QUANTIZATION-CONDITIONED NORMALIZING FLOWS FOR MULTI-GRANULARITY GRAPH ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-granularity Graph Anomaly Detection aims to design a unified framework for detecting abnormal patterns at node, edge, and subgraph granularities, occupying an important position in academia and industry. Existing methods usually focus on exploiting a unified representation space for different granularities, while overlooking the distinct characteristics and anomaly patterns across granularities, leading to poor identification ability. To address these limitations, a quantization-conditioned normalizing flow framework for multi-granularity GAD (QNMGAD) is proposed to tackle this challenge. On the one hand, QNMGAD introduces prototype quantization to capture discriminative semantics across granularities and utilizes residual patterns quantization to extract latent semantics within a specific granularity. On the other hand, QNMGAD introduces conditioned normalizing flows based on granularity prototypes to model the distribution of normal patterns and calculate density estimation to identify anomalies. Extensive experiments compared with state-of-the-art baselines show the superiority of QNMGAD.

1 INTRODUCTION

Graph Anomaly Detection (GAD) (Zhu et al., 2024; Fan et al., 2020) focuses on identifying elements that exhibit deviated normal patterns, which has been widely applied across multiple applications, such as cybersecurity (Deng et al., 2022; Gao et al., 2023; Duan et al., 2024), financial industry (Zhang et al., 2022; Rasul et al., 2024), and network analysis (Bloemheuvel et al., 2021). Most graph anomaly detection approaches typically concentrate on a single granularity, such as node or graph levels, neglecting the inherent interconnections among the granularities of anomalies. For instance, a suspicious money laundering transaction may involve both an anomalous account and the local interaction network. The pathological diagnosis of cells is often jointly determined by abnormalities in particular proteins and conjunctions.

Recently, multi-granularity GAD has been developed to address this limitation, aiming to unify node-, edge-, and graph-level anomaly detection into an integrated framework. Existing methods introduce sampling strategies (Lin et al., 2024), hypergraph networks (Liu et al., 2024), or subgraph construction (Song et al., 2025) to address the key challenges of multi-granularity GAD, and they are forced to capture normal patterns across different granularities within a unified framework, overlooking the distinct characteristics and distributions of different granularities. Actually, the anomaly patterns of different granularities may exhibit distinct inconsistencies, which indicates that using a completely unified local space may result in the loss of granularity-specific semantics of anomaly patterns. Take node and edge granularities as an example. From Fig. 1, two anomaly nodes can establish either normal or abnormal connections, and heterophilic connections also exhibit inconsistent category semantics. However, according to the existing unified frameworks, the connections between anomaly nodes are more easily identified as anomalies, and heterophilic connections between nodes of different types may lose their discrimination. This phenomenon indicates that granularity-specific semantics should not be overlooked for capturing critical abnormal patterns, and existing unified solutions may fail to explore discriminative spaces that contain granularity-specific anomaly patterns, which ultimately results in poor identification ability for multi-granularity GAD. Building upon these insights, this study aims to tackle granularity inconsistency problems in multi-granularity GAD and capture granularity-specific patterns to enhance the multi-granularity detection ability.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

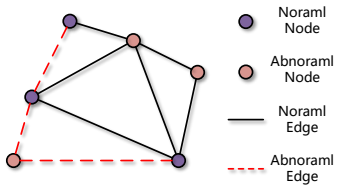


Figure 1: The inconsistency of anomaly patterns across granularity. Solutions based on unified spaces fail to extract granularity-inconsistent semantics.

We propose a quantization-conditioned normalizing flow framework for multi-granularity GAD (QNMGAD) to address this challenge via twofold measures. On the one hand, we attempt to explicitly model the semantic discrepancies across different granularities while preserving their unique semantic characteristics. Rather than enforcing a completely unified representation space, we differentiate distinct semantics of different granularities via prototype quantization. This idea utilizes discrete information to enhance discrimination across granularities, preserving the uniqueness of granularity-specific anomaly patterns and improving the poor ability to identify inconsistent anomaly patterns across different granularities under the unified framework. To capture latent semantic patterns within the granularity, we deploy residual pattern quantization. In this way, interaction with prototype concepts enhances semantic information extraction specific to granularity, and residual quantization further improves accurate feature learning.

On the other hand, the framework should establish reliable data distribution modeling conditioned on specific granularity in an unsupervised manner to achieve effective anomaly detection. Since anomaly labels are unavailable in practice, the model is expected to capture the intrinsic regularities shared by normal samples, thereby effectively identifying instances that deviate from these learned regularities. We turn our attention to normalizing flow (Chen et al., 2026b), which can provide an invertible mapping and exact density computation for modeling complex distributions compared with other generative solutions, to investigate its potential in multi-granularity GAD. To endow the normalizing flow with granularity discrimination, we leverage prototype concepts as a condition to establish a Gaussian distribution with granularity-specific semantics, employing granularity-discriminative Gaussian distributions to learn granularity-specific behavioral patterns.

By implementing these twofold measures, QNMGAD provides a principled direction for multi-granularity GAD: preserving granularity-specific semantic characteristics based on granularity discrimination, while constructing unified yet collaborative normality representations for multi-granularity GAD. The contributions are as follows:

- **A new problem**. To the best of our knowledge, we make the first attempt to explore inconsistent granularity-specific normal patterns in multi-granularity GAD problems instead of integrating them into a unified framework.
- **A novel model**. We propose a quantization-conditioned normalizing flow framework for multi-granularity GAD (QNMGAD). QNMGAD utilizes prototype quantization to discriminate granularities. Besides, QNMGAD captures latent semantics using residual pattern quantization and employs conditioned normalizing flows to extract the normal patterns within the specific granularity.
- **Advanced performance**. Extensive experiments demonstrate the effectiveness of QNMGAD compared with state-of-the-art baselines.

2 Related Work

Graph Anomaly Detection. Existing graph anomaly detection has a progressive ability to exploit structural and semantic information for anomalies, which can be mainly categorized into supervised and unsupervised methods. Supervised methods utilize label information to exploit critical characteristics, but they heavily rely on high-quality label information. For example, GE-GNN (Zhang & Luo, 2025) introduces augmented edge information and a gate mechanism to capture anomaly features. FLAG (Yang et al., 2025a) introduces an LLM to enhance the interactive semantics. Subsequently, unsupervised GAD methods are proposed to solve this problem, primarily including contrastive learning-based and reconstruction-based methods. MCLASt (Zhang & Luo, 2026) introduces multi-scale attributive contrastive learning and topological awareness to detect anomalies. SGAD-DCR (Chen et al., 2026c) introduces dual reconstruction and dual contrastive learning to calculate the anomaly scores. Beyond the above, some studies introduce advanced technologies such as diffusion (Wei et al., 2026; Xiao et al., 2025) and adversarial learning (Chen et al., 2026a) to capture the latent representations. Although they achieve promising progress, they only focus on specific

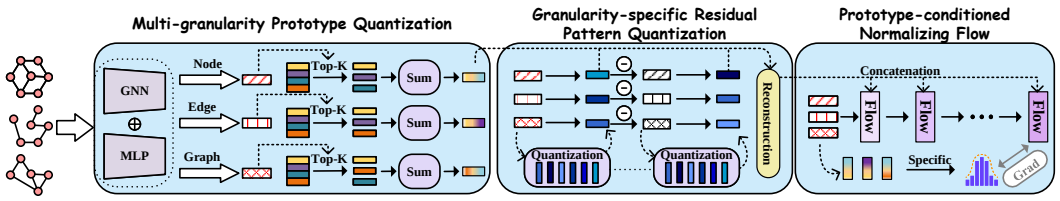


Figure 2: The framework of QNMGAD. QNMGAD first leverages prototype quantization to enhance the discrimination between granularities. Then, QNMGAD employs residual pattern quantization to capture latent semantics within the specific granularity through interaction with prototype concepts. Finally, QNMGAD introduces prototype-conditioned normalizing flows to fit a granularity-specific distribution, using density estimation to model specific normal patterns and calculate anomaly scores.

tasks, especially node-level anomaly detection, overlooking the identification of multi-granularity anomalies arising from collective behaviors.

Multi-granularity Graph Anomaly Detection. Some studies explore the multi-granularity GAD problem. BOURNE (Liu et al., 2024) jointly considers node- and edge-level anomalies by constructing dual hypergraph views and assessing embedding consistency. UniGAD (Lin et al., 2024) proposes a maximum Rayleigh quotient subgraph sampler to harmonize multi-level anomaly formats by mapping different objects into subgraph-level detection tasks. However, it depends on high-quality label information. Subsequently, UniFORM (Song et al., 2025) employs an energy-based GNN to model uncertainty of multi-granularity anomalies. However, it suffers from the computational burden of energy functions and a time-consuming sampling strategy. GFM-UAD (Han et al., 2026) unifies multi-granularity and cross-domain GAD using adversarial learning and a fine-tuning strategy. GRAND (Wang et al., 2026) unifies node- and edge-level anomaly detection by introducing dual diffusion strategies and utilizing GraphTransformers as a denoiser.

Despite this progress, existing multi-granularity GAD methods largely assume that anomalies follow the same data distribution and exhibit similar behavior patterns across granularities, while overlooking the discrepancies in behavior patterns between different granularities and the distinction between their corresponding distributions. In contrast, QNMGAD explores granularity-specific behavior patterns, modeling their distributions using prototype quantization and capturing intrinsic semantic and normal patterns with conditioned normalizing flows. These strategies avoid cross-granularity semantic overlap and enhance the discriminability for samples within the specific granularity.

3 PRELIMINARIES

Notations. We denote a graph as $G = (V, E, \mathbf{X}, \mathbf{A})$, where V is the node set, E is the edge set, $\mathbf{X} \in \mathbb{R}^{|V|\times d}$ is the node feature matrix, and $\mathbf{A} \in \mathbb{R}^{|V|\times|V|}$ is the adjacency matrix. For node $v_i \in V$, its feature vector is denoted as $\mathbf{x}_i \in \mathbb{R}^d$, where d is the dimension of node attributes. Unless otherwise specified, graphs are assumed to be undirected, in which case $\mathbf{A}$ is symmetric. $a_{ij} \in \mathbf{A} = 1$ denote there is an edge $e_{ij} \in E$ from node v_i to node v_j.

Problem Statement. Given a training set G_T, the aim is to use multi-granularity representations, such as node, edge, or graph levels, to train a unified model and then predict multi-granularity anomalies in a test set G_E, which contains arbitrary labels at any of these granularities.

Normalizing Flows. Normalizing flows transform a simple base probability distribution into a more complex one through a series of invertible mappings, leveraging the change of variables theorem. As the initial density is progressively modified via this "flow," the result remains a valid, well-defined probability distribution. Formally, we define a flow model $F = f_{K_f} \circ \cdots \circ f_1 : \mathbf{X} \leftrightarrow \mathbf{Z}$. The probability density function $p(\mathbf{X})$ can be formulated as:

$$p(\mathbf{X}) = p(\mathbf{Z})|\det \frac{\partial F}{\partial \mathbf{X}}|, \tag{1}$$

where $|\det \frac{\mathrm{d}F}{\mathrm{d}\mathbf{X}}|$ is the determinant of the Jacobian matrix, and K is the length of the flow. Normalizing flows can be trained by minimizing the negative log-likelihood of the latent $\mathbf{Z}$ on the base

distribution $\mathcal{Z}$:

$$\mathcal{L}_f = -\mathbb{E}_{x\sim p_{\mathcal{X}}}[\log p_{\mathcal{Z}}(\mathbf{Z}) + \log|\det \frac{\partial F}{\partial \mathbf{X}}|]. \tag{2}$$

With invertibility, F can capture the likelihood of the normal distribution, and the anomalies can be isolated as outliers.

4 METHODOLOGY

In this section, we introduce the framework of QNMGAD in detail. Unlike existing anomaly detection methods that overlook the granularity discrepancies, the core of QNMGAD is to model granularity-specific semantic information of normal patterns. To achieve this goal, QNMGAD first leverages prototype quantization to discriminate the discrepancies across granularities. To capture rich semantics within the granularity, QNMGAD employs residual pattern quantization, enhancing the interaction with prototype concepts and extracting residual information for precise quantization. Lastly, QNMGAD utilizes a prototype-conditioned normalizing flow to estimate the density of normal patterns. The framework of QNMGAD is shown in Fig. 2. The algorithm and complexity analysis are shown in Section K.

4.1 MULTI-GRANULARITY PROTOTYPE QUANTIZATION

Existing multi-granularity GAD methods try to unify different granularities within a universal local representation space, while ignoring the semantic discrepancies of anomaly patterns between different granularities. Portraying semantic information at specific granularities can help models better understand discriminative anomaly patterns across granularities. To this end, we introduce multi-granularity prototype quantization that utilizes initialized and learnable prototypes to capture the distinct concepts for different granularities.

Specifically, we first transform the node features into latent representation spaces and capture their local and global characteristics.

$$\mathbf{H}_1 = \mathcal{F}_1(\mathbf{X}), \quad \mathbf{H}_2 = \mathcal{F}_2(\mathbf{H}_1, \mathbf{A}), \tag{3}$$

where $\mathcal{F}_1$ is an MLP, $\mathcal{F}_2$ is a GCN, $\mathbf{H}_1, \mathbf{H}_2 \in \mathbb{R}^{|V|\times d'}$ and d' is latent dimension. To better capture the generalized representations for multi-granularity characteristics, we design the latent representations based on the encoded node features for different granularities. Specifically, given encoded local features $\mathbf{H}_1$ and global features $\mathbf{H}_2$, the node-granularity representation $\mathbf{H}^n \in \mathbb{R}^{|V|\times d'}$ can be defined:

$$\mathbf{H}^n = \mathcal{F}_g^n(\mathbf{H}_1||\mathbf{H}_2), \tag{4}$$

where $\mathcal{F}_g(\cdot)$ is an MLP. For graph-level anomalies, we leverage $\mathrm{MeanPool}(\cdot)$ to obtain the graph-granularity representation representations $\mathbf{H}^g \in \mathbb{R}^{|V|\times d'}$:

$$\mathbf{H}^g = \mathcal{F}_g^g(\mathrm{MeanPool}(\mathbf{H}_1)||\mathrm{MeanPool}(\mathbf{H}_2)). \tag{5}$$

We utilize the subtraction between the features of two connected nodes to represent edge-granularity representations. For example, give e_{ij}, its edge-granularity representations $\mathbf{h}^e \in \mathbb{R}^{d'}$ can be calculated as:

$$\mathbf{h}^{e_{ij}} = \mathcal{F}_g^e((\mathbf{H}_1[i,:] - \mathbf{H}_1[j,:])||(\mathbf{H}_2[i,:] - \mathbf{H}_2[j,:])). \tag{6}$$

where $\mathbf{H}_2[i,:]$ is the global features of node v_i, and $\mathbf{H}_1[i,:]$ is the local features of node v_i. With loss of generality, similar to $\mathbf{H}^n$ and $\mathbf{H}^g$, we denote the edge-granularity representations $\mathbf{H}^e \in \mathbb{R}^{|E|\times d'}$.

After getting the latent representations for different granularities, we generate a finite prototype concept set $\mathcal{C}_{pr} = \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_{K_c}\}$, where $K_c = |\mathcal{C}_{pr}|$ is the number of prototypes, and then we transform multi-granularity representations into a single vector using the $\mathrm{MeanPool}(\cdot)$ function:

$$\mathbf{h}_g^L = \mathcal{F}_g^m(\mathrm{MeanPool}(\mathbf{H}^L)), \tag{7}$$

where $L \in \{n, g, e\}$ denotes different superscripts of different granularities and $\mathcal{F}_g^m$ is an MLP. Then we calculate the similarity between the semantic-sensitive vectors $\mathbf{h}_g^L$ and concept prototypes, and generate a subset $\mathcal{C}_{pr}^L \subset \mathcal{C}_{pr}$ for granularity L. $\mathcal{C}_{pr}^L$ contains the top-K most similar prototypes using

Euclidean metric. Then we normalize the similarity to generate the dynamic aggregation weights $\mathbf{w}^L \in \mathbb{R}^{|\mathcal{C}_{pr}^L|}$ for evaluating the contributions of prototypes, and generate the concept prototype $\bar{\mathbf{c}}_{pr}$:

$$\bar{\mathbf{c}}_{pr}^L = \sum_{k=1}^{|\mathcal{C}_{pr}^L|} w_k \mathbf{c}_k. \tag{8}$$

Therefore, the training objective $\mathcal{L}_{pr}$ is formulated by evaluating the distance between $\bar{\mathbf{c}}_{pr}$ and $\mathbf{h}_g^L$:

$$\mathcal{L}_{pr}^L = \mathbb{E}[||\bar{\mathbf{c}}_{pr}^L - \mathbf{h}_g^L||^2], \tag{9}$$

where w_k is the k-th element in $\mathbf{w}$. For intuitive understanding, the concept set $\mathbf{C}_{pr}^L$ for each granularity serves as a granularity pointer for discriminating different granularities. Although these concept prototypes may not perfectly align with real semantic prototypes, each element in $\mathcal{C}_{pr}^L$ remains distinct from the other elements, thereby endowing our model with concept discrimination capability.

4.2 Granularity-specific Residual Pattern Quantization

Prototype quantization only captures coarse-grained and generalized concepts to discriminate the distinction between different granularities, which cannot achieve fine-grained, granularity-specific semantic perception to identify the margin between normal patterns and anomaly patterns. Therefore, we need to explore the granularity-specific normal patterns, which play important roles in GAD, especially where normal patterns vary across different distributions.

The direct way to extract normal patterns is to quantify latent representations with a pattern concept set $\mathcal{C}_s^L$ through instance-level mapping, similar to $\mathcal{C}_{pr}$. Different from multi-granularity prototype quantization that aims to record distinctive prototype conceptions for discriminating granularities, pattern quantization aims to capture feature-level rich semantic information for normal pattern modeling. However, vanilla quantization schemes have limited semantic perception because it lacks interaction with the concept set and precise feature modeling. To better capture normal patterns for each granularity, we introduce granularity-specific residual pattern quantization. We integrate the granularity-specific prototypes to enhance granularity awareness. Besides, we utilize the residual quantization to capture latent semantic characteristics and achieve more precise modeling.

Specifically, given the initialized concept set $\mathcal{C}_s^L = \{\mathbf{c}_1, \cdots, \mathbf{c}_{K_c}\}$, where $K_c = |\mathcal{C}_s^L|$, we first implement straight quantization on the instance-level features via integrating granularity-specific awareness using $\bar{\mathbf{c}}_{pr}^L$:

$$\hat{\mathbf{c}}_1^L = \arg\min_{\mathbf{c}_k \in \mathcal{C}_s^L} ||(\mathbf{h}_i^L - \bar{\mathbf{c}}_{pr}^L) - \mathbf{c}_k||^2, \tag{10}$$

where $\mathbf{h}_i^L = \mathbf{H}^L[i] \in \mathbb{R}^{d'}$ denotes the i-th row vector in $\mathbf{H}^L$. However, this straight quantization lacks interaction with prototype concepts. After obtaining the pattern concept for each instance, we calculate the residual between instances and their corresponding pattern concept:

$$\mathbf{r}^L = \mathbf{h}_i^L - \hat{\mathbf{c}}_1^L, \tag{11}$$

where $\mathbf{r}_1^L \in \mathbb{R}^{d'}$ is the residual, which can be quantified to further determine a more precise mapping. For simplicity, we utilize $\mathcal{C}_s^L$ as the pattern concept for residual quantization:

$$\hat{\mathbf{c}}_2^L = \arg\min_{\mathbf{c}_k \in \mathcal{C}_s^L} ||\mathbf{r}_1^L - \mathbf{c}_k||^2. \tag{12}$$

Then, we formulate the reconstruction using residual pattern quantization:

$$\hat{\mathbf{h}}_i^L = \hat{\mathbf{c}}_1^L + \bar{\mathbf{c}}_{pr}^L + \hat{\mathbf{c}}_2^L. \tag{13}$$

where $\hat{\mathbf{h}}_i^L$ is the reconstruction result of $\mathbf{h}_i^L$ using residual pattern quantization.

The training objective of residual pattern quantization can be formulated as:

$$\mathcal{L}_r^L = \mathbb{E}_i[||\hat{\mathbf{h}}_i^L - \mathbf{h}_i^L||^2]. \tag{14}$$

Table 1: The performance of node- and edge-level GAD on single-graph datasets

	Dataset	Reddit		Weibo		Amazon		Yelp		Tolokers		Questions		T-Finance	
	Metrics	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
Node-Level	AutoGAD	55.3 ± 1.6	5.5 ± 0.8	80.0 ± 1.8	22.4 ± 1.2	82.8 ± 1.4	30.4 ± 2.1	60.5 ± 1.9	8.5 ± 1.0	70.8 ± 1.5	30.8 ± 1.8	60.2 ± 1.7	12.5 ± 0.8	78.7 ± 1.6	35.7 ± 1.5
	HEC-GAD	58.0 ± 1.5	6.1 ± 1.2	82.6 ± 1.7	25.4 ± 1.3	80.2 ± 1.6	28.4 ± 2.1	77.0 ± 1.8	45.5 ± 1.2	73.3 ± 1.4	35.5 ± 1.5	65.1 ± 1.6	17.7 ± 0.7	72.4 ± 1.5	28.8 ± 1.5
	SGAD-DCR	60.1 ± 1.7	5.3 ± 1.4	81.7 ± 1.6	21.2 ± 1.0	84.5 ± 1.8	33.4 ± 1.5	69.7 ± 1.5	31.2 ± 1.2	72.5 ± 1.6	33.5 ± 2.3	64.5 ± 1.7	15.0 ± 1.2	78.2 ± 1.4	36.2 ± 1.8
	CVGAD	61.0 ± 1.9	6.5 ± 1.5	85.5 ± 1.8	27.6 ± 0.8	85.8 ± 1.6	34.0 ± 2.0	65.9 ± 2.0	28.7 ± 1.7	76.1 ± 1.7	38.2 ± 1.8	66.6 ± 1.5	18.1 ± 1.8	74.2 ± 1.6	38.1 ± 0.8
	AD-GCL	60.1 ± 1.4	6.0 ± 1.0	84.4 ± 1.6	26.0 ± 0.8	81.2 ± 1.5	25.4 ± 1.1	70.0 ± 1.7	37.8 ± 0.8	75.2 ± 1.6	36.5 ± 2.6	70.4 ± 1.5	19.7 ± 1.4	80.6 ± 1.4	40.0 ± 0.9
	UniFORM	62.9 ± 1.3	6.8 ± 1.7	83.9 ± 1.5	23.7 ± 2.5	80.6 ± 1.4	32.7 ± 1.7	74.0 ± 1.6	40.5 ± 1.5	79.0 ± 1.5	42.0 ± 1.1	68.3 ± 1.4	17.5 ± 0.7	82.4 ± 1.3	41.5 ± 0.8
	BOURNE	64.4 ± 1.2	7.2 ± 2.5	84.4 ± 1.4	26.4 ± 1.6	83.9 ± 1.3	30.1 ± 1.5	73.1 ± 1.5	42.5 ± 1.0	78.3 ± 1.4	34.5 ± 1.0	63.4 ± 1.3	14.9 ± 1.5	68.2 ± 1.2	22.5 ± 2.2
	GRAND	63.7 ± 1.1	8.3 ± 1.6	85.0 ± 1.3	29.4 ± 1.1	86.5 ± 1.2	36.8 ± 1.2	77.4 ± 1.4	44.5 ± 0.8	76.6 ± 1.3	39.9 ± 1.9	72.1 ± 1.2	21.8 ± 1.1	75.6 ± 1.1	30.5 ± 2.1
	QNMGAD	**69.5** ± 0.8	**10.3** ± 1.6	**89.4** ± 0.7	**32.0** ± 0.8	**90.9** ± 1.2	**40.5** ± 1.1	**82.4** ± 0.8	**51.7** ± 0.5	**83.3** ± 0.7	**46.6** ± 0.6	**76.8** ± 0.6	**25.5** ± 0.8	**85.2** ± 1.0	**45.6** ± 0.6
Edge-level	AutoGAD	52.4 ± 1.7	4.8 ± 0.3	84.0 ± 1.2	38.5 ± 1.2	84.9 ± 1.6	40.5 ± 1.2	62.4 ± 1.8	22.7 ± 1.3	66.8 ± 1.5	25.5 ± 1.0	70.2 ± 1.7	31.5 ± 2.4	70.7 ± 1.6	26.7 ± 2.0
	HEC-GAD	58.2 ± 1.6	5.3 ± 0.8	85.6 ± 1.8	40.2 ± 1.4	87.3 ± 1.7	48.5 ± 1.5	67.1 ± 1.9	26.3 ± 1.4	67.2 ± 1.5	28.3 ± 1.4	69.1 ± 1.6	28.5 ± 1.1	76.4 ± 1.5	33.2 ± 2.5
	SGAD-DCR	55.2 ± 1.8	5.0 ± 1.2	81.8 ± 1.7	35.7 ± 1.0	82.9 ± 1.9	37.5 ± 0.7	73.8 ± 1.6	35.3 ± 1.7	72.3 ± 1.7	30.6 ± 1.0	65.5 ± 1.8	20.5 ± 1.9	78.2 ± 1.6	35.7 ± 1.5
	CVGAD	55.0 ± 2.0	5.0 ± 1.0	82.7 ± 1.9	36.6 ± 0.6	81.2 ± 1.7	35.5 ± 1.5	66.0 ± 2.1	28.2 ± 1.0	72.2 ± 1.8	31.9 ± 1.9	68.6 ± 1.6	25.0 ± 1.0	74.2 ± 1.7	30.7 ± 1.4
	AD-GCL	53.2 ± 1.5	4.9 ± 1.1	86.4 ± 1.7	42.5 ± 2.1	88.2 ± 1.6	49.8 ± 1.8	60.1 ± 1.8	21.3 ± 1.1	71.3 ± 1.7	30.5 ± 1.4	70.4 ± 1.6	30.2 ± 2.2	80.6 ± 1.5	40.7 ± 2.0
	UniFORM	58.0 ± 1.4	5.4 ± 1.5	85.9 ± 1.6	40.5 ± 0.7	88.6 ± 1.5	47.2 ± 1.4	64.1 ± 1.7	25.3 ± 1.5	69.1 ± 1.6	29.5 ± 1.7	70.3 ± 1.5	29.5 ± 1.9	72.4 ± 1.4	27.7 ± 1.5
	BOURNE	56.8 ± 1.3	5.2 ± 1.1	82.4 ± 1.5	37.5 ± 1.2	84.0 ± 1.4	41.3 ± 1.3	72.5 ± 1.6	31.3 ± 1.0	72.4 ± 1.5	32.8 ± 1.5	71.8 ± 1.4	31.0 ± 2.0	68.2 ± 1.3	23.7 ± 2.3
	GRAND	60.8 ± 1.2	7.0 ± 1.5	86.4 ± 1.4	41.5 ± 1.2	85.0 ± 1.3	43.8 ± 0.6	71.5 ± 1.5	30.3 ± 0.7	71.4 ± 1.4	30.5 ± 1.8	72.8 ± 1.3	33.5 ± 2.43	75.2 ± 1.2	34.7 ± 1.3
	QNMGAD	**65.8** ± 0.5	**11.4** ± 0.4	**90.4** ± 0.6	**45.5** ± 1.4	**92.2** ± 0.5	**53.5** ± 0.5	**78.5** ± 0.7	**40.3** ± 0.6	**78.4** ± 0.6	**41.2** ± 1.1	**76.8** ± 0.5	**38.8** ± 0.7	**83.5** ± 0.5	**45.9** ± 0.9

4.3 Prototype-conditioned Normalizing Flows

Normalizing flows can flexibly preserve exact density computation via invertible transformations, exhibiting superiority over other generative models. Therefore, we utilize normalizing flows to model the patterns of normal samples. However, in multi-granularity GAD settings, it is important to faithfully model granularity-specific patterns and avoid cross-granularity semantic overlaps and noise. To better capture the complex posterior distribution characteristics for normal patterns within each granularity, we introduce prototype-conditioned normalizing flows, achieving granularity-aware posterior distribution approximation.

Instead of assuming a vanilla standard Gaussian distribution, we transform the target distribution based on $p_{\mathcal{Z}}(\mathbf{z})$ into a conditioned mixed Gaussian distribution $p_{\mathcal{Z}}(\mathbf{z}|\mathcal{C}_{pr}^L)$ based on the prototype concept set $\mathcal{C}_{pr}^L$. The intuition is that the prototype concept contains critical clues to discriminate granularity-specific patterns through quantization loss training, and can be employed to avoid semantic overlap across granularities. Specifically, the mixed posterior distribution $p_{\mathcal{Z}}(\mathbf{z}|\mathcal{C}_{pr}^L)$ can be formulated as the sum of $|\mathcal{C}_{pr}^L|$ prototype-specific Gaussian distribution $p_{\mathcal{Z}}(\mathbf{z}|\bar{\mathbf{c}}_{pr}^L)$:

$$p_Z^L(\mathbf{z}|\mathcal{C}_{pr}^L) = \mathcal{N}(\mathbf{z}; \mu(\bar{\mathbf{c}}_{pr}^L), \sigma^2(\bar{\mathbf{c}}_{pr}^L)), \tag{15}$$

where $\mu(\cdot)$ and $\sigma^2(\cdot)$ are MLP prediction functions, ensuring that the base distribution dynamically adapts to the underlying concept prototypes.

With the pattern-quantified semantic representations, we treat the quantized representations $\hat{\mathbf{h}}_i^L$ as condition features and integrate them with the corresponding latent representations $\mathbf{h}_i^L$ to feed into the flow model:

$$\mathbf{z}_i^L = F(\hat{\mathbf{h}}_i^L || \mathbf{h}_i^L) = f_{K_f} \circ \cdots \circ f_1(\hat{\mathbf{h}}_i^L || \mathbf{h}_i^L). \tag{16}$$

where $\mathbf{z}_i^L \sim \mathcal{N}(\mathbf{z}_i^L; \mu(\bar{\mathbf{c}}_{pr}^L), \sigma^2(\bar{\mathbf{c}}_{pr}^L))$ follows the Gaussian distribution with prototype-conditioned parameters. Correspondingly, the training objective can be formulated as:

$$\mathcal{L}_f^L = -\mathbb{E}_{\mathbf{h}\sim p_{\mathcal{H}}}[\log p_{\mathcal{Z}}(\mathbf{z}|\bar{\mathbf{c}}_{pr}^L) + \log|\det \frac{\partial F}{\partial \mathbf{h}}|]. \tag{17}$$

The instance-level computation can be formulated as:

$$p_{\mathcal{Z}}(\mathbf{z}|\bar{\mathbf{c}}_{pr}^L) = -\frac{1}{2}\log(2\pi) - \log\sigma(\bar{\mathbf{c}}_{pr}^L) - \frac{(\mathbf{z} - \mu(\bar{\mathbf{c}}_{pr}^L))^2}{2\sigma^2(\bar{\mathbf{c}}_{pr}^L)}. \tag{18}$$

4.4 Training and Inference

We illustrate the training process in detail. For a single-granularity task, such as node-level detection, we only need to learn the latent semantics within node granularity; therefore, the training function can be formulated as:

$$\mathcal{L}^n = \mathcal{L}_f^n + \theta\mathcal{L}_r^n + \epsilon\mathcal{L}_{pr}^n. \tag{19}$$

where θ and ϵ are hyperparameters. In addition, in transferability experiments for multi-granularity tasks, such as from node granularity to edge granularity detection, the training function can be formulated as:

$$\mathcal{L}^L = \mathcal{L}_f^n + \theta\mathcal{L}_r^n + \epsilon\frac{1}{|L|}\sum_{l\in L}\mathcal{L}_{pr}^l. \tag{20}$$

Table 2: The performance of node- and graph-level GAD on multi-graph dataset.

	Dataset	BM-MN		BM-MS		BM-MT		MUTAG		MNIST0		MNIST1		T-Group	
	Metries	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
Node-level	AutoGAD	81.2 ± 1.4	40.2 ± 1.4	73.8 ± 1.6	30.5 ± 1.0	65.6 ± 1.5	11.0 ± 1.1	86.1 ± 1.2	37.0 ± 0.6	90.9 ± 1.4	45.1 ± 1.6	92.2 ± 1.3	50.2 ± 1.2	85.0 ± 1.5	40.2 ± 0.8
	HEC-GAD	63.9 ± 1.8	25.5 ± 1.0	58.7 ± 1.9	15.2 ± 1.4	62.1 ± 1.7	10.2 ± 1.1	85.4 ± 1.3	36.5 ± 0.9	91.8 ± 1.5	50.2 ± 0.6	93.6 ± 1.4	55.1 ± 1.4	75.9 ± 1.6	31.2 ± 1.3
	SGAD-DCR	58.1 ± 1.9	18.2 ± 0.8	67.6 ± 2.0	24.8 ± 1.8	57.9 ± 1.8	9.5 ± 0.5	86.8 ± 1.4	37.5 ± 0.8	93.2 ± 1.1	55.7 ± 1.7	90.1 ± 1.0	45.0 ± 1.0	78.0 ± 1.6	35.2 ± 1.5
	CVGAD	59.4 ± 1.7	17.9 ± 1.4	59.8 ± 1.8	14.3 ± 1.0	57.2 ± 1.9	9.4 ± 0.4	82.8 ± 1.6	33.2 ± 1.5	90.1 ± 1.7	42.8 ± 1.3	90.0 ± 1.6	46.1 ± 0.5	81.9 ± 1.5	38.2 ± 1.9
	AD-GCL	66.7 ± 1.6	27.1 ± 1.1	69.1 ± 1.7	16.0 ± 1.0	71.4 ± 1.5	25.4 ± 1.2	85.9 ± 1.2	35.4 ± 1.8	92.8 ± 1.1	52.2 ± 1.5	91.5 ± 1.0	48.5 ± 1.4	79.6 ± 1.4	35.2 ± 2.1
	UniFORM	68.2 ± 1.5	28.2 ± 1.7	71.9 ± 1.6	28.2 ± 0.7	67.8 ± 1.7	16.2 ± 1.7	88.3 ± 1.3	40.8 ± 1.9	94.9 ± 1.0	60.2 ± 1.6	94.8 ± 0.9	55.1 ± 1.8	80.4 ± 1.3	40.0 ± 1.4
	BOURNE	79.8 ± 1.4	37.5 ± 2.5	62.9 ± 1.6	18.2 ± 1.8	71.2 ± 1.5	22.7 ± 1.7	88.9 ± 1.3	39.6 ± 1.6	95.7 ± 1.2	68.2 ± 1.5	92.6 ± 1.1	52.9 ± 2.4	76.4 ± 1.6	38.2 ± 2.8
	GRAND	83.4 ± 1.2	42.2 ± 0.7	80.1 ± 1.3	40.0 ± 1.0	72.3 ± 1.2	24.2 ± 0.9	84.4 ± 1.1	35.2 ± 1.5	95.2 ± 0.9	70.2 ± 1.8	92.1 ± 0.9	50.1 ± 1.6	85.7 ± 1.1	42.2 ± 1.7
	QNMGAD	**87.0** ± 0.6	**46.2** ± 0.6	**89.0** ± 0.7	**52.2** ± 0.5	**80.1** ± 0.6	**38.2** ± 0.6	**92.2** ± 0.5	**49.8** ± 0.5	**97.0** ± 0.5	**82.8** ± 1.2	**96.2** ± 0.5	**68.5** ± 0.6	**90.3** ± 0.6	**55.2** ± 1.1
Graph-level	RQGNN	79.3 ± 1.4	35.2 ± 1.8	70.2 ± 1.6	28.4 ± 1.4	81.3 ± 1.5	45.7 ± 1.7	89.4 ± 1.0	51.1 ± 1.8	85.1 ± 1.4	38.4 ± 1.2	83.8 ± 1.3	32.0 ± 1.1	71.8 ± 1.5	25.0 ± 1.5
	GUL	78.7 ± 1.8	32.1 ± 1.1	78.4 ± 1.9	35.8 ± 1.5	64.9 ± 1.7	18.5 ± 1.8	88.8 ± 1.5	54.0 ± 2.0	88.6 ± 1.5	40.0 ± 1.1	83.5 ± 1.4	31.3 ± 2.1	81.5 ± 1.6	38.0 ± 1.2
	MUSE	80.0 ± 1.9	38.5 ± 1.9	66.0 ± 2.0	24.7 ± 1.2	70.0 ± 1.8	22.0 ± 1.4	87.0 ± 1.4	50.8 ± 1.9	87.0 ± 1.1	38.8 ± 1.7	82.6 ± 1.0	30.5 ± 1.5	74.2 ± 1.6	27.8 ± 1.7
	GLAD	77.3 ± 1.7	32.5 ± 0.7	64.9 ± 1.8	22.4 ± 2.1	59.4 ± 1.9	11.7 ± 1.1	89.2 ± 1.4	52.2 ± 2.1	84.4 ± 1.5	33.7 ± 2.3	77.0 ± 1.6	24.0 ± 1.4	62.6 ± 1.5	11.0 ± 1.8
	GLADPro	81.5 ± 1.6	40.2 ± 0.9	75.5 ± 1.7	31.2 ± 1.0	65.7 ± 1.5	16.3 ± 1.5	87.4 ± 1.0	51.0 ± 1.0	87.9 ± 1.5	39.9 ± 1.1	85.7 ± 1.7	37.2 ± 1.4	78.2 ± 1.4	29.0 ± 1.6
	UniFORM	75.1 ± 1.5	30.2 ± 1.5	65.6 ± 1.6	20.8 ± 1.8	69.2 ± 1.7	20.9 ± 1.1	86.0 ± 1.7	48.2 ± 1.7	83.0 ± 1.3	37.8 ± 1.5	76.5 ± 0.9	26.4 ± 1.6	73.9 ± 1.3	18.0 ± 1.7
	BOURNE	73.0 ± 1.4	27.2 ± 1.7	59.2 ± 1.6	18.2 ± 0.5	75.6 ± 1.5	28.8 ± 1.0	87.8 ± 1.3	49.7 ± 1.9	88.2 ± 1.1	41.2 ± 2.1	79.9 ± 1.1	28.9 ± 1.7	65.7 ± 1.6	13.4 ± 2.3
	GRAND	80.1 ± 1.2	38.2 ± 2.1	72.2 ± 1.3	30.0 ± 1.1	79.0 ± 1.2	33.1 ± 1.3	90.1 ± 1.8	55.5 ± 1.7	85.6 ± 2.1	35.9 ± 1.7	80.8 ± 0.9	31.0 ± 0.8	74.8 ± 1.1	26.1 ± 0.8
	QNMGAD	**85.1** ± 0.6	**45.7** ± 1.2	**84.2** ± 0.7	**46.0** ± 0.9	**85.4** ± 0.6	**50.2** ± 1.1	**94.5** ± 0.9	**60.3** ± 1.2	**91.1** ± **1.1**	**53.7** ± 1.0	**90.0** ± **0.8**	**45.7** ± 0.8	**85.8** ± 0.6	**43.0** ± 1.4

where $L = \{n, e\}$. This is because $\mathcal{L}_f^n$ and $\mathcal{L}_r^n$ are used to capture latent semantics within the training granularity, while $\mathcal{L}_{pr}^L$ can extract granularity-specific distinctions. Therefore, in the multi-granularity transferability experiments, we introduce $\mathcal{L}_{pr}^n$ and $\mathcal{L}_{pr}^e$ for node and edge granularities.

Given that anomalous points tend to lie in sparse regions far from the main data cluster, we posit that they exhibit relatively low probability densities in the inference stage. Accordingly, we adopt the density estimate $p_{\mathcal{Z}}(\mathbf{z})$ as our scoring function for anomaly detection, with smaller density values corresponding to a stronger indication of abnormality, which can be formulated as:

$$p(\mathbf{z}_i) = -\log p_{\mathcal{Z}}(\mathbf{z}_i) - \log|\det \frac{\partial F}{\partial \mathbf{h}}|. \tag{21}$$

5 Experiments

5.1 Datasets and Baselines

We introduce 14 datasets (Lin et al., 2024), including 7 single-graph datasets for node- and edge-level anomaly detection and 7 multi-graph datasets for node- and graph-level anomaly detection. The single-graph datasets contain Reddit, Weibo, Amazon, Yelp, Tolokers, Questions, and T-finance. The multi-graph datasets contain BM-MN, BM-MS, BM-MT, MUTAG, MNIST0, MNIST1, and T-Group. For node- and edge-level detection on single-graph datasets, we introduce eight state-of-the-art unsupervised baselines, including AutoGAD (Yu et al., 2025), HEC-GAD (Hu et al., 2025), SGAD-DCR (Chen et al., 2026c), CVGAD (Jin et al., 2025), AD-GCL (Xu et al., 2025), UniFORM (Song et al., 2025), BOURNE (Liu et al., 2024), and GRAND (Wang et al., 2026), where UniFORM, BOURNE, and GRAND are multi-granularity GAD methods, and others are traditional node-level GAD methods. For node- and graph-level detection on multi-graph datasets, we introduce seven advanced unsupervised baselines, including RQGNN (Dong et al., 2024), GUL (Yang et al., 2026), MUSE (Kim et al., 2024), GLAD (Cai et al., 2025), GLADPro (Yang et al., 2025b), UniFORM (Song et al., 2025), BOURNE (Liu et al., 2024), and GRAND (Wang et al., 2026), where RQGNN, GUL, MUSE, GLAD, and GLADPro are traditional graph-level GAD methods. More details about datasets and baselines are in Section A.

5.2 Implementations

Every experiment is conducted 5 times, and the average performance is recorded with standard deviation. The hyperparameters d', ϵ, θ, K_c, K, and K_f are, respectively, set to 64, 1, 1, 128, 10, and 5. The training epoch is set to 500. The learning rate is 1e-4. The supervised baselines are extended to unsupervised settings by reconstruction using the latent representations. All methods are executed in a Python 3.9.12 environment, utilizing a single NVIDIA A40 GPU, 40GB of RAM, and a 2.60GHz Xeon (R) Gold 6240 CPU. We utilize three metrics to evaluate the performance: AUROC, AUPRC, and F1-score. More details are shown in Section B.

5.3 Experimental Results and Analysis

Multi-granularity Detection Performance. To validate the effectiveness of QNMGAD, we implement multi-granularity anomaly detection under two settings. First, we implement node- and edge-level anomaly detection on single-graph datasets. From the observations in Table 1, we discover that

QNMGAD achieves the top AUROC and AUPRC compared with state-of-the-art baselines, including task-specific models and all multi-granularity GAD methods, indicating that QNMGAD can capture more granularity-specific semantic information and has better discrimination across granularity. For example, QNMGAD exhibits AUROC improvements of at least 3.6%, 8.9%, 7.8%, 3.3%, 1.3%, 1.4%, 4.6%, respectively, on Reddit, Weibo, Amazon, Yelp, Tolokers, Questions, and T-Finance datasets with respect to the node-level task. Second, we conduct node- and graph-level tasks on multi-graph datasets. From Table 2, QNMGAD consistently outperforms baselines, and QNMGAD demonstrates an average AUPRC improvement of at least 7.29% on the graph-level task. The comprehensive advancement of QNMGAD can be attributed to two-fold reasons. First, QNMGAD explores the discrepancies and inconsistencies between granularities, preventing the semantic overlap across different granularities. Second, QNMGAD captures granularity-specific semantics using residual pattern quantization that interacts with prototype concepts to the discriminability of abnormal patterns within each granularity, and utilizes conditioned normalizing flows based on prototypes to model the real posterior data distribution with granularity perception.

Table 3: The transferability of node- and edge-level GAD on a single dataset on Reddit, Weibo, and Amazon.

	Dataset	Reddit		Weibo		Amazon	
	Metrics	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
N→E	UniFORM	50.8 ± 1.5	4.5 ± 1.3	71.9 ± 1.7	11.8 ± 1.9	68.3 ± 1.6	10.6 ± 1.4
	BOURNE	52.1 ± 1.4	4.8 ± 1.8	72.4 ± 1.6	13.2 ± 1.4	65.2 ± 1.5	8.0 ± 1.3
	GRAND	47.6 ± 1.3	3.1 ± 1.2	70.0 ± 1.5	11.5 ± 0.9	66.2 ± 1.4	9.3 ± 1.0
	QNMGAD	**56.2** ± 0.9	**6.2** ± 1.1	**82.8** ± 0.8	**28.5** ± 0.7	**78.5** ± 1.3	**20.6** ± 1.0
E→N	UniFORM	46.5 ± 1.6	2.5 ± 1.2	70.8 ± 1.8	12.6 ± 0.6	76.1 ± 1.7	25.8 ± 1.3
	BOURNE	45.3 ± 1.5	2.2 ± 1.0	66.6 ± 1.7	7.8 ± 1.0	71.4 ± 1.6	17.1 ± 1.2
	GRAND	48.9 ± 1.4	3.4 ± 1.3	72.2 ± 1.6	14.3 ± 1.1	72.5 ± 1.5	17.7 ± 0.6
	QNMGAD	**53.7** ± 0.7	**6.7** ± 0.4	**80.9** ± 0.7	**30.8** ± 1.2	**80.5** ± 0.6	**25.4** ± 0.6

Transferability. We implement a transferability experiment across different specific tasks compared with multi-granularity GAD baselines. Table 3 demonstrates the transferability of QNMGAD on single-graph datasets from node level to edge level detection and from edge level to node level detection. Table 4 demonstrates the transferability of QNMGAD on multi-graph datasets from node level to graph level detection and from graph level to node level detection. Notably, QNMGAD has superior transferability compared with the corresponding baselines. Experiment results on other datasets are shown in Section C.

Table 4: The transferability of node- and graph-level GAD on multi-graph datasets on BM-MN, BM-MS, and BM-MT.

	Dataset	BM-MN		BM-MS		BM-MT	
	Metrics	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
N→G	UniFORM	51.0 ± 1.5	4.4 ± 1.2	72.1 ± 1.7	14.9 ± 1.9	68.4 ± 1.6	11.5 ± 1.4
	BOURNE	52.6 ± 1.4	4.8 ± 1.8	72.6 ± 1.6	16.1 ± 1.4	71.0 ± 1.5	18.1 ± 1.3
	GRAND	53.2 ± 1.3	5.2 ± 1.3	73.2 ± 1.5	18.6 ± 0.9	74.0 ± 1.4	22.1 ± 1.0
	QNMGAD	**56.4** ± 0.9	**6.1** ± 1.1	**76.9** ± 0.8	**20.6** ± 0.7	**78.6** ± 1.3	**25.4** ± 1.0
G→N	UniFORM	46.7 ± 1.6	3.6 ± 1.2	73.6 ± 1.8	24.7 ± 0.6	76.3 ± 1.7	28.6 ± 1.3
	BOURNE	45.1 ± 1.5	3.4 ± 1.0	70.8 ± 1.7	22.7 ± 1.0	71.2 ± 1.6	25.0 ± 1.2
	GRAND	49.1 ± 1.4	4.5 ± 1.3	74.1 ± 1.6	25.4 ± 1.1	72.3 ± 1.5	26.6 ± 0.6
	QNMGAD	**53.5** ± 0.7	**6.6** ± 0.4	**78.7** ± 0.7	**27.9** ± 1.2	**80.3** ± 0.6	**32.2** ± 0.6

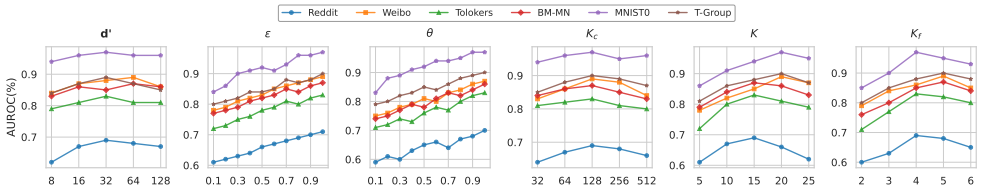


Figure 3: The sensitivity experiments on six selected baselines with respect to AUROC.

Sensitivity. We explore the sensitivity of hyperparameters, including d', ϵ, θ, K_c, K, and K_f. Fig. 3 exhibits the performance of node-level GAD tasks on some single- and multi-graph datasets. For d', excessively small values may lead to loss of semantic representation, while overly large values can introduce semantic redundancy. Empirically, settings such as 32 or 64 are relatively appropriate. For the loss-function hyperparameters ϵ and θ, they control the model's learning capacity; therefore, relatively larger values can enhance the model's ability to capture semantic information. K_c and K influence the discriminability of semantics across granularities, and overly small values may result in weak granularity distinc-

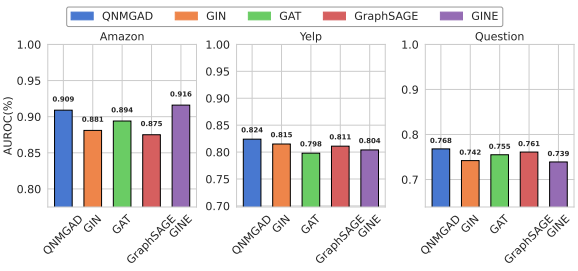


Figure 4: The utility of different graph encoders with respect to AUROC.

tion. K_f reflects the capability of the normalizing flow to fit the posterior distribution, and complex distributions typically require more layers to achieve convergence. More experiments are shown in Section E.

Components Utility. We progressively validate the utility of components in QNMGAD, including prototype quantization (PQ), residual pattern quantization (RPQ), and the prototype condition for normalizing flows (PC). The single and double ablations on three datasets are shown in Table 5. All variants show performance degradation compared with the baseline (ID.6). Double ablation variants (ID.0-ID.2) achieve relatively poor performance compared with single ablation variants (ID.3-ID.5). In addition, PQ shows a significant contribution to performance because, in the single ablation study, ID. 4 obtains the worst performance. In addition, compared with PQ and PC, the contribution of RPQ is relatively less. For example, performance has less degradation when RPQ is removed compared to other single ablation variants. More ablations are shown in Section D.

Table 5: The ablation study results on node- and edge-level GAD on single-graph datasets.

ID	PQ	RPQ	PC	Reddit		Weibo		Amazon	
				Node	Edge	Node	Edge	Node	Edge
0	✓			57.3 ± 1.5	55.1 ± 1.8	77.1 ± 1.7	80.6 ± 2.0	73.5 ± 1.3	79.6 ± 1.8
1		✓		55.1 ± 1.6	54.7 ± 2.0	76.0 ± 1.6	78.2 ± 1.9	71.8 ± 1.7	77.4 ± 1.5
2			✓	59.2 ± 1.5	56.4 ± 1.7	79.6 ± 1.5	81.4 ± 1.3	77.2 ± 1.6	80.2 ± 1.7
3	✓	✓		61.9 ± 1.0	59.9 ± 1.2	85.1 ± 1.1	84.3 ± 1.2	81.9 ± 1.7	81.6 ± 1.1
4		✓	✓	61.7 ± 1.4	57.9 ± 1.9	84.9 ± 1.7	83.2 ± 1.5	80.9 ± 1.4	84.1 ± 1.5
5	✓		✓	63.6 ± 1.8	60.7 ± 1.2	87.5 ± 1.0	87.6 ± 1.0	84.4 ± 1.1	86.5 ± 1.3
6	✓	✓	✓	**69.5** ± 0.8	**65.8** ± 0.5	**89.4** ± 0.7	**90.4** ± 0.6	**90.9** ± 1.2	**92.2** ± 0.5

Graph Encoder. We explore different graph encoder ($\mathcal{F}_2$), including GIN (Xu et al., 2019), GAT (Velickovic et al., 2018), GraphSAGE (Hamilton et al., 2017), and GINE (Hu et al., 2020). From the observations in Fig. 4, we discover that deploying different graph encoders results in performance degradation. For example, GCN brings an average AUROC improvement of 1.88% on the Question dataset. However, GINE achieves the best performance on the Amazon dataset. More experiments are in Section F.

Modules Strategies. We validate the strategies in modules. QNMGAD$_{f_1}$ denotes using only an MLP to encode the graph. QNMGAD$_{f_2}$ denotes using only a GCN to encode the graph. QNMGAD$_{mean}$ denote generating $\bar{\mathbf{c}}_{pr}$ using mean value. QNMGAD$_r$ denote removing residual quantization $\mathbf{r}^L$ and $\hat{\mathbf{c}}_2^L$ in RPQ. QNMGAD$_{int}$ denote removing the interaction with $\bar{\mathbf{c}}_{pr}$ in Eq. (10). The observations in Fig. 5 demonstrate the significant performance degradation of the variants, validating the effectiveness of the strategies employed in the modules. Specifically, using local representations with MLP and interactive representations with GCN can enhance the performance. The Top-K strategy effectively boosts the discrimination of granularities. Besides, the designs in RPQ help to capture semantic information.

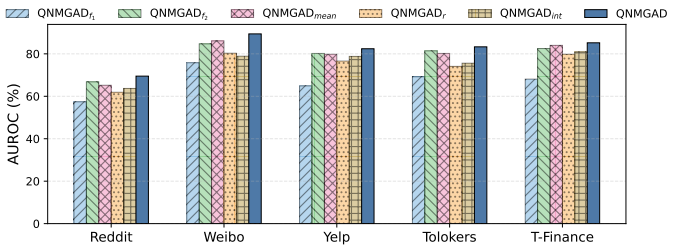


Figure 5: The utility of different graph encoders with respect to AUROC.

Other experiments. To validate the effectiveness of normalizing flow, we implement other encoding tools as variants, and the results are shown in Section H. We implement robustness experiments to validate the model's ability to resist noise in Section G. We implement scalability experiments in Section I to validate the performance with different training rates. Efficiency experiments are in Section J.

6 Conclusion

This paper explores the inconsistencies between granularities for multi-granularity GAD and proposes a quantization-conditioned normalizing flow framework for multi-granularity GAD (QNMGAD). QNMGAD utilizes quantization strategies to portray the discrepancies across granularities and extract granularity-specific semantics. QNMGAD models the posterior data distribution of normal patterns using conditioned normalizing flows and evaluates the likelihood estimation as anomaly scores. Extensive experiments on comprehensive public datasets exhibit the superiority of QNMGAD over state-of-the-art baselines. In the future, we will advance the scalability and robustness of multi-granularity GAD.

AI USE STATEMENT

During manuscript preparation, the authors used ChatGPT solely for grammatical correction and English-language polishing. The tool was not used for research design, methodological development, code generation, experimentation, data analysis, or the generation of scientific results or conclusions. The authors take full responsibility for the content, accuracy, and integrity of the manuscript.

ETHICS STATEMENT

This study does not involve human subjects, practices related to dataset releases, potentially harmful insights, methodologies and applications, potential conflicts of interest and sponsorship, discrimination/bias/fairness concerns, privacy and security issues, legal compliance, and research integrity issues (e.g., IRB, documentation, research ethics).

REPRODUCIBILITY STATEMENT

Sections 3 and 4 define the problem, the proposed framework, and the corresponding objective. The supplement specifies more details about experimental configuration and results. All evaluated datasets are public and cited; no private data are used.

REFERENCES

Stefan Bloemheuvel, Jurgen van den Hoogen, and Martin Atzmueller. A computational framework for modeling complex sensor network data using graph signal processing and graph neural networks in structural health monitoring. *Applied Network Science*, 6(1):97, 2021.

Jinyu Cai, Yunhe Zhang, Fusheng Liu, and See-Kiong Ng. Leveraging diffusion model as pseudo-anomalous graph generator for graph-level anomaly detection. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Zm2M92TZyO.

Qian Chen, Huiying Xu, Ruidong Wang, Yue Liu, and Xinzhong Zhu. Structure adversarial augmented graph anomaly detection via multi-view contrastive learning. *Knowl. Based Syst.*, 338: 115455, 2026a. doi: 10.1016/J.KNOSYS.2026.115455.

Yang Chen, Xiaowei Xu, Shuai Wang, Chenhui Zhu, Ruxue Wen, Xubin Li, Tiezheng Ge, and Limin Wang. Flowing backwards: Improving normalizing flows via reverse representation alignment. In *Fortieth AAAI Conference on Artificial Intelligence*, pp. 3074–3082, 2026b. doi: 10.1609/AAAI.V40I4.37300.

Yujia Chen, Jian Li, Linfei Sun, Tongcun Liu, and Guanjun Liu. Semi-supervised graph anomaly detection via dual-channel reconstruction. *Neurocomputing*, 668:132401, 2026c. doi: 10.1016/J.NEUCOM.2025.132401.

Leyan Deng, Defu Lian, Zhenya Huang, and Enhong Chen. Graph convolutional adversarial networks for spatiotemporal anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2416–2428, 2022.

Xiangyu Dong, Xingyi Zhang, and Sibo Wang. Rayleigh quotient graph neural networks for graph-level anomaly detection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4UIBysXjVq.

Guanghan Duan, Hongwu Lv, Huiqiang Wang, Guangsheng Feng, and Xiaoli Li. Practical cyber attack detection with continuous temporal graph in dynamic network system. *IEEE Transactions on Information Forensics and Security*, 19:4851–4864, 2024.

Haoyi Fan, Fengbin Zhang, and Zuoyong Li. Anomalydae: Dual autoencoder for anomaly detection on attributed networks. In *the Proceeding of ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5685–5689. IEEE, 2020.

Mengnan Gao, Lifa Wu, Qi Li, and Wei Chen. Anomaly traffic detection in iot security using graph neural networks. *Journal of Information Security and Applications*, 76:103532, 2023.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pp. 1024–1034, 2017.

Renda Han, Xiaobao Wang, Luzhi Wang, Wenxin Zhang, Guangzhen Yao, and Hongxiang Liang. A graph foundation model for unified anomaly detection. In *Proceedings of the ACM Web Conference 2026*, pp. 523–534, 2026. doi: 10.1145/3774904.3792108.

Jingtao Hu, Siwei Wang, Jingcan Duan, Hu Jin, Xinwang Liu, and En Zhu. Higher-order enhanced contrastive-based graph anomaly detection without graph augmentation. *Pattern Recognition*, 167:111666, 2025. URL https://doi.org/10.1016/j.patcog.2025.111666.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay S. Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *8th International Conference on Learning Representations*, 2020.

Di Jin, Jingyi Cao, Xiaobao Wang, Bingdao Feng, Dongxiao He, Longbiao Wang, and Jianwu Dang. Rethinking contrastive learning in graph anomaly detection: a clean-view perspective. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 3009–3017, 2025. URL https://doi.org/10.24963/ijcai.2025/335.

Sunwoo Kim, Soo Yong Lee, Fanchen Bu, Shinhwan Kang, Kyungho Kim, Jaemin Yoo, and Kijung Shin. Rethinking reconstruction-based graph-level anomaly detection: Limitations and a simple remedy. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024*, 2024.

Yiqing Lin, Jianheng Tang, Chenyi Zi, H Vicky Zhao, Yuan Yao, and Jia Li. Unigad: Unifying multi-level graph anomaly detection. *Advances in neural information processing systems*, 37: 136120–136148, 2024.

Jie Liu, Mengting He, Xuequn Shang, Jieming Shi, Bin Cui, and Hongzhi Yin. Bourne: Bootstrapped self-supervised learning framework for unified graph anomaly detection. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pp. 2820–2833. IEEE, 2024.

Iftekhar Rasul, SM Iftekhar Shaboj, Mainuddin Adel Rafi, Md Kauser Miah, Md Redwanul Islam, and Abir Ahmed. Detecting financial fraud in real-time transactions using graph neural networks and anomaly detection. *Journal of Economics, Finance and Accounting Studies*, 6(1):131–142, 2024.

Chuancheng Song, Xixun Lin, Hanyang Shen, Yanmin Shang, and Yanan Cao. Uniform: Towards unified framework for anomaly detection on graphs. In *the Association for the Advancement of Artificial Intelligence*, pp. 12559–12567, 2025. URL https://doi.org/10.1609/aaai.v39i12.33369.

Petar Velickovic, Guillem Cucurull, Alberto Casanova, Javier Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, 2018.

Maolin Wang, Beining Bao, Hongyu Chen, Zichun Liu, Lang Fu, Jun Chu, Langzhang Liang, and Zenglin Xu. GRAND: A robust diffusion framework for multi-granularity graph anomaly detection in web platforms. In *Proceedings of the ACM Web Conference 2026*, pp. 9572–9580, 2026. doi: 10.1145/3774904.3793012.

Chunyu Wei, Siyuan He, Yu Wang, Yueguo Chen, Yunhai Wang, Bing Bai, Yidong Zhang, Yong Xie, Shunming Zhang, and Fei Wang. Balanced anomaly-guided ego-graph diffusion model for inductive graph anomaly detection. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1542–1553, 2026. doi: 10.1145/3770854.3780240.

Chunjing Xiao, Yao Tong, Jiahui Lu, Yuxia Xue, Wenxin Tai, Zhangtao Cheng, and Fan Zhou. Graph-enhanced multi-scale contrastive learning for graph anomaly detection with adaptive diffusion models. *IEEE Trans. Big Data*, 11(6):2946–2957, 2025.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations*, 2019.

Yiming Xu, Zhen Peng, Bin Shi, Xu Hua, Bo Dong, Song Wang, and Chen Chen. Revisiting graph contrastive learning on anomaly detection: a structural imbalance perspective. In *the Association for the Advancement of Artificial Intelligence*, pp. 12972–12980, 2025. URL https://doi.org/10.1609/aaai.v39i12.33415.

Chengdong Yang, Hongrui Liu, Daixin Wang, Zhiqiang Zhang, Cheng Yang, and Chuan Shi. FLAG: fraud detection with llm-enhanced graph neural network. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5150–5160, 2025a.

Zhenyu Yang, Ge Zhang, Jia Wu, Jian Yang, Shan Xue, Amin Beheshti, Hao Peng, and Quan Z. Sheng. Global interpretable graph-level anomaly detection via prototype. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3586–3597, 2025b.

Zhenyu Yang, Ge Zhang, Shan Xue, Xiaoxiao Ma, Jian Yang, Hao Peng, Amin Beheshti, and Jia Wu. Revisiting graph-level anomaly detection: From partially to fully unsupervised learning. In *Proceedings of the ACM Web Conference 2026*, pp. 901–912, 2026. doi: 10.1145/3774904.3792307.

Jiaqi Yu, Yang Gao, Hong Yang, Zhihong Tian, Peng Zhang, and Xingquan Zhu. Automated graph anomaly detection with large language models. *Knowledge-based Systems*, 324:113809, 2025. URL https://doi.org/10.1016/j.knosys.2025.113809.

Ge Zhang, Zhao Li, Jiaming Huang, Jia Wu, Chuan Zhou, Jian Yang, and Jianliang Gao. efraudcom: An e-commerce fraud detection system via competitive graph neural networks. *ACM Transactions on Information Systems (TOIS)*, 40(3):1–29, 2022.

Muhan Zhang. Graph neural networks: link prediction. In *Graph Neural Networks: Foundations, Frontiers, and Applications*, pp. 195–223. Springer, 2022.

Wenxin Zhang and Cuicui Luo. GE-GNN: gated edge-augmented graph neural network for fraud detection. *IEEE Trans. Big Data*, 11(4):1664–1676, 2025.

Wenxin Zhang and Cuicui Luo. Mclast: Multi-hierarchy contrastive learning graph anomaly detection with structure-awareness. *Neurocomputing*, 669:132480, 2026.

Jiawen Zhu, Choubo Ding, Yu Tian, and Guansong Pang. Anomaly heterogeneity learning for open-set supervised anomaly detection. In *the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17616–17626, 2024.

A Datasets and Baselines

First, the statistics of datasets are shown in Table 6. Second, we introduce the details of baselines:

- AutoGAD first designs a novel search space tailored for GAD tasks based on the characteristics of the graph autoencoder framework and then leverages a large language model as the controller of GNAS, guiding the LLM to rapidly generate architectures suitable for GAD within the search space through well-designed prompts.
- HEC-GAD samples from ego-net and expands it to high-order neighbors to generate a multi-view subgraph of the target node. To more efficiently mine anomalies in topology and attribute information, HEC-GAD conducts intra-view node-subgraph and inter-view subgraph-subgraph contrastive learning, and identifies abnormal nodes through the reconstruction error of nodes.
- SGAD-DCR employs dual-filter attention encoder to construct dual aggregation channels for low-frequency and high-frequency signals and mitigates the homophily trap via semantic-level alignment.

Table 6: Detailed statistics of the datasets used in our experiments.

Dataset	Train%	# Graphs	# Edges	# Nodes	# Dims	Nodes_{ab}	Edges_{ab}	Graphs_{ab}
Reddit	40%	1	168,016	10,984	64	3.33%	2.72%	/
Weibo	40%	1	416,368	8,405	400	10.33%	5.71%	/
Amazon	50%	1	8,847,096	11,944	25	6.87%	2.49%	/
Yelp	50%	1	7,739,912	45,954	32	14.53%	13.89%	/
Tolokers	50%	1	530,758	11,758	10	21.82%	33.44%	/
Questions	50%	1	202,461	48,921	301	2.98%	7.50%	/
T-Finance	40%	1	21,222,543	39,357	10	4.58%	2.77%	/
BM-MN	40%	700	40,032	12,911	1	48.91%	/	14.29%
BM-MS	40%	700	30,238	9,829	1	31.99%	/	14.29%
BM-MT	40%	700	32,042	10,147	1	34.49%	/	14.29%
MUTAG	40%	2,951	179,732	88,926	14	4.81%	/	34.40%
MNIST0	10%	70,000	41,334,380	4,939,668	5	35.46%	/	9.86%
MNIST1	10%	70,000	41,334,380	4,939,668	5	35.46%	/	11.25%
T-Group	40%	37,402	93,367,082	11,015,616	10	0.64%	/	4.26%

- CVGAD includes a multi-scale anomaly awareness module to identify key sources of interference in the contrastive learning process. Moreover, to mitigate bias from the one-step edge removal process, CVGAD introduces a novel progressive purification module, which incrementally refines the graph by iteratively identifying and removing interfering edges.
- AD-GCL devises the neighbor pruning strategy to filter noisy edges for head nodes and facilitate the detection of genuine tail nodes by aligning from head nodes to forged tail nodes. Moreover, AD-GCL actively explores potential neighbors to enlarge the receptive field of tail nodes through anomaly-guided neighbor completion.
- UniFORM unifies node-, edge-, and graph-level tasks from a subgraph perspective, leveraging an energy-based GNN for iterative multi-granular anomaly detection, and enhances meta-learning through contrastive learning and employs Langevin dynamics to generate phantom samples as substitutes for anomalous data, reducing reliance on labeled data.
- BOURNE extracts a subgraph centered on each target node as node context and transforms it into a dual hypergraph (hypergraph view) as edge context, and enables the mutual detection of node and edge anomalies by swapping the context embeddings between nodes and edges and measuring the agreement in the embedding space.
- GRAND introduces a dual-diffusion strategy that contains continuous Gaussian diffusion for node features and discrete diffusion for edges to detect the subtle anomaly signals typical of web data.
- RQGNN explores the inherent spectral features of anomalous graphs for graph-level anomaly detection, comprised of the Rayleigh Quotient learning component and Chebyshev Wavelet GNN with RQ-pooling.
- GUL evaluates the uncertainty through Gaussian distributions over the detectors' predictions and optimizes anomaly detection and uncertainty modeling via an Expectation-Maximization algorithm.
- MUSE takes multifaceted summaries of reconstruction errors as graph features for graph-level anomaly detection, reports non-trivial counter-examples, a phenomenon we call reconstruction flip, and highlights the limitations of the existing Graph-AE-based methods.
- GLAD leverages the latent diffusion framework to incorporate subtle perturbations into graph representations, and jointly trains a classifier to distinguish these generated graph anomalies from normal graphs.
- GLADPro incorporates prototype learning with the information bottleneck principle, enabling prototypes to capture the most significant subgraph patterns as global-level explanations through persistent interactions with key subgraphs from input graphs.

B IMPLEMENTATION

For edge-level anomalies, we generate abnormal edges based on the probability of nodes $p(e_{ij}) = \text{Avg}(p_i, p_j)$ (Lin et al., 2024). We ignore edges between the training nodes and the testing nodes

Table 7: The transferability of node- and edge-level GAD on a single dataset on Yelp, Tolokers, Questions, and T-Finance.

	Dataset	Yelp		Tolokers		Questions		T-Finance	
	Metrics	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
N→ E	UniFORM	58.2 ± 1.8	7.3 ± 1.2	66.8 ± 1.7	18.5 ± 0.9	56.1 ± 1.6	10.1 ± 0.6	70.5 ± 1.5	20.8 ± 0.7
	BOURNE	51.5 ± 1.7	4.8 ± 0.9	65.9 ± 1.6	15.7 ± 0.9	51.9 ± 1.5	8.7 ± 1.2	56.4 ± 1.4	8.5 ± 1.8
	GRAND	54.8 ± 1.6	6.1 ± 0.7	64.1 ± 1.5	14.8 ± 1.6	58.5 ± 1.4	10.9 ± 0.9	63.8 ± 1.3	15.9 ± 1.7
	QNMGAD	**65.9** ± 0.9	**10.1** ± 0.6	**72.4** ± 0.8	**30.9** ± 0.6	**64.7** ± 0.7	**15.1** ± 0.7	**74.3** ± 1.1	**27.4** ± 0.6
E→N	UniFORM	51.8 ± 1.9	3.9 ± 1.2	57.4 ± 1.8	10.6 ± 1.4	54.2 ± 1.7	4.2 ± 1.6	61.1 ± 1.6	14.5 ± 1.3
	BOURNE	60.9 ± 1.8	9.6 ± 0.9	60.3 ± 1.7	14.5 ± 1.3	55.4 ± 1.6	5.0 ± 1.7	56.2 ± 1.5	10.1 ± 2.0
	GRAND	59.6 ± 1.7	7.1 ± 0.6	59.8 ± 1.6	12.3 ± 1.5	58.1 ± 1.5	10.4 ± 1.8	65.0 ± 1.4	16.0 ± 1.2
	QNMGAD	**67.1** ± 0.8	**14.9** ± 0.6	**66.8** ± 0.7	**25.7** ± 1.0	**65.2** ± 0.6	**16.1** ± 0.6	**73.1** ± 0.6	**27.6** ± 0.8

Table 8: The transferability of node- and graph-level GAD on multi-graph datasets on MUTAG, MNIST0, MNIST1, and T-Group.

	Dataset	MUTAG		MNIST0		MNIST1		T-Group	
	Metrics	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
N→ G	UniFORM	61.0 ± 1.8	15.4 ± 1.3	66.2 ± 1.7	25.1 ± 0.9	56.4 ± 1.6	10.2 ± 0.6	70.6 ± 1.5	24.6 ± 0.7
	BOURNE	60.4 ± 1.7	13.7 ± 1.0	65.4 ± 1.6	20.5 ± 0.9	52.1 ± 1.5	8.9 ± 1.2	56.7 ± 1.4	13.4 ± 1.8
	GRAND	64.6 ± 1.6	18.4 ± 0.7	63.9 ± 1.5	23.6 ± 1.6	61.7 ± 1.4	13.5 ± 0.9	63.6 ± 1.3	18.8 ± 1.7
	QNMGAD	**70.7** ± 0.9	**30.0** ± 0.6	**72.1** ± 0.8	**28.7** ± 0.6	**64.9** ± 0.7	**15.0** ± 0.7	**74.1** ± 1.1	**27.2** ± 0.6
G→N	UniFORM	51.9 ± 1.9	3.8 ± 1.2	57.2 ± 1.8	17.5 ± 1.4	60.1 ± 1.7	18.3 ± 1.6	61.0 ± 1.6	16.4 ± 1.3
	BOURNE	60.7 ± 1.8	12.4 ± 0.9	60.1 ± 1.7	19.3 ± 1.3	61.3 ± 1.6	18.8 ± 1.7	56.1 ± 1.5	14.0 ± 2.0
	GRAND	59.5 ± 1.7	10.0 ± 0.6	59.6 ± 1.6	18.1 ± 1.5	62.0 ± 1.5	20.3 ± 1.8	64.8 ± 1.4	20.8 ± 1.2
	QNMGAD	**66.9** ± 0.8	**21.8** ± 0.6	**66.5** ± 0.7	**25.6** ± 1.0	**65.0** ± 0.6	**23.0** ± 0.6	**72.9** ± 0.6	**27.5** ± 0.8

to prevent data leakage; for graph-level anomaly detection, we construct training and testing sets with different graphs and corresponding nodes. Considering the limited studies for the edge-level detection, we integrate a concatenated strategy (Zhang, 2022) based on link prediction into the above baselines.

C TRANSFERABILITY

We implement additional transferability experiments on the remaining datasets. Results are shown in Table 7 and Table 8. It can be seen that QNMGAD achieves the best transferability performance on all datasets with respect to different transfer tasks, including node level to edge level, edge level to node level, node level to graph level, and graph level to node level, demonstrating the effectiveness and advancement of QNMGAD.

D ABLATION STUDY

We exhibit the ablation study on the rest of the datasets with respect to comprehensive multi-granularity GAD tasks. Table 9 shows the performance of Tolokers, Questions, and T-Finance datasets on node- and edge-level GAD. Table 10 shows the performance of BM-MN, BM-MS, BM-MT, MUTAG, MNIST0, MNIST1, and T-Group on node- and graph-level GAD. From these results, we can see that QNMGAD achieves the best performance compared with all ablation variants consistently, which demonstrates the effectiveness of each component in QNMGAD.

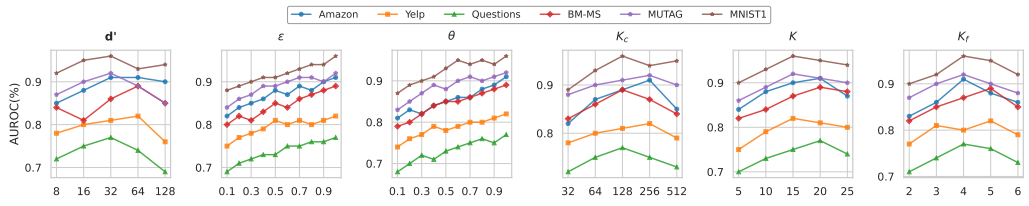


Figure 6: The extensive sensitivity experiments at node-level GAD with respect to AUROC.

Table 9: The ablation study results on node- and edge-level GAD across four datasets.

ID	PQ	RPQ	PC	Yelp		Tolokers		Questions		T-Finance	
				Node	Edge	Node	Edge	Node	Edge	Node	Edge
0	✓			76.9 ± 0.9	73.6 ± 1.0	76.5 ± 0.9	73.9 ± 1.0	72.4 ± 0.9	72.8 ± 0.9	79.8 ± 0.9	78.4 ± 0.9
1		✓		75.2 ± 1.0	73.1 ± 1.1	75.8 ± 1.0	73.0 ± 1.1	71.6 ± 1.0	72.0 ± 1.1	79.1 ± 1.0	77.8 ± 1.0
2			✓	74.8 ± 1.1	72.6 ± 1.2	75.2 ± 1.2	72.8 ± 1.2	70.9 ± 1.2	71.5 ± 1.3	78.6 ± 1.0	77.1 ± 1.1
3	✓	✓		76.9 ± 1.5	74.8 ± 1.6	79.0 ± 1.6	76.6 ± 1.7	73.5 ± 1.7	74.1 ± 1.8	80.4 ± 1.5	79.0 ± 1.6
4		✓	✓	77.6 ± 1.4	75.1 ± 1.5	79.5 ± 1.5	76.8 ± 1.6	74.2 ± 1.5	74.6 ± 1.7	81.0 ± 1.4	79.6 ± 1.5
5	✓		✓	78.8 ± 1.2	76.0 ± 1.4	80.1 ± 1.3	77.4 ± 1.4	75.3 ± 1.4	75.8 ± 1.5	82.1 ± 1.2	80.7 ± 1.3
6	✓	✓	✓	$\mathbf{82.4} \pm 0.8$	$\mathbf{78.5} \pm 0.7$	$\mathbf{83.3} \pm 0.7$	$\mathbf{78.4} \pm 0.6$	$\mathbf{76.8} \pm 0.6$	$\mathbf{76.8} \pm 0.5$	$\mathbf{85.2} \pm 1.0$	$\mathbf{83.5} \pm 0.5$

Table 10: The ablation study results on node- and graph-level GAD across seven datasets.

ID	PQ	RPQ	PC	BM-MN		BM-MS		BM-MT		MUTAG		MNIST0		MNIST1		T-Group	
				Node	Graph	Node	Graph	Node	Graph	Node	Graph	Node	Graph	Node	Graph	Node	Graph
0	✓			80.1 ± 1.1	79.4 ± 1.0	77.8 ± 1.2	78.1 ± 1.3	78.5 ± 1.1	79.2 ± 1.2	91.2 ± 0.8	91.6 ± 0.7	84.8 ± 1.0	85.3 ± 0.9	85.6 ± 0.9	86.2 ± 0.9	82.1 ± 1.0	82.8 ± 1.1
1		✓		77.8 ± 1.3	76.9 ± 1.2	74.6 ± 1.4	75.2 ± 1.5	75.9 ± 1.3	76.5 ± 1.4	88.1 ± 0.9	88.7 ± 0.8	81.9 ± 1.2	82.6 ± 1.1	83.4 ± 1.1	84.0 ± 1.0	79.2 ± 1.2	80.0 ± 1.3
2			✓	78.3 ± 1.4	77.5 ± 1.5	75.1 ± 1.6	74.4 ± 1.4	76.4 ± 1.5	75.8 ± 1.6	88.6 ± 1.0	89.0 ± 0.9	82.3 ± 1.3	82.9 ± 1.2	83.8 ± 1.2	84.2 ± 1.1	79.8 ± 1.4	80.5 ± 1.5
3	✓	✓		83.6 ± 0.9	83.1 ± 0.9	81.2 ± 1.0	80.6 ± 1.1	81.5 ± 1.0	82.0 ± 0.9	93.1 ± 0.6	93.6 ± 0.6	87.1 ± 0.8	87.6 ± 0.8	88.0 ± 0.8	88.6 ± 0.7	84.6 ± 0.9	85.1 ± 0.8
4		✓	✓	83.1 ± 1.0	82.4 ± 0.9	80.7 ± 1.1	80.1 ± 1.2	80.9 ± 1.1	81.4 ± 1.0	92.6 ± 0.7	93.0 ± 0.7	86.5 ± 0.9	87.1 ± 0.8	87.4 ± 0.8	88.0 ± 0.8	84.0 ± 0.9	84.6 ± 0.9
5	✓		✓	84.5 ± 0.7	85.0 ± 0.7	82.0 ± 0.8	82.6 ± 0.9	83.1 ± 0.8	83.7 ± 0.8	94.6 ± 0.5	95.0 ± 0.5	88.6 ± 0.7	89.2 ± 0.6	89.8 ± 0.6	90.3 ± 0.6	86.2 ± 0.7	86.8 ± 0.7
6	✓	✓	✓	$\mathbf{87.0} \pm 0.6$	$\mathbf{85.1} \pm 0.6$	$\mathbf{89.0} \pm 0.7$	$\mathbf{84.2} \pm 0.7$	$\mathbf{80.1} \pm 0.6$	$\mathbf{85.4} \pm 0.6$	$\mathbf{92.2} \pm 0.5$	$\mathbf{94.5} \pm 0.9$	$\mathbf{97.0} \pm 0.5$	$\mathbf{91.1} \pm 1.1$	$\mathbf{96.2} \pm 0.5$	$\mathbf{90.0} \pm 0.8$	$\mathbf{90.3} \pm 0.6$	$\mathbf{85.8} \pm 0.6$

E SENSITIVITY

Hyperparameters We implement comprehensive hyperparameter sensitivity experiments. Fig. 6 shows the AUROC performance at node-level GAD with some other single- and multi-graph datasets, including Amazon, Yelp, Questions, BM-MS, MUTAG, and MNIST1. Fig. 7 shows the AUROC performance at edge-level GAD with single-graph datasets (Amazon, Yelp, and Questions) and at graph-level with multi-graph datasets (BM-MS, MUTAG, and MNIST1). From the extensive sensitivity experiments, we can more explicitly explore the influence of each hyperparameter on the performance.

Joint Effect We further test the joint effect of K_c and K, as shown in Fig. 8. Two important conclusions can be drawn from these results. First, even within the same task granularity (e.g., node-level GAD), the interaction between the two hyperparameters exhibits noticeable variations across different datasets. Second, for the same dataset, tasks at different granularities require different optimal hyperparameter configurations. These findings further validate the existence of cross-granularity inconsistency.

F UTILITY OF ENCODER

We further validate the utility of encoders on Reddit, Weibo, Tolokers, BM-MS, BM-MT, and T-Group datasets as shown in Fig. 9. We can observe the varying performance with different graph encoders. First, using different encoders may result in performance fluctuation according to specific tasks. For example, on the Reddit dataset, using GCN achieves the best performance with respect to node-level detection, while using GINE achieves the best performance with respect to edge-level detection. Second, encoders have distinct performance on different datasets. For example, with respect to node-level GAD, GCN achieves the best performance on the Weibo dataset, while GIN outperforms other encoders on the Tolokers dataset.

G ROBUSTNESS

We validate the robustness of QNMGAD compared with state-of-the-art multi-granularity GAD baselines by adding standard Gaussian noise to samples at different rates. The experiments are conducted on node-level detection with single-graph datasets (Reddit, Weibo, and Tolokers) and on graph-level detection with multi-graph datasets (BM-MN, MNIST0, and T-Group). For single-graph datasets, we introduce noise to specific nodes, while for multi-graph datasets, we select specific graphs and add noise to 50% nodes in each graph. The observations are shown in Fig. 10. We can see that QNMGAD remain superiority when adding noise at different rates, which demonstrates the robustness of QNMGAD. Besides, all methods exhibit performance degradation as the noise rates increase. This is because noise cover part critical characteristics of samples. However, QNMGAD leverages Dirichlet-based evidential learning to avoid the over-confidence and model the uncertainty of detection, which moderates the performance deterioration.

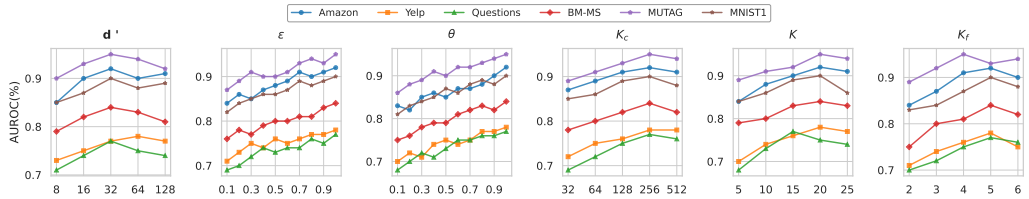


Figure 7: The extensive sensitivity experiments at edge- and graph-level with respect to AUROC.

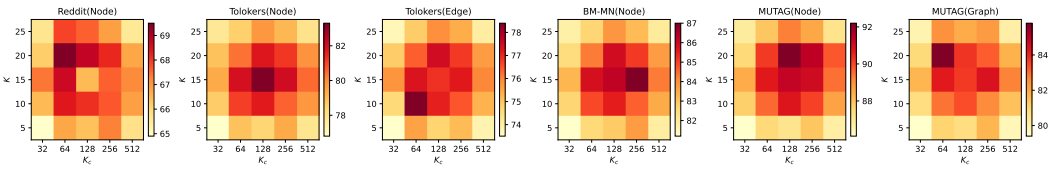


Figure 8: The joint effect of K_c and K on selected baselines with respect to AUROC.

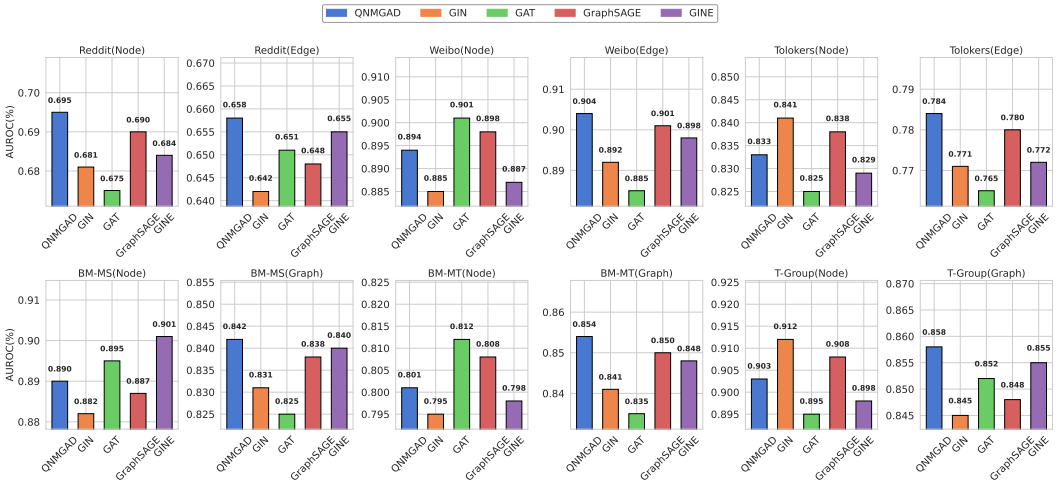


Figure 9: The utility of different graph encoders with respect to AUROC.

We implement extensive robustness experiments on edge-level single-graph datasets (Amazon, Yelp, and Questions) and on graph-level multi-graph datasets (BM-MS, BM-MT, and MUTAG), as shown in Fig. 11. From the observations, we can see that QNMGAD maintains the superiority compared with baselines and shows less influence from the added noise.

H Normalizing Ablation

In this section, we validate the effectiveness of normalizing flow and introduce some traditional distribution modeling schemes as variants, including VAE, VGAE, and diffusion in DiffGAD. From the observations in Table 14, QNMGAD achieves the best performance across datasets on multi-granularity GAD tasks, which demonstrates the effectiveness of using normalizing flows as distribution modeling schemes. In addition, we can see that variants show unstable performance. For example, diffusion achieves the best performance among the variants on the Reddit datasets, while VAE achieves the best performance among the variants on the Weibo datasets.

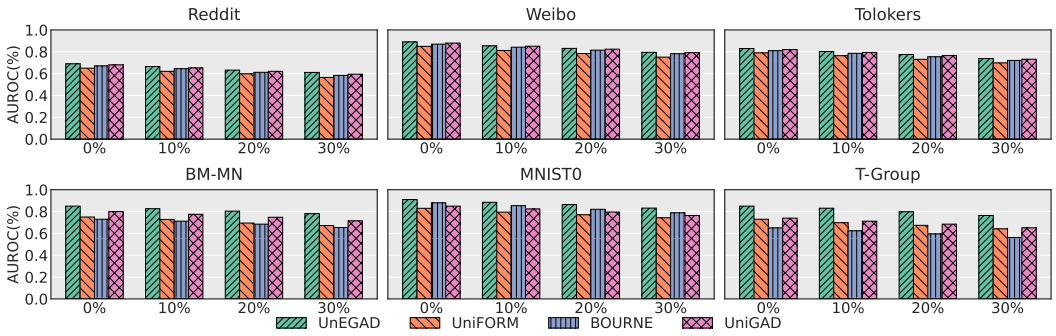


Figure 10: The robustness experiments on Reddit, Weibo, Tolokers, BM-MN, MNIST0, T-Group datasets with respect to AUROC.

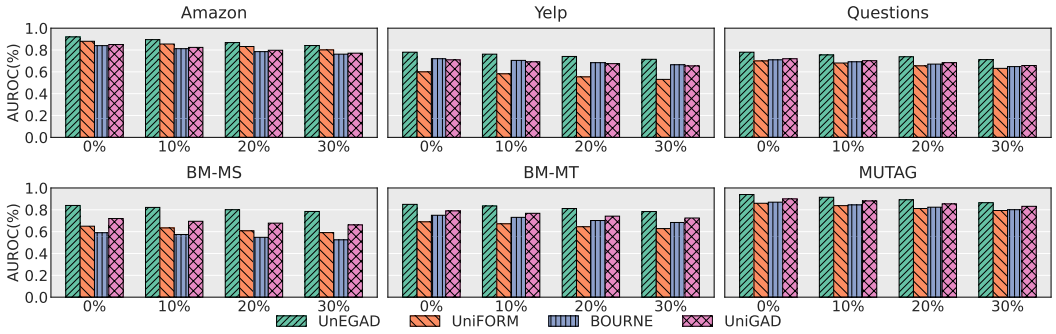


Figure 11: The extensive robustness experiments with respect to AUROC.

Table 11: Initial training ratios and the training ratio used in the scalability experiments.

Dataset	Initial Train%	Scalability Train%
Reddit	40%	20%
Weibo	40%	20%
Amazon	50%	20%
Yelp	50%	20%
Tolokers	50%	20%
Questions	50%	20%
T-Finance	40%	20%
BM-MN	40%	20%
BM-MS	40%	20%
BM-MT	40%	20%
MUTAG	40%	20%
MNIST0	10%	5%
MNIST1	10%	5%
T-Group	40%	20%

Table 12: The scalability performance of node- and edge-level GAD on single-graph datasets

	Dataset	Reddit		Weibo		Amazon		Yelp		Tolokers		Questions		T-Finance	
	Metrics	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
Node-Level	AutoGAD	47.5 ± 1.9	3.8 ± 0.9	62.4 ± 2.1	16.8 ± 1.4	77.6 ± 1.7	24.5 ± 2.3	49.8 ± 2.2	5.6 ± 1.2	63.5 ± 1.8	24.1 ± 2.0	54.6 ± 1.9	9.4 ± 1.1	73.5 ± 1.8	30.8 ± 1.8
	HEC-GAD	49.9 ± 1.8	4.3 ± 1.3	64.9 ± 1.9	19.2 ± 1.5	75.3 ± 1.8	22.5 ± 2.3	64.5 ± 2.0	34.2 ± 1.5	65.7 ± 1.7	27.5 ± 1.8	58.8 ± 1.8	13.6 ± 0.9	67.2 ± 1.7	24.5 ± 1.8
	SGAD-DCR	52.6 ± 1.9	3.7 ± 1.5	63.5 ± 1.8	16.1 ± 1.2	79.1 ± 2.0	27.1 ± 1.8	58.3 ± 1.8	22.7 ± 1.4	66.8 ± 1.8	26.4 ± 2.5	59.2 ± 1.9	11.8 ± 1.4	73.0 ± 1.7	31.1 ± 2.0
	CVGAD	53.2 ± 2.1	4.5 ± 1.7	68.2 ± 2.0	20.9 ± 1.0	80.5 ± 1.8	28.0 ± 2.2	54.7 ± 2.3	20.8 ± 1.9	70.5 ± 1.9	30.5 ± 2.0	61.4 ± 1.8	14.5 ± 2.0	69.0 ± 1.8	32.2 ± 1.0
	AD-GCL	52.1 ± 1.7	4.2 ± 1.2	67.0 ± 1.8	19.6 ± 1.0	75.9 ± 1.7	20.6 ± 1.4	58.8 ± 1.9	28.5 ± 1.0	69.2 ± 1.8	28.8 ± 2.8	64.5 ± 1.7	16.1 ± 1.6	75.4 ± 1.7	34.2 ± 1.1
	UniFORM	55.4 ± 1.6	4.9 ± 1.9	66.4 ± 1.7	18.2 ± 2.8	75.5 ± 1.6	26.3 ± 1.9	62.3 ± 1.8	30.6 ± 1.7	72.8 ± 1.7	33.7 ± 1.3	62.0 ± 1.6	13.8 ± 0.9	77.6 ± 1.6	35.4 ± 1.0
	BOURNE	56.0 ± 1.5	5.2 ± 2.8	66.9 ± 1.6	20.3 ± 1.8	79.0 ± 1.5	24.5 ± 1.8	61.0 ± 1.7	31.9 ± 1.2	72.1 ± 1.6	27.0 ± 1.2	57.5 ± 1.5	11.5 ± 1.7	63.8 ± 1.4	18.8 ± 2.4
	GRAND	55.8 ± 1.4	6.0 ± 1.8	68.6 ± 1.5	22.6 ± 1.3	81.2 ± 1.4	30.8 ± 1.5	65.4 ± 1.6	34.0 ± 1.0	70.6 ± 1.5	31.6 ± 2.1	66.7 ± 1.4	17.8 ± 1.3	71.2 ± 1.3	25.9 ± 2.3
	QNMGAD	**57.8** ± 1.0	**7.3** ± 1.8	**71.2** ± 0.9	**24.5** ± 1.0	**84.6** ± 1.4	**33.2** ± 1.3	**68.7** ± 1.0	**38.8** ± 0.8	**74.5** ± 0.9	**37.2** ± 0.8	**70.4** ± 0.8	**20.8** ± 1.0	**80.3** ± 1.2	**38.6** ± 0.8
Edge-Level	AutoGAD	41.3 ± 2.0	3.1 ± 0.5	59.5 ± 1.5	28.2 ± 1.5	76.8 ± 1.8	31.5 ± 1.5	48.1 ± 2.0	15.6 ± 1.5	57.8 ± 1.8	18.8 ± 1.2	62.6 ± 2.0	24.8 ± 2.6	64.2 ± 1.9	20.4 ± 2.2
	HEC-GAD	46.5 ± 1.9	3.5 ± 1.0	61.7 ± 2.0	29.8 ± 1.6	79.0 ± 1.9	38.2 ± 1.8	52.4 ± 2.1	18.5 ± 1.6	58.6 ± 1.8	20.8 ± 1.6	61.0 ± 1.8	22.5 ± 1.3	69.0 ± 1.8	26.4 ± 2.8
	SGAD-DCR	43.2 ± 2.1	3.2 ± 1.4	57.8 ± 1.9	26.4 ± 1.2	74.9 ± 2.1	29.8 ± 1.0	58.6 ± 1.8	25.6 ± 1.9	63.0 ± 1.9	22.8 ± 1.2	57.5 ± 2.0	16.2 ± 2.1	71.2 ± 1.8	28.7 ± 1.7
	CVGAD	42.9 ± 2.3	3.3 ± 1.2	58.6 ± 2.1	27.5 ± 0.8	73.8 ± 1.9	28.3 ± 1.8	51.5 ± 2.3	20.6 ± 1.2	63.5 ± 2.0	24.0 ± 2.2	60.8 ± 1.8	20.2 ± 1.2	67.5 ± 1.9	24.8 ± 1.6
	AD-GCL	41.6 ± 1.8	3.2 ± 1.2	63.5 ± 1.9	31.2 ± 2.4	80.0 ± 1.8	39.6 ± 2.0	46.2 ± 2.0	15.0 ± 1.3	62.4 ± 1.9	22.7 ± 1.7	62.5 ± 1.8	24.1 ± 2.4	74.8 ± 1.7	33.6 ± 2.2
	UniFORM	46.1 ± 1.7	3.7 ± 1.8	62.8 ± 1.8	29.7 ± 0.9	80.4 ± 1.7	37.5 ± 1.7	50.2 ± 1.9	17.8 ± 1.8	60.5 ± 1.8	21.8 ± 1.9	62.8 ± 1.7	23.5 ± 2.1	67.8 ± 1.6	21.5 ± 1.7
	BOURNE	45.3 ± 1.6	3.5 ± 1.3	59.2 ± 1.7	27.3 ± 1.4	75.8 ± 1.6	32.8 ± 1.5	57.3 ± 1.8	22.5 ± 1.2	63.1 ± 1.7	24.6 ± 1.8	64.0 ± 1.6	24.5 ± 2.2	63.6 ± 1.5	18.6 ± 2.5
	GRAND	48.4 ± 1.5	4.8 ± 1.8	63.7 ± 1.6	30.5 ± 1.5	77.0 ± 1.5	35.2 ± 0.8	56.5 ± 1.7	21.8 ± 0.9	62.2 ± 1.6	22.5 ± 2.1	65.5 ± 1.5	26.2 ± 2.6	70.5 ± 1.4	27.4 ± 1.6
	QNMGAD	**50.5** ± 0.7	**7.2** ± 0.6	**67.6** ± 0.8	**33.6** ± 1.6	**81.5** ± 0.7	**41.8** ± 0.7	**61.7** ± 0.9	**29.5** ± 0.8	**65.8** ± 0.8	**30.5** ± 1.3	**66.5** ± 0.7	**30.2** ± 0.9	**75.8** ± 0.7	**36.2** ± 1.1

I SCALABILITY

In this section, we test the influence of training rate on the performance. Table 11 shows the change in training rate. As shown in Table 12 and Table 13, we conduct comprehensive scalability experiments on multi-granularity GAD with baselines. From the observations, we can conclude that QNMGAD still outperforms other baselines when reducing the scale of the training data, which demonstrates the effectiveness of QNMGAD. In addition, we notice that the performance inevitably shows degradation compared with the performance with the original training rate. This is because the reduction in training ratio leads to a decrease in the available semantic information of the model, resulting in a degradation of discriminability.

J EFFICIENCY

In this section, we record the running time of QNMGAD to demonstrate the efficiency of QNMGAD. As shown in Table 15 and Table 16, QNMGAD has the most efficient observations on most datasets, which is enough to prove the utility of the architecture of QNMGAD. The reason may be that quantization strategies provide lightweight computational costs. Although normalizing flows may be regarded as a time-consuming tool in consensus, the total computational cost remains at a relatively low level.

K ALGORITHM

We analyze the computational complexity. For multi-granularity prototype quantization, the complexity is $\mathcal{O}(\bar{N}dd' + \bar{N}dd' + |E|d' + \bar{N}d'K_c)$, where $\bar{N}$ is the number of samples within specific

Table 13: The scalability performance of node- and graph-level GAD on multi-graph dataset.

	Dataset	BM-MN		BM-MS		BM-MT		MUTAG		MNIST0		MNIST1		T-Group	
	Metrics	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
Node-level	RQGNN	71.8 ± 1.7	32.4 ± 1.6	57.6 ± 1.9	22.0 ± 1.2	53.1 ± 1.8	8.2 ± 1.3	81.5 ± 1.5	31.5 ± 0.9	87.6 ± 1.7	36.8 ± 1.8	88.5 ± 1.5	41.6 ± 1.5	76.8 ± 1.8	33.2 ± 1.0
	GUL	56.2 ± 2.0	19.8 ± 1.2	45.6 ± 2.1	10.8 ± 1.7	50.5 ± 1.9	7.8 ± 1.3	80.8 ± 1.6	30.8 ± 1.1	88.6 ± 1.8	41.5 ± 0.9	90.5 ± 1.6	46.0 ± 1.7	67.8 ± 1.8	24.5 ± 1.5
	MUSE	50.6 ± 2.1	13.5 ± 1.0	53.4 ± 2.2	18.2 ± 2.0	46.5 ± 2.0	7.1 ± 0.8	82.2 ± 1.7	32.0 ± 1.0	90.2 ± 1.4	46.5 ± 1.9	86.2 ± 1.3	37.2 ± 1.2	70.2 ± 1.9	28.5 ± 1.7
	GLAD	51.8 ± 1.9	13.2 ± 1.6	46.5 ± 2.0	10.4 ± 1.2	45.8 ± 2.1	7.0 ± 0.7	78.6 ± 1.8	27.5 ± 1.7	86.9 ± 1.9	35.6 ± 1.6	86.3 ± 1.8	38.6 ± 0.8	73.9 ± 1.8	31.2 ± 2.1
	GLADPro	59.8 ± 1.8	21.4 ± 1.3	55.8 ± 1.9	11.6 ± 1.2	60.3 ± 1.8	19.8 ± 1.5	81.3 ± 1.5	29.8 ± 2.0	89.9 ± 1.4	43.6 ± 1.8	88.0 ± 1.3	41.2 ± 1.6	72.0 ± 1.6	29.0 ± 2.4
	UniFORM	61.2 ± 1.7	22.5 ± 1.9	58.6 ± 1.8	21.0 ± 0.9	56.2 ± 1.9	12.4 ± 2.0	84.7 ± 1.5	35.2 ± 2.1	91.8 ± 1.2	51.0 ± 1.9	91.0 ± 1.2	46.0 ± 2.0	73.1 ± 1.5	33.5 ± 1.6
	BOURNE	70.8 ± 1.6	29.6 ± 2.8	49.6 ± 1.8	13.2 ± 2.0	60.5 ± 1.8	17.0 ± 1.9	85.2 ± 1.5	34.5 ± 1.8	92.4 ± 1.5	58.5 ± 1.8	88.8 ± 1.3	43.5 ± 2.6	68.5 ± 1.8	30.2 ± 3.0
	GRAND	74.8 ± 1.4	34.8 ± 0.9	65.2 ± 1.5	28.5 ± 1.2	61.6 ± 1.4	18.8 ± 1.1	80.5 ± 1.3	29.6 ± 1.8	92.0 ± 1.1	60.0 ± 2.0	88.3 ± 1.1	42.6 ± 1.9	77.2 ± 1.3	35.5 ± 2.0
	QNMGAD	$\mathbf{77.6} \pm 0.8$	$\mathbf{38.8} \pm 0.8$	$\mathbf{69.5} \pm 0.9$	$\mathbf{39.8} \pm 0.8$	$\mathbf{67.4} \pm 0.8$	$\mathbf{28.6} \pm 0.8$	$\mathbf{88.5} \pm 0.7$	$\mathbf{41.8} \pm 0.7$	$\mathbf{94.3} \pm 0.6$	$\mathbf{69.8} \pm 1.4$	$\mathbf{92.1} \pm 0.7$	$\mathbf{55.0} \pm 0.8$	$\mathbf{82.8} \pm 0.8$	$\mathbf{43.8} \pm 1.3$
Graph-level	RQGNN	67.4 ± 1.7	27.6 ± 2.0	54.8 ± 1.9	20.8 ± 1.7	65.8 ± 1.8	33.5 ± 2.0	84.6 ± 1.3	43.5 ± 2.0	79.6 ± 1.7	31.0 ± 1.5	77.5 ± 1.6	25.6 ± 1.4	60.5 ± 1.8	18.6 ± 1.8
	GUL	66.5 ± 2.0	25.2 ± 1.4	61.5 ± 2.1	26.5 ± 1.8	51.2 ± 1.9	13.2 ± 2.0	84.0 ± 1.8	46.0 ± 2.2	83.2 ± 1.8	32.2 ± 1.4	77.0 ± 1.6	24.8 ± 2.3	69.6 ± 1.8	29.0 ± 1.5
	MUSE	68.2 ± 2.1	30.6 ± 2.1	50.8 ± 2.2	18.2 ± 1.5	56.8 ± 2.0	15.8 ± 1.6	82.2 ± 1.7	43.2 ± 2.1	81.8 ± 1.4	30.6 ± 1.9	76.0 ± 1.3	24.2 ± 1.8	63.5 ± 1.9	20.2 ± 2.0
	GLAD	64.2 ± 1.9	25.6 ± 1.0	49.0 ± 2.0	16.5 ± 2.4	46.8 ± 2.1	8.5 ± 1.4	84.0 ± 1.7	44.2 ± 2.3	78.6 ± 1.8	26.5 ± 2.5	70.2 ± 1.8	18.6 ± 1.7	51.8 ± 1.8	8.2 ± 2.0
	GLADPro	69.5 ± 1.8	32.5 ± 1.2	58.5 ± 1.9	23.5 ± 1.3	52.5 ± 1.8	11.8 ± 1.8	82.8 ± 1.3	43.0 ± 1.3	82.5 ± 1.8	31.8 ± 1.4	79.5 ± 1.9	29.6 ± 1.6	66.8 ± 1.7	21.5 ± 1.8
	UniFORM	61.6 ± 1.7	22.8 ± 1.8	50.5 ± 1.8	15.5 ± 2.0	55.8 ± 1.9	15.5 ± 1.4	81.5 ± 2.0	40.6 ± 1.9	77.8 ± 1.6	30.0 ± 1.8	69.5 ± 1.2	20.5 ± 1.8	62.0 ± 1.6	13.2 ± 2.0
	BOURNE	59.8 ± 1.6	20.5 ± 2.0	44.8 ± 1.8	13.6 ± 0.8	60.6 ± 1.8	21.5 ± 1.3	82.8 ± 1.6	42.0 ± 2.1	83.0 ± 1.4	33.2 ± 2.3	73.5 ± 1.4	22.5 ± 2.0	55.5 ± 1.8	9.8 ± 2.6
	GRAND	68.6 ± 1.5	30.5 ± 2.4	56.6 ± 1.6	22.8 ± 1.4	63.8 ± 1.5	25.5 ± 1.5	85.8 ± 2.0	47.5 ± 1.9	80.2 ± 2.3	29.5 ± 1.9	74.8 ± 1.2	24.6 ± 1.1	63.8 ± 1.4	19.2 ± 1.1
	QNMGAD	$\mathbf{74.8} \pm 0.8$	$\mathbf{36.5} \pm 1.4$	$\mathbf{65.8} \pm 0.9$	$\mathbf{34.8} \pm 1.1$	$\mathbf{70.5} \pm 0.8$	$\mathbf{38.5} \pm 1.3$	$\mathbf{90.8} \pm 1.0$	$\mathbf{51.6} \pm 1.4$	$\mathbf{86.2} \pm 1.3$	$\mathbf{41.8} \pm 1.2$	$\mathbf{84.8} \pm 1.0$	$\mathbf{35.8} \pm 1.0$	$\mathbf{74.5} \pm 0.8$	$\mathbf{31.8} \pm 1.6$

Table 14: The ablation of normalizing flows on node-, edge-, and graph-level GAD.

Task	Method	Single-Graph Datasets						Multi-Graph Datasets					
		Reddit		Weibo		Amazon		BM-MN		BM-MS		BM-MT	
		AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
Node	VAE	$61.8_{\pm 1.2}$	$7.4_{\pm 1.5}$	$82.5_{\pm 1.4}$	$24.6_{\pm 1.1}$	$77.6_{\pm 1.1}$	$27.8_{\pm 0.9}$	$74.5_{\pm 1.3}$	$34.8_{\pm 1.2}$	$79.4_{\pm 0.9}$	$41.8_{\pm 1.4}$	$69.2_{\pm 1.5}$	$28.6_{\pm 1.1}$
	VGAE	$58.3_{\pm 1.6}$	$6.3_{\pm 1.1}$	$79.8_{\pm 1.2}$	$21.7_{\pm 1.6}$	$82.9_{\pm 1.5}$	$34.9_{\pm 1.3}$	$77.9_{\pm 1.0}$	$39.6_{\pm 1.6}$	$73.5_{\pm 1.4}$	$36.8_{\pm 1.0}$	$72.8_{\pm 1.2}$	$30.3_{\pm 1.5}$
	diffusion	$64.2_{\pm 0.9}$	$8.1_{\pm 1.7}$	$75.3_{\pm 1.7}$	$18.5_{\pm 1.3}$	$73.8_{\pm 1.4}$	$24.1_{\pm 1.6}$	$81.3_{\pm 0.8}$	$42.1_{\pm 1.1}$	$76.6_{\pm 1.6}$	$39.5_{\pm 1.5}$	$74.6_{\pm 1.1}$	$33.5_{\pm 1.2}$
Edge / Graph	VAE	$57.9_{\pm 1.1}$	$7.8_{\pm 1.4}$	$80.3_{\pm 1.5}$	$35.8_{\pm 1.2}$	$79.1_{\pm 1.0}$	$41.4_{\pm 0.8}$	$71.8_{\pm 1.6}$	$35.9_{\pm 1.3}$	$75.2_{\pm 1.2}$	$35.1_{\pm 1.5}$	$75.0_{\pm 1.5}$	$39.7_{\pm 1.0}$
	VGAE	$53.8_{\pm 1.7}$	$5.2_{\pm 1.1}$	$84.7_{\pm 1.1}$	$39.2_{\pm 1.7}$	$83.6_{\pm 1.4}$	$45.3_{\pm 1.4}$	$78.9_{\pm 1.2}$	$40.3_{\pm 1.1}$	$68.9_{\pm 1.6}$	$29.6_{\pm 1.4}$	$79.8_{\pm 1.2}$	$43.0_{\pm 1.3}$
	diffusion	$60.6_{\pm 0.8}$	$8.6_{\pm 1.6}$	$76.8_{\pm 1.8}$	$31.4_{\pm 1.0}$	$87.0_{\pm 0.9}$	$48.1_{\pm 1.1}$	$80.2_{\pm 0.9}$	$42.8_{\pm 1.5}$	$72.7_{\pm 1.3}$	$33.5_{\pm 1.2}$	$67.8_{\pm 1.8}$	$31.8_{\pm 1.6}$
	QNMGAD	$\mathbf{65.8}_{\pm 0.5}$	$\mathbf{11.4}_{\pm 0.4}$	$\mathbf{90.4}_{\pm 0.6}$	$\mathbf{45.5}_{\pm 1.4}$	$\mathbf{92.2}_{\pm 0.5}$	$\mathbf{53.5}_{\pm 0.5}$	$\mathbf{85.1}_{\pm 0.6}$	$\mathbf{45.7}_{\pm 1.2}$	$\mathbf{84.2}_{\pm 0.7}$	$\mathbf{46.0}_{\pm 0.9}$	$\mathbf{85.4}_{\pm 0.6}$	$\mathbf{50.2}_{\pm 1.1}$

granularity. For granularity-specific residual pattern quantization, the complexity is $\mathcal{O}(\bar{N}d'K_c)$. For prototype-conditioned normalizing flow, the complexity is $\mathcal{O}(K_f\bar{N}d')$. The algorithm is shown in Algorithm 1.

Table 15: Efficiency comparison on single-graph datasets in terms of training time per epoch (seconds).

Method	Reddit	Weibo	Amazon	Yelp	Tolokers	Questions	T-Finance
Node-level anomaly detection							
AutoGAD	24.1	19.2	27.3	35.2	21.4	30.5	45.2
HEC-GAD	29.8	22.5	32.1	40.7	25.6	36.4	52.7
SGAD-DCR	26.3	20.4	29.2	37.8	23.3	33.2	48.1
CVGAD	28.1	21.8	30.8	39.5	25.4	35.1	50.5
AD-GCL	25.8	19.8	28.1	36.6	22.5	32.1	46.9
UniFORM	31.5	23.7	33.9	42.8	26.8	38.2	55.1
BOURNE	27.6	21.3	30.3	38.9	24.5	34.5	49.9
GRAND	30.5	23.1	33.6	42.1	26.1	37.3	53.6
QNMGAD	**18.4**	**14.2**	**20.1**	**26.7**	**16.3**	**22.8**	**33.5**
Edge-level anomaly detection							
AutoGAD	38.1	30.2	32.3	24.1	33.5	37.8	49.2
HEC-GAD	26.5	24.8	39.7	33.1	40.1	46.4	60.5
SGAD-DCR	31.2	32.4	35.4	38.9	36.5	41.7	63.8
CVGAD	33.7	34.2	37.8	30.9	39.6	34.2	57.3
AD-GCL	30.3	31.1	43.5	46.2	35.4	49.5	62.2
UniFORM	28.9	27.2	32.2	36.3	42.7	38.9	54.1
BOURNE	33.2	33.5	26.8	22.1	38.2	43.4	46.5
GRAND	37.5	26.3	30.9	45.4	40.8	37.6	52.8
QNMGAD	**19.8**	**15.6**	**21.7**	**28.4**	**27.5**	**24.0**	**35.1**

Table 16: Efficiency comparison on multi-graph datasets in terms of training time per epoch (seconds).

Method	BM-MN	BM-MS	BM-MT	MUTAG	MNIST0	MNIST1	T-Group
Node-level anomaly detection							
AutoGAD	20.5	24.8	30.0	32.0	37.4	40.1	48.0
HEC-GAD	24.3	28.1	34.0	36.5	41.6	44.3	52.8
UniFORM	23.5	27.6	33.8	35.6	39.5	42.5	50.4
CoCo	22.3	25.5	31.5	33.5	38.5	41.7	49.6
CVGAD	23.0	26.5	32.5	34.5	39.2	42.4	50.5
BOURNE	22.5	26.1	32.2	34.0	38.5	41.5	49.5
GRAND	25.1	28.5	35.7	37.4	42.8	45.0	53.1
QNMGAD	**18.2**	**22.0**	**28.5**	**30.5**	**35.0**	**38.0**	**45.0**
Graph-level anomaly detection							
RQGNN	19.8	23.5	30.2	32.5	36.0	39.9	46.4
GUL	22.5	25.8	32.8	35.7	38.5	41.5	48.5
MUSE	23.5	27.1	34.5	36.8	40.2	43.7	50.8
GLAD	20.8	24.3	31.5	33.8	37.2	40.3	47.1
GLADPro	21.5	25.1	32.3	34.5	38.0	40.8	47.5
UniFORM	25.7	28.8	37.2	39.5	43.5	46.5	53.2
BOURNE	22.8	26.3	33.8	36.1	39.8	42.5	49.6
GRAND	25.8	29.5	38.0	40.5	44.5	47.5	54.4
QNMGAD	**17.5**	**21.0**	**28.0**	**14.0**	**24.8**	**27.2**	**34.5**

Algorithm 1 Algorithm of QNMGAD.

Input: Graph $\mathcal{G} = (\mathcal{V}, \mathbf{X}, \mathbf{A})$; loss weights θ and ϵ; Epochs E.

1: Initialize the parameters of QNMGAD.
2: **for** $e \in (1, 2, \cdots, E)$ **do**
3: Obtain $\mathbf{H}_1$ and $\mathbf{H}_2$ via Eq. 3;
4: Initialize prototype concepts $\mathcal{C}_{pr}$ and pattern concept set $\mathcal{C}_s^L$;
5: Obtain the granularity-specific representations and find the top-K prototypes via Eq. (4)-Eq. (8);
6: Calculate prototype quantization loss via Eq. 9;
7: capture granularity-specific semantics using residual pattern quantization based on $\mathcal{C}_s^L$ via Eq. (10)-Eq. (13);
8: Calculate pattern quantization loss via Eq. (14);
9: Encode the latent representation using normalizing flows to fit the posterior distributions via Eq. (15)-Eq. (16);
10: Calculate the loss function of normalizing flows via Eq. (17);
11: Calculate the total loss functions via Eq. (19);
12: Update the model's parameters.
13: **end for**
14: **return** Scores