pm 0.001}$	$0.0265_{\pm 0.002}$	$0.3103_{\pm 0.013}$	$0.0541_{\pm 0.005}$	$0.8142_{\pm 0.045}$	$\mathbf{0.2015}_{\pm 0.015}$	$0.0561_{\pm 0.028}$	0.2126
	XGBGraph	$0.0268_{\pm 0.006}$	$0.0315_{\pm 0.000}$	$0.4116_{\pm 0.040}$	$0.0500_{\pm 0.003}$	$\mathbf{0.8673}_{\pm 0.000}$	$\underline{0.1994}_{\pm 0.012}$	$0.0541_{\pm 0.005}$	$\underline{0.2344}$
	GraphPrompt	$0.0169_{\pm 0.000}$	$0.0298_{\pm 0.000}$	$0.0812_{\pm 0.000}$	$\mathbf{0.0975}_{\pm 0.000}$	$0.1368_{\pm 0.000}$	$0.1481_{\pm 0.000}$	$\mathbf{0.1157}_{\pm 0.000}$	0.0894
	Our	$\mathbf{0.2291}_{\pm 0.023}$	$0.0340_{\pm 0.001}$	$\underline{0.4746}_{\pm 0.033}$	$0.0610_{\pm 0.003}$	$0.7329_{\pm 0.042}$	$0.1767_{\pm 0.004}$	$\underline{0.0933}_{\pm 0.013}$	**0.2574**

Table 3: AUROC and AUPRC results on seven real-world GAD datasets with the generalist model trained on Amazon. For each dataset and metric, the best performance per column is boldfaced, with the second-best underlined. "Avg" denotes the average performance of each method.

Datasets	5	10	15	20	25	30	35	40	45	50
Facebook	0.8859	0.9125	0.9218	0.9231	0.9210	0.9272	0.9340	0.9379	0.9369	0.9255
Reddit	0.5667	0.5754	0.5802	0.5824	0.5942	0.5871	0.5864	0.6067	0.5906	0.5828
Amazon	0.6264	0.6960	0.7152	0.7393	0.7394	0.7463	0.7462	0.7335	0.7392	0.7389

Table 4: AUROC results of UNPrompt with different thresholds (%) in unsupervised GAD.

the optimization and subsequently degrade the GAD performance. By contrast, increasing the threshold would lift the acceptance bar of normal nodes, allowing the optimization to focus on high-confident normal nodes and effectively mitigate the adverse effects caused by the wrongly labeled normal nodes.

6 Time Complexity Analysis of UNPrompt

Theoretical Analysis. In this section, we analyze the time complexity of training UNPrompt. As discussed in the main paper, UNPrompt first pre-trains the aggregation network with graph contrastive learning. Then, the model remains frozen when optimizing neighborhood graph prompts and the transformation layer to capture the generalized normal and abnormal graph patterns. In the experimental section, we employ a one-layer aggregation network, denoting the number of hidden units as d_h. The time complexity of the graph contrastive learning is $\mathcal{O}(4E_1(|A|d_h + Nd_hd^{'} + 6Nd_h^2))$, where $|A|$ returns of the number of edges of the $\mathcal{G}_{\text{train}}$, N is the number of nodes, $d^{'}$ represents the predefined dimensionality of node attributes, and E_1 is the number of training epoch. After that, we freeze the learned model and learn the learnable neighborhood prompt tokens and the transformation layer to capture the shared anomaly patterns. In our experiments, we set the size of each graph prompt to K and implement the classification head as a single-layer MLP with the same hidden units d_h. Given the number of the training epoch E_2, the time complexity of optimizing the graph prompt and the transformation layer is $\mathcal{O}((4KNd^{'} + 2Nd_h^2)E_2)$, which includes both the forward and backward propagation. Note that, despite the neighborhood aggregation model being frozen, the forward and backward propagations of the model are still needed to optimize the task-specific graph prompts and the transformation layer. Therefore, the overall time complexity of UNPrompt is $\mathcal{O}(4E_1(|A|d_h + Nd_hd^{'} + 6Nd_h^2) + 2E_2(|A|d_h + Nd_hd^{'} + 2KNd^{'} + Nd_h^2))$, which is linear to the number of nodes, the number of edges, and the number of node attributes of the training graph. Note that, after the training, the learned generalist model is directly utilized to perform anomaly detection on various target graphs without any further training.

Methods	AnomalyDAE	TAM	GAT	BWGNN	UNPrompt
Training Time	86.04	479.70	2.43	4.86	2.08
Inference Time	264.29	521.92	300.90	330.99	58.95

Table 5: Training time and inference time (seconds) for different methods.

Empirical Analysis. In Table 5, we report the training time and inference time of different methods, where two representative unsupervised methods (AnomalyDAE and TAM) and two supervised methods (GAT and BWGNN) are used for

Data set	Type	Nodes	Edges	Attributes	Anomalies(Rate)
Facebook	Social Networks	1,081	55,104	576	25(2.31%)
Reddit	Social Networks	10,984	168,016	64	366(3.33%)
Weibo	Social Networks	8,405	407,963	400	868(10.30%)
Amazon	Co-review	10,244	175,608	25	693(6.66%)
YelpChi	Co-review	24,741	49,315	32	1,217(4.91%)
Amazon-all	Co-review	11,944	4,398,392	25	821(6.87%)
YelpChi-all	Co-review	45,941	3,846,979	32	6,674(14.52%)
Disney	Co-purchase	124	335	28	6(4.84%)

Table 6: Key statistics of the real-world GAD datasets with real anomalies.

comparison to our method UNPrompt. The results show that the proposed method requires much less training and inference time compared to other baselines, demonstrating the efficiency of the proposed UNPrompt. Note that, TAM has the highest time consumption, which can be attributed to that it performs multiple graph truncation and learns multiple local affinity maximization networks.

7 Experimental Setup

7.1 Details on Datasets

We conduct the experiments using eight real-world with genuine anomalies in diverse online shopping services, social networks and co-purchase networks, including Facebook [Xu *et al.*, 2022], Reddit [Kumar *et al.*, 2019], Weibo [Zhao *et al.*, 2020], Amazon [McAuley and Leskovec, 2013], YelpChi [Rayana and Akoglu, 2015] as well as two large-scale graph datasets including Amazon-all [McAuley and Leskovec, 2013] and YelpChi-all [Rayana and Akoglu, 2015] and Disney [Sánchez *et al.*, 2013]. The statistical information including the number of nodes, edge, the dimension of the feature, and the anomaly rate of the datasets can be found in Table 6.

7.2 More Implementation Details

Generalist GAD. For the graph contrastive learning-based pre-training, the probabilities of edge removal and attribute masking are by default set to 0.2 and 0.3 respectively. Besides, the learning rate is set to 0.001 with the Adam optimizer, the training epoch is set to 200 and the temperature τ is 0.5.

For the neighborhood prompt learning, the learning rate is also set to 0.001 with the Adam optimizer, and the training epoch is set to 900. Note that, since we focus on generalist GAD, we do not perform any hyperparameter search for specific target graphs. Instead, the results of all target graphs are obtained with the same hyperparameter settings.

Unsupervised GAD. Similar to the generalist GAD setting, the hidden units of the neighborhood aggregation network and the transformation layer are set to 128 for all graphs. The threshold t is determined by the 40th percentile of the normal scores obtained by the pre-trained model g, and the scaling parameter α is set to 10 for all graphs. Besides, we utilize random search to find the optimal hyperparameters of the size of neighborhood prompts K and the trade-off parameter λ.

For both generalist and unsupervised GAD, the code is implemented with Pytorch (version: 1.13.1), DGL (version: 1.0.1), OGB (version: 1.3.6), and Python 3.8.19. All experiments are conducted on a Linux server with an Intel CPU (Intel Xeon Gold 6346 3.1GHz) and a Nvidia A40 GPU.

8 More Discussions on UNPrompt

8.1 Difference between Inductive learning and Generalist GAD

Similar to the generalist setting, inductive graph learning [Hamilton *et al.*, 2017; Ding *et al.*, 2021; Li *et al.*, 2023; Huang *et al.*, 2023; Fang *et al.*, 2023] also focuses on inference on unseen graph data. However, these methods are not applicable to the generalist setting. Specifically, inductive graph learning trains the model on partial data of the whole graph dataset [Hamilton *et al.*, 2017; Ding *et al.*, 2019; Li *et al.*, 2023] or the previously observed data of dynamic graphs [Fang *et al.*, 2023]. Then, the learned model is evaluated on the unseen data of the whole dataset or the future graph. These unseen testing data are from the same source of the training data with the same dimensionality and semantics. In contrast, the unseen data in our method are from different distributions/domains with significantly different dimensionality and semantics. This cross-dataset nature, specifically referred to as a zero-shot problem [Jeong *et al.*, 2023; Zhou *et al.*, 2024], makes our setting significantly different from the current inductive graph learning setting.

8.2 Difference between ARC and UNPrompt

While both ARC [Liu *et al.*, 2024] and our UNPrompt focus on generalist graph anomaly detection, there exist significant differences between them. First, ARC addresses a few-shot setting, whereas UNPrompt addresses a zero-shot setting. Second, ARC performs anomaly scoring using a traditional anomaly measure, reconstruction error. It calculates the reconstruction errors of query node embeddings w.r.t. in-context embeddings of labeled nodes. Differently, UNPrompt proposes a novel generalized anomaly measure based on latent attribute predictability. Third, the training datasets for ARC consist of the largest graphs from all multiple graph types/domains. The test graphs from the same types/domains are used during inference. By contrast, UNPrompt requires only one single dataset to effectively learn the generalist model, and the testing datasets are from unseen datasets and domains.

8.3 Future works

Despite the proposed method achieving promising performance for generalist GAD, there are several interesting future works that can be explored to further enhance the contribution of UNPrompt. First, like most existing methods, UNPrompt cannot process very large graphs with millions of nodes, as it requires loading the full graph during inference. A subgraph-based inference may be a solution, and we will investigate this problem in future work. Second, we assume that normal nodes can be well predicted by their neighbors, and thus we focus on their one-hop neighbors. The experiments verify the effectiveness of this assumption. However, it is still worth exploring the utilization of multi-hop neighbors. Third, it would be interesting and practical to extend

the proposed method to dynamic graph anomaly detection [Jiang *et al.*, 2024; Chen *et al.*, 2024; Jiang *et al.*, 2022; Liu *et al.*, 2021b].

References

[Chen *et al.*, 2024] Yuhang Chen, Jiaxin Jiang, Shixuan Sun, Bingsheng He, and Min Chen. Rush: Real-time burst subgraph detection in dynamic graphs. *Proceedings of the VLDB Endowment*, 17(11):3657–3665, 2024.

[Ding *et al.*, 2019] Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. Deep anomaly detection on attributed networks. In *Proceedings of the 2019 SIAM international conference on data mining*, pages 594–602. SIAM, 2019.

[Ding *et al.*, 2021] Kaize Ding, Jundong Li, Nitin Agarwal, and Huan Liu. Inductive anomaly detection on attributed networks. In *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence*, pages 1288–1294, 2021.

[Fang *et al.*, 2023] Lanting Fang, Kaiyu Feng, Jie Gui, Shanshan Feng, and Aiqun Hu. Anonymous edge representation for inductive anomaly detection in dynamic bipartite graph. *Proceedings of the VLDB Endowment*, 16(5):1154–1167, 2023.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[Huang *et al.*, 2022] Tianjin Huang, Yulong Pei, Vlado Menkovski, and Mykola Pechenizkiy. Hop-count based self-supervised anomaly detection on attributed networks. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 225–241. Springer, 2022.

[Huang *et al.*, 2023] Yihong Huang, Liping Wang, Fan Zhang, and Xuemin Lin. Unsupervised graph outlier detection: Problem revisit, new insight, and superior method. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 2565–2578. IEEE, 2023.

[Jeong *et al.*, 2023] Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. Winclip: Zero-/few-shot anomaly classification and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19606–19616, 2023.

[Jiang *et al.*, 2022] Jiaxin Jiang, Yuan Li, Bingsheng He, Bryan Hooi, Jia Chen, and Johan Kok Zhi Kang. Spade: A real-time fraud detection framework on evolving graphs. *Proceedings of the VLDB Endowment*, 16(3):461–469, 2022.

[Jiang *et al.*, 2024] Jiaxin Jiang, Yuhang Chen, Bingsheng He, Min Chen, and Jia Chen. Spade+: A generic real-time fraud detection framework on dynamic graphs. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[Kumar *et al.*, 2019] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278, 2019.

[Li *et al.*, 2023] Xujia Li, Yuan Li, Xueying Mo, Hebing Xiao, Yanyan Shen, and Lei Chen. Diga: guided diffusion model for graph recovery in anti-money laundering. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4404–4413, 2023.

[Liu *et al.*, 2012] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012.

[Liu *et al.*, 2021a] Yixin Liu, Zhao Li, Shirui Pan, Chen Gong, Chuan Zhou, and George Karypis. Anomaly detection on attributed networks via contrastive self-supervised learning. *IEEE transactions on neural networks and learning systems*, 33(6):2378–2392, 2021.

[Liu *et al.*, 2021b] Yixin Liu, Shirui Pan, Yu Guang Wang, Fei Xiong, Liang Wang, Qingfeng Chen, and Vincent CS Lee. Anomaly detection in dynamic graphs via transformer. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12081–12094, 2021.

[Liu *et al.*, 2024] Yixin Liu, Shiyuan Li, Yu Zheng, Qingfeng Chen, Chengqi Zhang, and Shirui Pan. Arc: A generalist graph anomaly detector with in-context learning. *arXiv preprint arXiv:2405.16771*, 2024.

[Luo *et al.*, 2022] Xuexiong Luo, Jia Wu, Amin Beheshti, Jian Yang, Xiankun Zhang, Yuan Wang, and Shan Xue. Comga: Community-aware attributed graph anomaly detection. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 657–665, 2022.

[McAuley and Leskovec, 2013] Julian John McAuley and Jure Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*, pages 897–908, 2013.

[Peng *et al.*, 2018] Zhen Peng, Minnan Luo, Jundong Li, Huan Liu, Qinghua Zheng, et al. Anomalous: A joint modeling approach for anomaly detection on attributed networks. In *IJCAI*, volume 18, pages 3513–3519, 2018.

[Qiao and Pang, 2023] Hezhe Qiao and Guansong Pang. Truncated affinity maximization: One-class homophily modeling for graph anomaly detection. *Advances in Neural Information Processing Systems*, 36, 2023.

[Rayana and Akoglu, 2015] Shebuti Rayana and Leman Akoglu. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, pages 985–994, 2015.

[Sánchez *et al.*, 2013] Patricia Iglesias Sánchez, Emmanuel Müller, Fabian Laforet, Fabian Keller, and Klemens

Böhm. Statistical selection of congruent subspaces for mining attributed graphs. In *2013 IEEE 13th international conference on data mining*, pages 647–656. IEEE, 2013.

[Xu *et al.*, 2022] Zhiming Xu, Xiao Huang, Yue Zhao, Yushun Dong, and Jundong Li. Contrastive attributed network anomaly detection with data augmentation. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 444–457. Springer, 2022.

[Zhao *et al.*, 2020] Tong Zhao, Chuchen Deng, Kaifeng Yu, Tianwen Jiang, Daheng Wang, and Meng Jiang. Error-bounded graph anomaly loss for gnns. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1873–1882, 2020.

[Zheng *et al.*, 2021] Yu Zheng, Ming Jin, Yixin Liu, Lianhua Chi, Khoa T Phan, and Yi-Ping Phoebe Chen. Generative and contrastive self-supervised learning for graph anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12220–12233, 2021.

[Zhou *et al.*, 2024] Qihang Zhou, Guansong Pang, Yu Tian, Shibo He, and Jiming Chen. AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection. In *The Twelfth International Conference on Learning Representations*, 2024.

[Zhu *et al.*, 2020] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.