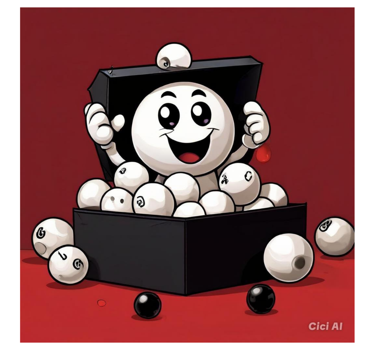
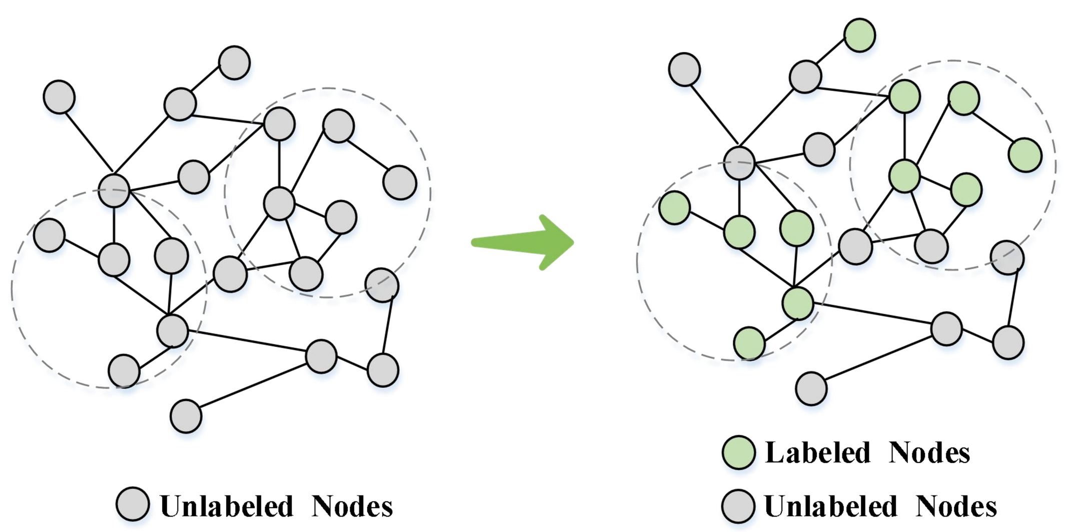
Generative Semi-supervised Graph Anomaly Detection

Hezhe Qiao, Qingsong Wen, Xiaoli Li, Ee-Peng Lim, Guansong Pang

Motivation

This work considers a practical **semi-supervised graph anomaly detection (GAD) scenario**, where part of the nodes in a graph are known to be normal, contrasting to the extensively explored unsupervised setting with a fully unlabeled graph.



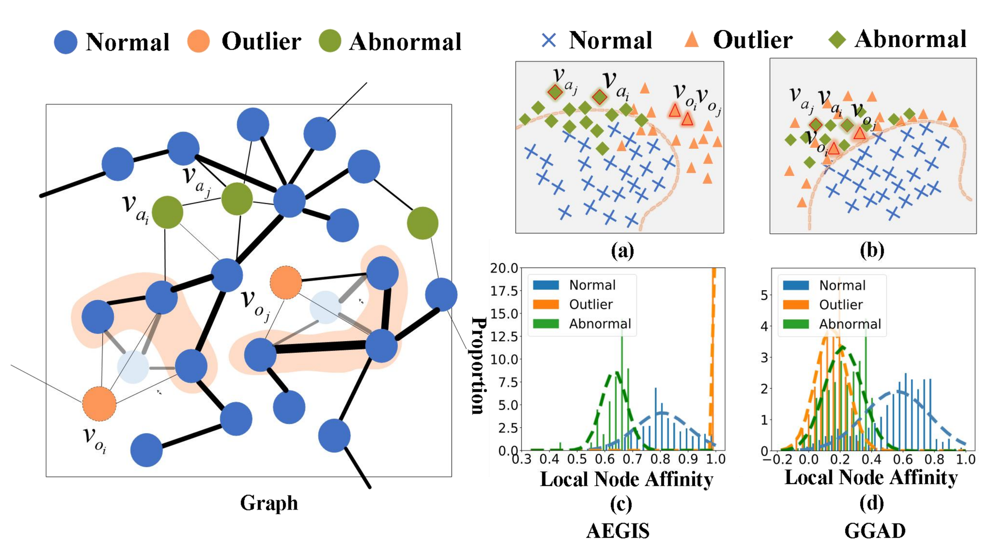


Unsupervised

Partial Normal Labeled Supervised

Randomly sample some nodes from the graph as the normal nodes, without any human manual labeling (the same labeling cost as unsupervised GAD).

Two Important Priors about Anomalies



Asymmetric local affinity

The affinity between normal nodes is typically significantly stronger than that between normal and abnormal nodes.

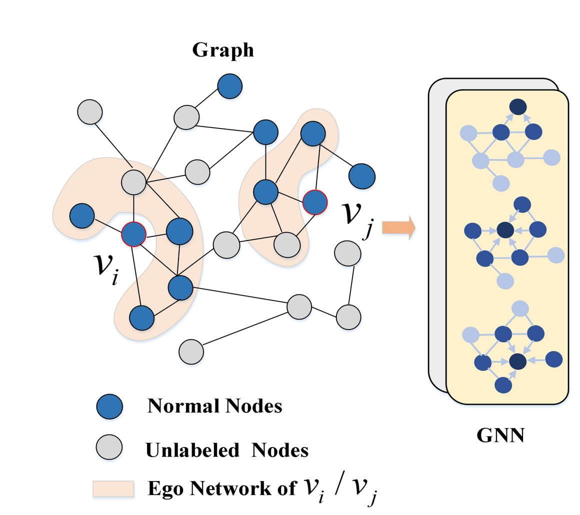
Egocentric closeness

The outliers should be closed to the normal nodes that share similar local structure due to subtle abnormality or adversarial camouflage.

Key idea

The key idea is to generate pseudo anomaly nodes, referred to as 'outlier nodes', for providing effective negative node samples in training a discriminative one-class classifier.

Graph Neural Network for Node Representation



$$\mathbf{H}^{(\ell)} = GNN(\mathbf{A}, \mathbf{H}^{(\ell-1)}; \mathbf{W}^{(\ell-1)}) \qquad \mathbf{H}^{(\ell)} \in \mathbb{R}^{N \times h^{(\ell)}}, \mathbf{H}^{(\ell-1)} \in \mathbb{R}^{N \times h^{(\ell-1)}}$$

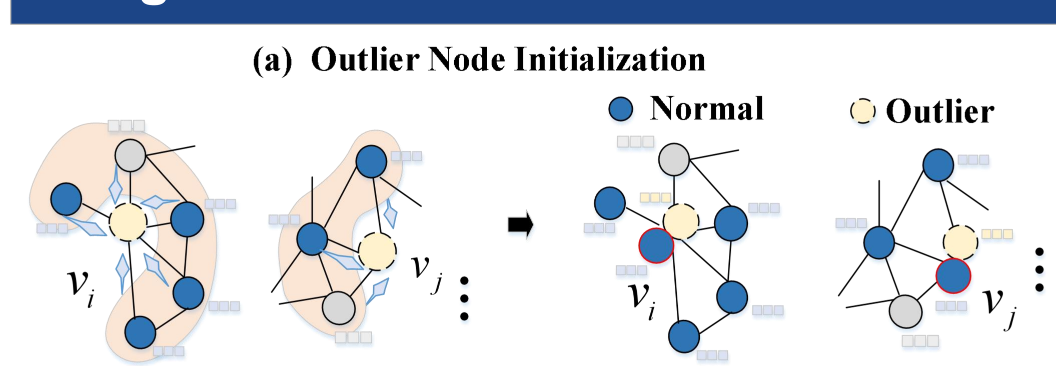
$\mathbf{W}^{(\ell)}$ learnable parameters $\quad \mathbf{H}^{(0)} = \mathbf{X}$

$\mathbf{H}_i^{(\ell)}$ are the embeddings of node v_i

Employ a 2-layer GCN due to its high efficiency

$$\mathbf{H}^{(\ell)} = \phi\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(\ell-1)}\mathbf{W}^{(\ell-1)}\right)$$

Neighborhood-aware outlier initialization

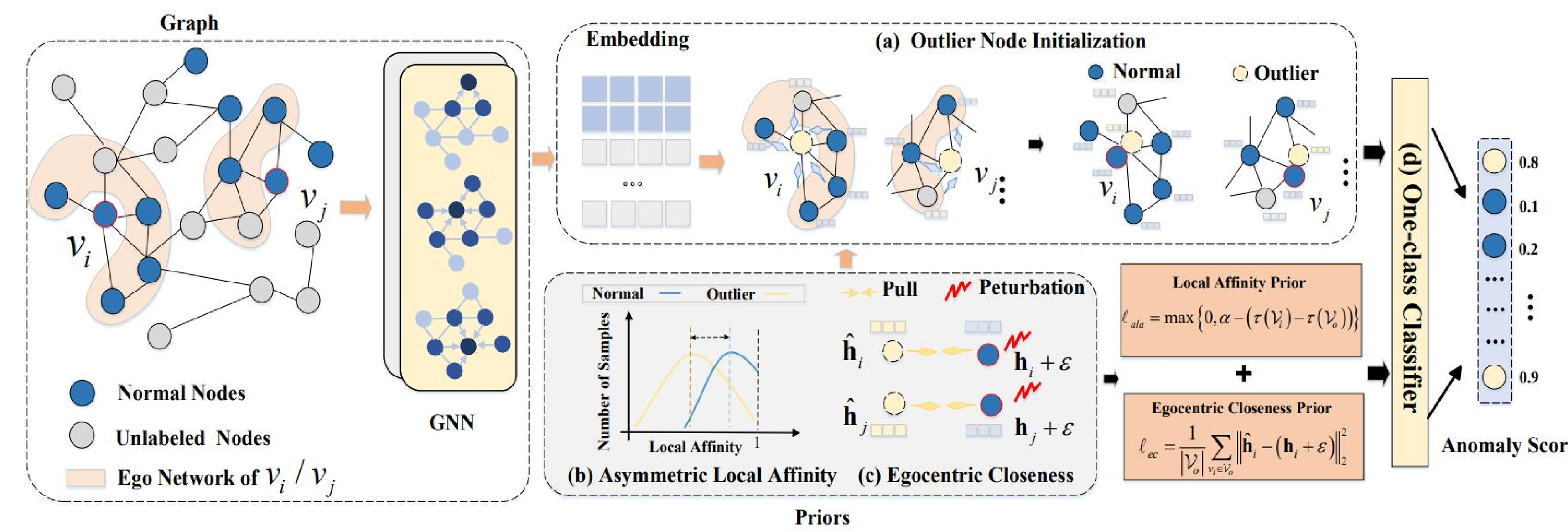


We sample a set of normal nodes from normal set and respectively generate an outlier node for each of them based on its ego network.

$$\hat{\mathbf{h}}_i = \Psi(v_i, \mathcal{N}(v_i); \Theta_g) = \frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j \in \mathcal{N}(v_i)} \sigma(\widetilde{\mathbf{W}}\mathbf{h}_j)$$

Ψ is a mapping function determined by parameters Θ_g and $\widetilde{\mathbf{W}} \in \mathbb{R}^{d \times d}$ are the learnable parameter

The Framework of GGAD



Incorporating the Asymmetric Local Affinity Prior

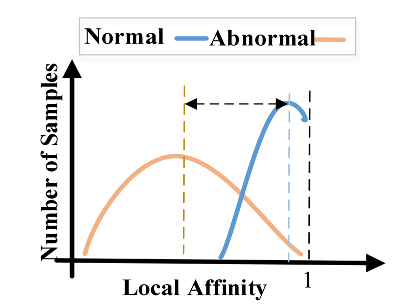
- **Local Node Affinity Calculation**

$$\tau(v_i) = \frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j \in \mathcal{N}(v_i)} \text{sim}(\mathbf{h}_i, \mathbf{h}_j) \qquad \tau(\mathcal{V}_l) = \frac{1}{|\mathcal{V}_l|}\sum_{v_i \in \mathcal{V}_l} \tau(v_i) \qquad \tau(\mathcal{V}_o) = \frac{1}{|\mathcal{V}_o|}\sum_{v_i \in \mathcal{V}_o} \tau(v_i)$$

$\mathcal{V}_o$ and $\mathcal{V}_l$ are the sets of abnormal nodes and normal nodes

- **Enforcing the Structural Affinity Prior**

$$\ell_{ala} = \max\{0, \alpha - (\tau(\mathcal{V}_l) - \tau(\mathcal{V}_o))\}$$ Margin loss function



Asymmetric Local Affinity

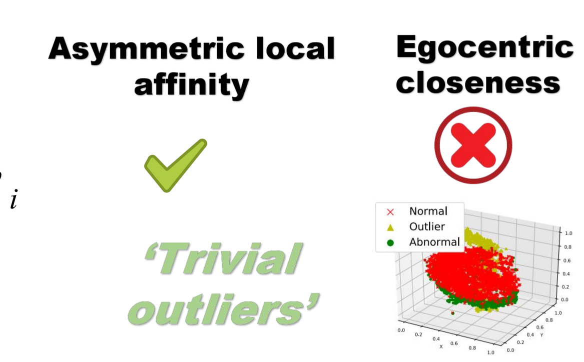
Incorporating the Egocentric Closeness Prior

Solely using this local affinity prior may distribute far away from the normal nodes in the representation space.

$$\ell_{ec} = \frac{1}{|\mathcal{V}_o|}\sum_{v_i \in \mathcal{V}_o} \left\| \hat{\mathbf{h}}_i - (\mathbf{h}_i + \varepsilon) \right\|_2^2$$

$\mathbf{h}_i$ and $\hat{\mathbf{h}}_i$ are the representations of the normal node v_i and its corresponding generated outliers

ε is a noise perturbation generated from a Gaussian distribution to guarantee separability



Training and Inference

- **Binary cross-entropy loss function**

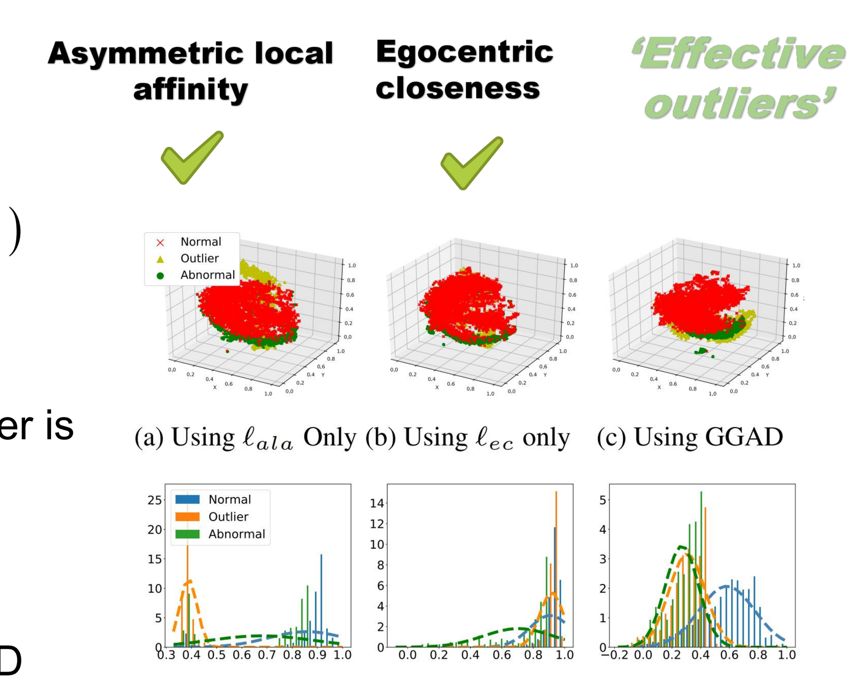
$$\ell_{bce} = \sum_{i}^{|\mathcal{V}_o|+|\mathcal{V}_l|} y_i \log(p_i) + (1 - y_i)\log(1 - p_i)$$

$$\ell_{total} = \ell_{bce} + \beta\ell_{ala} + \lambda\ell_{ec}$$

The inverse prediction of the one-class classifier is used as the anomaly score

$$\text{score}(v_j) = 1 - \eta(\mathbf{h}_j; \Theta^*)$$

where Θ^* is the learned parameters of GGAD



(a) Using ℓ_{ala} Only (b) Using ℓ_{ec} only (c) Using GGAD

(d) Using ℓ_{ala} Only (e) Using ℓ_{ec} Only (f) Using GGAD

Main Comparison Result

Setting	Method	AUROC Amazon	AUROC T-Finance	AUROC Reddit	AUROC Elliptic	AUROC Photo	AUROC DGraph	AUPRC Amazon	AUPRC T-Finance	AUPRC Reddit	AUPRC Elliptic	AUPRC Photo	AUPRC DGraph
Unsupervised	DOMINANT	0.7025	0.6087	0.5105	0.2960	0.5136	0.5738	0.1315	0.0536	0.0380	0.0454	0.1039	0.0075
	AnomalyDAE	0.7783	0.5809	0.5091	0.4963	0.5069	0.5763	0.1429	0.0491	0.0319	0.0872	0.0987	0.0070
	OCGNN	0.7165	0.4732	0.5246	0.2581	0.5307	/	0.1352	0.0392	0.0375	0.0616	0.0965	/
	AEGIS	0.6059	0.6496	0.5349	0.4553	0.5516	0.4509	0.1200	0.0622	0.0413	0.0827	0.0972	0.0053
	GAAN	0.6513	0.3091	0.5216	0.2590	0.4296	/	0.0852	0.0283	0.0348	0.0436	0.0767	/
	TAM	0.8303	0.6175	0.6062	0.4039	0.5675	/	0.4024	0.0547	0.0437	0.0502	0.1013	/
Semi-supervised	DOMINANT	0.8867	0.6167	0.5194	0.3256	0.5314	0.5851	0.7289	0.0542	0.0414	0.0652	0.1283	0.0076
	AnomalyDAE	0.9171	0.6027	0.5280	0.5409	0.5272	0.5866	0.7748	0.0538	0.0362	0.0949	0.1177	0.0071
	OCGNN	0.8810	0.5742	0.5622	0.2881	0.6461	/	0.7538	0.0492	0.0400	0.0640	**0.1501**	/
	AEGIS	0.7593	0.6728	0.5605	0.5132	0.5936	0.4450	0.2616	0.0685	0.0441	0.0912	0.1110	0.0058
	GAAN	0.6531	0.3636	0.5349	0.2724	0.4355	/	0.0856	0.0324	0.0362	0.0611	0.0768	/
	TAM	0.8405	0.5923	0.5829	0.4150	0.6013	/	0.5183	0.0551	0.0446	0.0552	0.1087	/
	GGAD (Ours)	**0.9443**	**0.8228**	**0.6354**	**0.7290**	**0.6476**	**0.5943**	**0.7922**	**0.1825**	**0.0610**	**0.2425**	0.1442	**0.0082**

Ablation Study

- By unifying both priors through the two losses, the generated outlier nodes can be thought as hard anomalies that lie at the fringe of normal nodes in the feature representation space

Ablation study on our two priors

Metric	ℓ_{ala}	ℓ_{ec}	Amazon	T-Finance	Reddit	Elliptic	Photo	DGraph
AUROC		✓	0.8871	0.8149	0.5839	0.6863	0.5762	0.5891
	✓		0.7250	0.6994	0.5230	0.7001	0.6103	0.5513
	✓	✓	**0.9324**	**0.8228**	**0.6354**	**0.7290**	**0.6476**	**0.5943**
AUPRC		✓	0.6643	0.1739	0.0409	0.1954	0.1137	0.0076
	✓		0.1783	0.0800	0.0398	**0.2683**	0.1186	0.0063
	✓	✓	**0.7843**	**0.1924**	**0.0610**	0.2425	**0.1442**	**0.0087**

- ❖ Random
- ❖ Nonlearnable Outliers (NLO)
- ❖ Gaussian Perturbation
- ❖ Noise and GaussianP
- ❖ VAE and GAN

GGAD vs. alternative outlier generators

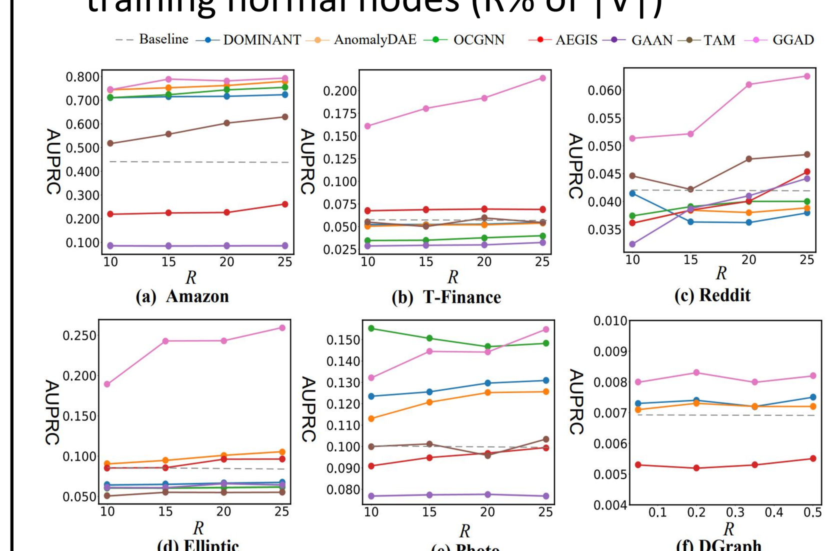
Metric	Method	Amazon	T-Finance	Reddit	Elliptic	Photo	DGraph
AUROC	Random	0.7263	0.4613	0.5227	0.6856	0.5678	0.5712
	NLO	0.8613	0.6179	0.5638	0.6787	0.5307	0.5538
	Noise	0.8508	0.8204	0.5285	0.6786	0.5940	0.5779
	GaussianP	0.2279	0.6659	0.5235	0.6715	0.5925	0.5862
	VAE	0.8984	0.6674	0.6175	0.7055	0.6222	0.5801
	GAN	0.8288	0.5487	0.5378	0.6256	0.6032	0.5101
	GGAD (Ours)	**0.9324**	**0.8228**	**0.6354**	**0.7290**	**0.6476**	**0.5943**
AUPRC	Random	0.1755	0.0402	0.0394	0.1981	0.1063	0.0061
	NLO	0.4696	0.1364	0.0495	0.1750	0.1092	0.0065
	Noise	0.5384	0.1762	0.0381	0.1924	0.1200	0.0076
	GaussianP	0.0397	0.0677	0.0376	0.1682	0.1194	0.0078
	VAE	0.6111	0.0652	0.0528	0.2344	0.1272	0.0063
	GAN	0.3715	0.0461	0.0433	0.1263	0.1143	0.0051
	GGAD (Ours)	**0.7843**	**0.1924**	**0.0610**	**0.2425**	**0.1442**	**0.0087**

Performance w.r.t. Training Size and Anomaly Contamination

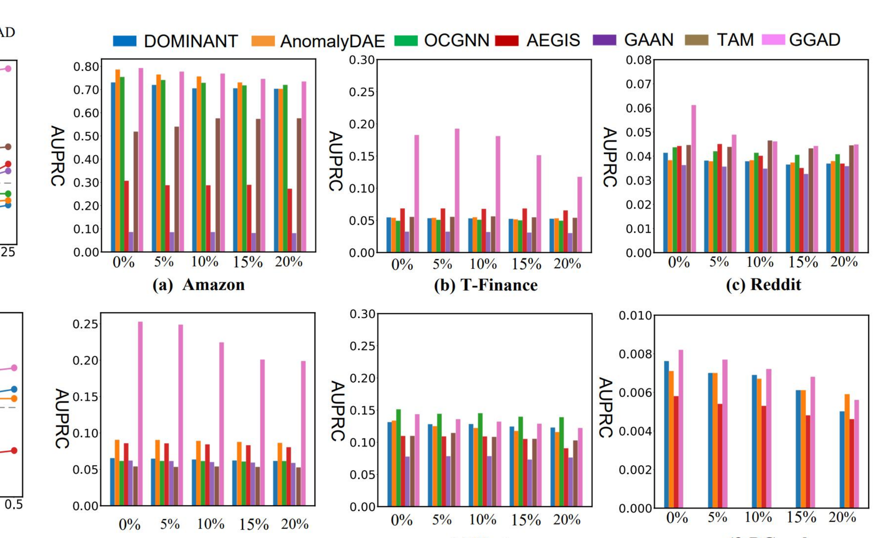
❑ With increasing training samples of normal nodes, the performance of all methods generally gets improved

❑ GGAD consistently maintains the best performance under different contamination rates, showing good robustness w.r.t. the contamination

AUPRC results w.r.t. different number of training normal nodes (R% of |V|)



AUPRC w.r.t. different anomaly contamination



GGAD enabled Unsupervised Methods

- We incorporate the outlier generation into existing unsupervised method demonstrating the generation in GGAD can also benefit the existing unsupervised methods
- The distribution of the generated outliers have much smaller MMD distance

GGAD enabled unsupervised methods

Metric	Method	Amazon	T-Finance	Elliptic
#Anomalies/#Top-K Nodes		387/500	351/1000	1448/2000
AUROC	DOMINATE	0.7025	0.6087	0.2960
	GGAD-enabled DOMINATE	0.8186	0.6275	0.2986
	OCGNN	0.7165	0.4732	0.2581
	GGAD-enabled OCGNN	0.8692	0.5931	0.2638
	AEGIS	0.6059	0.6496	0.4553
	GGAD-enabled AEGIS	0.8395	0.7024	0.5036
	GGAD	0.9431	0.8108	0.7225
AUPRC	DOMINATE	0.1315	0.0536	0.0454
	GGAD-enabled DOMINATE	0.3462	0.0585	0.0613
	OCGNN	0.1352	0.0392	0.0616
	GGAD-enabled OCGNN	0.3950	0.0480	0.0607
	AEGIS	0.1200	0.0622	0.0827
	GGAD-enabled AEGIS	0.3833	0.0784	0.0910
	GGAD	0.7769	0.1734	0.2484

Analysis of the generated outliers using MMD distance

Metric	Amazon	T-Finance	Elliptic	Photo	Reddit
with Abnormal Node	**0.1980**	**0.0784**	**0.1094**	**0.3703**	**0.3409**
with Normal Node	0.2318	0.1040	0.1304	0.3880	0.3605

Computational Efficiency Analysis

- GGAD is generally more efficient than existing unsupervised GAD methods

Runtimes (in seconds) on the six datasets on CPU

Method	Amazon	T-Finance	Reddit	Elliptic	Photo	DGraph
DOMINANT	1592	10721	125	1119	437	388
AnomalyDAE	1656	18560	161	8296	445	457
OCGNN	765	5717	162	3517	125	/
AEGIS	1121	15258	166	5638	417	1022
GAAN	1678	12120	94	1866	307	/
TAM	4516	17360	432	13200	165	/
GGAD (Ours)	658	9345	368	5146	106	488