NEURAL INFORMATION PROCESSING SYSTEMS

# Generative Semi-supervised Graph Anomaly Detection

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

NeurIPS2024@Vancouver, Canada
12 Decemeber, 2024

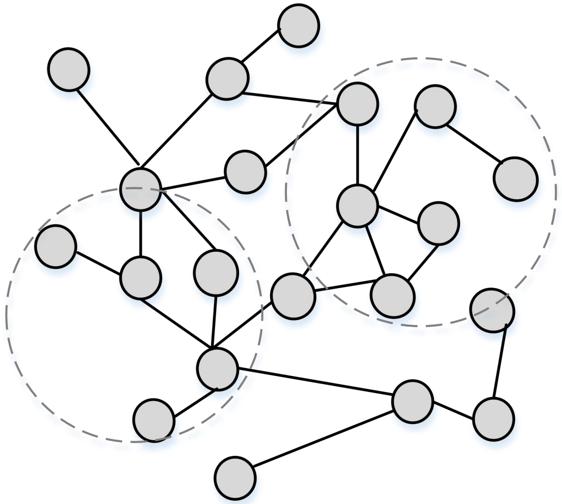
SMU
SINGAPORE MANAGEMENT UNIVERSITY

Squirrel Ai Learning

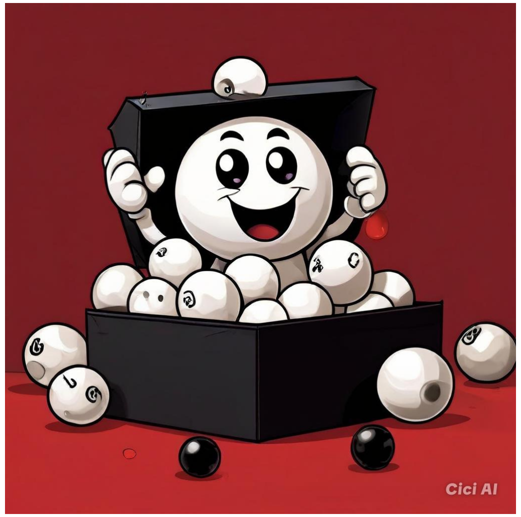
Agency for Science, Technology and Research
SINGAPORE

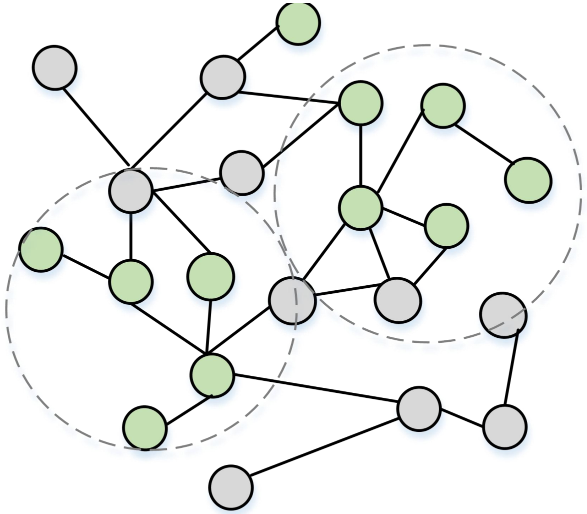
# Motivation

**Overwhelming normal samples**



Unlabeled Nodes



Labeled Nodes

Unlabeled Nodes

***Labels of normal samples is relatively easy to obtain***

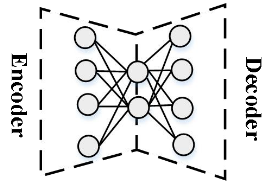
**Unsupervised** → **Partial Normal Labeled Supervised**

# Existing Unsupervised GAD Methods

- DOMINATE
- AnomalyDAE

## Reconstruction

$$\mathcal{L} = (1-\alpha)\| A - \hat{A} \|_F^2 + \alpha \| X - \hat{X} \|_F^2$$

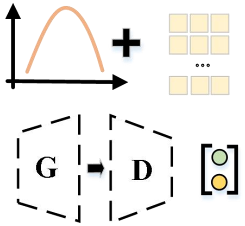



Reconstruction

- GAAD
- AEGIS

## Generative Adversarial Network

$$\mathcal{L} = \mathcal{L}_{AE} + \mathcal{L}_{GAN}$$

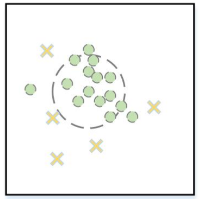



Generative Adversarial Network

- OCGNN

## One Class SVM

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} \left\| \mathbf{h}_i - \mathbf{c} \right\|_2^2 + \frac{\lambda}{2} \| \Theta \|_F^2$$

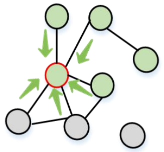


One Class SVM

- TAM

## Affinity Maxmization

$$-\frac{1}{\left|\mathcal{N}(v_i)\right|} \sum_{v_j \in \mathcal{N}(v_i)} \mathrm{sim}\left(\mathbf{h}_i, \mathbf{h}_j\right)$$



Affinity Maxmization

# Existing Unsupervised GAD Methods

## Disadvantages

- Fail to analyze the problem from the partially labeled normal samples
- Fail to fully take advantage of the  two important priors about anomaly nodes – asymmetric local affinity and egocentric closeness

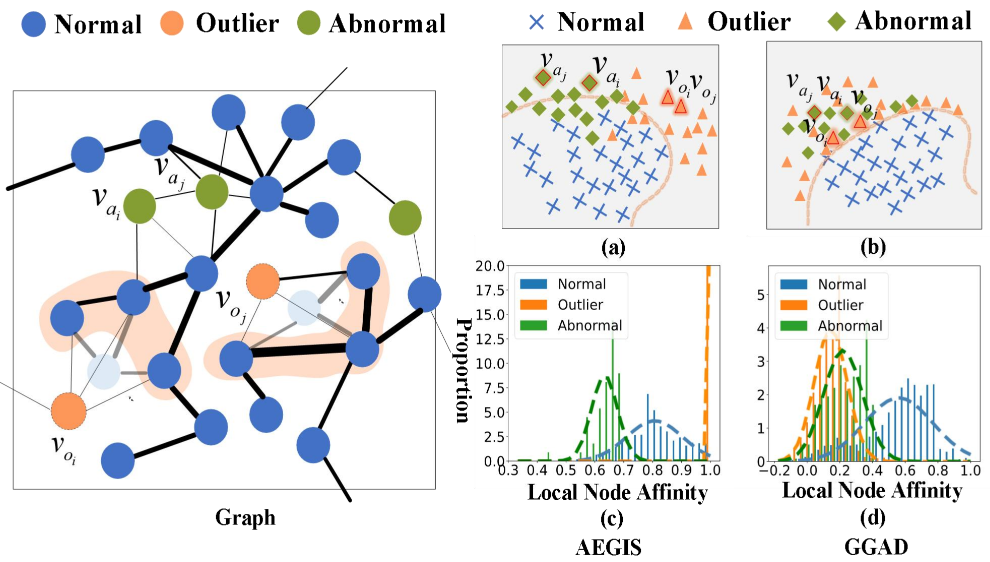
# Two Important Priors about Anomalies

## ☐ Asymmetric local affinity

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

## ☐ Egocentric closeness

The representation of the outlier nodes should be closed to the normal nodes that share similar local structure as the outlier nodes




**Left:** An exemplar graph with the edge width indicates the level of affinity connecting two nodes. **Right:** GGAD aims to generate outliers (*e.g.*, $v_{o_i}$ and $v_{o_j}$) that can well assimilate the anomaly nodes.

# Insight

- Construct a new experimental setting, semi-supervised GAD (training on exclusively normal nodes) and establish a new benchmark by adapting existing unsupervised anomaly detection methods to this setting.

- An outlier node generation based on the two important priors is proposed to enable the semi-supervised graph anomaly detection.
  - These generated outlier nodes and the given normal nodes can then be used to build a binary classifier for the GAD task.

**Our success will rely on how much the outlier nodes are analogous to the real anomalies**

# Notation and Problem Statement

## Notation

An attributed graph can be denoted by $\mathcal{G}=(\mathcal{V},\mathcal{E},\mathbf{X})$， where $\mathcal{V}=\{v_1,\cdots,v_N\}$ denotes the node set，$\mathcal{E}\subseteq\mathcal{V}\times\mathcal{V}$ denotes the edge set. $\mathbf{X}\in\mathbb{R}^{N\times F}$ and $\mathbf{A}\in\{0,1\}^{N\times N}$ are node attribute and adjacency matrix.

## Problem Statement

The goal of semi-supervised GAD is to learn an anomaly scoring function $f:\mathcal{G}\rightarrow\mathbb{R}$ such that $f(v)<f\left(v'\right),\forall v\in\mathcal{V}_n, v'\in\mathcal{V}_a$ given a set of labeled normal nodes $\mathcal{V}_l\subset\mathcal{V}_n$ and no access to labels of anomaly nodes.

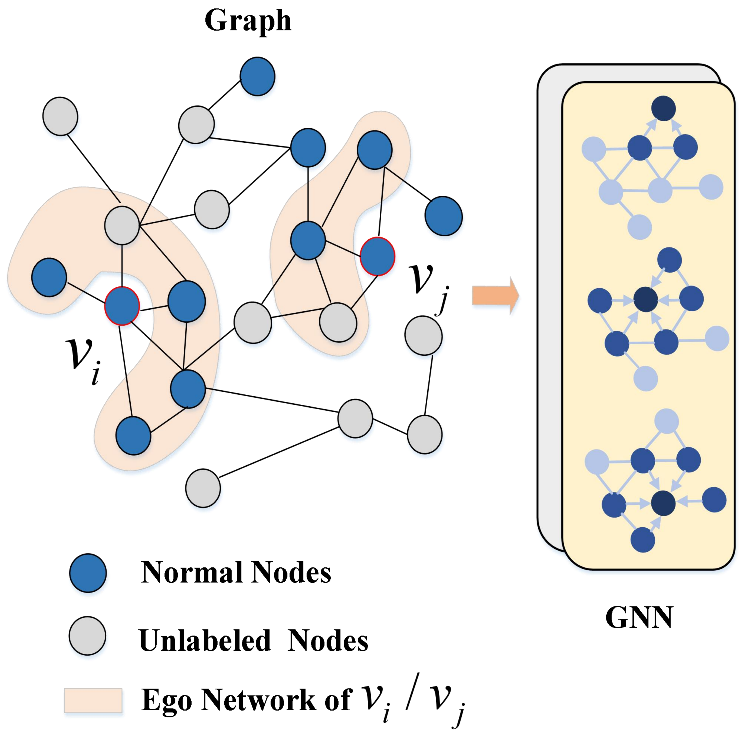
All other unlabeled nodes, denoted by $\mathcal{V}_u=\mathcal{V}\setminus\mathcal{V}_l$ , comprise the test data set.

- Evaluation Metric AUROC, AUPRC

# Methodology – GNN for Node Representation Learning

- Obtain the embedding of nodes

$$\mathbf{H}_i^{(l)} = GNN(\mathbf{A}, \mathbf{H}_i^{(l-1)}; \mathbf{W}^{(l-1)})$$




$$\mathbf{H}^{(\ell)} \in \mathbb{R}^{N \times h^{(l)}}, \mathbf{H}^{(\ell-1)} \in \mathbb{R}^{N \times h^{(l-1)}}$$

$\mathbf{H}$ are the embeddings of nodes

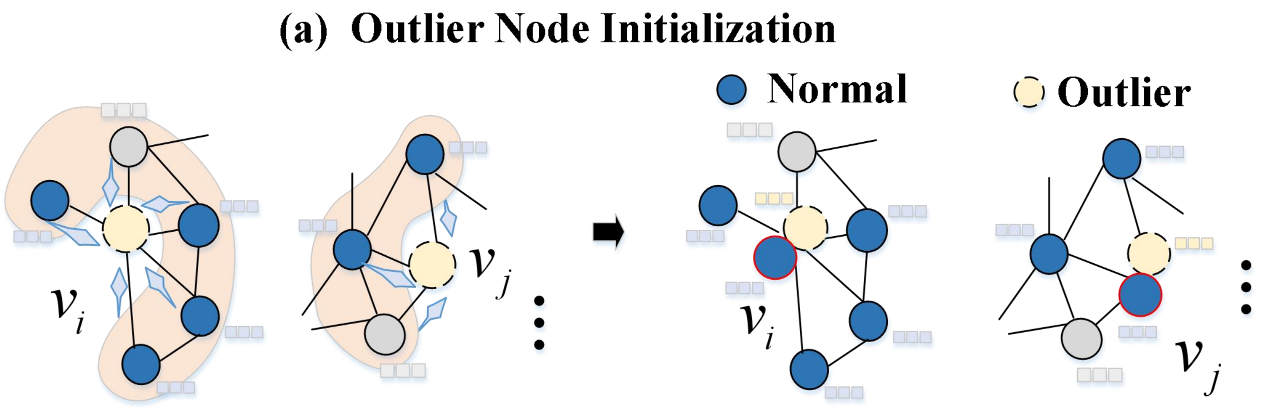
$\mathbf{W}^{(\ell)}$ are learnable parameters $\mathbf{H}^{(0)} = X$

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

Employ a GCN due to its high efficiency

# Methodology – Outlier node generation

- Neighborhood-aware outlier initialization




We sample a set of normal nodes from $\mathcal{V}_l$ and respectively generate an outlier node for each of them based on its ego network.

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

$\Psi$ is a mapping function determined by parameters $\Theta_g$ that contain the learnable parameter $\widetilde{\mathbf{W}} \in \mathbb{R}^{d \times d}$

# Methodology – Incorporating the Asymmetric Local Affinity Prior

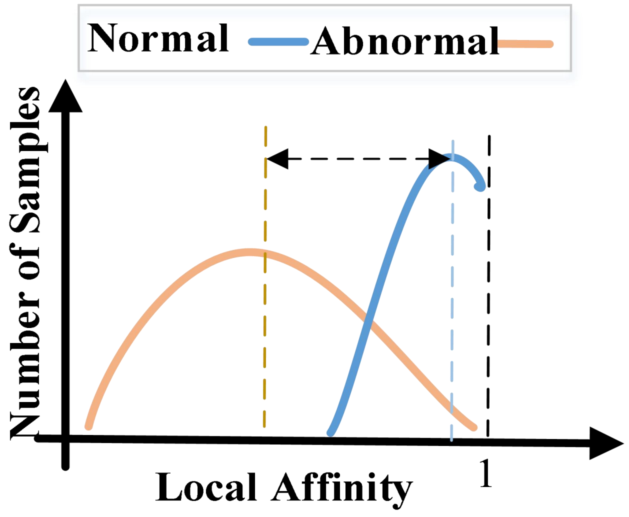
- **Local Node Affinity Calculation**

$$\tau(v_i) = \frac{1}{|\mathcal{N}(v_i)|} \sum_{v_j \in \mathcal{N}(v_i)} \mathrm{sim}(\mathbf{h}_i, \mathbf{h}_j)$$

- **Enforcing the Structural Affinity Prior**

$$\ell_{\text{ala}} = \max\left\{0, \alpha - \left(\tau(\mathcal{V}_l) - \tau(\mathcal{V}_o)\right)\right\}$$

$$\tau(\mathcal{V}_o) = \frac{1}{|\mathcal{V}_o|} \sum_{v_i \in \mathcal{V}_o} \tau(v_i) \qquad \tau(\mathcal{V}_l) = \frac{1}{|\mathcal{V}_l|} \sum_{v_i \in \mathcal{V}_l} \tau(v_i)$$




Asymmetric Local Affinity

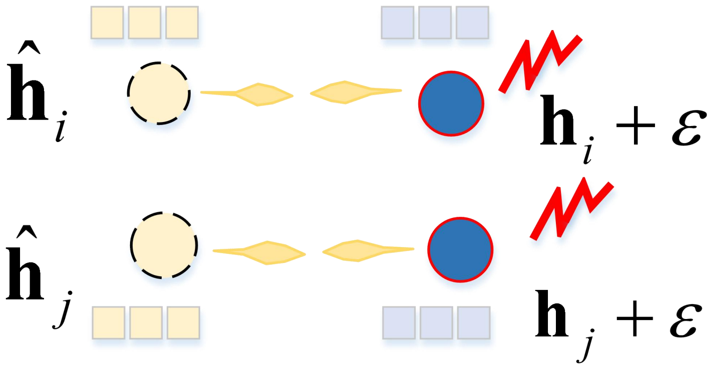
$\mathcal{V}_o$ and $\mathcal{V}_l$ are the sets of abnormal nodes and normal nodes

# Methodology – Incorporating the Egocentric Closeness Prior

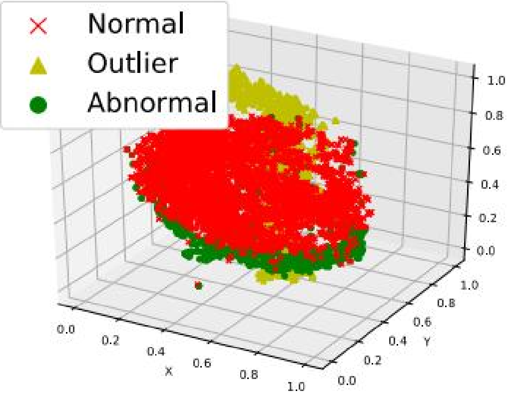
- Solely using this local affinity prior may distribute far away from the normal nodes in the representation space.

**Asymmetric local affinity** ✓

**Egocentric closeness** ✗




**Trivial outliers**




- Egocentric closeness prior-based loss

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

# Methodology – Training

- **Structural Affinity Prior**

$$\ell_{\text{ala}} = \max\left\{0, \alpha - \left(\tau\left(\mathcal{V}_l\right) - \tau\left(\mathcal{V}_o\right)\right)\right\}$$

- **Egocentric Closeness Prior**

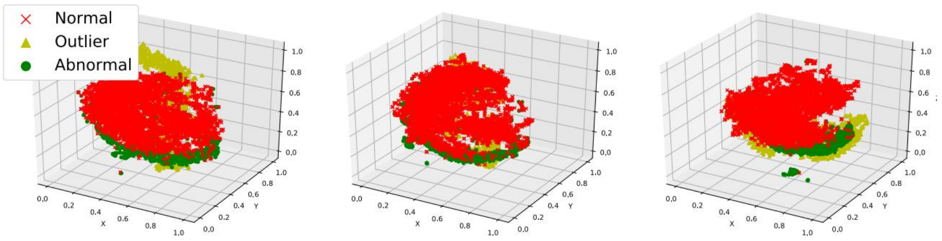
$$\ell_{ec} = \frac{1}{\left|\mathcal{V}_o\right|}\sum_{v_i \in \mathcal{V}_o}\left\|\hat{\mathbf{h}}_i - \left(\mathbf{h}_i + \varepsilon\right)\right\|_2^2$$

- **Binary cross-entropy loss function**

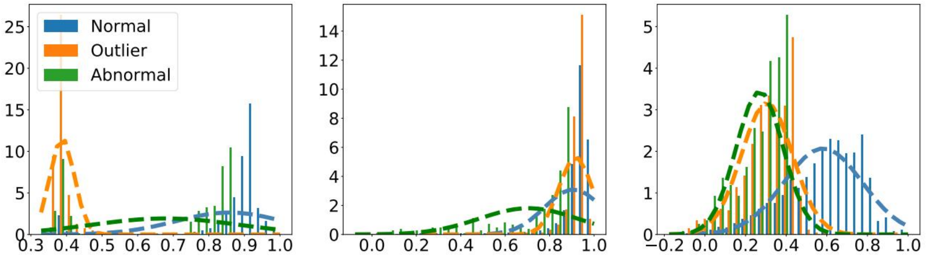
$$\ell_{bce} = \sum_{i}^{\left|\mathcal{V}_o\right|+\left|\mathcal{V}_l\right|} y_i \log\left(p_i\right) + \left(1 - y_i\right)\log\left(1 - p_i\right)$$

- **Total loss function**

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




(a) Using $\ell_{ala}$ Only (b) Using $\ell_{ec}$ only (c) Using GGAD

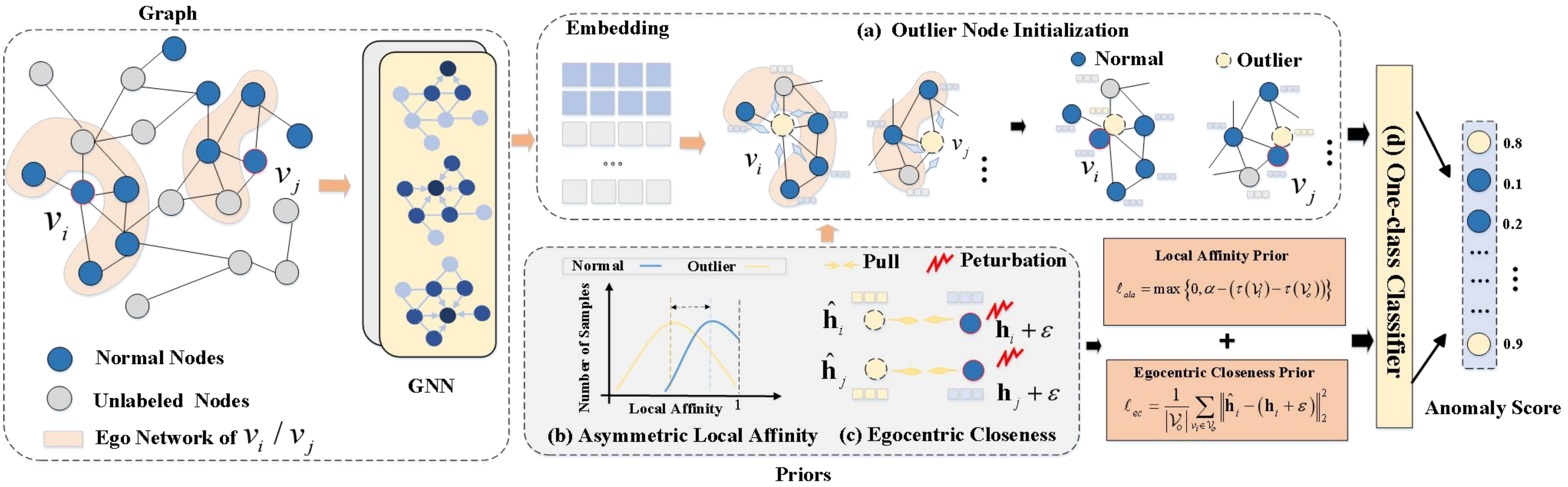



(d) Using $\ell_{ala}$ Only (e) Using $\ell_{ec}$ Only (f) Using GGAD

(a-c) t-SNE visualization of the node representations and (d-f) histograms of local affinity yielded by GGAD and its two variants on a GAD dataset T-Finance.

$\lambda$ and $\beta$ are the weights parameters.

# Overall Framework




The overview of GGAD

- The generated outlier nodes are treated as negative samples to train a discriminative one-class classifier

# Methodology – Inference

During inference, we can directly use the inverse of the prediction of the one-class classifier as the anomaly score:

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

where $\Theta^*$ is the learned parameters of GGAD.

Since our outlier nodes well assimilate the real abnormal nodes, they are expected to receive high anomaly scores from the one-class classifier.

# Datasets

Table 1. Key statistics of the six datasets used in our experiments

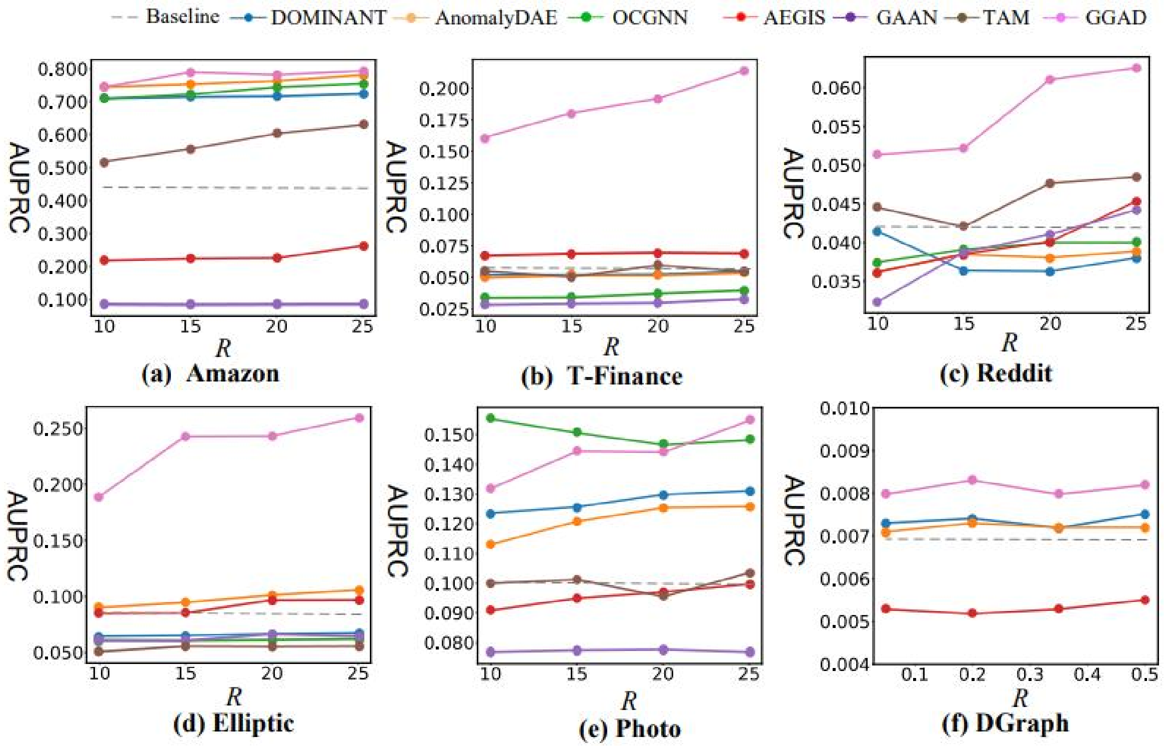
| Datasets | Type | #Node | #Edge | #Attribute | Anomaly Rate |
|---|---|---|---|---|---|
| Amazon | Co-review | 11,944 | 4,398,392 | 25 | 6.9% |
| T-Finance | Transaction | 39,357 | 21,222,543 | 10 | 4.6% |
| Reddit | Social Media | 10,984 | 168,016 | 64 | 3.3% |
| Elliptic | Bitcoin Transaction | 46,564 | 73,248 | 93 | 9.76% |
| Photo | Co-purchase | 7,535 | 119,043 | 745 | 9.2% |
| DGraph | Financial Networks | 3,700,550 | 73,105,508 | 17 | 1.3% |

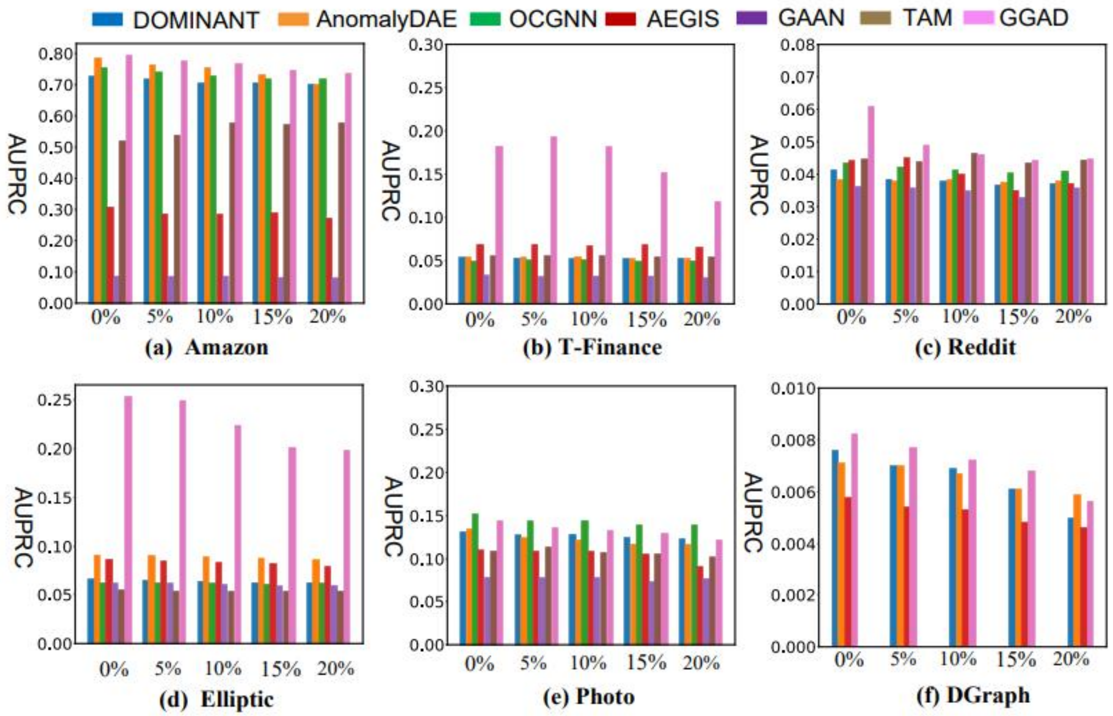
# Main Experimental Results

Table 2. AUROC and AUPRC on six GAD datasets. The best performance per dataset is boldfaced, with the second-best underlined. '/' indicates that the model cannot handle the DGraph dataset

| Setting | Method | Dataset | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | AUROC | | | | | | AUPRC | | | | | |
| | | Amazon | T-Finance | Reddit | Elliptic | Photo | DGraph | Amazon | T-Finance | Reddit | Elliptic | Photo | 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** |

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




AUPRC results w.r.t the size of training normal nodes. ‘Baseline’ denotes the performance of the best unsupervised GAD method

AUPRC w.r.t. contamination

# Ablation Study

- ❑ **Importance of the Two Anomaly Node Priors**

Table 3. Ablation study on our two priors

| Metric | Component $\ell_{ala}$ | Component $\ell_{ec}$ | 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** |

- ❑ **GGAD vs. Alternative Outlier Node Generation Approaches**

Table 4. 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 | <u>0.8204</u> | 0.5285 | 0.6786 | 0.5940 | 0.5779 |
| | GaussianP | 0.2279 | 0.6659 | 0.5235 | 0.6715 | 0.5925 | <u>0.5862</u> |
| | VAE | <u>0.8984</u> | 0.6674 | <u>0.6175</u> | <u>0.7055</u> | <u>0.6222</u> | 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 | <u>0.1762</u> | 0.0381 | 0.1924 | 0.1200 | 0.0076 |
| | GaussianP | 0.0397 | 0.0677 | 0.0376 | 0.1682 | 0.1194 | <u>0.0078</u> |
| | VAE | <u>0.6111</u> | 0.0652 | <u>0.0528</u> | <u>0.2344</u> | <u>0.1272</u> | 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** |

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

# GGAD vs. GGAD enabled Unsupervised Methods

Table 5. GGAD enabled unsupervised methods

| Metric | Method | Dataset | | |
|---|---|---|---|---|
| | | Amazon | T-Finance | Elliptic |
| #Anomalies/#Top-K Nodes | | 387/500 | 351/1000 | 1448/2000 |
| AUROC | DOMINANT | 0.7025 | 0.6087 | 0.2960 |
| | GGAD-enabled DOMINANT | 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 | DOMINANT | 0.1315 | 0.0536 | 0.0454 |
| | GGAD-enabled DOMINANT | 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** |

We incorporate the outlier generation into existing unsupervised methods to demonstrate the generation in GGAD can also benefit the existing unsupervised methods

# Conclusion

- We investigate a new semi-supervised GAD scenario where part of normal nodes are known during training.

- To fully exploit those normal nodes, we introduce a novel outlier generation approach GGAD that leverages two important priors about anomalies in the graph to learn outlier nodes that well assimilate real anomalies in both graph structure and feature representation space.

- The quality of these outlier nodes is justified by their effectiveness in training a discriminative one-class classifier together with the given normal nodes.