ANOMALY DETECTION THROUGH CONDITIONAL DIFFUSION PROBABILITY MODELING ON GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing Graph Neural Network-based anomaly detection methods suffer from over-smoothing issues during feature aggregation. Moreover, most existing methods are discriminative models that learn the boundaries between anomalous and normal data points, allowing malicious nodes in a dynamic adversarial environment to bypass detection boundaries. We propose an advanced Conditional Graph Anomaly Diffusion Model (CGADM) to model and capture the joint distribution of anomalies on the whole graph, thereby enabling generative graph anomaly detection. By iteratively refining node anomaly distributions during the denoising process, CGADM effectively mitigates over-smoothing and reconstructs obfuscated features by leveraging contextual neighborhood information. To avoid starting the diffusion process from a random state, CGADM introduces a prior-guided denoising diffusion probability model. To circumvent the need for iterative denoising samplings for each node on large-scale graphs, we adopt a prior confidence-aware mechanism to dynamically adjust the reverse sampling steps for each node, significantly reducing the computationaql burden on large-scale graphs. We conducted experiments on CGADM using standard benchmarks, and the results demonstrated excellent performance in graph anomaly detection tasks. Ablation studies confirmed our framework's computational advantages. [1]

1 INTRODUCTION

Anomaly detection is aimed at identifying objects that deviate significantly from the majority within a vast array of objects. With the massive flow of information on Internet, it is inherently suitable to use the non-Euclidean graphs for modeling. Examples include social networks formed by users on social media, transaction networks formed by mobile payments, and bipartite graphs formed by users and contents. Consequently, graph anomaly detection (GAD) has emerged as a crucial research field, achieving successful applications, such as financial fraud detection (Huang et al., 2022; Dou et al., 2020), and telecommunication fraud detection (Yang et al., 2021), among others.

Among the methods employed for GAD, Graph Neural Networks (GNNs) have ascended to prominence, chiefly due to their exceptional capability to model topological structures. GNNs excel in their iterative refinement of node representations, operating by focusing on a particular node and aggregating attributes from neighboring nodes via the Message Passing (MP) paradigm (Kipf & Welling, 2017; Hamilton et al., 2017; Velickovic et al., 2018; Xu et al., 2019). Subsequent to this feature aggregation, node representations, now enriched with information from their neighboring nodes, are fed into a classifier to determine whether they are outliers or anomalies. This process effectively leverages the power of GNNs in capturing high-order information within the graph, providing a common paradigm for anomaly detection (Li et al., 2019; Wang et al., 2021; Liu et al., 2021b; Zhu et al., 2020; He et al., 2021).

However, discriminative models based on feature aggregation exhibit inherent shortcomings. From a **topology**-level perspective, vanilla GNNs suffer from the over-smoothing problem. As a low-pass filter, GNNs with feature aggregation tend to average the representations of anomalies, making them less distinguishable. As illustrated in the left part of Figure 1, some fraudulent nodes can manipulate their representations by intentionally connecting with a large number of carefully selected

[1] The code is available on https://github.com/CGADManonymous/CGADM

neighbors. For instance, in money laundering transactions, fraudsters can distribute transactions or create numerous interactions with bot accounts to blend in with the crowd. From a **feature**-level perspective, discriminative models perform anomaly detection by learning the boundaries between anomalous and normal data points. This approach may lead to a lack of generalization, as fraudulent nodes always co-evolve with the detection system. By continuously obfuscating their node features, these deceptive entities can cross the classifier's boundary and masquerade as normal nodes.

To address these issues, contemporary research can be summarized along two lines. The first line of work focuses on enhancing the generalizability of GNN models, such as applying attention mechanisms (Wang et al., 2019a; Liu et al., 2021a), designing auxiliary losses (Zhao et al., 2022), and utilizing contrastive learning (Chen et al., 2023a). The second line of work involves leveraging generative models, such as Generative Adversarial Networks (GANs), to perform data augmentation, thereby enriching the diversity of training samples (Chen et al., 2020b). However, these methods primarily focus on enhancing the discriminative boundary for each individual node, rather than considering the interdependencies of node anomalies from a holistic graph perspective. Inspired by the recent powerful capabilities of diffusion models (DMs) in generating high-dimensional data, such as high-resolution images (Dhariwal & Nichol, 2021), we propose the use of diffusion models to model the joint distribution of anomaly on the whole graph, capturing the the interdependencies of node anomalies. To address **topology**-level flaw, we leverage the **iterative refinement** of diffusion models. Instead of increasing GNN depth to aggregate distant information, which risks over-smoothing, our approach applies GNN-based denoiser within each denoising iteration to refine anomaly modeling. Each iterative refinement step incorporates neighborhood information while preserving node-specific high-frequency anomaly information via a residual propagation mechanism, thereby preventing oversmoothing and effectively capturing long-range dependencies. To address **feature**-level flaw, we leverage the **denoising reconstruction** of diffusion models. This reconstruction process ensures that even when malicious nodes disguise their features to blend in with normal nodes, their underlying anomaly patterns can be recovered.

To achieve these goal, we need to address two notable challenges, as shown in right part of Figure 1:

- **Effectiveness.** Traditional denoising models have primarily focused on unconditional generative modeling (Song & Ermon, 2019; Song et al., 2021b; Ramesh et al., 2022). While many tasks in the image or video domain have introduced guided-diffusion models to generate high-resolution photo-realistic images that match the semantic meanings or content of the label, text, or corrupted images, most work in the graph domain has started generating from white noise or empty or fully connected graphs. However, for anomaly detection on graphs, due to various deceptive and obfuscating tactics employed by anomalous nodes, directly recovering the underlying true distribution from a random noise distribution may not yield satisfactory results.
- **Efficiency.** The reverse process of DMs requires numerous iterative denoising samplings (Yi et al., 2023; Chen et al., 2023b). Existing graph diffusion models utilize a GNN-based encoder to update all nodes at time step t during each iterative refinement to obtain the nodes at time step $t-1$. While this approach is feasible for standard graph generation tasks, it becomes computationally prohibitive for anomaly detection tasks on extremely large graphs. Performing such iterative operations generation across potentially millions of nodes in the entire graph can significantly increase computational overhead, thereby affecting the practical applicability of the algorithm.

In this paper, we propose a novel Conditional Graph Anomaly Diffusion Model (CGADM) for graph anomaly detection to address the aforementioned challenges synergistically.

To tackle the effectiveness issue, we propose a prior-guided diffusion process, which injects a pre-trained conditional anomaly estimator into both the forward and reverse diffusion chains. This approach constructs a denoising diffusion probabilistic model for more accurate anomaly detection. Specifically, we introduce a lightweight model to estimate an anomaly prior for each node, serving as the endpoint for our forward noise addition process and the starting point for our reverse denoising process. Based on this new probabilistic model, we redesign the probability model and optimization objective of our CGADM.

To tackle the efficiency issue, we build on the intuition that normal nodes are generally farther from the decision boundary compared to anomalous nodes that have narrowly evaded detection. Therefore, in the reverse process, we introduce a prior confidence-aware mechanism to adaptively determine the reverse time step for each node. Nodes with high confidence in their anomaly prior

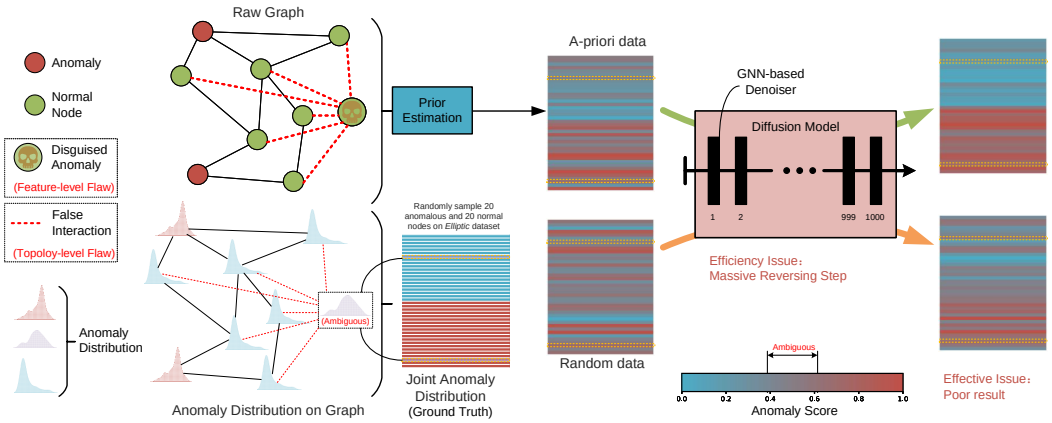

Figure 1: An illustration of Generative Graph Anomaly Detection.

require fewer time steps, while those with lower confidence require more sampling time steps. To facilitate inference over arbitrary numbers of steps, we propose a conditional non-Markovian reverse process, and derive its closed-form expression within the framework of the CGADM. This approach not only accurately estimates the anomaly probability for each node but also reduces the number of predictions in the reverse process, thereby decreasing computational time.

Our main contributions can be summarized as follows:

- We innovatively propose CGADM, which employs a prior-guided denoising diffusion probabilistic model to capture the joint distribution of anomalies on the whole graph, thereby enabling generative graph anomaly detection.
- We propose a prior confidence-aware mechanism to dynamically allocate disparate sampling time steps during the inference process. In support of this mechanism, we derive a conditional non-Markovian reverse process within the framework of the CGADM. This approach significantly mitigates the computational burden associated with anomaly detection in large-scale graphs.
- Through experiments on benchmarks for graph anomaly detection, CGADM achieves state-of-the-art results. Additional studies confirm the computational advantages of our framework.

2 Related Work

2.1 Graph Anomaly Detection

Graph anomaly detection (Duan et al., 2023) aims to identify nodes that deviate significantly from most other nodes. FdGars (Wang et al., 2019b) utilized a predefined tagging system to classify users according to their content and behavioral characteristics, and employed a multi-layered GNN to identify fraudulent users. CARE-GNN (Dou et al., 2020) proposed to adjust the threshold in the process of aggregating neighbors through reinforcement learning, thereby addressing the inconsistency issue. FRAUDRE (Zhang et al., 2021) aggregates different relational neighbors of nodes by applying an imbalanced loss function, addressing the class imbalance problem. PC-GNN (Liu et al., 2021b) resolves the class imbalance issue by selecting training nodes using a label-balanced sampler. AMNet (Chai et al., 2022) captures features of normal and abnormal frequency bands using a dual filter based on Bernstein polynomials and aggregates them through an attention mechanism. BWGNN (Tang et al., 2022) adopts a Beta-kernel-based GNN model, effectively dealing with abnormal high-frequency features by applying multiple filters to various frequency bands. GHRN (Gao et al., 2023b) eliminates harmful heterogeneous connections on any qualified fraud detection model through approximating pre-training labels. Recent advancements in graph anomaly detection have tackled various challenges. Gao et al. (2023a) addressed structural distribution shifts through feature-specific constraints in Graph Decomposition Networks (GDN), while Xu et al. (2024) proposed SEC-GFD to handle heterophily and label imbalance via spectral filtering. Qiao et al. (2024) introduced a semi-supervised generative framework (GGAD) that leverages labeled normal nodes to generate pseudo-anomalies, and He et al. (2024) developed ADA-GAD to mit-

igate anomaly overfitting through anomaly-denoised graph augmentation. Unlike these methods, our CGADM adopts a novel generative diffusion approach to model the joint anomaly distribution over the graph, enabling holistic and scalable anomaly detection without reliance on augmentation strategies.

However, the aforementioned methods predominantly rely on discriminative models based on feature aggregation, which are susceptible to the over-smoothing problem inherent in GNNs and the camouflage deception of fraudulent nodes. We departs from this traditional perspective and proposes a novel generative model to jointly model the anomaly distribution of each node on the graph.

2.2 Diffusion Model

Denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020; Song et al., 2021a), or simply diffusion models, are a class of probabilistic generative models that transform noise into data samples, hence primarily used for generative tasks (Dhariwal & Nichol, 2021; Rombach et al., 2022). Diffusion-based generative models have demonstrated strong capabilities in generating high-quality graphs (Niu et al., 2020; Liu et al., 2019; Jo et al., 2022; Haefeli et al., 2022; Chen et al., 2022; Vignac et al., 2023; Kong et al., 2023). Haefeli et al. (2022) designed a model limited to graphs without attributes and similarly observed the benefits of discrete diffusion for graph generation. Previous graph diffusion models were based on Gaussian noise. Niu et al. (2020) generated adjacency matrices indicating the presence of edges by thresholding continuous values, while Jo et al. (2022) extended this model to handle node and edge attributes. Digress (Vignac et al., 2023) was the first to propose a discrete diffusion model for graphs. Regarding the severe label imbalance problem in anomaly detection, many existing anomaly detection methods improve datasets by generating synthetic anomalies (Chen et al., 2020b; Ding et al., 2020), creating a more balanced environment.

We approaches from a different angle, using diffusion models to model the distribution of anomalies on large-scale graphs for more precise and robust anomaly detection. To the best of our knowledge, there is currently no work on modeling the distribution of anomalies based on diffusion models.

3 Preliminaries

Attributed Graph We typically characterize an attributed graph as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ represents the set of all N nodes on graph $\mathcal{G}$, and $\mathcal{E} = \{e_{ij} | v_i, v_j \in \mathcal{V}\}$ signifies the set of edges, indicating the existence of an edge between nodes v_i and v_j. For each node v_i, there exists a d-dimensional feature vector, $x_i \in \mathbb{R}^d$. The feature vectors of all nodes together form the feature matrix of the graph, denoted as $\mathbf{X} = [x_1, x_2, \ldots, x_N] \in \mathbb{R}^{N \times d}$. For convenience, An adjacency matrix $\mathbf{A}$ records the relationships between nodes on graph $\mathcal{G}$. Each entry $\mathbf{A}_{ij} = 1$ if there exists $e_{ij} \in \mathcal{E}$, otherwise, $\mathbf{A}_{ij} = 0$. Additionally, the degree matrix $\mathbf{D} \in \mathbb{N}^{N \times N}$ is a diagonal matrix, in which each entry $\mathbf{D}_{ii}$ denotes the number of nonzero entries in the i-th row of $\mathbf{A}$.

Anomaly Detection on Graph Consider two disjoint subsets of $\mathcal{V}$, namely $\mathcal{V}_a$ and $\mathcal{V}_n$, such that $\mathcal{V}_a \cap \mathcal{V}_n = \emptyset$. $\mathcal{V}_a$ contains all nodes labeled as anomalous, and $\mathcal{V}_n$ comprises all normal nodes. The goal of graph anomaly detection (GAD) is to compute anomaly probability $p(\mathbf{y}|\mathcal{E}, \mathbf{X})$ of the unlabeled nodes with partial node labels. Please refer Appendix E for challenges of GAD.

Diffusion Probabilistic Model To construct an efficient diffusion model, it must satisfy three key properties: (1) The conditional distribution $q(z_t|x)$ should possess a closed-form equation to circumvent the recursive application of noise during training. (2) The posterior $q(z_{t-1}|z_t, x)$ should also have a closed-form solution to serve as the neural network's target. (3) The limiting distribution $q_\infty = \lim_{T \to \infty} q(z_T|x)$ should be independent of x, enabling its use as a prior distribution for inference. These properties are all met when the noise follows a Gaussian distribution. The common steps in the diffusion model are shown in Appendix A.

4 Methodology

We formulate the GAD problem as a task of modeling the joint conditional distribution of anomalies on the graph. Given an attributed graph, a lightweight mean estimator is used to compute a prior distribution of the anomaly. This prior distribution serves as the endpoint for adding noise

and the starting point for inference. CGADM gradually transforms the ground truth anomaly distribution into the prior distribution instead of the conventional Guassian distribution. By utilizing a topological-guided denoising network, CGADM is capable of simultaneously modeling the topological information and features of nodes to iteratively recover the ground truth. To expedite the inference process, we introduce a prior-aware strided sampling strategy. To enable inference over arbitrary numbers of steps, we propose a conditional non-Markovian reverse process.

4.1 Diffuse Ground Truth to Prior

In light of Section 3, we propose to cast the graph anomaly detection problem as a generative task. We set $\mathbf{y}_0$ as the anomaly ground truth and $\mathbf{y}_{1:T}$ as the intermediate predictions generated in the forward process of the diffusion model. The objective of graph anomaly detection then becomes the maximization of the log-likelihood $p(\mathbf{y}_0|\mathcal{E}, \mathbf{X})$. Consequently, Equation 2 can be restructured as the following Conditional Evidence Lower Bound (CELBO) to serve as our new optimization target:

$$\log p_\theta(\mathbf{y}_0|\mathcal{E}, \mathbf{X}) = \log \int p_\theta(\mathbf{y}_{0:T}|\mathcal{E}, \mathbf{X}) d\mathbf{y}_{1:T} \geq \mathbb{E}_{q(\mathbf{y}_{1:T}|\mathbf{y}_0, \mathcal{E}, \mathbf{X})} \left[\log \frac{p_\theta(\mathbf{y}_{0:T}|\mathcal{E}, \mathbf{X})}{q(\mathbf{y}_{1:T}|\mathbf{y}_0, \mathcal{E}, \mathbf{X})} \right], \tag{1}$$

where $p_\theta(\mathbf{y}_{0:T}|\mathcal{E}, \mathbf{X})$ is the joint distribution of the target and the predictions under the denoising model parameters θ, and $q(\mathbf{y}_{1:T}|\mathbf{y}_0, \mathcal{E}, \mathbf{X})$ is the conditional distribution of forward or diffusion process given the ground truth and the input data.

By substituting Equation 1 into Equation 16, we can express our optimization objective as follows:

$$\begin{aligned} \mathcal{L} = &\mathbb{E}_q \left[-\log p_\theta(\mathbf{y}_0|\mathbf{y}_1, \mathcal{E}, \mathbf{X}) \right] + \mathbb{E}_q \left[\mathbb{D}_{KL} \left(q(\mathbf{y}_T|\mathbf{y}_0, \mathcal{E}, \mathbf{X}) \| \, p(\mathbf{y}_T|\mathcal{E}, \mathbf{X}) \right) \right] \\ &+ \sum_{t=2}^{T} \mathbb{E}_q \left[\mathbb{D}_{KL} \left(q(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{y}_0, \mathcal{E}, \mathbf{X}) \| \, p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathcal{E}, \mathbf{X}) \right) \right]. \end{aligned} \tag{2}$$

Following the conventions of Denoising Diffusion Probabilistic Models (DDPM) (Ho et al., 2020), we respectively name the first, second, and third terms of the above objective function as the reconstruction term $\mathcal{L}_{recon}$, the prior matching term $\mathcal{L}_{prior}$, and the consistency term $\mathcal{L}_{con}$.

To avoid our CGADM recovering the joint anomaly distribution starting from random noise (Han et al., 2022b), we modify the endpoint of the diffusion process from the conventional Guassian distribution $N(0, I)$ to:

$$p(\mathbf{y}_T|\mathcal{E}, \mathbf{X}) = N(g_\phi(\mathcal{E}, \mathbf{X}), I), \tag{3}$$

where $g_\phi(\mathcal{E}, \mathbf{X})$ is a parameterized network pretrained on training set D to estimate the mean value of the final normal distribution. By doing so, we effectively utilize the condition $\mathcal{E}, \mathbf{X}$ in the distribution $p(\mathbf{y}_T|\mathcal{E}, \mathbf{X})$ to help us establish a prior understanding of the joint anomaly distribution.

The prior matching term $\mathcal{L}_{prior}$ is a parameter-free term. In order to make it close to zero, we need to adjust the forward process in combination with the calculation of the prior $g_\phi(\mathcal{E}, \mathbf{X})$. Following the practice of Pandey et al. (2022), we define the noise-adding process at each step as follows:

$$q(\mathbf{y}_t|\mathbf{y}_{t-1}, g_\phi(\mathcal{E}, \mathbf{X})) = \mathcal{N}(\mathbf{y}_t; \sqrt{1-\beta_t}\mathbf{y}_{t-1} + (1 - \sqrt{1-\beta_t}) g_\phi(\mathcal{E}, \mathbf{X}), \beta_t I), \tag{4}$$

where $\mathcal{N}$ represents the Gaussian Distribution, and $\beta_t \in (0, 1)$ regulates the noise scales added at step t. This noise-adding step allows for a closed-form sampling distribution at any arbitrary timestep t, according to the additivity of the Gaussian distribution:

$$q(\mathbf{y}_t|\mathbf{y}_0, \mathcal{E}, \mathbf{X}) = q(\mathbf{y}_t|\mathbf{y}_0, g_\phi(\mathcal{E}, \mathbf{X})) = \mathcal{N}(\mathbf{y}_t; \sqrt{\bar{\alpha}_t}\mathbf{y}_0 + (1 - \sqrt{\bar{\alpha}_t}) g_\phi(\mathcal{E}, \mathbf{X}), (1 - \bar{\alpha}_t) I), \tag{5}$$

where $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_t \alpha_t$. This sampling distribution enables $\mathcal{L}_{prior}$ to be close to zero when $t = T$. Intuitively, the noise-adding process defined by Equation 5 can be interpreted as an interpolation between the true data $\mathbf{y}_0$ and the estimated prior $g_\phi(\mathcal{E}, \mathbf{X})$, which exhibits a gradual transition from the true data towards the estimated prior over the course of the forward process.

With the above formulation, we can derive a tractable posterior that serves as the target for our denoising network. It can be expressed as follows:

$$q(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{y}_0, \mathcal{E}, \mathbf{X}) = q(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{y}_0, g_\phi(\mathcal{E}, \mathbf{X})) = \mathcal{N}\left(\mathbf{y}_{t-1}; \tilde{\mu}(\mathbf{y}_t, \mathbf{y}_0, g_\phi(\mathcal{E}, \mathbf{X})), \tilde{\beta}_t \mathbf{I}\right), \tag{6}$$

where $\tilde{\mu} := \gamma_0 \mathbf{y}_0 + \gamma_1 \mathbf{y}_t + \gamma_2 g_\phi(\mathcal{E}, \mathbf{X})$ and $\tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$, with:

$$\gamma_0 = \sqrt{\beta_t \bar{\alpha}_{t-1}}, \quad \gamma_1 = \frac{(1 - \bar{\alpha}_{t-1})\sqrt{\alpha_t}}{(\alpha_t - 1)(\sqrt{\alpha_t} + \sqrt{\bar{\alpha}_{t-1}})}, \quad \gamma_2 = \frac{1}{1 - \bar{\alpha}_t}. \tag{7}$$

For detailed derivation, please refer to Appendix B.

4.2 TOPOLOGICAL-GUIDED DENOISING NETWORK

According to Equation 4, we define $p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathcal{E}, \mathbf{X})$ as $N(\mathbf{y}_{t-1}; \mu_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}), \Sigma_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}))$ for $1 < t \leq T$. Following the setup of DDPM, we set $\Sigma_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) = \sigma_t^2 \mathbf{I}$ to untrained time-dependent constants and set $\sigma_t^2 = \tilde{\beta}_t$. For the parameterization, we may select:

$$\mu_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) = \frac{1}{\sqrt{\alpha_t}}(\mathbf{y}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X})), \tag{8}$$

where ϵ_θ is a parameterized network intended to predicts the forward diffusion noise ϵ sampled for anomaly scores $\mathbf{y}_t$.

An anomalous node is typically strongly correlated not only with its node features but also with the its local topological structure. The bias brought about by a few anomalous nodes is high-frequency information in the frequency domain. Most existing GNNs act as low-pass filters and cannot effectively capture the high-frequency signals carried by anomalous nodes. Borrowing the idea from GCNII (Chen et al., 2020a), we adopt a residual propagation mechanism that prevents the high-frequency information of nodes from being overlooked due to over-smoothing in the multi-layer graph convolution process:

$$\mathbf{h}_v^l = \sigma\left(\mathbf{W}^{l-1}\left(\mathbf{h}_v^{l-1} - \frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\mathbf{h}_u^{l-1}\right)\right), \quad \mathbf{h}^{final} = AGG(\mathbf{h}_v^0, \mathbf{h}_v^1, \ldots, \mathbf{h}_v^L), \tag{9}$$

where L is the number of graph convolution layers and $AGG(\cdot)$ can be a simple aggregation function such as summation or concatenation. With this message-passing mechanism, we define our topological-aware denoising network as $\epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) = \epsilon_\theta(\mathbf{y}_t, t, \mathbf{H}^{final})$. For more details about the denoising network, please refer to Appendix G.

To execute our training, we sample $\mathbf{y}_t$ according to Equation 5. Through the reparameterization trick, we can derive:

$$\mathbf{y}_t = \sqrt{\bar{\alpha}_t}\mathbf{y}_0 + (1-\sqrt{\bar{\alpha}_t})g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1-\bar{\alpha}_t}\epsilon. \tag{10}$$

We simplify $\mathcal{L}_{recon}$ and $\mathcal{L}_{con}$ to obtain the final loss $\mathcal{L}$ as follows:

$$\mathcal{L}_\epsilon = ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{y}_0 + (1-\sqrt{\bar{\alpha}_t})g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1-\bar{\alpha}_t}\epsilon, t, \mathcal{E}, \mathbf{X})||^2 \tag{11}$$

Where elements in $\mathbf{t}$ is uniformly distributed between 1 and T. The case of $t = 1$ corresponds to $\mathcal{L}_{recon}$. Similar to DDPM, the cases where $t > 1$ correspond to an unweighted version of $\mathcal{L}_{con}$. The whole process of training is shown in Appendix H.

4.3 INFERENCE FOR ANOMALY DETECTION

In image synthesis tasks, DMs draw random Gaussian noises for reverse generation, and the generation results are guided by a pre-trained classifier or other signals such as textual queries. However, for generating anomaly scores on graphs, due to various deceptive and obfuscating tactics employed by anomalous nodes, generating directly from pure noise may not yield accurate anomaly detection results. Therefore, we propose a simple inference strategy that aligns with the CGADM training for anomaly inference, which is shown in Algorithm 1.

4.4 PRIOR-AWARE STRIDED SAMPLING

As can be seen from Equation 11, our training actually results in a topological-aware denoising network capable of denoising the predicted prior score at arbitrary time step t. Inspired by Song et al. (2021a), we can use this denoising network to perform time-step skipping sampling, greatly reducing the number of sampling steps. By discarding the Markov constraint brought by Equation 4, we can obtain the conditional non-Markovian reverse process different from Equation 6 as follows:

$$\mathbf{y}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{y}}_0 + (1-\sqrt{\bar{\alpha}_{t-1}})g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) + \sigma_t\epsilon_t \tag{13}$$

where $\hat{\mathbf{y}}_0$ is the denoised score in Equation 12. For detailed derivation, please refer to Appendix C. By substituting Equation 12 into Equation 13, we can obtain:

$$\begin{aligned}\mathbf{y}_{t-1} =& \sqrt{\frac{\bar{\alpha}_{t-1}}{\bar{\alpha}_t}}\left(\mathbf{y}_t - (1-\sqrt{\bar{\alpha}_t})g_\phi(\mathcal{E}, \mathbf{X}) - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X})\right) \\ &+ (1-\sqrt{\bar{\alpha}_{t-1}})g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) + \sigma_t\epsilon_t,\end{aligned} \tag{14}$$

Algorithm 1 Inference for Anomaly Detection

1: Initialize $\mathbf{y}_T \sim \mathcal{N}(g_\phi(\mathcal{E}, \mathbf{X}), I)$
2: **for** $t = T$ to 1 **do**
3: Calculate reparameterized $\hat{\mathbf{y}}_0$ according to Equation 10:

$$\hat{\mathbf{y}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{y}_t - (1 - \sqrt{\bar{\alpha}_t}) g_\phi(\mathcal{E}, \mathbf{X}) - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) \right) \tag{12}$$

4: **if** $t > 1$ **then**
5: Draw $z \sim \mathcal{N}(0, I)$
6: $\mathbf{y}_{t-1} = \gamma_0 \hat{\mathbf{y}}_0 + \gamma_1 \mathbf{y}_t + \gamma_2 g_\phi(\mathcal{E}, \mathbf{X}) + \tilde{\beta}_t z$, according to Equation 6.
7: **else**
8: Set $\mathbf{y}_{t-1} = \hat{\mathbf{y}}_0$
9: **end if**
10: **end for**
11: **return** y_0

This allows the use of a forward process defined only on a subset of the latent variables $\mathbf{y}_{\tau_1}, \ldots, \mathbf{y}_{\tau_t}$ where $\tau_1, \ldots, \tau_t$ is an increasing subsequence of 1, ..., T with length S, where S could be much smaller than T. To reduce the number of sampling steps from T to K, we use K evenly spaced real numbers between 1 and T (inclusive), and then round each resulting number to the nearest integer, as follows: $\{\tau_i\}_{i=1}^K = \left\{1 + \frac{(T-1)(i-1)}{K-1}\right\}_{i=1}^K$.

Intuitively, when our prior is more confident, our model can use fewer sampling steps, or a smaller K, and vice versa. We propose a heuristic strategy to dynamically adjust the size of K according to the confidence of different prior scores of anomalies. We choose the inverse sigmoid function to simulate the decay of the ratio as the confidence $|_\phi(\mathcal{E}, \mathbf{X}) - 0.5|$ increases:

$$K = \frac{r}{1 + \exp\left(\frac{|g_\phi(\mathcal{E}, \mathbf{X}) - 0.5|}{0.5}\right)} \times T \tag{15}$$

Typically, with r set to 2, our framework adjusts the sampling steps K to around 1000 for ambiguous priors near 0.5, and reduces it to about 500 for high-confidence priors close to 1. Notably, most nodes on the graph are associated with high prior confidence, which leads to a substantial decrease in computational demand. Conversely, for anomalous nodes that are adept at camouflage, the lower prior confidence necessitates a larger number of diffusion steps, facilitating their accurate detection. Our method thus strikes a balance between computational efficiency and thorough identification. We show the inference process with our prior-aware strided sampling in Appendix I.

5 Experiments

5.1 Experimental Setup

Datasets We have extensively employed five diverse datasets from various domains to verify our method. They are the e-finance category dataset Elliptic (Weber et al., 2019), crowd-sourcing category datasets Tolokers (Platonov et al., 2023) and YelpChi (Rayana & Akoglu, 2015), and Social media datasets Question (Platonov et al., 2023) and Reddit (Kumar et al., 2019). For the detail of dataset statistics and processing, please refer to Appendix F.

Baselines We have compared our CGADM with two categories of methods in the context of graph anomaly detection: (1) Standard GNNs, which include GCN (Kipf & Welling, 2017), GIN (Xu et al., 2019), GraphSAGE (Hamilton et al., 2017), and GAT (Velickovic et al., 2018), and (2) GNNs specifically designed for anomaly detection, such as GAS (Li et al., 2019), PCGNN (Liu et al., 2021b), BWGNN (Tang et al., 2022), and GHRN (Gao et al., 2023b). For detailed descriptions of these methods, please refer to Appendix D.

Metrics Following the evaluation setup employed by most anomaly detection works (Han et al., 2022a), we have chosen the Area Under the Receiver Operating Characteristic Curve (AUROC) and the Area Under the Precision-Recall Curve (AUPRC) as our metrics for graph anomaly detection.

Both of these metrics range between 0 and 1, and we record them as percentages for convenience. For both metrics, a higher value indicates better performance.

Implementation Details For CGADM, the layer number of graph convolution is set to three, a value considered reasonable by most works (Liu et al., 2021b). For our diffusion process, the noise levels at the initial and final time steps, β_1 and β_T, are set to 1e-4 and 0.02, respectively. Additionally, we employ linear interpolation to divide the time steps between them, which is consistent with DDPM (Ho et al., 2020). For other implementation details, please refer to Appendix J.

5.2 Overall Comparison

Table 1: Performance Comparison on Graph Anomaly Detection

	Ellip		Tolo		Yelp		Quest		Reddit	
Model	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC
GCN	80.19	95.12	41.44	73.58	23.59	59.89	10.27	67.73	5.65	62.55
GIN	83.88	96.21	37.89	74.02	38.13	77.40	11.23	68.07	5.38	<u>65.25</u>
Graphsage	86.16	96.61	43.73	77.30	<u>50.23</u>	<u>83.24</u>	13.86	<u>70.64</u>	5.78	63.67
GAT	87.59	97.11	42.18	76.66	46.64	80.95	13.19	68.19	5.42	63.55
GAS	87.54	<u>97.14</u>	42.39	74.55	39.18	78.63	12.41	66.09	5.66	61.23
PCGNN	67.29	93.88	36.76	71.28	45.32	79.61	13.79	69.12	4.13	54.58
BWGNN	87.90	96.99	45.02	77.80	49.15	81.85	<u>14.64</u>	69.96	5.42	60.63
GHRN	<u>88.13</u>	97.04	<u>45.25</u>	<u>77.98</u>	49.78	82.36	14.61	69.32	<u>5.85</u>	63.51
CGADM	**97.03**	**99.30**	**46.02**	**79.68**	**76.54**	**92.69**	**18.51**	69.41	5.79	**65.85**

† **Boldface** denotes the highest score, and underline indicates the best result of the baselines.

We summarize the performance of all algorithms in terms of AUROC and AUPRC across different datasets in Table 1. The results demonstrate that our CGADM outperforms most other baselines across all metrics. We conduct two-sample t-tests, and $p-$ value < 0.05 indicates that the improvements are statistically significant. In addition to these findings, we make the following observations:

- GAD methods such as GHRN and BWGNN represent state-of-the-art methods. This indicates that GAD, with its unique challenges of data imbalance, data heterogeneity, and deliberate node obfuscation, cannot be adequately addressed by general GNNs and requires specialized design.
- No single baseline method consistently outperforms on all datasets. We believe this is because these discriminative models identify anomalous nodes through decision boundaries. Many anomalous nodes manage to cross these boundaries by obfuscating their features, making it difficult for these methods to adapt to various scenarios. In contrast, our CGADM consider the joint distribution of anomaly in a generative way, making it difficult for anomalous nodes to obfuscate.
- Among standard GNN methods, GraphSage and GAT perform better than the other two methods, especially on the YelpChi dataset, which has significantly more edges. This aligns with our analysis in the introduction, where GNN, as a low-pass filter, blurs the distinctive features of anomalies in its inherent feature aggregation mechanism, a problem that worsens with an increased number of edges. GraphSage and GAT to some extent mitigate the over-smoothing issue by sampling neighbors or amplifying the weight of important neighbors, respectively.
- Our method performs exceptionally well on the edge-dense YelpChi dataset. This may be due to our topological-guided denoising network's use of a residual propagation mechanism. This mechanism effectively overcoming the over-smoothing problem during the generation process and ensuring that each node's anomaly distribution is influenced by its neighborhood distribution.

5.3 Ablation Studies

5.3.1 Comparison with Different Prior Model

In generating the final anomaly value with CGADM, to ensure effectiveness, we do not start the reverse process from a random state. Instead, we opt for a conditional anomaly estimator to guide the reverse process of the model. For efficiency, we employ a lightweight ensemble trees model as the estimator. Here, we explore both Random Forest (RF) and Extreme Gradient Boosting Tree (XGBT) as estimator. We denote CGADM using RF and XGBT as conditional anomaly estimators as CGADM$_{RF}$ and CGADM$_{XGBT}$, respectively. Figure 2 records the performance of these models

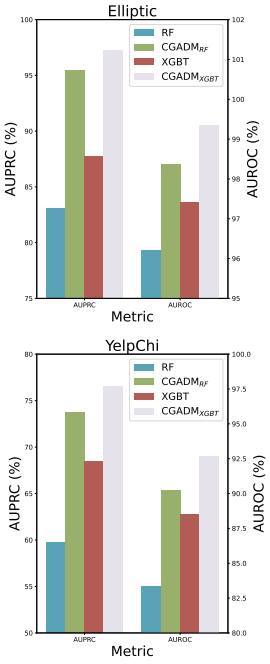

Figure 2: Performance w.r.t. Different Prior Models

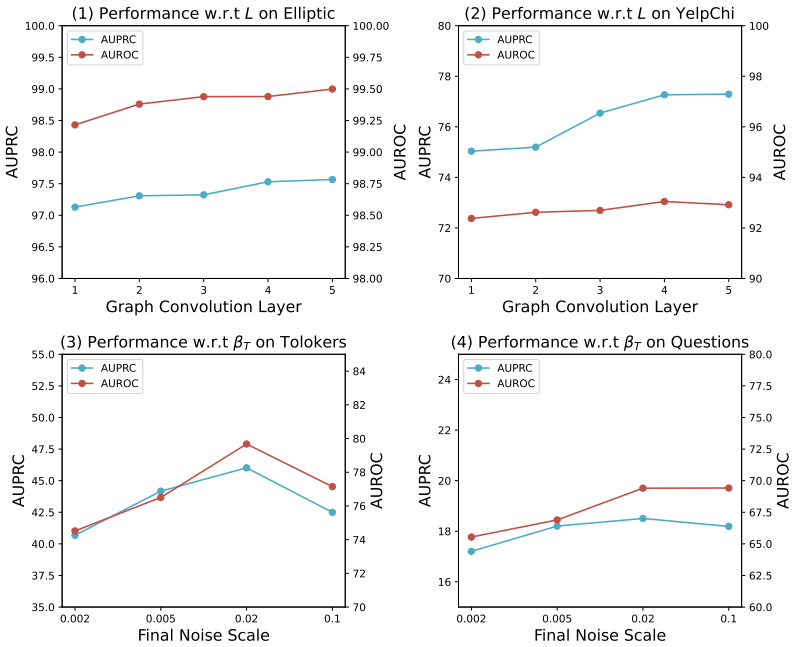

Figure 3: Parameter Sensitivity on Different Datasets

on the Elliptic and YelpChi datasets. Two observations can be made from figure 2. Firstly, both CGADM$_{RF}$ and CGADM$_{XGBT}$ outperform their corresponding initial priors. This proves that our CGADM's diffusion process can significantly enhance the performance of GAD. Secondly, the performance gap between CGADM$_{RF}$ and CGADM$_{XGBT}$ is significantly smaller than that between RF and XGBT. This indicates that our CGADM possesses strong robustness. Even in the face of initially inaccurate prior estimates, our CGADM can effectively correct the results under the iterative refinement of the topological-guided denoising network.

5.3.2 Parameter Sensitivity

Impact of Graph Convolution Layer L In order to better capture the topological information surrounding nodes for joint distribution modeling, we employ a GNN-based encoder in our topological-guided denoising network. We explored the impact of the number of graph convolution layers on the Elliptic and YelpChi datasets. The results are shown in Figures 3 (1) and (2). From the results, we can observe a slowly gradual improvement in performance as the number of layers increases, reaching farther topological structure information. Even at a depth of five layers, there is no performance degradation. This suggests that our CGADM can effectively overcome the over-smoothing problem commonly encountered in traditional discriminative methods based on GNNs. We attribute this mainly to two factors. First, the paradigm shift to generating the joint distribution of anomaly on the graph allows considering the influence of surrounding neighbor nodes. Second, our residual propagation mechanism prevents the high-frequency information of nodes, thereby retaining more valuable information for anomaly value generation.

Impact of the Final Noise Scale β_T We modify the endpoint of CGADM's diffusion process from the conventional Gaussian distribution $N(0, I)$ to $N(g_\phi(\mathcal{E}, \mathbf{X}), I)$. Intuitively, β_T represents the maximum degree to which our noise-added $\mathbf{y}_t$ can deviate from the ground truth. It also represents the maximum scale at which our denoising network can correct the prior. We studied the magnitude of this degree on the Tolokers and Questions datasets, with the results shown in Figure 3 (3) and (4). We can observe that as the maximum correction scale increases, the performance initially improves. This suggests that the bias of the prior can be better corrected at this point. However, when the correction scale exceeds 0.02, the performance begins to decline as the maximum correction scale continues to increase. This may because the maximum correction scale has already surpassed the

maximum bias produced by the prior. Overcorrection of the prior could prevent CGADM from modeling the true distribution. Therefore, we recommend using $\beta_T = 0.02$ in our cases,

5.4 Efficiency Analysis

In Section 4.4, we designed a prior-aware strided sampling strategy to adaptively reduce the reverse steps needed to generate anomaly values. To verify its efficiency, we designed the following two ablation experiments. In the first experiment, we tested the computation time and corresponding model performance of our CGADM with different sampling steps during generation. The results are shown in Figure 4. As can be seen, as our striding magnitude increases, i.e., the reverse steps of sampling become fewer, both computation time and model performance decrease. However, the decline in computation time is much greater than the decline in graph anomaly detection performance. Even when the striding is not large at the beginning, the decline in performance is not significant. This implies that sacrificing a little performance can result in substantial savings in computation time. Therefore, we designed another ablation experiment. Here, we denote CGADM configured with prior-aware strided sampling as CGADM$_s$ and present its model performance and average reverse steps during inference in Table 2. Compared to the original 1000 sampling steps, our method reduces the average sampling steps for all nodes to 583, while ensuring only a slight drop in model performance, which remains highly competitive.

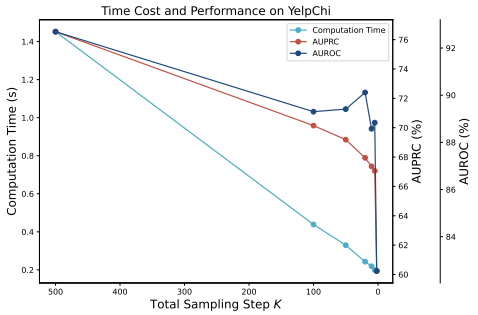

Figure 4: Time cost and Accuracy w.r.t. Sampling Steps K

	CGADM	CGADM$_S$
Average Reverse Step	1000	583.0256
AUPRC (%)	76.5424	73.6636
AUROC (%)	92.6930	91.9423

Table 2: Performance Metrics

6 Conclusions

Existing GNN-based graph anomaly detection methods are susceptible to fraudulent nodes in the network due to their inherent feature aggregation and discriminative characteristics. Therefore, we propose an advanced Conditional Graph Anomaly Diffusion Model (CGADM) that considers the interdependencies of node anomalies from a holistic graph perspective, thereby generating a distribution of anomaly values across the entire graph. To address the issue of effectiveness, we propose a prior-guided diffusion process, which injects a pre-trained conditional anomaly estimator to constrain the entire diffusion process. Based on this, we redesign the forward and reverse processes. To solve the efficiency issue, we introduce a prior confidence-aware mechanism to adaptively determine the reverse time step for each node, thus significantly saving computational expenses. Through experiments on standard benchmarks for graph anomaly detection, we demonstrate that CGADM achieves state-of-the-art results.

References

Ziwei Chai, Siqi You, Yang Yang, Shiliang Pu, Jiarong Xu, Haoyang Cai, and Weihao Jiang. Can abnormality be detected by graph neural networks? In Luc De Raedt (ed.), *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pp. 1945–1951. ijcai.org, 2022.

Bo Chen, Jing Zhang, Xiaokang Zhang, Yuxiao Dong, Jian Song, Peng Zhang, Kaibo Xu, Evgeny Kharlamov, and Jie Tang. GCCAD: graph contrastive coding for anomaly detection. *IEEE Trans. Knowl. Data Eng.*, 35(8):8037–8051, 2023a.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1725–1735. PMLR, 2020a.

Nan Chen, Zemin Liu, Bryan Hooi, Bingsheng He, Rizal Fathony, Jun Hu, and Jia Chen. Consistency training with learnable data augmentation for graph anomaly detection with limited supervision. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (eds.), *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 785–794. ACM, 2016.

Xiaohui Chen, Yukun Li, Aonan Zhang, and Li-Ping Liu. Nvdiff: Graph generation through the diffusion of node vectors. *CoRR*, abs/2211.10794, 2022.

Xiaohui Chen, Jiaxing He, Xu Han, and Liping Liu. Efficient and degree-guided graph generation via discrete diffusion modeling. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 4585–4610. PMLR, 2023b.

Zhenxing Chen, Bo Liu, Meiqing Wang, Peng Dai, Jun Lv, and Liefeng Bo. Generative adversarial attributed network anomaly detection. In Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (eds.), *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1989–1992. ACM, 2020b.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 8780–8794, 2021.

Kaize Ding, Jundong Li, Nitin Agarwal, and Huan Liu. Inductive anomaly detection on attributed networks. In Christian Bessiere (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pp. 1288–1294. ijcai.org, 2020.

Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (eds.), *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 315–324. ACM, 2020.

Jingcan Duan, Siwei Wang, Pei Zhang, En Zhu, Jingtao Hu, Hu Jin, Yue Liu, and Zhibin Dong. Graph anomaly detection via multi-scale contrastive learning networks with augmented view. In Brian Williams, Yiling Chen, and Jennifer Neville (eds.), *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pp. 7459–7467. AAAI Press, 2023.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *CoRR*, abs/1903.02428, 2019.

Yuan Gao, Xiang Wang, Xiangnan He, Zhenguang Liu, Huamin Feng, and Yongdong Zhang. Alleviating structural distribution shift in graph anomaly detection. In Tat-Seng Chua, Hady W. Lauw, Luo Si, Evimaria Terzi, and Panayiotis Tsaparas (eds.), *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023, Singapore, 27 February 2023 - 3 March 2023*, pp. 357–365. ACM, 2023a.

Yuan Gao, Xiang Wang, Xiangnan He, Zhenguang Liu, Huamin Feng, and Yongdong Zhang. Addressing heterophily in graph anomaly detection: A perspective of graph spectrum. In Ying Ding, Jie Tang, Juan F. Sequeda, Lora Aroyo, Carlos Castillo, and Geert-Jan Houben (eds.), *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pp. 1528–1538. ACM, 2023b.

Kilian Konstantin Haefeli, Karolis Martinkus, Nathanaël Perraudin, and Roger Wattenhofer. Diffusion models for graphs benefit from discrete state spaces. *CoRR*, abs/2210.01549, 2022.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 1024–1034, 2017.

Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022a.

Xizewen Han, Huangjie Zheng, and Mingyuan Zhou. CARD: classification and regression diffusion models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022b.

Junwei He, Qianqian Xu, Yangbangyan Jiang, Zitai Wang, and Qingming Huang. ADA-GAD: anomaly-denoised autoencoders for graph anomaly detection. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 8481–8489. AAAI Press, 2024.

Mingguo He, Zhewei Wei, Zengfeng Huang, and Hongteng Xu. Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 14239–14251, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Xuanwen Huang, Yang Yang, Yang Wang, Chunping Wang, Zhisheng Zhang, Jiarong Xu, Lei Chen, and Michalis Vazirgiannis. Dgraph: A large-scale financial dataset for graph anomaly detection. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10362–10383. PMLR, 2022.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Lingkai Kong, Jiaming Cui, Haotian Sun, Yuchen Zhuang, B. Aditya Prakash, and Chao Zhang. Autoregressive diffusion model for graph generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17391–17408. PMLR, 2023.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (eds.), *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pp. 1269–1278. ACM, 2019.

Ao Li, Zhou Qin, Runshi Liu, Yiqun Yang, and Dong Li. Spam review detection with graph convolutional networks. In Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (eds.), *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pp. 2703–2711. ACM, 2019.

Xuan Li, Chunjing Xiao, Ziliang Feng, Shikang Pang, Wenxin Tai, and Fan Zhou. Controlled graph neural networks with denoising diffusion for anomaly detection. *Expert Syst. Appl.*, 237(Part B): 121533, 2024.

Can Liu, Li Sun, Xiang Ao, Jinghua Feng, Qing He, and Hao Yang. Intention-aware heterogeneous graph attention networks for fraud transactions detection. In Feida Zhu, Beng Chin Ooi, and Chunyan Miao (eds.), *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pp. 3280–3288. ACM, 2021a.

Jenny Liu, Aviral Kumar, Jimmy Ba, Jamie Kiros, and Kevin Swersky. Graph normalizing flows. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 13556–13566, 2019.

Kay Liu, Hengrui Zhang, Ziqing Hu, Fangxin Wang, and Philip S. Yu. Data augmentation for supervised graph outlier detection with latent diffusion models. *CoRR*, abs/2312.17679, 2023.

Kay Liu, Huijun (Lona) Yu, Yao Yan, Ziqing Hu, Pankaj Rajak, Amila Weerasinghe, Olcay Boz, Deepayan Chakrabarti, and Fei Wang. Graph diffusion models for anomaly detection. In *WSDM 2024*, 2024. URL `https://www.amazon.science/publications/graph-diffusion-models-for-anomaly-detection`.

Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. Pick and choose: A gnn-based imbalanced learning approach for fraud detection. In Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (eds.), *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pp. 3168–3177. ACM / IW3C2, 2021b.

Zhiwei Liu, Yingtong Dou, Philip S. Yu, Yutong Deng, and Hao Peng. Alleviating the inconsistency problem of applying graph neural network to fraud detection. In Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (eds.), *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pp. 1569–1572. ACM, 2020.

Xiaoxiao Ma, Ruikun Li, Fanzhen Liu, Kaize Ding, Jian Yang, and Jia Wu. Graph anomaly detection with few labels: A data-centric approach. In Ricardo Baeza-Yates and Francesco Bonchi (eds.), *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pp. 2153–2164. ACM, 2024a.

Xiaoxiao Ma, Ruikun Li, Fanzhen Liu, Kaize Ding, Jian Yang, and Jia Wu. New recipes for graph anomaly detection: Forward diffusion dynamics and graph generation, 2024b.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In Silvia Chiappa and

Roberto Calandra (eds.), *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pp. 4474–4484. PMLR, 2020.

Kushagra Pandey, Avideep Mukherjee, Piyush Rai, and Abhishek Kumar. Diffusevae: Efficient, controllable and high-fidelity generation from low-dimensional latents. *Trans. Mach. Learn. Res.*, 2022, 2022.

Shikang Pang, Chunjing Xiao, Wenxin Tai, Zhangtao Cheng, and Fan Zhou. Graph anomaly detection with diffusion model-based graph enhancement (student abstract). In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 23610–23612. AAAI Press, 2024.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 8024–8035, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, 2011.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Hezhe Qiao, Qingsong Wen, Xiaoli Li, Ee-Peng Lim, and Guansong Pang. Generative semi-supervised graph anomaly detection. *CoRR*, abs/2402.11887, 2024.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *CoRR*, abs/2204.06125, 2022.

Shebuti Rayana and Leman Akoglu. Collective opinion spam detection: Bridging review networks and metadata. In Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams (eds.), *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pp. 985–994. ACM, 2015.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 10674–10685. IEEE, 2022.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021a.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 11895–11907, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021b.

Jianheng Tang, Jiajin Li, Ziqi Gao, and Jia Li. Rethinking graph neural networks for anomaly detection. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 21076–21089. PMLR, 2022.

Jianheng Tang, Fengrui Hua, Ziqi Gao, Peilin Zhao, and Jia Li. Gadbench: Revisiting and benchmarking supervised graph anomaly detection. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Daixin Wang, Yuan Qi, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, and Shuang Yang. A semi-supervised graph attentive network for financial fraud detection. In Jianyong Wang, Kyuseok Shim, and Xindong Wu (eds.), *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, pp. 598–607. IEEE, 2019a.

Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, and Jian Xiong. Fdgars: Fraudster detection via graph convolutional networks in online app review system. In Sihem Amer-Yahia, Mohammad Mahdian, Ashish Goel, Geert-Jan Houben, Kristina Lerman, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia (eds.), *Companion of The 2019 World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pp. 310–316. ACM, 2019b.

Yanling Wang, Jing Zhang, Shasha Guo, Hongzhi Yin, Cuiping Li, and Hong Chen. Decoupling representation learning and classification for gnn-based anomaly detection. In Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (eds.), *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pp. 1239–1248. ACM, 2021.

Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I. Weidele, Claudio Bellei, Tom Robinson, and Charles E. Leiserson. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. *CoRR*, abs/1908.02591, 2019.

Chunjing Xiao, Shikang Pang, Xovee Xu, Xuan Li, Goce Trajcevski, and Fan Zhou. Counterfactual data augmentation with denoising diffusion for graph anomaly detection. *CoRR*, abs/2407.02143, 2024.

Fan Xu, Nan Wang, Hao Wu, Xuezhi Wen, Xibin Zhao, and Hai Wan. Revisiting graph-based fraud detection in sight of heterophily and spectrum. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 9214–9222. AAAI Press, 2024.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Yang Yang, Yuhong Xu, Yizhou Sun, Yuxiao Dong, Fei Wu, and Yueting Zhuang. Mining fraudsters and fraudulent strategies in large-scale mobile social networks. *IEEE Trans. Knowl. Data Eng.*, 33(1):169–179, 2021.

Kai Yi, Bingxin Zhou, Yiqing Shen, Pietro Lió, and Yuguang Wang. Graph denoising diffusion for inverse protein folding. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Ge Zhang, Jia Wu, Jian Yang, Amin Beheshti, Shan Xue, Chuan Zhou, and Quan Z. Sheng. FRAUDRE: fraud detection dual-resistant to graph inconsistency and imbalance. In James Bailey, Pauli Miettinen, Yun Sing Koh, Dacheng Tao, and Xindong Wu (eds.), *IEEE International Conference on Data Mining, ICDM 2021, Auckland, New Zealand, December 7-10, 2021*, pp. 867–876. IEEE, 2021.

Tong Zhao, Tianwen Jiang, Neil Shah, and Meng Jiang. A synergistic approach for graph anomaly detection with pattern mining and feature learning. *IEEE Trans. Neural Networks Learn. Syst.*, 33(6):2393–2405, 2022.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

A Common Process of Diffusion Probabilistic Model

Here we show the common steps in the diffusion model as follows:

- **Forward process:** Given an input data sample $x_0 \sim q(x_0)$, the forward process constructs the latent variables $x_{1:T}$ in a Markov chain by progressively adding Gaussian noises over T steps. Specifically, the forward transition $x_{t-1} \to x_t$ is defined as $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$, where $t \in \{1, ..., T\}$ refers to the diffusion step, $\mathcal{N}$ denotes the Gaussian distribution, and $\beta_t \in (0, 1)$ regulates the noise scales added at step t. If $T \to \infty$, x_T approaches a standard Gaussian distribution (Ho et al., 2020).
- **Reverse process:** Diffusion models (DMs) aim to remove the added noises from x_t to recover x_{t-1} in the reverse step, striving to capture minor alterations in the complex generation process. Formally, taking x_T as the initial state, DMs learn the denoising process $x_t \to x_{t-1}$ iteratively by $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$, where $\mu_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$ are the mean and covariance of the Gaussian distribution predicted by a neural network with parameters θ.
- **Optimization:** DMs are optimized by maximizing the Evidence Lower Bound (ELBO) of the likelihood of observed input data x_0. Denote $\mathbb{D}_{KL}(p||q)$ as the Kullback–Leibler (KL) divergence from distribution p to distribution q:

$$\begin{aligned}
\log p(x_0) &= \log \int p(x_{0:T}) dx_{1:T} = \log \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p(x_{0:T})}{q(x_{1:T}|x_0)} \right] \\
&\geq \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p(x_{0:T})}{q(x_{1:T}|x_0)} \right] \\
&= \mathbb{E}_{q(x_1|x_0)} \left[\log p_\theta(x_0|x_1) \right] - \mathbb{D}_{KL}(q(x_T|x_0)||p(x_T)) \\
&\quad - \sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)} \left[\mathbb{D}_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \right]
\end{aligned} \tag{16}$$

- **Inference:** After training θ, DMs can draw $x_T \sim \mathcal{N}(0, I)$ and use $p_\theta(x_{t-1}|x_t)$ to iteratively repeat the generation process $x_T \to x_{T-1} \to \ldots \to x_0$.

B Posterior Coefficients Derivation

Similar to Han et al. (2022b), here we give the detailed derivation of Equation 6 and 7.

$$
\begin{aligned}
q(\mathbf{y}_{t-1}|\mathbf{y}_t,\mathbf{y}_0,\mathcal{E},\mathbf{X}) &= q(\mathbf{y}_{t-1}|\mathbf{y}_t,\mathbf{y}_0,g_\phi(\mathcal{E},\mathbf{X})) \propto q(\mathbf{y}_t|\mathbf{y}_{t-1},g_\phi(\mathcal{E},\mathbf{X}))q(\mathbf{y}_{t-1}|\mathbf{y}_0,g_\phi(\mathcal{E},\mathbf{X})) \\
&\propto \exp\left(-\frac{1}{2}\left(\frac{\left(\boldsymbol{y}_t-\left(1-\sqrt{\alpha_t}\right)g_\phi(\mathcal{E},\mathbf{X})-\sqrt{\alpha_t}\boldsymbol{y}_{t-1}\right)^2}{\beta_t}\right.\right. \\
&\left.\left.+\frac{\left(\boldsymbol{y}_{t-1}-\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{y}_0-\left(1-\sqrt{\bar{\alpha}_{t-1}}\right)g_\phi(\mathcal{E},\mathbf{X})\right)^2}{1-\bar{\alpha}_{t-1}}\right)\right) \\
&\propto \exp\left(-\frac{1}{2}\left(\frac{\alpha_t\boldsymbol{y}_{t-1}^2-2\sqrt{\alpha_t}\left(\boldsymbol{y}_t-\left(1-\sqrt{\alpha_t}\right)g_\phi(\mathcal{E},\mathbf{X})\right)\boldsymbol{y}_{t-1}}{\beta_t}\right.\right. \\
&\left.\left.+\frac{\boldsymbol{y}_{t-1}^2-2\left(\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{y}_0+\left(1-\sqrt{\bar{\alpha}_{t-1}}\right)g_\phi(\mathcal{E},\mathbf{X})\right)\boldsymbol{y}_{t-1}}{1-\bar{\alpha}_{t-1}}\right)\right) \\
&= \exp(-\frac{1}{2}((\underbrace{\frac{\alpha_t}{\beta_t}+\frac{1}{1-\bar{\alpha}_{t-1}}}_{\text{Term 1}})\boldsymbol{y}_{t-1}^2 \\
&\quad \underbrace{-2(\frac{\sqrt{\alpha_{t-1}}}{1-\bar{\alpha}_{t-1}}\boldsymbol{y}_0+\frac{\sqrt{\alpha_t}}{\beta_t}\boldsymbol{y}_t+\left(\frac{\sqrt{\alpha_t}\left(\sqrt{\alpha_t}-1\right)}{\beta_t}+\frac{1-\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\right)g_\phi(\mathcal{E},\mathbf{X}))\boldsymbol{y}_{t-1}))}_{\text{Term 2}},
\end{aligned}
\tag{17}
$$

where

$$
\text{Term 1} = \frac{\alpha_t\left(1-\bar{\alpha}_{t-1}\right)+\beta_t}{\beta_t\left(1-\bar{\alpha}_{t-1}\right)} = \frac{1-\bar{\alpha}_t}{\beta_t\left(1-\bar{\alpha}_{t-1}\right)}, \tag{18}
$$

$$
\tilde{\beta}_t = \frac{1}{(1)} = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t, \tag{19}
$$

Afterwards, we divide each coefficient in Term 2 by Term 1.

$$
\gamma_0 = \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}/1 = \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t}\beta_t \tag{20}
$$

$$
\gamma_1 = \frac{\sqrt{\alpha_t}}{\beta_t}/1 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\sqrt{\alpha_t}, \tag{21}
$$

and

$$
\begin{aligned}
\gamma_2 &= \left(\frac{\sqrt{\alpha_t}\left(\sqrt{\alpha_t}-1\right)}{\beta_t}+\frac{1-\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\right)/1 \\
&= \frac{\alpha_t-\bar{\alpha}_t-\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)+\beta_t-\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t} \\
&= 1+\frac{\left(\sqrt{\bar{\alpha}_t}-1\right)\left(\sqrt{\alpha_t}+\sqrt{\bar{\alpha}_{t-1}}\right)}{1-\bar{\alpha}_t}.
\end{aligned}
\tag{22}
$$

Finally, we put every γ_0, γ_1, and γ_2 together and obtain Equation 6 and 7.

$$
\tilde{\boldsymbol{\mu}}\left(\boldsymbol{y}_t,\boldsymbol{y}_0,g_\phi(\mathcal{E},\mathbf{X})\right) = \gamma_0\boldsymbol{y}_0+\gamma_1\boldsymbol{y}_t+\gamma_2 g_\phi(\mathcal{E},\mathbf{X}) \tag{23}
$$

C Derivation of conditional non-Markovian reverse process

Following DDIM, we formally carry out the derivation of discarding the Markov constraint introduced by Equation 4 in our prior-conditional reverse step Equation 6. First, let's organize our target: given $q\left(\mathbf{y}_t \mid \mathbf{y}_0, g_\phi(\mathcal{E},\mathbf{X})\right)$ and $q\left(\mathbf{y}_{t-1} \mid \mathbf{y}_0, g_\phi(\mathcal{E},\mathbf{X})\right)$, without $q\left(\mathbf{y}_t \mid \mathbf{y}_{t-1}\right)$, we aim to find $q\left(\mathbf{y}_{t-1} \mid \mathbf{y}_t, \mathbf{y}_0, g_\phi(\mathcal{E},\mathbf{X})\right)$.

Here we assume that $\mathbf{y}_{t-1}$ is a linear combination of $\mathbf{y}_t$, $\mathbf{y}_0$ and prior $g_\phi(\mathcal{E}, \mathbf{X})$ with coefficients denoted as m_t, n_t and o_t, respectively. That is,

$$\mathbf{y}_{t-1} = m_t \mathbf{y}_t + n_t \mathbf{y}_0 + o_t g_\phi(\mathcal{E}, \mathbf{X}) + \sigma_t \epsilon_1 \tag{24}$$

We also know that

$$\mathbf{y}_t = \sqrt{\bar{\alpha}_t}\mathbf{y}_0 + (1 - \sqrt{\bar{\alpha}_t}) g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1 - \bar{\alpha}_t}\epsilon_2, \tag{25}$$

$$\mathbf{y}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{y}_0 + (1 - \sqrt{\bar{\alpha}_{t-1}}) g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1 - \bar{\alpha}_{t-1}}\epsilon_3. \tag{26}$$

Here, the subscripts of ϵ_n are used to distinguish different samples from the Gaussian distribution. Substituting Equation 25 into Equation 24, we get

$$\begin{aligned}
\mathbf{y}_{t-1} &= m_t \left(\sqrt{\bar{\alpha}_t}\mathbf{y}_0 + (1 - \sqrt{\bar{\alpha}_t}) g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1 - \bar{\alpha}_t}\epsilon_2\right) + n_t \mathbf{y}_0 + o_t g_\phi(\mathcal{E}, \mathbf{X}) + \sigma_t \epsilon_1 && (27)\\
&= \left(m_t\sqrt{\bar{\alpha}_t} + n_t\right)\mathbf{y}_0 + (m_t - m_t\sqrt{\bar{\alpha}_t} + o_t) g_\phi(\mathcal{E}, \mathbf{X}) + m_t\sqrt{1 - \bar{\alpha}_t}\epsilon_2 + \sigma_t \epsilon_1 && (28)
\end{aligned}$$

Therefore, we have

$$m_t\sqrt{\bar{\alpha}_t} + n_t = \sqrt{\bar{\alpha}_{t-1}}, \tag{29}$$

$$m_t^2 (1 - \alpha_t) + \sigma_t^2 = 1 - \bar{\alpha}_{t-1}, \tag{30}$$

$$m_t - m_t\sqrt{\bar{\alpha}_t} + o_t = 1 - \sqrt{\bar{\alpha}_{t-1}} \tag{31}$$

Immediately, we can calculate m_t and n_t:

$$m_t = \sqrt{\frac{1 - \bar{\alpha}_{t-1} - \sigma_t^2}{1 - \bar{\alpha}_t}}, \tag{32}$$

$$n_t = \sqrt{\bar{\alpha}_{t-1}} - \sqrt{\frac{\bar{\alpha}_t}{1 - \bar{\alpha}_t}(1 - \bar{\alpha}_{t-1} - \sigma_t^2)}, \tag{33}$$

$$o_t = 1 - \sqrt{\bar{\alpha}_{t-1}} - \sqrt{\frac{1 - \bar{\alpha}_{t-1} - \sigma_t^2}{1 - \bar{\alpha}_t}}(1 - \sqrt{\bar{\alpha}_t}). \tag{34}$$

Substituting back into Equation 24, we have

$$\begin{aligned}
\mathbf{y}_{t-1} &= \sqrt{\frac{1 - \bar{\alpha}_{t-1} - \sigma_t^2}{1 - \bar{\alpha}_t}}\mathbf{y}_t + \left(\sqrt{\bar{\alpha}_{t-1}} - \sqrt{\frac{\bar{\alpha}_t}{1 - \bar{\alpha}_t}(1 - \bar{\alpha}_{t-1} - \sigma_t^2)}\right)\mathbf{y}_0 \\
&\quad + (1 - \sqrt{\bar{\alpha}_{t-1}} - \sqrt{\frac{1 - \bar{\alpha}_{t-1} - \sigma_t^2}{1 - \bar{\alpha}_t}}(1 - \sqrt{\bar{\alpha}_t})) g_\phi(\mathcal{E}, \mathbf{X}) + \sigma_t \epsilon && (35)\\
&= \sqrt{\bar{\alpha}_{t-1}}\mathbf{y}_0 + (1 - \sqrt{\bar{\alpha}_{t-1}}) g_\phi(\mathcal{E}, \mathbf{X}) \\
&\quad + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\left(\frac{1}{\sqrt{1 - \bar{\alpha}_t}}\mathbf{y}_t - \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1 - \bar{\alpha}_t}}\mathbf{y}_0 - \frac{1 - \sqrt{\bar{\alpha}_t}}{\sqrt{1 - \bar{\alpha}_t}} g_\phi(\mathcal{E}, \mathbf{X})\right) + \sigma_t \epsilon && (36)\\
&= \sqrt{\bar{\alpha}_{t-1}}\mathbf{y}_0 + (1 - \sqrt{\bar{\alpha}_{t-1}}) g_\phi(\mathcal{E}, \mathbf{X}) \\
&\quad + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\frac{\mathbf{y}_t - \sqrt{\bar{\alpha}_t}\mathbf{y}_0 - (1 - \sqrt{\bar{\alpha}_t}) g_\phi(\mathcal{E}, \mathbf{X})}{\sqrt{1 - \bar{\alpha}_t}} + \sigma_t \epsilon && (37)
\end{aligned}$$

Substituting the model's predicted value, we have

$$\mathbf{y}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{y}}_{0|t} + (1 - \sqrt{\bar{\alpha}_{t-1}}) g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) + \sigma_t \epsilon \tag{38}$$

At this point, the derived result Equation 38 is completely consistent with Equation 14. That is, we use the two conditions $q(\mathbf{y}_t \mid \mathbf{y}_0, g_\phi(\mathcal{E}, \mathbf{X}))$ and $q(\mathbf{y}_{t-1} \mid \mathbf{y}_0, g_\phi(\mathcal{E}, \mathbf{X}))$, without $q(\mathbf{y}_t \mid \mathbf{y}_{t-1})$, and obtain $q(\mathbf{y}_{t-1} \mid \mathbf{y}_t, \mathbf{y}_0, g_\phi(\mathcal{E}, \mathbf{X}))$. DDPM removes the condition $q(\mathbf{y}_t \mid \mathbf{y}_{t-1})$, leading to the more general DDIM sampling formula.

D BASELINES

In this section, we introduce the baseline models, which can be broadly bifurcated into two categories: (1) General-purpose graph neural networks, and (2) Techniques specifically designed for graph anomaly detection. We have annotated each model with their respective categories for easy differentiation.

- **GCN** (Kipf & Welling, 2017) (1): This technique employs the convolution operation on graphs to propagate information from a node to its adjacent nodes. This allows the network to learn a representation for each node, grounded on its local neighborhood.
- **GIN** (Xu et al., 2019) (1): A variant of GNN, GIN is designed to encapsulate the graph's structure while maintaining graph isomorphism. This implies that it yields identical embeddings for graphs that are structurally indistinguishable, irrespective of permutations in their node labels.
- **GraphSAGE** (Hamilton et al., 2017) (1): This is an inductive learning framework that generates node embeddings by sampling and aggregating features from a node's local neighborhood.
- **GAT** (Velickovic et al., 2018) (1): This GNN framework incorporates the attention mechanism, assigning varying degrees of importance to different nodes during the neighborhood information aggregation process. This enables the model to concentrate on the most informative neighbors.
- **GAS** (Li et al., 2019) (2): This is a highly scalable technique for detecting spam reviews. It expands GCN to manage heterogeneous and heterophilic graphs and adapts to the graph structure of specific GAD applications using the KNN algorithm.
- **PCGNN** (Liu et al., 2021b) (2): This framework is designed for imbalanced GNN learning in fraud detection. It employs a label-balanced sampler to select nodes and edges for training, leading to a balanced label distribution in the induced sub-graph. Additionally, it uses a learnable parameterized distance function to select neighbors, filtering out superfluous links and incorporating beneficial ones for fraud prediction.
- **BWGNN** (Tang et al., 2022) (2): This technique is proposed to address the 'right-shift' phenomenon of graph anomalies, where the spectral energy distribution focuses less on low frequencies and more on high frequencies. It utilizes the Beta kernel to tackle higher frequency anomalies through multiple flexible, spatial/spectral-localized, and band-pass filters.
- **GHRN** (Gao et al., 2023b) (2): This approach addresses the heterophily issue in the spectral domain of graph anomaly detection by pruning inter-class edges to highlight and outline the graph's high-frequency components.

E CHALLENGE OF GRAPH ANOMALY DETECTION

Although GAD is essentially a binary node classification problem, it presents several unique challenges. Firstly, anomalous nodes typically constitute a small fraction of the total nodes, leading to a significant data imbalance (Liu et al., 2021b). Secondly, graphs containing anomalies often exhibit strong heterophily, where connected nodes possess diverse features and labels (Gao et al., 2023b; Tang et al., 2023). This heterophily necessitates the development of methods that can effectively handle neighborhood feature disparities during message passing. Lastly, anomalous nodes tend to camouflage their features and connections, striving to blend in by mimicking normal patterns within the graph (Liu et al., 2020).

F DETAILS OF THE DATASETS

The detailed statistics of the datasets we used are in Table 3. In line with the data characteristics of anomaly detection, the selected datasets each contain over 100 anomaly points, and the proportion of anomalies does not exceed 25%, satisfying the inherent imbalance problem in graph anomaly detection (Tang et al., 2023). For each dataset, we randomly selected 20% of the points as training data, 10% of the points as validation data, and the remaining points as test data.

Table 3: Descriptive statistics of the datasets.

	#Nodes	#Edges	Feature Dim	Anomaly Ratio	Feature Type
Elliptic	203,769	234,355	166	9.8%	Timestamps and transaction information
Tolokers	11,758	519,000	10	21.8%	User profile with task performance statistics
YelpChi	45,954	3,846,979	32	14.5%	Hand-crafted review features and statistics
Questions	48,921	153,540	301	3.0%	FastText embeddings for user descriptions
Reddit	10,984	168,016	64	3.3%	Hand-crafted review features and statistics

G Implementation of Topological-guided Denoising Network

Reflecting upon Equation 9, we initially extend the formula of graph convolution to matrix form to facilitate computation across the entire graph, as shown below:

$$\mathbf{H}^{l} = \sigma(\mathbf{W}^{l-1}(\mathbf{I} - \mathbf{D}^{-1}\mathbf{A}\mathbf{H}^{l-1}))$$

After conducting L rounds of convolution, we use weighted summation as our aggregation function for the hidden representations obtained from each layer of graph convolution. The formula is as follows:

$$\mathbf{H}^{final} = AGG(\mathbf{H}^{1}, \mathbf{H}^{2}, \ldots, \mathbf{H}^{L}) = \sum_{l=0}^{L} \alpha_l \mathbf{H}^{l}$$

Here, α_l are the weights for each layer's representation, which can be learned during training. Having obtained the representation of nodes that integrates both topological structure and node features, we construct our denoising function $\epsilon_\theta(\mathbf{y}_t, t, \mathbf{H}^{final})$ through a Multilayer Perceptron (MLP). Following the original DDPM Ho et al. (2020), we also adopt position embedding to encode time t. Therefore, the denoising function ϵ_θ is as follows:

$$\epsilon_\theta = MLP(\text{Concat}[Pos(\mathbf{t}), \mathbf{y}_t, \mathbf{H}^{final}])$$

In this equation, $Pos(\mathbf{t})$ represents the position embedding of time $mathbft$, $\mathbf{y}_t$ is the current representation of the nodes, and $\mathbf{H}^{final}$ is the final aggregated representation after L layers of graph convolution.

H Training of CGADM

According to the loss in Equation 11, the pseudo algorithm for training is shown in Algorithm 2

Algorithm 2 CGADM Training

1: Pre-train $g_\phi(\mathcal{E}, \mathbf{X})$ that predicts the anomaly prior
2: **repeat**
3: Draw $\mathbf{t} \sim \text{Uniform}(\{1, \ldots, T\})$
4: Draw $\epsilon \sim \mathcal{N}(0, I)$
5: Compute the noise estimation loss:

$$\mathcal{L}_\epsilon = ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{y}_0 + (1 - \sqrt{\bar{\alpha}_t})g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1 - \bar{\alpha}_t}\epsilon, t, \mathcal{E}, \mathbf{X})||^2$$

6: Take a numerical optimization step on $\nabla_\theta L_\epsilon$
7: **until** Convergence

I Inference with Prior-aware Strided Sampling

We show the complete pseudo algorithm for inference with our prior-aware strided sampling strategy in Algorithm 3

Algorithm 3 Inference for Anomaly Detection with Sampling Strategy

1: Initialize $\mathbf{y}_T \sim \mathcal{N}(g_\phi(\mathcal{E}, \mathbf{X}), I)$
2: Compute K based on the prior confidence $|g_\phi(\mathcal{E}, \mathbf{X}) - 0.5|$ using:

$$K = \frac{r}{1 + \exp\left(\frac{|g_\phi(\mathcal{E}, \mathbf{X}) - 0.5|}{0.5}\right)} \times T$$

where r is a hyperparameter.
3: Generate sampling time steps $\{\tau_i\}_{i=1}^{K}$:

$$\tau_i = \left\lfloor 1 + \frac{(T-1)(i-1)}{K-1} \right\rfloor, \quad i = 1, \ldots, K$$

4: **for** $i = K$ to 1 **do**
5: Set $t = \tau_i$
6: Calculate reparameterized $\hat{\mathbf{y}}_0$ using Equation 12:

$$\hat{\mathbf{y}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{y}_t - (1 - \sqrt{\bar{\alpha}_t}) g_\phi(\mathcal{E}, \mathbf{X}) - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) \right)$$

7: **if** $i > 1$ **then**
8: Draw $z \sim \mathcal{N}(0, I)$
9: Update $\mathbf{y}_{t-1}$ using the modified non-Markovian reverse process:

$$\mathbf{y}_{t-1} = \sqrt{\bar{\alpha}_{\tau_{i-1}}} \hat{\mathbf{y}}_0 + (1 - \sqrt{\bar{\alpha}_{\tau_{i-1}}}) g_\phi(\mathcal{E}, \mathbf{X}) + \sqrt{1 - \bar{\alpha}_{\tau_{i-1}} - \sigma_t^2} \epsilon_\theta(\mathbf{y}_t, t, \mathcal{E}, \mathbf{X}) + \sigma_t z$$

10: **else**
11: Set $\mathbf{y}_{t-1} = \hat{\mathbf{y}}_0$
12: **end if**
13: **end for**
14: **return** $\mathbf{y}_0$

J Implementation Detail

All experiments were conducted on a Linux machine equipped with an Nvidia GeForce RTX 3090. The CUDA version used was 11.1, and the driver version was 455.45.01. We implemented our algorithm and the corresponding baseline methods using PyTorch (Paszke et al., 2019) and the graph computation framework Pytorch-Geometric (Fey & Lenssen, 2019). For the Random Forest (RF) and Extreme Gradient Boosting Tree (XGBT) that serve as conditional anomaly estimators, we used the RF version implemented in the Scikit-Learn library Pedregosa et al. (2011). For XGBoost Chen & Guestrin (2016), we utilized its official implementation.

We initialize the latent vectors for all models with a Gaussian Distribution, having a mean value of 0 and a standard deviation of 0.01. To ensure a level playing field, the dimension of the hidden layer for all baseline models, as well as our CGADM, is set to 64. We conducted a grid search for hyper-parameter tuning. The learning rates were selected from the set [0.005, 0.01, 0.02, 0.05]. To prevent overfitting, we incorporated an L2 norm with the coefficient tuned from the set [0.001, 0.005, 0.01, 0.02, 0.1]. For all methods, we selected the best models by implementing early stopping when the AUROC on the validation set did not increase for five consecutive epochs.

K Efficacy in Highly Imbalanced Scenarios

We conducted additional experiments on the **DGraph** dataset Huang et al. (2022), a highly imbalanced real-world financial fraud detection dataset where anomalies constitute only **1.3%** of the data. The results are presented in Table 4:

As Table 4 illustrates, CGADM consistently outperforms all baseline methods on both AUPRC and AUROC metrics in this **extremely imbalanced setting**. Notably, the AUPRC metric demonstrates CGADM's ability to handle rare event detection by excelling in anomaly-specific precision and recall. Similarly, the superior AUROC indicates robust overall discriminative performance.

Method	AUPRC	AUROC
GCN	3.66	74.97
GIN	3.22	73.14
GraphSAGE	3.43	73.81
GAT	3.65	75.17
GAS	2.91	71.21
PCGNN	2.82	71.78
BWGNN	3.63	75.16
GHRN	3.68	75.15
CGADM	**3.83**	**76.43**

Table 4: Performance comparison on the DGraph dataset.

Metric	Dataset	GODM	CGenGA	CGADM (Ours)
AUPRC	Ellip	85.89	87.36	**97.03**
	Tolo	**46.15**	44.89	46.02
	Yelp	51.77	52.76	**76.54**
	Quest	15.11	15.34	**18.51**
	Reddit	5.55	5.78	**5.79**
AUROC	Ellip	93.92	96.07	**99.34**
	Tolo	76.42	78.95	**79.68**
	Yelp	84.33	85.65	**92.69**
	Quest	68.86	68.46	**69.41**
	Reddit	62.10	64.78	**65.85**

Table 5: Comparisons with Diffusion-based Data-centric Approaches

L Comparisons with Diffusion-based Data-centric Approaches

We have conducted experiments to compare CGADM against the methods in GODM (Ma et al., 2024a) and CGenGA (Liu et al., 2023) on five benchmark datasets (*Elliptic*, *Tolo*, *Yelp*, *Quest*, and *Reddit*). For fair comparisons, we implemented the diffusion-based data-centric approaches following the settings and optimal detector configurations specified in their respective papers. We summarize the results in terms of **AUPRC** and **AUROC** in Table 5:

Our results demonstrate that CGADM consistently outperforms GODM Ma et al. (2024a) and CGenGA Liu et al. (2023) across almost all datasets in both AUPRC and AUROC metrics. This superior performance underscores the advantages of our generative framework in directly modeling the joint anomaly distribution, as opposed to relying on downstream discriminative classifiers.

M Empirical Results on Efficiency

To provide concrete evidence, we conducted experiments to compare memory usage and inference time with all the baselines specifically designed for anomaly detection on the *Elliptic* dataset, which contains 203,769 nodes and 234,355 edges. The results are summarized in Table 6:

Model	Memory (MB)	Inference Time (s)
GAS	1418	2.3865
PCGN	914	0.0827
BWGNN	446	0.1185
GHRN	924	0.1249
CGADM (ours)	**1048**	**0.5691**

Table 6: Memory usage and inference time comparison on the Elliptic dataset.

We have the following observation:

- **Memory Efficiency:** The use of sparse matrix computations ensures that CGADM remains efficient in terms of memory usage, even for large-scale graphs. The marginal increase in memory usage is negligible compared to the scalability benefits.
- **Inference Time:** While our inference time is higher than most discriminative methods, the increase is justified given the novel generative anomaly detection paradigm. Considering the already low baseline inference time of anomaly detection tasks, the additional time overhead is acceptable, especially in scenarios where performance improvements are critical.

N Additional Experiment Results

We have also conducted experiments comparing our Conditional Graph Anomaly Diffusion Model (CGADM) with XGBGraph (Tang et al., 2023) and CONSISGAD (Chen et al., 2024) on the same datasets. Below, we present the results in terms of **AUPRC** and **AUROC** in Table 7 and 8, two widely used metrics in the anomaly detection domain:

Model	Ellip	Tolo	Yelp	Quest	Reddit
XGBGraph	90.47	44.47	75.91	14.33	4.59
CONSISGAD	86.42	40.59	41.74	12.85	5.57
Ours (CGADM)	**97.03**	**46.02**	**76.54**	**18.51**	**5.79**

Table 7: Comparison of AUPRC results with XGBGraph and CONSISGAD.

Model	Ellip	Tolo	Yelp	Quest	Reddit
XGBGraph	94.35	77.28	91.85	64.90	60.58
CONSISGAD	96.38	76.03	79.35	70.54	66.99
Ours (CGADM)	**99.34**	**79.68**	**92.69**	69.41	65.85

Table 8: Comparison of AUROC results with XGBGraph and CONSISGAD.

We computed the **F1-scores** for our model and baseline methods across all datasets. These results further confirm the superior performance of our model. Table 9 presents the F1-scores, which show consistency with the experiment results in Table 1.

Model	Ellip	Tolo	Yelp	Quest	Reddit
GCN	73.672	47.376	27.658	6.856	7.794
GIN	75.338	49.443	42.214	10.288	6.443
GraphSAGE	81.096	50.226	43.949	12.041	**10.075**
GAT	80.498	50.878	48.891	11.157	8.432
GAS	77.844	48.253	43.404	10.867	9.071
PCGNN	45.090	47.213	44.608	5.796	6.981
BWGNN	83.134	49.983	47.323	12.788	6.501
GHRN	85.678	51.493	45.970	12.696	6.702
xGBGraph	87.555	51.079	65.121	16.088	2.954
CONSISGAD	79.120	49.762	41.606	9.848	6.443
Ours (CGADM)	**93.390**	**51.595**	**69.396**	**17.162**	9.754

Table 9: F1-scores comparison across datasets.

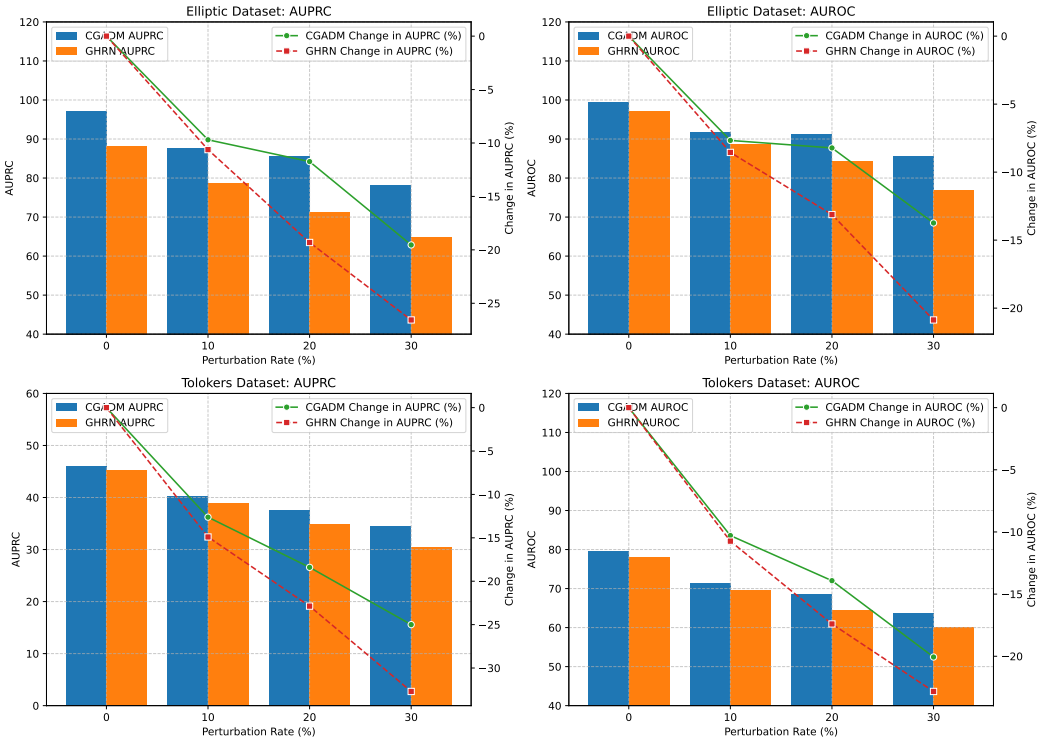

Figure 5: Robustness against Feature Manipulation

O Robustness of CGADM against Feature Manipulation

To evaluate the robustness of CGADM against feature manipulation, we introduced feature perturbations in the **Elliptic** and **Tolokers** datasets. Specifically, we randomly perturbed the features of nodes with varying proportions (10%, 20%, and 30%) by randomly selecting values from their possible ranges with uniform probability. We then compared the performance of CGADM with GHRN (the best-performing baseline from our original experiments) under these conditions.

The results are summarized in Figure 5. As the proportion of perturbed nodes increases, the performance of both models decreases. However, CGADM consistently exhibits a slower decline compared to GHRN. This highlights CGADM's superior robustness to feature perturbations, which we attribute to its denoising reconstruction mechanism. This mechanism leverages information from neighboring nodes during the reverse diffusion process to iteratively restore the true anomaly signals.

P Effect of High- and Low-frequency Signals

To further substantiate that the high-frequency components are indeed reflected in the residual propagations, we designed an ablation study comparing our original CGADM (denoted as $CGADM_{HP}$) with a variant (denoted as $CGADM_{LP}$) that only propagates low-frequency signals. In $CGADM_{LP}$, the graph convolution operation is replaced with the standard GCN:

$$\frac{1}{|\mathcal{N}(v)|+1}\left(\mathbf{h}_v^{l-1}+\sum_{u\in\mathcal{N}(v)}\mathbf{h}_u^{l-1}\right), \tag{39}$$

where the feature representation is averaged across the node and its neighbors, propagating only low-frequency signals.

We conducted experiments on the *Elliptic* and *YelpChi* datasets, varying the number of GNN layers in the denoiser module. The results are shown in Table 10:

GNN Layers	Model	AUPRC (Elliptic) %	AUROC (Elliptic) %	AUPRC (YelpChi) %	AUROC (YelpChi) %
1	$CGADM_{HP}$	97.13	99.22	75.04	92.37
	$CGADM_{LP}$	95.71	98.43	72.23	91.88
2	$CGADM_{HP}$	97.31	99.38	75.20	92.62
	$CGADM_{LP}$	93.73	97.60	70.92	90.88
3	$CGADM_{HP}$	97.32	99.44	76.54	92.69
	$CGADM_{LP}$	90.83	95.58	71.43	89.64
4	$CGADM_{HP}$	97.53	99.44	77.27	93.05
	$CGADM_{LP}$	87.12	92.60	69.98	87.71
5	$CGADM_{HP}$	97.57	99.50	77.29	92.92
	$CGADM_{LP}$	81.20	89.49	68.71	86.08

Table 10: Performance comparison of $CGADM_{HP}$ and $CGADM_{LP}$ with varying GNN layers.

According to Table 10, we have the following observations:

1. **High-Frequency Signal Preservation Matters:** $CGADM_{HP}$, which retains high-frequency signals through residual propagation, consistently outperforms $CGADM_{LP}$ across all metrics and datasets. This highlights the importance of preserving high-frequency information for anomaly detection, as anomalies often manifest as local deviations that are captured by these components.
2. **Sensitivity to GNN Layers:** For $CGADM_{LP}$, performance declines significantly as the number of GNN layers increases. This is indicative of the well-known over-smoothing issue, where stacking multiple low-pass filters causes node representations to converge, losing discriminative information. Conversely, $CGADM_{HP}$ remains robust, and its performance even improves slightly with additional layers, demonstrating the effectiveness of residual propagation in mitigating over-smoothing.
3. **Iterative Refinement Amplifies Over-Smoothing:** In the context of our diffusion model, the iterative refinement process repeatedly aggregates neighborhood information, exacerbating the impact of over-smoothing in $CGADM_{LP}$. This leads to a failure to capture new anomaly-relevant signals at each stage of refinement. In contrast, $CGADM_{HP}$ avoids this issue by leveraging high-frequency signals to refine anomaly detection throughout the iterative process.

Q Comparison with Data-augmentation Methods

The main distinction between CGADM and the existing data-augmentation methods lies in the underlying approach to anomaly detection. While prior works focus on using diffusion models for **data augmentation** to improve detection performance, CGADM adopts a **generative, model-centric paradigm** to directly model the joint distribution of anomalies on the entire graph. Below, we summarize the key differences:

- CAGAD (Xiao et al., 2024): Uses a graph-specific diffusion model to generate counterfactual representations by transforming normal neighbors into anomalous ones. This is a classic **data augmentation** technique to enhance anomaly distinguishability.
- DEGAD (Pang et al., 2024): Employs diffusion models to generate manipulated neighbors, enhancing graphs by creating augmented data. This technique is used as a **data enhancement module** within a contrastive learning framework.
- ConGNN (Li et al., 2024): Introduces a generator based on diffusion models to control neighborhood aggregation and **create augmented data** for better anomaly detection performance.
- GD (Liu et al., 2024): Tackles the label imbalance problem by **generating positive examples** using a diffusion model in the latent space. The primary goal is to balance datasets, not directly detect anomalies.
- Diffad (Ma et al., 2024b): Investigates denoising diffusion models to **synthesize graph structures** and enhance existing methods. This approach focuses on data synthesis rather than directly detecting anomalies.

We have conducted a detailed experimental comparison of our proposed Conditional Graph Anomaly Diffusion Model (CGADM) with some diffusion-based data augmentation methods CAGAD (Xiao et al., 2024), DEGAD (Pang et al., 2024), ConGNN (Li et al., 2024), GD (Liu et al., 2024), and Diffad (Ma et al., 2024b). We analyzed their performance across several standard benchmark datasets (`Elliptic`, `Tolokers`, and `YelpChi`), and the key results are summarized below:

Metric	Model	Ellip	Tolo	Yelp
AUPRC	CAGAD	89.75	40.80	72.30
	DEGAD	93.86	43.51	75.11
	ConGNN	91.60	42.22	73.60
	GD	88.63	39.90	68.01
	Diffad	90.05	41.75	71.28
	CGADM	**97.28**	**45.11**	**76.54**
AUROC	CAGAD	94.82	72.22	90.34
	DEGAD	97.88	76.20	92.22
	ConGNN	95.60	74.56	91.33
	GD	93.53	70.70	83.84
	Diffad	92.72	73.31	88.21
	CGADM	**99.34**	**78.11**	**92.69**

Table 11: AUPRC and AUROC comparison with Data Augmentation Methods

As shown in the Table 11, CGADM consistently outperforms the data-augmentation methods in both AUPRC and AUROC across all datasets. This underscores the efficacy of our generative framework in addressing graph anomaly detection challenges.