RGDM: Robust Graph Diffusion Model

Yancheng Wang [1], **Ping Li** [2], **Dongfang Sun** [1], **Yingzhen Yang** [1]

[1] School of Computing and Augmented Intelligence
Arizona State University
Tempe, AZ 85281, USA
{yancheng.wang,dsun30,yingzhen.yang}@asu.edu

[2] VecML Inc.
Bellevue, WA 98004, USA
pingli98@gmail.com

Abstract

Diffusion models represent a powerful class of generative models known for their solid theoretical foundations and remarkable performance across diverse tasks and domains. Existing node level diffusion methods synthesize nodes and edges within a graph, but are not designed to learn a robust generator from adversarially corrupted graphs. In this study, we introduce the Robust Graph Diffusion Model (RGDM), designed to generate labeled synthetic graph structures consisting of nodes and edges that integrate seamlessly into a given graph. The RGDM consists of a Robust Graph Autoencoder (RGAE) and a Latent Diffusion Model (LDM). Leveraging an edge selection mechanism and an innovative low-rank regularization on the latent feature, the RGDM produces clean and high-quality synthetic graph structures, even when trained on graphs subject to adversarial attacks. Comprehensive experimental evaluations reveal that Graph Neural Networks (GNNs) trained on the augmented graph, which is formed by merging the original attacked graph with the synthetic graph structures, exhibit significantly improved robustness against various graph adversarial attacks in the context of semi-supervised node classification. The code of the RGDM is available at https://github.com/Statistical-Deep-Learning/RGDM.

1 Introduction

Building on the success of diffusion and latent diffusion models, recent works use diffusion-generated data to augment training sets [1, 2]. However, extensions to graphs [3, 4, 5, 6, 7] mainly target graph-level augmentation. Most node-level methods expand labels through mixup [8, 9, 10, 11, 12, 13] or GAN-based synthesis for imbalance [14, 15, 16, 17, 18]. Node level diffusion methods DoG [19] and LR-FGDM [20] share the autoencoder and diffusion template, but do not train the generator specifically for adversarial robustness. Moreover, GANs can be unstable and produce distributions misaligned with real data [21]. As a result, the challenge of generating synthetic graph structures (SGS) within a given graph, which are the synthetic nodes and their corresponding edges, in a robust manner to enhance node classification under attacks, remains underexplored. In this work, we tackle the problem of node-level graph generation and propose the Robust Graph Diffusion Model (RGDM), designed to synthesize high-quality graph structures within a single graph, even under adversarial attacks. These synthetic structures include both synthetic nodes and edges connecting them within the same graph, forming an augmented graph as illustrated in Figure 1. Robust GNNs will be trained on the augmented graph, and superior performance is expected due to the enlarged set of labeled nodes.

Generating high-quality SGS in adversarial scenarios for improved performance is challenging, important, and unaddressed in the robust graph learning literature. To address this challenge, RGDM integrates a Robust Graph Autoencoder (RGAE) with a Latent Diffusion Model (LDM). The RGAE incorporates an edge selection mechanism and a low-rank regularization on the latent features to

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

counteract adversarial perturbations in the attacked graph. The low-rank regularization is motivated by the Low Frequency Property (LFP) [22, 23, 24, 25, 26, 19]: in the evaluated latent spaces, label information is more concentrated in leading eigendirections than adversarial perturbations (Section F.1). Theorem 3.1 provides a complementary, conditional justification through the spectral complexity of a bounded linear probe on this latent space. It is not a guarantee for the downstream GNN trained on the augmented graph.

Figure 1 illustrates the 2-hop neighborhood of a real node and a synthetic node that have similar node features, evidenced by the t-SNE visualization in Figure 5 in Section F.10 of the appendix. The fidelity evaluations in Tables 13 and 14 compare synthetic structures with clean reference graphs through Frechet Node Distance (FND) and Frechet Edge Distance (FED). RGDM has lower distances than vanilla DDPM [27] and the compared node augmentation methods. On Pubmed under 25% Metattack, its FND/FED are $3.95/5.34$, compared with $3.79/5.15$ for the clean reference. These are empirical fidelity measurements. Clean graphs are used for evaluation, not for training RGDM. The augmentation pipeline retains the original attacked adjacency. Edge selection acts inside the generator, reducing the reproduction of unreliable connections. The generated labeled structures supply additional supervision and relatively clean message passing paths. On Citeseer, adding 785 synthetic edges changes the normalized perturbation density from 25% to approximately 20.6%. This is a denominator effect, not removal of the original perturbations or a causal robustness guarantee (Section F.4). **Difference from diffusion based purification.** Diffusion models have also been used for data and graph purification [28, 29, 30, 31, 32]. Purifying an observed graph and generating additional labeled nodes with incident edges are different tasks. Graph level purification methods do not directly provide the node level augmentation interface of RGDM. Conversely, vanilla node generation trained on a corrupted graph can reproduce its noise. RGDM trains its robust generator without paired clean graphs. Section F.13 compares GDDM and DiffSP preprocessing with RGDM augmentation using the same SG-GSR backbone, while Section F.12 separately evaluates a purification adaptation of RGDM.

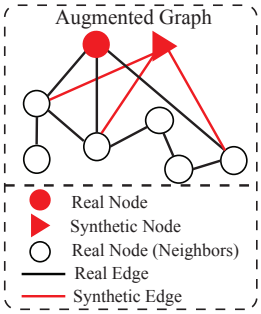


Figure 1: Illustration of the 2-hop neighborhood of a real node and a synthetic node in the augmented graph. Figure 5 in Section F.10 illustrates the t-SNE visualization of the augmented graph.

Contributions. First, RGDM learns a robust node level generator through edge selection, sparse hierarchical reconstruction, and low rank regularization of the latent features used by diffusion. Its Sparse Hierarchical Edge Decoder (SHED) evaluates selected clusters instead of globally scoring every candidate neighbor. Controlled comparisons with DoG and LR-FGDM, componentwise ablations, and wall clock measurements distinguish these contributions from the shared autoencoder and diffusion architecture. Second, RGDM augments existing classifiers without changing their architecture or objective. We evaluate its incremental benefit with each backbone fixed, including SG-GSR [33], and report paired variability. The gains over this strong backbone are modest in absolute magnitude, but consistent in the tested settings. Fidelity, heterophily, and adaptive attack evaluations support this empirical conclusion. The linear probe theorem motivates the latent regularizer and does not establish an end to end robustness guarantee.

2 Related Works

Data Augmentation for Graph Learning. To improve the performance of GNNs, node-level graph data augmentation methods have been studied to modify the structure [34, 35, 36, 37, 38], features [39, 40, 41], or node labels [39, 40, 41]. Studies have demonstrated that increasing the number of labeled nodes based on Mix-up strategies [8, 9, 10, 11, 12, 13] can greatly enhance the performance of the GNNs. Recent works address node and edge imbalance by generating synthetic nodes and edges [42, 43, 44, 45, 46], but predominantly rely on Generative Adversarial Networks (GANs). However, the computer vision literature [21] has demonstrated that GANs often exhibit instability during training and that the distribution of synthetic data generated by GANs poorly matches the distribution of real data. Graph level diffusion models [4, 3, 5, 6, 7] generate entire graphs. DoG [19] and LR-FGDM [20] instead generate nodes and associated edges within an existing graph, as does RGDM. DoG reconstructs observed neighborhoods with a Bi-Level Neighborhood Decoder (BLND) and applies low rank regularization to its downstream GNN. LR-FGDM addresses few shot

classification using a Hierarchical Graph Autoencoder (HGAE) with prototypical regularization, followed by a low rank transductive classifier. RGDM places edge selection, a sparsification objective, and low rank regularization inside the generator. Its TNN penalty acts on the latent features used by diffusion. SHED reconstructs selected sparse neighborhoods rather than all observed connections. The resulting augmented graph can be used with GCN, Pro-GNN, or SG-GSR. Tables 4 and 5 compare these methods with a fixed backbone. Table 33 additionally compares hierarchical reconstruction variants.

Graph Adversarial Attacks and Defense. Despite the success of GNNs in graph-structured tasks [47, 48], recent studies reveal their vulnerability to adversarial attacks [49]. Depending on the attack objective, threat models are categorized as: targeted attacks [50], which mislead predictions on specific nodes, and untargeted attacks [51, 52], which reduce overall model accuracy. To defend against these threats, robust learning strategies are grouped into three categories: adversarial training, graph processing, and model robustification. Adversarial training [53, 54] augments training data with crafted adversarial samples. Graph processing [55, 56, 57, 58] seeks to denoise perturbed graphs. Model robustification [59, 60, 61, 62, 63, 64, 65, 33] enhances model resilience.

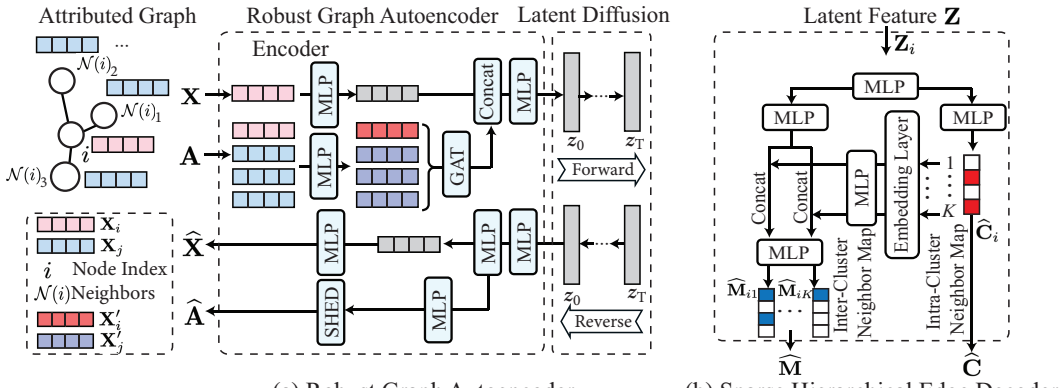
3 Methods

3.1 Preliminaries of the Attributed Graph

An attributed graph of N nodes is represented by $\mathcal{G} = (\mathcal{V}, \mathbf{X}, \mathbf{A})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denote the nodes and edges respectively. $\mathbf{X} \in \mathbb{R}^{N \times D}$ contains the node attributes. $\mathbf{X}_i \in \mathbb{R}^D$, the i-th row of $\mathbf{X}$, denotes the attributes of the i-th node. $\mathbf{A} \in \{0,1\}^{N \times N}$ is the adjacency matrix of the graph $\mathcal{G}$. $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix of a graph with self-loops added. $\tilde{\mathbf{D}}$ denotes the diagonal degree matrix of $\tilde{\mathbf{A}}$. We use $\mathcal{N}(i) = \left\{ j \mid \tilde{\mathbf{A}}_{i,j} = 1 \right\}$ to denote the set of nodes connected to v_i. $\mathbf{Z} \in \mathbb{R}^{N \times D'}$ contains the latent features of all the nodes in the graph, where D' is the latent dimension. Let $\mathcal{V}_\mathcal{L}$ and $\mathcal{V}_\mathcal{U}$ denote the set of labeled nodes and unlabeled test nodes, respectively, and $|\mathcal{V}_\mathcal{L}| = m$, $|\mathcal{V}_\mathcal{U}| = u$. Let $Y_L = \{y_i \mid v_i \in \mathcal{V}_L,\ y_i \in [C]\}$ denote the labels of nodes in $\mathcal{V}_\mathcal{L}$, where C is the number of classes. Let $\mathbf{Y} = [\mathbf{y}_1; \mathbf{y}_2; \ldots \mathbf{y}_N] \in \mathbb{R}^{N \times C}$ be the ground truth label matrix of all the nodes in $\mathcal{G}$, where $\mathbf{y}_i$ is the one-hot label vector of node v_i. Let $\mathbf{u} \in \mathbb{R}^N$ be a vector. We use $[\mathbf{u}]_\mathcal{A}$ to denote a vector formed by elements of $\mathbf{u}$ with indices in $\mathcal{A}$ for $\mathcal{A} \subseteq [N]$. If $\mathbf{u}$ is a matrix, then $[\mathbf{u}]_\mathcal{A}$ denotes a submatrix formed by rows of $\mathbf{u}$ with row indices in $\mathcal{A}$. $\|\cdot\|_\mathrm{F}$ denotes the Frobenius norm of a matrix, and $\|\cdot\|_p$ denotes the p-norm of a vector.

3.2 Robust Graph Autoencoder (RGAE)

In this section, we propose the Robust Graph Autoencoder (RGAE) capable of encoding node attributes and associated edges into continuous latent features and decoding these features to reconstruct the node attributes and edges in the graph. Beyond reconstruction, the RGAE is designed to be robust to adversarial perturbations in the node attributes and the edges in the graph, enhancing the resilience against potential graph adversarial attacks. We encode each node's attributes together



(a) Robust Graph Autoencoder (b) Sparse Hierarchical Edge Decoder

Figure 2: Figure (a) illustrates the synthetic node generation process in Robust Graph Diffusion Model. Figure (b) illustrates the structure of the Sparse Hierarchical Edge Decoder (SHED).

with its neighborhood. Positional embeddings give $\mathbf{X}'_j = \mathbf{X}_j + \text{pos}(j)$, and GAT layers [66] aggregate these features. For $j \in \mathcal{N}(i)$, the attention weight is $\mathbf{B}_{ij} = \exp(a_{ij}) / \sum_{\ell \in \mathcal{N}(i)} \exp(a_{i\ell})$, where $a_{ij} = \sigma(\mathbf{X}'_i\mathbf{Q} \| \mathbf{X}'_j\mathbf{Q})\mathbf{V}$, $\|$ denotes concatenation, σ is the sigmoid function, and $\mathbf{Q}, \mathbf{V}$ are learnable matrices. Attention is restricted to observed neighbors. To suppress unreliable edges during RGAE training, a learned binary mask $\mathbf{S}$ selects connections. We use the sigmoid relaxation $\widehat{\mathbf{S}}_{ij} = \sigma(\boldsymbol{\theta}_{ij}/\tau)$ associated with binary Gumbel Softmax training [67, 68]. A linear layer applied to $\mathbf{X}'_i \| \mathbf{X}'_j$ predicts $\boldsymbol{\theta}_{ij}$. The forward pass uses $\mathbf{S}_{ij} = \mathbb{1}_{\{\widehat{\mathbf{S}}_{ij} > 1/2\}}$, while a straight through estimator differentiates through $\widehat{\mathbf{S}}_{ij}$. We use $\tau = 0.5$ unless stated otherwise. The selected neighbor aggregation is $\mathbf{Z}'_i = \text{ReLU}(\sum_{j \in \mathcal{N}(i)} \mathbf{S}_{ij}\mathbf{B}_{ij}\mathbf{X}'_j\mathbf{W})$, with learnable $\mathbf{W}$. Two GAT layers yield the neighborhood representation. Concatenating it with an MLP of the node attributes gives $\mathbf{Z}_i = \mathbf{Z}'_i \| f(\mathbf{X}_i)$. The mask changes the generator's training representation, not the original adjacency retained for downstream augmentation. **Sparse Hierarchical Edge Decoder (SHED).** Decoding edges from latent features can be computationally intensive, as each node may connect to any other node. To mitigate this, we propose a Sparse Hierarchical Edge Decoder (SHED), which decodes each latent feature into an inter-cluster and an intra-cluster neighbor map, as illustrated in Figure 2 (b). The SHED begins by partitioning the nodes into K balanced clusters using balanced K-means clustering [69] based on node attributes. Each cluster has a maximum capacity of $M = \lceil \frac{N}{K} \rceil$. Within each cluster, nodes are indexed by their original graph order. For a node v_i, SHED first reconstructs an inter-cluster neighbor map $\mathbf{C} \in \{0,1\}^{N \times K}$, where $\mathbf{C}_{ik} = 1$ if v_i is connected to at least one node in cluster k. It then reconstructs the intra-cluster neighbor map as a tensor $\mathbf{M} \in \{0,1\}^{N \times K \times M}$, where $\mathbf{M}_{ikm} = 1$ indicates that v_i is connected to the m-th node in cluster k. The SHED begins by computing the inter-cluster neighbor assignment $\widehat{\mathbf{C}}_i$ for each node v_i using a single-layer MLP. At generation time, the active cluster indices, $\mathcal{C}(i) = \left\{ k \in [K] \mid \widehat{\mathbf{C}}_{ik} > 1/2 \right\}$, are extracted and passed through a cluster index embedding module $g(\cdot)$, which consists of an MLP, inspired by the classifier-free guidance technique [70]. This produces a set of embeddings $\mathcal{Z}(i) = \left\{ g(k) \in \mathbb{R}^{D'} \mid k \in \mathcal{C}(i) \right\}$. Each embedding $g(k)$ is then concatenated with the latent representation $\mathbf{Z}_i$ from the alternate branch, and the result is processed by a decoding MLP, $g'(\cdot)$, to estimate the intra-cluster neighbor relation as $\widehat{\mathbf{M}}_{ik} = g'(\mathbf{Z}_i \| g(k))$. Compared to the Bi-Level Neighborhood Decoder (BLND) in DoG [19], the reconstructed neighbor map $\widehat{\mathbf{M}}$ is sparse for robust graph learning, which is enforced by the training of the RGAE detailed later.

Optimization of the RGAE. RGAE incorporates an edge selection mechanism in the encoder that gradually removes the adversarial edges. By learning an edge selection mask $\mathbf{S}$, RGAE purifies the adjacency matrix of the graph as $\mathbf{A} \circ \mathbf{S}$. By integrating the edge selection mechanism with SHED, we decompose $\mathbf{S}$ into two edge selection masks, $\mathbf{S}_{\text{inter}} \in \{0,1\}^{N \times K}$ and $\mathbf{S}_{\text{intra}} \in \{0,1\}^{N \times K \times M}$, for edge selection on the inter-cluster neighbor map $\mathbf{C}$ and intra-cluster neighbor map $\mathbf{M}$. At the end of each epoch, $\mathbf{S}_{\text{inter}}$ and $\mathbf{S}_{\text{intra}}$ are computed from $\mathbf{S}$. In particular, $[\mathbf{S}_{\text{inter}}]_{ik} = 1$ if and only if there is a node v_j from the cluster k, such that $\mathbf{S}_{ij} = 1$. $[\mathbf{S}_{\text{intra}}]_{ikm} = 1$ if and only if $\mathbf{S}_{ij} = 1$, where v_j is the m-th node in the cluster k.

We regularize the latent feature matrix $\mathbf{Z} \in \mathbb{R}^{N \times D'}$ through its Gram matrix $\mathbf{K} \coloneqq \mathbf{Z}\mathbf{Z}^\top$. Put $q \coloneqq \min\{N, D'\}$, and let $\widehat{\lambda}_1 \geq \cdots \geq \widehat{\lambda}_N \geq 0$ be the eigenvalues of $\mathbf{K}$. For $0 \leq r_0 < q$, its truncated nuclear norm is $\|\mathbf{K}\|_{r_0} \coloneqq \sum_{i=r_0+1}^{N} \widehat{\lambda}_i$. A zero tail implies $\text{rank}(\mathbf{K}) \leq r_0$. A soft penalty encourages, but does not impose, this rank. The RGAE loss at epoch t is

$$\mathcal{L}^{(t)}_{\text{RGAE}}(\mathbf{S}^{(t-1)}_{\text{intra}}, \mathbf{S}^{(t-1)}_{\text{inter}}) = \mathcal{L}^{(t)}_{\text{C}} + \tau_{\text{S}}\mathcal{L}^{(t)}_{\text{S}} + \tau_{\text{T}}\|\mathbf{K}\|_{r_0}, \tag{1}$$

where

$$\mathcal{L}^{(t)}_{\text{C}} = \left\|\mathbf{X} - \widehat{\mathbf{X}}\right\|^2_{\text{F}} + \left\|\mathbf{C} - \widehat{\mathbf{C}}\right\|^2_{\text{F}} + \left\|\mathbf{M} - \widehat{\mathbf{M}}\right\|^2_{\text{F}},$$
$$\mathcal{L}^{(t)}_{\text{S}} = \left\|\mathbf{C} \circ \mathbf{S}^{(t-1)}_{\text{inter}} - \widehat{\mathbf{C}}\right\|^2_{\text{F}} + \left\|\text{vec}\left(\mathbf{M} \circ \mathbf{S}^{(t-1)}_{\text{intra}} - \widehat{\mathbf{M}}\right)\right\|_1.$$

Here $\circ$ denotes the Hadamard product. $\mathcal{L}^{(t)}_{\text{RGAE}}$ is the overall training loss of the RGAE at the t-th epoch, where $t \in [t_{\text{train}}]$ is the current epoch number and t_{train} is the total number of training epochs. We initialize $\mathbf{S}^{(0)}_{\text{intra}} = \mathbf{M}$ and $\mathbf{S}^{(0)}_{\text{inter}} = \mathbf{C}$ before the first epoch of training. For tensors, $\|\cdot\|_{\text{F}}$ denotes

the Euclidean norm after vectorization. τ_{S} and τ_{T} are the scale factors for the sparsification loss and the TNN. $\mathcal{L}_{\mathrm{C}}^{(t)}$ and $\mathcal{L}_{\mathrm{S}}^{(t)}$ are the consistency loss and the sparsification loss at the t-th epoch. The starting rank r_0 is selected by cross-validation as described in Section 4.1. The training of RGDM is described in Algorithm 1 in Section C of the appendix. We further show in Section D.1 of the appendix that the TNN regularizer can be efficiently computed from the singular values of the latent representation $\mathbf{Z}$ with a complexity of $O(ND'^2)$.

The node reconstruction loss and the hierarchical edge reconstruction loss ensure consistency between the reconstructed features and adjacency matrix and their inputs. The sparsification loss is motivated by prior work on graph structure learning for robust GNNs [57, 71], which aims to purify noisy or perturbed graph structures. The entrywise ℓ_1 term in $\mathcal{L}_{\mathrm{S}}^{(t)}$ discourages nonzero reconstructions outside the selected sparse neighbor map. Although the BLND in DoG [19] also performs edge reconstruction in a two-level manner, it is trained to precisely recover all edges connected to each node in the given graph, including those corrupted by adversarial attacks. In contrast, our SHED reconstructs a sparse and robust graph structure by selectively preserving the clean edges, motivated by prior studies showing that graph sparsification enhances the robustness of GNNs against adversarial attacks by removing noisy connections [57, 58, 71, 72]. Moreover, existing studies on adversarial defense reveal that perturbations typically reside in high-rank residual components, whereas clean signals lie in low-rank subspaces [73, 74]. Projecting representations onto low-rank subspaces [73, 74, 75] or enforcing low-rank weight matrices [76] has been proven to be effective for enhancing robustness. However, research in graph adversarial defense has focused predominantly on reducing the rank of the adjacency matrix [77, 56, 78, 79, 80]. Our penalty instead regularizes the diffusion latent. Theorem 3.1 analyzes a bounded linear probe on fixed latent features.

Motivation and scope of low rank regularization. The eigen projection study (Figures 3 and 4, Section F.1) finds that leading eigendirections of $\mathbf{K}$ retain more label information than perturbation information. This is an empirical property of the evaluated latent kernels, not an assumption that adjacent nodes share labels. The soft TNN penalty balances compression against reconstruction. We choose $r_0 = \lceil \gamma q \rceil$ by five fold cross validation over $\gamma \in \{0.1, 0.2, \ldots, 0.9\}$ (Section E.4). A larger rank can preserve useful heterophilic information.

Theoretical justification and its limits. For a fixed latent matrix $\mathbf{Z}$, consider the linear probe $\mathbf{F}(\mathbf{W}) \coloneqq \mathbf{Z}\mathbf{W}$ with training loss $L_{\mathcal{L}}(\mathbf{W}) \coloneqq m^{-1}\|[\mathbf{Z}\mathbf{W} - \mathbf{Y}]_{\mathcal{L}}\|_{\mathrm{F}}^2$. The following finite pool bound makes the boundedness and sampling requirements explicit based on the sharp transductive bounds by Transdcutive Local Complexity (TLC) [81, 82, 83]. Section A gives a self contained proof and the squared loss gradient descent specialization.

Theorem 3.1. For a positive integer k, let $[k] \coloneqq \{1, \ldots, k\}$, and let log denote the natural logarithm. Use the graph notation of Section 3.1 and the latent Gram matrix $\mathbf{K}$, rank bound q, and eigenvalues $\widehat{\lambda}_1, \ldots, \widehat{\lambda}_N$ defined above. Write $\mathcal{L} \coloneqq \{i \in [N] : v_i \in \mathcal{V}_{\mathcal{L}}\}$ and $\mathcal{U} \coloneqq \{i \in [N] : v_i \in \mathcal{V}_{\mathcal{U}}\}$. Assume $D', C \geq 1$, $N = m + u$, $m, u \geq 2$, $\mathcal{U} = [N] \setminus \mathcal{L}$, and $\mathcal{L}$ is uniform among subsets with m elements, with $\mathbf{Z}, \mathbf{Y}$ fixed before sampling. For fixed $B > 0$ and $\tau_0 \geq 0$, suppose $\max_{i \in [N]} \|\mathbf{Z}_i\|_2 \leq \tau_0$. Put $s \coloneqq \min\{m, u\}$. For $r \in [q]$, define $\|\mathbf{K}\|_r \coloneqq \sum_{j=r+1}^{N} \widehat{\lambda}_j$, with an empty sum equal to zero, and put

$$\mathrm{KC}(\mathbf{K}) \coloneqq \min_{r \in [q]} \left[r\left(\frac{1}{u} + \frac{1}{m}\right) + \sqrt{\frac{\|\mathbf{K}\|_r}{N}} \left(\frac{1}{\sqrt{u}} + \frac{1}{\sqrt{m}}\right) \right].$$

There is a constant $c_0 > 0$, depending only on C, B, τ_0, such that, for every $x > 0$, with probability at least $1 - \exp(-x)$, simultaneously for all $\mathbf{W} \in \mathbb{R}^{D' \times C}$ with $\|\mathbf{W}\|_{\mathrm{F}} \leq B$,

$$\mathcal{U}_{\text{test}}(\mathbf{W}) \coloneqq \frac{\|[\mathbf{Z}\mathbf{W} - \mathbf{Y}]_{\mathcal{U}}\|_{\mathrm{F}}^2}{u} \leq 2L_{\mathcal{L}}(\mathbf{W}) + c_0 \left[\mathrm{KC}(\mathbf{K}) + \frac{x + \log C}{s} \right], \tag{2}$$

For a fixed rank and feature scale bound, reducing the spectral tail reduces its contribution to the complexity bound in Equation (2). It need not reduce the label fitting term. Strong heterophily may increase either term and loosen the bound. The theorem concerns a bounded linear probe under an independent random split, not class conditional generation, a label dependent adaptive attack, or the downstream GNN. The latter connection is empirical, through the low rank ablation and FND/FED evaluations. Main text heterophily results appear in Section 4.5.

3.3 Conditional Generation of Synthetic Graph Structures in Latent Space

To generate synthetic graph structures, consisting of the synthetic nodes and their edges connecting to the original graph, we train a latent diffusion model (LDM) [84] with Classifier-Free Guidance (CFG) [70]. The trained LDM generates latent features for the synthetic structures, which are then decoded into node features and associated edges. Let $\mathcal{V}_L \subseteq \mathcal{V}$ denote the set of the labeled nodes in the original graph $\mathcal{G}$, and $\mathcal{V}_{\text{syn}}$ denotes the set of the generated synthetic nodes by the RGDM. Let the node attributes of $\mathcal{V}_{\text{syn}}$ generated by the RGDM be $\mathbf{X}_{\text{syn}}$. Let C be the number of classes, and N_0 be the number of synthetic nodes for each class. $N' = CN_0 = |\mathcal{V}_{\text{syn}}|$ is the number of the synthetic nodes. Let $Y_{\text{syn}} = \{\widehat{y}_i\}_{i=1}^{N'}$ denote the set of the labels of the synthetic nodes. Let $\widehat{\mathbf{Z}} = \left\{\widehat{\mathbf{Z}}_i\right\}_{i=1}^{N'}$ denote the latent features of the synthetic graph structures. Let $\mathbf{A}_{\text{syn}} \in \mathbb{R}^{N' \times N}$ denote the edges of the synthetic nodes generated by our RGDM. Then the augmented synthetic graph structures are $(\mathcal{V}_{\text{syn}}, \mathbf{X}_{\text{syn}}, \mathbf{A}_{\text{syn}})$. There are no generated edges between synthetic nodes in this construction. The adjacency matrix is given by Equation (3):

$$\mathbf{A}_{\text{aug}} := \begin{bmatrix} \mathbf{A} & \mathbf{A}_{\text{syn}}^{\top} \\ \mathbf{A}_{\text{syn}} & \mathbf{0}_{N' \times N'} \end{bmatrix} \in \mathbb{R}^{(N+N') \times (N+N')}. \tag{3}$$

The node feature matrix is $\mathbf{X}_{\text{aug}} = [\mathbf{X}; \mathbf{X}_{\text{syn}}] \in \mathbb{R}^{(N+N') \times D}$. The augmented graph, which is the combination of the original graph $\mathcal{G}$ and the synthetic graph structures, is then denoted by $\mathcal{G}_{\text{aug}} = (\mathcal{V} \cup \mathcal{V}_{\text{syn}}, \mathbf{X}_{\text{aug}}, \mathbf{A}_{\text{aug}})$. The set of labeled nodes in $\mathcal{G}_{\text{aug}}$ is $\mathcal{V}_L \cup \mathcal{V}_{\text{syn}}$ and their labels are $Y_L \cup Y_{\text{syn}}$.

3.4 Training GNNs on the Augmented Graph

After obtaining the augmented graph $\mathcal{G}_{\text{aug}}$, we then train either the vanilla GCN [47] or the robust GNNs, Pro-GNN [57] and SG-GSR [33], on $\mathcal{G}_{\text{aug}}$. Results in Section 4.1 evidence that the GNNs trained on the augmented graph exhibit much better performance and robustness for the semi-supervised node classification task than the models trained on the original attacked graphs perturbed by graph adversarial attacks, including Metattack [51], Nettack [50], and Topology Attack [85].

Complexity Analysis. We perform a detailed complexity analysis of the RGDM in Section D.2 of the appendix, demonstrating the improved efficiency of the proposed SHED.

4 Experiments

We evaluate the incremental benefit of RGDM with each downstream backbone fixed. In addition to the standard attacks, the main text includes direct node level diffusion comparisons, component-wise ablations, heterophily, adaptive attacks, and decoder timings. The appendix provides spectral diagnostics (Section F.1), complete attack results (Section F.2), FND/FED (Section F.3), timings and cross validation (Sections F.5 to F.6), large graph edge selection ablations (Section F.7), sensitivity and heterophily details (Sections F.8 to F.9), visualization and paired statistics (Sections F.10 to F.11), purification variants (Sections F.12 to F.13), node count, DDPM, and label ratio studies (Sections F.14 to F.16), hierarchical reconstruction comparisons (Section F.17), and adaptive attack details (Section F.18).

4.1 Semi-Supervised Node Classification under Graph Adversarial Attacks

Settings. We evaluate RGDM on four public graph benchmarks: Cora, Citeseer, Pubmed [86], and Polblogs [87]. Dataset statistics are provided in Table 9 in Section E.1 of the appendix. The training settings of the RGAE and LDM in RGDM are described in Section E.2. The number of added synthetic nodes is selected by cross-validation following Section F.6, and the starting rank r_0 is selected by cross-validation as detailed in Section E.4 of the appendix.

We conduct experiments on semi-supervised node classification under the Metattack [51], the Nettack [50], and the Topology Attack [85]. The compared methods are detailed in Section E.6 of the appendix. We apply the RGDM on vanilla GCN [47] and state-of-the-art robust GNNs, which are Pro-GNN [57] and SG-GSR [33]. The results for Metattack under the perturbation rates of 15% and 25% are shown in Table 1. Additional results with perturbation rates from 5% to 25% with a step of 5% for the Metattack are deferred to Table 10 in Section F.2 of the appendix. The results for Nettack [50] under attack budgets of 3 and 5 are shown in Table 2. Detailed results for Nettack, with attack budgets varying from 1 to 5 in increments of 1, are deferred to Table 11 in Section F.2 of the

Table 1: Metattack accuracy ± marginal SD (%). Parentheses give the augmentation gain. A further ± is its SD from the separate paired evaluation (Table 26). A 0% rate is a clean graph. Bold and underline mark the best and second best results across both panels for each setting. Complete results appear in Table 10.

(a) Original graph

Dataset	Ptb Rate (%)	GCN	GCORNs	HANG	STABLE	Pro-GNN	SG-GSR
Cora	0 (clean)	83.5±0.4	82.6±0.5	80.1±0.3	84.7±0.5	83.0±0.2	85.5±1.3
	15	65.1±0.7	72.6±1.2	72.9±1.0	74.4±0.9	76.4±1.3	78.5±1.6
	25	47.5±1.9	69.5±1.9	68.7±1.9	69.6±0.6	69.7±1.7	77.8±1.3
Citeseer	0 (clean)	72.0±0.6	72.6±0.4	73.3±0.4	74.8±0.5	73.3±0.7	75.8±1.4
	15	64.5±1.1	65.5±1.0	70.8±0.9	67.5±0.4	72.0±1.1	73.1±1.4
	25	56.9±1.1	65.2±1.3	66.4±1.6	65.6±1.9	69.0±1.8	72.7±1.4
Polblogs	0 (clean)	95.7±0.4	95.3±0.9	94.8±1.1	95.5±0.2	93.2±0.6	95.9±1.8
	15	65.0±1.9	82.1±1.4	71.7±1.3	87.8±0.3	86.0±1.2	90.0±1.6
	25	49.2±1.4	66.6±1.7	65.8±1.3	80.0±1.9	63.2±1.4	87.8±1.1
Pubmed	0 (clean)	87.2±0.1	86.4±0.7	85.1±0.2	87.7±0.0	87.3±0.2	87.7±1.7
	15	78.7±0.1	76.8±1.8	85.0±0.2	87.3±0.1	87.2±0.1	87.3±1.4
	25	75.5±0.2	70.8±1.9	85.1±0.2	86.0±0.1	86.8±0.2	87.2±1.4

(b) RGDM augmentation with the same backbone

Dataset	Ptb Rate (%)	RGDM (GCN)	RGDM (Pro-GNN)	RGDM (SG-GSR)
Cora	0 (clean)	85.2±0.4 (↑ 1.7)	84.3±0.5 (↑ 1.3)	**85.9±1.1** (↑ 0.4 ± 0.1)
	15	69.4±0.4 (↑ 4.3)	78.3±1.1 (↑ 1.9)	**80.0±1.1** (↑ 1.5 ± 0.3)
	25	61.4±0.8 (↑ 13.9)	72.3±1.6 (↑ 2.6)	**79.1±1.6** (↑ 1.3 ± 0.3)
Citeseer	0 (clean)	73.7±0.8 (↑ 1.7)	74.5±0.8 (↑ 1.2)	**76.6±1.4** (↑ 0.8)
	15	68.7±0.4 (↑ 4.2)	73.7±0.4 (↑ 1.7)	**76.0±1.1** (↑ 2.9 ± 0.4)
	25	62.8±0.7 (↑ 5.9)	70.2±0.7 (↑ 1.2)	**75.3±1.0** (↑ 2.6 ± 0.4)
Polblogs	0 (clean)	**96.1±0.1** (↑ 0.4)	95.7±1.2 (↑ 2.5)	**96.1±1.5** (↑ 0.2)
	15	69.9±0.4 (↑ 4.9)	88.0±1.1 (↑ 2.0)	**91.1±1.9** (↑ 1.1 ± 0.3)
	25	56.5±0.6 (↑ 7.3)	66.9±1.4 (↑ 3.7)	**88.8±1.8** (↑ 1.0 ± 0.3)
Pubmed	0 (clean)	88.1±0.2 (↑ 0.9)	88.4±0.3 (↑ 1.1)	**90.3±1.5** (↑ 2.6)
	15	85.0±0.3 (↑ 6.3)	88.1±0.2 (↑ 0.9)	**88.7±1.7** (↑ 1.4 ± 0.3)
	25	81.5±0.2 (↑ 6.0)	87.6±0.2 (↑ 0.8)	**88.6±1.3** (↑ 1.4 ± 0.3)

appendix. In addition, the results for defending against Topology Attack are deferred to Table 12 in Section F.2 of the appendix. The original accuracy tables report means and marginal standard deviations over 10 runs. In a separate paired evaluation, each backbone and its RGDM augmented counterpart use the same perturbed graph for each attack seed. Their difference measures the incremental gain of augmentation. These paired summaries are not obtained by subtracting the marginal SD in the original tables, whose runs need not coincide with the paired reevaluation. Table 26 reports the smaller standard deviations of the paired gains for all reported paired evaluation rows. Selected paired deviations are also shown alongside the accuracies in Table 1. Section F.11 distinguishes reported significance levels from exact p values inferred from rounded summaries. **Controlled gains and clean graphs.** The relevant comparison is RGDM(A) against the same classifier A: RGDM changes its training graph, not its architecture or objective. Under 25% Metattack, the SG-GSR gains are 1.3 ± 0.3, 2.6 ± 0.4, 1.0 ± 0.3, and 1.4 ± 0.3 percentage points on Cora, Citeseer, Polblogs, and Pubmed, respectively (paired gain ± paired SD). These increments are modest relative to the accuracy of the strong backbone. The much larger improvement over vanilla GCN does not measure improvement over SG-GSR. Reported significance levels across Metattack, Nettack, and Topology Attack are given in Table 27. The 0% perturbation and zero budget rows are explicitly clean graph evaluations. RGDM improves clean GCN by $1.7/1.7/0.4/0.9$ points and clean SG-GSR by $0.4/0.8/0.2/2.6$ points across these datasets. Thus the observed augmentation benefit also occurs when there are no adversarial edges to filter.

Table 2: Nettack accuracy ± marginal SD (%), with augmentation gains in parentheses. Budget 0 is a clean graph. The highlighting convention is that of Table 1. Complete results appear in Table 11.

(a) Original graph

Dataset	Budget	GCN	GCORNs	HANG	STABLE	Pro-GNN	SG-GSR
Cora	0 (clean)	81.4±1.0	82.6±0.4	80.8±1.3	85.3 ± 1.8	84.8±0.6	85.5±1.7
	3	68.0±1.7	73.2±0.7	73.1±1.9	73.8 ±1.8	72.4±0.5	74.9± 1.2
	5	55.5±1.7	66.9±1.1	68.8±1.6	68.4 ± 1.6	66.9±0.7	68.0±1.5
Citeseer	0 (clean)	81.0±1.4	81.9±1.2	81.1±1.1	82.4 ± 1.8	82.1±0.8	82.8± 1.2
	3	64.0±1.7	78.1±1.1	77.1±1.5	80.9 ± 1.5	79.7±2.0	81.7± 1.2
	5	47.6±1.2	71.7±1.8	73.5±1.5	72.5 ± 1.5	71.3±2.0	73.8±1.0
Polblogs	0 (clean)	97.0±0.2	97.5±0.3	97.4±0.5	98.0 ±1.4	97.1±0.2	97.5±1.3
	3	95.4±0.2	95.7±1.0	96.7±0.2	96.8 ± 1.4	96.9±0.1	95.4± 1.9
	5	93.0±0.5	93.4±0.9	95.9±0.3	96.2 ±1.8	96.1±0.3	96.8±1.4
Pubmed	0 (clean)	88.1±1.4	84.6±1.4	85.4±1.2	90.4 ±1.5	88.5±1.2	91.0± 1.1
	3	81.3±1.7	82.3±1.5	84.0±1.1	85.6 ±1.1	84.4±1.1	86.4±1.2
	5	68.3±1.2	72.3±1.1	70.6±1.3	73.9 ± 1.8	72.2±1.3	73.5± 1.4

(b) RGDM augmentation with the same backbone

Dataset	Budget	RGDM (GCN)	RGDM (Pro-GNN)	RGDM (SG-GSR)
Cora	0 (clean)	83.4±1.1 (↑ 2.0)	**86.0±1.1** (↑ 1.2)	**86.0 ±1.3** (↑ 0.5)
	3	70.5±1.1 (↑ 2.5)	74.3±1.3 (↑ 1.9)	**76.4 ±1.5** (↑ 1.5)
	5	59.4±1.2 (↑ 3.9)	**69.8±1.0** (↑ 2.9)	**69.8 ±1.8** (↑ 1.8)
Citeseer	0 (clean)	82.9±1.1 (↑ 1.9)	83.1±1.0 (↑ 1.0)	**83.7 ±1.3** (↑ 0.9)
	3	68.5±1.2 (↑ 4.5)	80.9±2.0 (↑ 1.2)	**83.6 ±1.8** (↑ 1.9)
	5	55.7±1.0 (↑ 8.1)	74.6±1.3 (↑ 3.3)	**75.6 ±1.4** (↑ 1.8)
Polblogs	0 (clean)	97.7±0.1 (↑ 0.7)	97.8±0.3 (↑ 0.7)	**98.5 ±0.2** (↑ 1.0)
	3	97.0±0.3 (↑ 1.6)	**97.2±0.6** (↑ 0.3)	96.4 ±0.1 (↑ 1.0)
	5	94.9±0.4 (↑ 1.9)	96.9±1.4 (↑ 0.8)	**97.7 ±0.2** (↑ 0.9)
Pubmed	0 (clean)	89.2±1.4 (↑ 1.1)	89.6±1.1 (↑ 1.1)	**92.5 ±1.7** (↑ 1.5)
	3	83.4±1.6 (↑ 2.1)	85.8±1.3 (↑ 1.4)	**89.0 ±1.0** (↑ 2.6)
	5	71.4±1.4 (↑ 3.1)	**76.3±1.0** (↑ 4.1)	75.7 ±1.3 (↑ 2.2)

4.2 Ablation Study on the Sparsification Loss and the Low-Rank Regularization

To validate the effectiveness of the sparsification loss and the low-rank regularization term in Equation (1), we conduct an ablation study on Cora, Citeseer, Polblogs, and Pubmed under the Metattack with the perturbation rate of 25%. We evaluate two variants of the RGDM based on the GCN, which are the RGDM without the sparsification loss and the RGDM without the low-rank regularization. We also compare RGDM with existing low-rank adversarial defense methods, [76, 77, 56, 78, 79, 80], with details provided in Section E.5 of the appendix. As shown in Table 3, RGDM outperforms both comparison regularizers across all datasets. Removing the sparsification loss reduces accuracy by $4.70/3.87/3.23/3.60$ points across the four datasets. The low rank penalty is complementary. Table 22 in Section F.7 extends the edge selection ablation to ogbn-arxiv and Reddit, where restoring the module adds 0.87 and 0.99 points under 20% improved DICE.

Table 3: Ablation study on the sparsification loss and the low-rank regularization term.

Method	Cora	Citeseer	Polblogs	Pubmed
RGDM (GCN) w/o Sparsification Loss ($\mathcal{L}_{\mathrm{S}}^{(t)}$)	56.72	58.97	53.24	77.92
RGDM (GCN) w/o Low-Rank Regularization ($\|\mathbf{K}\|_{r_0}$)	60.04	60.38	54.85	80.85
RGDM (GCN, Weight Orthogonality [76])	59.88	61.12	55.17	80.16
RGDM (GCN, Low-Rank Adjacency Matrix [56])	61.01	60.25	55.03	80.02
RGDM (GCN)	**61.42**	**62.84**	**56.47**	**81.52**

4.3 Comparison with Existing Node-Level Graph Augmentation Methods

To demonstrate the superiority of RGDM over existing graph augmentation methods, we compare it with the mix-up based conventional augmentation method iGraphMix [12], and GAN-based SGS

generation methods, including ImGAGN [42], Graphsmote [43], and SNS [45]. The evaluation is conducted on Cora, Citeseer, Polblogs, and Pubmed under Metattack with a perturbation rate of 25%. For a fair comparison, we compare GCN augmented by the competing node-level graph augmentation methods with RGDM (GCN), where the number of synthetic nodes added by each baseline is selected via cross-validation following Section F.6. As shown in Table 4, RGDM (GCN) significantly outperforms GCNs augmented by all competing baselines. We also include DoG [19] and LR-FGDM [20], using their generated structures with the same GCN classifier. RGDM exceeds DoG by $2.02/2.74/1.77/2.32$ points and LR-FGDM by $2.52/3.24/2.27/2.62$ points on Cora/Citeseer/Polblogs/Pubmed. These are augmentation comparisons with a common classifier, rather than comparisons with LR-FGDM's specialized few shot classifier.

Table 4: Comparisons between RGDM and existing node-level graph augmentation methods. The study is performed for Metattack with a perturbation rate of 25%.

Method	Cora	Citeseer	Polblogs	Pubmed
GCN	47.53	56.94	49.23	75.50
iGraphMix (GCN) [12]	53.25	57.11	51.65	76.10
ImGAGN (GCN) [42]	53.58	56.89	50.95	75.32
Graphsmote (GCN) [43]	55.10	58.06	52.44	76.25
SNS (GCN) [45]	58.35	59.17	54.03	78.28
DoG (GCN) [19]	59.40	60.10	54.70	79.20
LR-FGDM (GCN) [20]	58.90	59.60	54.20	78.90
RGDM (GCN)	**61.42**	**62.84**	**56.47**	**81.52**

4.4 Componentwise Comparison with DoG

Table 5 starts from DoG augmentation with BLND, keeps SG-GSR and 25% Metattack fixed, and successively adds RGDM components. Edge selection adds $1.0/1.1/1.0/0.8$ points. Latent TNN adds another $0.8/1.0/0.9/0.7$. Replacing BLND with SHED adds $1.7/2.2/1.6/2.0$. Every step improves all four datasets, and the complete generator is needed to exceed plain SG-GSR consistently. These are conditional increments along this sequence, not additive effects independent of component interactions. The HGAE reconstruction variant is evaluated in Table 33.

Table 5: Successive generator components. SG-GSR and 25% Metattack are fixed. Accuracy in percent.

Configuration	Cora	Citeseer	Polblogs	Pubmed
SG-GSR (no augmentation)	77.8	72.7	87.8	87.2
DoG augmentation (BLND)	75.6	71.0	85.3	85.1
+ Edge selection	76.6	72.1	86.3	85.9
+ Latent TNN (RGDM with BLND)	77.4	73.1	87.2	86.6
+ SHED (full RGDM)	**79.1**	**75.3**	**88.8**	**88.6**

4.5 Heterophily, Adaptive Attacks, and Efficiency

Heterophily. Theorem 3.1 makes no homophily assumption. Heterophily can nevertheless make a low rank bias inappropriate if it discards label relevant directions. RGDM uses a soft penalty and selects its starting rank by cross validation. Table 6 reports the LHS [88] comparison on Chameleon [89] and Actor [90]. Gains are positive at every tested perturbation rate, including 2.38 and 2.53 points at 25%. This supports the chosen penalty in these settings, not the claim that stronger rank reduction always helps. Section F.9 reports the standard deviations. **Adaptive threat**

Table 6: Heterophilic graphs under Metattack. Each entry is LHS / RGDM(LHS) accuracy (%).

Dataset	0% (clean)	15%	25%
Chameleon	72.31 / 74.02	71.87 / 72.98	70.03 / 72.41
Actor	38.87 / 41.80	36.79 / 39.01	35.37 / 37.90

model. Following [91], the white box attacker knows the generator, edge mask, TNN regularization,

and downstream SG-GSR. It perturbs the final augmented graph, including candidate edges incident to synthetic nodes, under the same absolute edge flip budget as 25% Metattack on the original graph. Thus augmentation does not protect synthetic structures by evaluation order. A second variant also differentiates the meta gradient through $\widehat{\mathbf{S}}_{ij} = \sigma(\boldsymbol{\theta}_{ij}/\tau)$, with a thresholded forward mask, straight through backward gradient, and $\tau = 0.5$. Table 7 gives both variants over 10 seeds. The gains remain positive, but these fixed budget tests do not establish immunity to arbitrary adaptive attacks. Equation (12) and Tables 34 to 35 in Section F.18 give the admissible perturbations and detailed results. **Wall clock efficiency.** Table 8 compares SHED with a global decoder on one NVIDIA A100.

Table 7: Adaptive Metattack: baseline SG-GSR / RGDM(SG-GSR) accuracy (%). The budget is fixed in absolute edge flips.

Dataset	Final graph attack	Through edge selection
Cora	74.8 / 77.8	74.5 / 76.9
Citeseer	69.5 / 73.4	69.2 / 72.4
Polblogs	84.7 / 87.6	84.4 / 86.6
Pubmed	84.9 / 87.2	84.6 / 86.3

SHED reduces RGAE training by $3.14\times$ on Pubmed and $5.56\times$ on ogbn-arxiv. Per node generation includes LDM sampling and decoding. Its speedups are $1.10\times$ and $3.27\times$, respectively. These are not downstream GNN inference times. The decoder operation count reduction therefore should not be identified with end to end generation speedup. Table 15 in Section F.5 gives all five datasets, and Section D.2 explains the sparse decoding cost.

Table 8: SHED / global decoder timings: RGAE training and complete synthetic node generation.

Dataset	Training (min)	Generation (s/node)
Pubmed	153 / 480	0.073 / 0.080
ogbn-arxiv	452 / 2511	0.15 / 0.49

5 Conclusions

We present the Robust Graph Diffusion Model (RGDM), marking the first robust graph diffusion model designed to generate synthetic graph structures, which are the synthetic nodes and synthetic edges connecting to them in the graph, for the node classification task. In addition, the RGDM significantly mitigates the negative effects of graph adversarial attacks and generates faithful synthetic graph structures, benefiting from the edge selection mechanism and the low-rank latent feature learning in the RGAE. The experiment results show that incorporating the synthetic graph structures into the original attacked graph for the training of GNNs significantly improves the classification accuracy for the semi-supervised node classification under various types of graph adversarial attacks.

Acknowledgments and Disclosure of Funding

This work was supported by National Institute of Health 1OT2OD037955-01.

References

[1] Brandon Trabucco, Kyle Doherty, Max A Gurinas, and Ruslan Salakhutdinov. Effective data augmentation with diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

[2] Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi, and David J Fleet. Synthetic data from diffusion models improves imagenet classification. *Transactions on Machine Learning Research*, 2023.

[3] Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *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*, pages 4474–4484. PMLR, 2020.

[4] 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.

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

[6] 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*, 2023.

[7] Stratis Limnios, Praveen Selvaraj, Mihai Cucuringu, Carsten Maple, Gesine Reinert, and Andrew Elliott. Sagess: Sampling graph denoising diffusion model for scalable graph generation. *CoRR*, abs/2306.16827, 2023.

[8] Xiaotian Han, Zhimeng Jiang, Ninghao Liu, and Xia Hu. G-mixup: Graph data augmentation for graph classification. In *International Conference on Machine Learning (ICML)*, pages 8230–8248. PMLR, 2022.

[9] Yiwei Wang, Wei Wang, Yuxuan Liang, Yujun Cai, and Bryan Hooi. Mixup for node and graph classification. In *Proceedings of the Web Conference 2021*, pages 3663–3674, 2021.

[10] Vikas Verma, Meng Qu, Kenji Kawaguchi, Alex Lamb, Yoshua Bengio, Juho Kannala, and Jian Tang. Graphmix: Improved training of gnns for semi-supervised learning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 10024–10032. AAAI Press, 2021.

[11] Wentao Zhao, Qitian Wu, Chenxiao Yang, and Junchi Yan. Geomix: Towards geometry-aware data augmentation. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 4500–4511, 2024.

[12] Jongwon Jeong, Hoyeop Lee, Hyui Geon Yoon, Beomyoung Lee, Junhee Heo, Geonsoo Kim, and Kim Jin Seon. igraphmix: Input graph mixup method for node classification. In *The Twelfth International Conference on Learning Representations*, 2024.

[13] Weigang Lu, Ziyu Guan, Wei Zhao, Yaming Yang, and Long Jin. Nodemixup: Tackling under-reaching for graph neural networks. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *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*, pages 14175–14183. AAAI Press, 2024.

[14] Nan Jia, Xiaolin Tian, Wenxing Gao, and Licheng Jiao. Deep graph-convolutional generative adversarial network for semi-supervised learning on graphs. *Remote Sensing*, 15(12):3172, 2023.

[15] Cheng Wu, Chaokun Wang, Jingcao Xu, Ziyang Liu, Kai Zheng, Xiaowei Wang, Yang Song, and Kun Gai. Graph contrastive learning with generative adversarial network. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 2721–2730, 2023.

[16] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Graphgan: Graph representation learning with generative adversarial nets. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 2508–2515. AAAI Press, 2018.

[17] Ming Liang, Yao Meng, Jiyu Wang, David L Lubkeman, and Ning Lu. Feedergan: Synthetic feeder generation via deep graph adversarial nets. *IEEE Transactions on Smart Grid*, 12(2):1163–1173, 2020.

[18] Carl Yang, Peiye Zhuang, Wenhan Shi, Alan Luu, and Pan Li. Conditional structure generation through graph variational generative adversarial nets. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

[19] Yancheng Wang, Changyu Liu, and Yingzhen Yang. Diffusion on graph: Augmentation of graph structure for node classification. *Trans. Mach. Learn. Res.*, 2025, 2025.

[20] Yancheng Wang, Chengshuai Zhao, Dongfang Sun, Huan Liu, and Yingzhen Yang. Low-rank few-shot node classification by node-level graph diffusion. In *The Fourteenth International Conference on Learning Representations*, 2026.

[21] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:8780–8794, 2021.

[22] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, volume 97, pages 5301–5310. PMLR, 09–15 Jun 2019.

[23] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 322–332. PMLR, 2019.

[24] Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards understanding the spectral bias of deep learning. In Zhi-Hua Zhou, editor, *International Joint Conference on Artificial Intelligence*, pages 2205–2211. ijcai.org, 2021.

[25] Moulik Choraria, Leello Tadesse Dadi, Grigorios Chrysos, Julien Mairal, and Volkan Cevher. The spectral bias of polynomial neural networks. In *International Conference on Learning Representations*. OpenReview.net, 2022.

[26] Yancheng Wang, Rajeev Goel, Utkarsh Nath, Alvin C. Silva, Teresa Wu, and Yingzhen Yang. Learning low-rank feature for thorax disease classification. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[27] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[28] Xin He, Wenqi Fan, Yili Wang, Chengyi Liu, Rui Miao, Xin Juan, and Xin Wang. Graph defense diffusion model. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 416–427, 2026.

[29] Jiayi Luo, Qingyun Sun, Haonan Yuan, Xingcheng Fu, and Jianxin Li. Robust graph learning against adversarial evasion attacks via prior-free diffusion-based structure purification. In *Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025- 2 May 2025*, pages 2098–2110. ACM, 2025.

[30] Chu Zhao, Enneng Yang, Yuliang Liang, Jianzhe Zhao, Guibing Guo, and Xingwei Wang. Distributionally robust graph out-of-distribution recommendation via diffusion model. In *Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025- 2 May 2025*, pages 2018–2031. ACM, 2025.

[31] Ganglin Xie, Haobo Lu, Xinze Zhang, and Kun He. Advancing SAR image robustness: Integrating diffusion models for adversarial purification and speckle noise suppression. In *2025 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2025, Hyderabad, India, April 6-11, 2025*, pages 1–5. IEEE, 2025.

[32] Xinquan Chen, Xitong Gao, Juanjuan Zhao, Kejiang Ye, and Cheng-Zhong Xu. Advdiffuser: Natural adversarial example synthesis with diffusion models. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 4539–4549. IEEE, 2023.

[33] Yeonjun In, Kanghoon Yoon, Kibum Kim, Kijung Shin, and Chanyoung Park. Self-guided robust graph structure refinement. In *Proceedings of the ACM Web Conference 2024*, pages 697–708, 2024.

[34] Johannes Gasteiger, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. *Advances in neural information processing systems*, 32, 2019.

[35] Tong Zhao, Yozen Liu, Leonardo Neves, Oliver J. Woodford, Meng Jiang, and Neil Shah. Data augmentation for graph neural networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 11015–11023. AAAI Press, 2021.

[36] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2020.

[37] Wenzheng Feng, Yuxiao Dong, Tinglin Huang, Ziqi Yin, Xu Cheng, Evgeny Kharlamov, and Jie Tang. GRAND+: scalable graph random neural networks. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 3248–3258. ACM, 2022.

[38] Yurui Lai, Xiaoyang Lin, Renchi Yang, and Hongtao Wang. Efficient topology-aware data augmentation for high-degree graph neural networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1463–1473, 2024.

[39] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[40] Kezhi Kong, Guohao Li, Mucong Ding, Zuxuan Wu, Chen Zhu, Bernard Ghanem, Gavin Taylor, and Tom Goldstein. Robust optimization as data augmentation for large-scale graphs. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 60–69. IEEE, 2022.

[41] Amitoz Azad and Yuan Fang. A learned generalized geodesic distance function-based approach for node feature augmentation on graphs. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 49–58, 2024.

[42] Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin. Imgagn: Imbalanced network embedding via generative adversarial graph networks. In Feida Zhu, Beng Chin Ooi, and Chunyan Miao, editors, *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 1390–1398. ACM, 2021.

[43] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 833–841, 2021.

[44] Hung Chun Hsu, Ting-Le Lin, Bo-Jun Wu, Ming-Yi Hong, Che Lin, and Chih-Yu Wang. Fincgan: A gan framework of imbalanced node classification on heterogeneous graph neural network. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5750–5754. IEEE, 2024.

[45] Xinyi Gao, Wentao Zhang, Tong Chen, Junliang Yu, Hung Quoc Viet Nguyen, and Hongzhi Yin. Semantic-aware node synthesis for imbalanced heterogeneous information networks. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 545–555, 2023.

[46] Hung Chun Hsu, Bo-Jun Wu, Ming-Yi Hong, Che Lin, and Chih-Yu Wang. A GAN approach for node embedding in heterogeneous graphs using subgraph sampling. *CoRR*, abs/2312.06519, 2023.

[47] 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.

[48] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, 2018.

[49] Enyan Dai, Tianxiang Zhao, Huaisheng Zhu, Junjie Xu, Zhimeng Guo, Hui Liu, Jiliang Tang, and Suhang Wang. A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability. *CoRR*, abs/2204.08570, 2022.

[50] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2847–2856, 2018.

[51] Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *International Conference on Learning Representations (ICLR)*, 2019.

[52] Yiwei Sun, Suhang Wang, Xianfeng Tang, Tsung-Yu Hsieh, and Vasant Honavar. Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach. In *Proceedings of the Web Conference 2020*, pages 673–683, 2020.

[53] Jintang Li, Jiaying Peng, Liang Chen, Zibin Zheng, Tingting Liang, and Qing Ling. Spectral adversarial training for robust graph neural network. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[54] Fuli Feng, Xiangnan He, Jie Tang, and Tat-Seng Chua. Graph adversarial training: Dynamically regularizing based on graph structure. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[55] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. Adversarial examples for graph data: Deep insights into attack and defense. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI*, 2019.

[56] Negin Entezari, Saba A Al-Sayouri, Amirali Darvishzadeh, and Evangelos E Papalexakis. All you need is low (rank) defending against adversarial attacks on graphs. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 169–177, 2020.

[57] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 66–74, 2020.

[58] Runlin Lei, Zhen Wang, Yaliang Li, Bolin Ding, and Zhewei Wei. Evennet: Ignoring odd-hop neighbors improves robustness of graph neural networks. *Advances in Neural Information Processing Systems*, 2022.

[59] Beini Xie, Heng Chang, Ziwei Zhang, Xin Wang, Daixin Wang, Zhiqiang Zhang, Rex Ying, and Wenwu Zhu. Adversarially robust neural architecture search for graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8143–8152, 2023.

[60] T Konstantin Rusch, Ben Chamberlain, James Rowbottom, Siddhartha Mishra, and Michael Bronstein. Graph-coupled oscillator networks. In *International Conference on Machine Learning*, pages 18888–18909. PMLR, 2022.

[61] Yang Song, Qiyu Kang, Sijie Wang, Kai Zhao, and Wee Peng Tay. On the robustness of graph neural diffusion to topology perturbations. *Advances in Neural Information Processing Systems*, 35:6384–6396, 2022.

[62] Kai Zhao, Qiyu Kang, Yang Song, Rui She, Sijie Wang, and Wee Peng Tay. Adversarial robustness in graph neural networks: A hamiltonian approach. *Advances in Neural Information Processing Systems*, 2023.

[63] Yaning Jia, Dongmian Zou, Hongfei Wang, and Hai Jin. Enhancing node-level adversarial defenses by lipschitz regularization of graph neural networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 951–963, 2023.

[64] Songtao Liu, Jinghui Chen, Tianfan Fu, Lu Lin, Marinka Zitnik, and Dinghao Wu. Graph adversarial diffusion convolution. In *International Conference on Machine Learning*, 2024.

[65] Yassine Abbahaddou, Sofiane Ennadir, Johannes F. Lutzeyer, Michalis Vazirgiannis, and Henrik Boström. Bounding the expected robustness of graph neural networks subject to node feature attacks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[66] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[67] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[68] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[69] Mikko I Malinen and Pasi Fränti. Balanced k-means for clustering. In *Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, S+ SSPR 2014, Joensuu, Finland, August 20-22, 2014. Proceedings*, pages 32–41. Springer, 2014.

[70] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[71] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pages 11458–11468. PMLR, 2020.

[72] Subhajit Dutta Chowdhury, Zhiyu Ni, Qingyuan Peng, Souvik Kundu, and Pierluigi Nuzzo. Unveiling adversarially robust graph lottery tickets. *Transactions on Machine Learning Research*, 2024.

[73] Pranjal Awasthi, Himanshu Jain, Ankit Singh Rawat, and Aravindan Vijayaraghavan. Adversarial robustness via robust low rank representations. *Advances in Neural Information Processing Systems*, 33:11391–11403, 2020.

[74] Pranjal Awasthi, Vaggos Chatziafratis, Xue Chen, and Aravindan Vijayaraghavan. Adversarially robust low dimensional representations. In *Conference on Learning Theory*, pages 237–325. PMLR, 2021.

[75] Nanqing Xu, Weiwei Feng, Tianzhu Zhang, and Yongdong Zhang. Fd-gan: Generalizable and robust forgery detection via generative adversarial networks. *International Journal of Computer Vision*, 132(12):5801–5819, 2024.

[76] Dayana Savostianova, Emanuele Zangrando, Gianluca Ceruti, and Francesco Tudisco. Robust low-rank training via approximate orthonormal constraints. *Advances in Neural Information Processing Systems*, 36:66064–66083, 2023.

[77] Zhebin Wu, Lin Shu, Ziyue Xu, Yaomin Chang, Chuan Chen, and Zibin Zheng. Robust tensor graph convolutional networks via t-svd based graph augmentation. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2090–2099, 2022.

[78] Chenhui Deng, Xiuyu Li, Zhuo Feng, and Zhiru Zhang. Garnet: Reduced-rank topology learning for robust and scalable graph neural networks. In *Learning on Graphs Conference*, pages 3–1. PMLR, 2022.

[79] Chenyu Zhou, Wei Huang, Xinyuan Miao, Yabin Peng, Xianglong Kong, Yi Cao, and Xi Chen. Fortifying graph neural networks against adversarial attacks via ensemble learning. *Knowledge-Based Systems*, 309:112867, 2025.

[80] Jianfu Zhang, Yan Hong, Dawei Cheng, Liqing Zhang, and Qibin Zhao. Defending adversarial attacks in graph neural networks via tensor enhancement. *Pattern Recognition*, 158:110954, 2025.

[81] Yingzhen Yang. A new concentration inequality for sampling without replacement and its application for transductive learning. In *Forty-second International Conference on Machine Learning*, 2025.

[82] Yingzhen Yang. Improved generalization bounds for transductive learning by transductive local complexity and its applications. 2025.

[83] Yingzhen Yang. Even sharper bounds for transductive learning and its applications. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026.

[84] 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*, pages 10674–10685. IEEE, 2022.

[85] Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. Topology attack and defense for graph neural networks: An optimization perspective. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3961–3967. ijcai.org, 2019.

[86] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Mag.*, 29(3):93–106, 2008.

[87] Lada A Adamic and Natalie Glance. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*, pages 36–43, 2005.

[88] Chenyang Qiu, Guoshun Nan, Tianyu Xiong, Wendi Deng, Di Wang, Zhiyang Teng, Lijuan Sun, Qimei Cui, and Xiaofeng Tao. Refining latent homophilic structures over heterophilic graphs for robust graph convolution networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 8930–8938, 2024.

[89] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[90] Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 807–816. ACM, 2009.

[91] Felix Mujkanovic, Simon Geisler, Stephan Günnemann, and Aleksandar Bojchevski. Are defenses for graph neural networks robust? *Advances in Neural Information Processing Systems*, 35:8954–8968, 2022.

[92] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *KDD*, 2019.

[93] Boyuan Feng, Yuke Wang, and Yufei Ding. Uag: Uncertainty-aware attention graph neural network for defending adversarial attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7404–7412, 2021.

[94] Kuan Li, Yang Liu, Xiang Ao, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. Reliable representations make a stronger defender: Unsupervised structure refinement for robust gnn. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 925–935, 2022.

[95] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[96] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[97] DC Dowson and BV666017 Landau. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 1982.

[98] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in Neural Information Processing Systems*, 2021.

[99] 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, editors, *Advances in Neural Information Processing Systems 30, NeurIPS 2017, December 4-9, 2017*, pages 1024–1034, 2017.

[100] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*, 2020.

[101] Kuan Li, Yang Liu, Xiang Ao, and Qing He. Revisiting graph adversarial attack and defense from a data distribution perspective. In *The Eleventh International Conference on Learning Representations*, 2022.

A Theoretical Results

We apply the general STLC inequality in [83, Theorem 4.2], rather than its empirical risk minimizer bound. This distinction retains a uniform guarantee for bounded probes and hence for every gradient descent iterate that satisfies the stated norm bound. All probabilities below are over the probe split. $\mathbf{Z}$ and $\mathbf{Y}$ are fixed. For a function $h : [N] \to \mathbb{R}$, write $P_N h := N^{-1} \sum_{i=1}^{N} h(i)$ and $P_{\mathcal{A}} h := |\mathcal{A}|^{-1} \sum_{i \in \mathcal{A}} h(i)$ for nonempty $\mathcal{A} \subseteq [N]$.

Proof of Theorem 3.1. Let $i, j \in [N]$ be indices and $c \in [C]$ a class index. We use $\langle \cdot, \cdot \rangle$ for the Euclidean inner product, range$(\cdot)$ for the column space, and tr $(\cdot)$ for the trace of a square matrix. The superscript $\top$ denotes transpose, and $\mathbf{0}$ denotes the zero matrix of the required dimensions. We retain the row and restriction conventions of Section 3.1. Thus $\mathbf{Y}_{ic} = (\mathbf{y}_i)_c$, $\mathbf{Z}_i$ is a row vector, and $\mathbf{W} \in \mathbb{R}^{D' \times C}$ is the coefficient matrix of the probe $\mathbf{F}(\mathbf{W}) = \mathbf{Z}\mathbf{W}$ introduced before the theorem. The sets $\mathcal{L}, \mathcal{U}$ are the indices of $\mathcal{V}_{\mathcal{L}}, \mathcal{V}_{\mathcal{U}}$ and have cardinalities m, u.

Residual enlargement. Put $\mu := \sqrt{B^2 + 1}$ and $R := \mu\sqrt{\tau_0^2 + 1}$, so $R \geq \mu \geq 1$. For each class $c \in [C]$ and each $i \in [N]$, define $\mathbf{x}_i^{(c)} := [\mathbf{Z}_i, \mathbf{Y}_{ic}]^\top \in \mathbb{R}^{D'+1}$. For $\mathbf{v} \in \mathbb{R}^{D'+1}$, define $f_{\mathbf{v}} : [N] \to \mathbb{R}$ by $f_{\mathbf{v}}(i) := \left\langle \mathbf{v}, \mathbf{x}_i^{(c)} \right\rangle$. The finite linear kernel class is

$$\mathcal{F}_c := \left\{ f_{\mathbf{v}} : \mathbf{v} \in \mathbb{R}^{D'+1},\ \|\mathbf{v}\|_2 \leq \mu \right\}.$$

Write $\mathbf{Y}^{(c)} \in \mathbb{R}^N$ for column c of $\mathbf{Y}$. The Gram matrix of $\mathcal{F}_c$ is $\mathbf{G}_c := \mathbf{K} + \mathbf{Y}^{(c)}\mathbf{Y}^{(c)\top}$. Projection onto the span of the feature vectors preserves evaluations and does not increase the coefficient norm. Thus $\mathcal{F}_c$ is the evaluation class of the radius μ transductive reproducing kernel Hilbert space (RKHS) ball for this kernel, including when $\mathbf{G}_c$ is singular. Every $f \in \mathcal{F}_c$ satisfies $|f(i)| \leq R$. Write $\mathbf{w}_c \in \mathbb{R}^{D'}$ for column c of $\mathbf{W}$. If $\|\mathbf{W}\|_{\mathrm{F}} \leq B$, the vector $\left[\mathbf{w}_c^\top, -1\right]^\top$ has norm at most μ, so the residual function $i \mapsto (\mathbf{Z}\mathbf{W})_{ic} - \mathbf{Y}_{ic}$ belongs to $\mathcal{F}_c$. The label coordinate is used only to enlarge the loss class in the proof. It is not supplied to the learner.

Envelope and localization. Powers of functions are taken pointwise. Let $\mathcal{H}_c := \left\{ f^2 : f \in \mathcal{F}_c \right\}$. For $h \in \mathcal{H}_c$, put $\tilde{T}_c(h) := R^2 P_N h$. Since $0 \leq h(i) \leq R^2$, we have $P_N(h^2) \leq \tilde{T}_c(h)$. For $\rho > 0$, put $\mathcal{H}_c(\rho) := \left\{ h \in \mathcal{H}_c : \tilde{T}_c(h) \leq \rho \right\}$ and $\mathcal{H}_c^2(\rho) := \left\{ h^2 : h \in \mathcal{H}_c(\rho) \right\}$. These classes contain zero. For a nonempty class $\mathcal{F}$ of functions on $[N]$ and $p \in \{u, m\}$, let $J_1, \ldots, J_p$ be independent uniform indices in $[N]$, and let $\sigma_1, \ldots, \sigma_p$ be independent Rademacher variables, each taking values $+1$ and -1 with equal probability and independent of these indices. Put $\boldsymbol{\sigma} := (\sigma_1, \ldots, \sigma_p)^\top$ and define

$$\mathcal{R}_p(\mathcal{F}) := \mathbb{E}_{J_1, \ldots, J_p, \boldsymbol{\sigma}} \left[\sup_{f \in \mathcal{F}} \frac{1}{p} \sum_{k=1}^{p} \sigma_k f(J_k) \right], \qquad p \in \{u, m\}.$$

These auxiliary variables may be taken independent of the probe split. Let $\lambda_{c,1} \geq \cdots \geq \lambda_{c,N} \geq 0$ be the eigenvalues of $\mathbf{G}_c/N$, and, for each integer $Q \in \{0, \ldots, N\}$, put $S_{c,Q} := \sum_{j=Q+1}^{N} \lambda_{c,j}$, with $S_{c,N} = 0$. This auxiliary normalized spectrum will be compared with the eigenvalues $\widehat{\lambda}_j$ of $\mathbf{K}$ defined in the main text. The spectral estimate in [83, Lemma C.11] gives, for $v \geq 0$,

$$\begin{aligned} \phi_{c,p}(v) &:= \min_{0 \leq Q \leq N} \left[\sqrt{\frac{vQ}{p}} + \mu \sqrt{\frac{S_{c,Q}}{p}} \right], \\ \mathcal{R}_p\left(\left\{ f \in \mathcal{F}_c : P_N(f^2) \leq v \right\} \right) &\leq \phi_{c,p}(v). \end{aligned} \tag{4}$$

The minima in Equation (4) are over integers. Let $\mathcal{A}_p \subseteq [N]$ be a uniformly sampled subset with p elements, and let $\varepsilon \in \{+1, -1\}$. Denote by $\mathrm{TC}_p^{\varepsilon}(\mathcal{F})$ the expected supremum of $\varepsilon(P_{\mathcal{A}_p} f - P_N f)$ over $f \in \mathcal{F}$. The comparison and contraction inequalities in [83, Theorems A.4 and A.1] imply

$$\begin{aligned} \mathrm{TC}_p^{\varepsilon}(\mathcal{H}_c(\rho)) &\leq 2\mathcal{R}_p(\mathcal{H}_c(\rho)) \leq 4R\phi_{c,p}(\rho/R^2), \\ \mathrm{TC}_p^{\varepsilon}(\mathcal{H}_c^2(\rho)) &\leq 4R^2\mathcal{R}_p(\mathcal{H}_c(\rho)) \leq 8R^3\phi_{c,p}(\rho/R^2). \end{aligned}$$

For contraction, the globally Lipschitz maps $a \mapsto \min\{|a|, R\}^2$ and $a \mapsto \min\{|a|, R^2\}^2$ agree with squaring on the respective bounded classes, vanish at zero, and have Lipschitz constants $2R$ and $2R^2$. Thus no global Lipschitz assumption on the unbounded squared loss function is needed.

Subroot majorant and fixed point. A function $\psi : [0,\infty) \to [0,\infty)$ is called subroot if it is nondecreasing and $\psi(\rho)/\sqrt{\rho}$ is nonincreasing for $\rho > 0$. Set $A := 8R^3$ and, for $\rho \geq 0$, define

$$\psi_c(\rho) := A\left[\phi_{c,u}(\rho/R^2) + \phi_{c,m}(\rho/R^2)\right].$$

Finite minima and positive sums of the displayed functions combining spectral head and tail terms are subroot, so ψ_c is subroot and majorizes all four localized complexities and their squared class counterparts. If $\mathbf{G}_c = \mathbf{0}$, set $\rho_c^* := 0$. Every function in $\mathcal{H}_c$ is then zero and its loss bound is deterministic. Otherwise ψ_c is nonzero, and we define $\rho_c^* > 0$ by $\psi_c(\rho_c^*) = \rho_c^*$. For any integer $Q \in \{0,\dots,N\}$, put $D_Q := R^{-1}(\sqrt{Q/u} + \sqrt{Q/m})$ and $E_{c,Q} := \mu\sqrt{S_{c,Q}}(u^{-1/2} + m^{-1/2})$. Put $\kappa := 128R^4$. The fixed point identity and Equation (4) yield

$$\begin{aligned}
&\rho_c^* \leq AD_Q\sqrt{\rho_c^*} + AE_{c,Q} \leq \tfrac{1}{2}\rho_c^* + \tfrac{1}{2}A^2D_Q^2 + AE_{c,Q},\\
&\rho_c^* \leq A^2D_Q^2 + 2AE_{c,Q} \leq \kappa\left[Q\left(\frac{1}{u}+\frac{1}{m}\right) + \sqrt{S_{c,Q}}\left(\frac{1}{\sqrt{u}}+\frac{1}{\sqrt{m}}\right)\right],
\end{aligned} \tag{5}$$

Indeed, $D_Q^2 \leq 2Q(R^{-2})(1/u + 1/m)$, $2A^2/R^2 = 128R^4$, and $2A\mu \leq 16R^4$. Minimizing Equation (5) over Q bounds the fixed point uniformly in the split sizes and the spectrum.

Rank one comparison with the latent kernel. For $r \in [q]$, let $\mathbf{K}_r$ be the spectral truncation of $\mathbf{K}$ at rank r, and set $Q_r := \min\{r+1, N\}$. Project onto the span of range$(\mathbf{K}_r)$ and $\mathbf{Y}^{(c)}$. This span has dimension at most Q_r, and the normalized trace of the residual of $\mathbf{G}_c$ is at most $\operatorname{tr}(\mathbf{K} - \mathbf{K}_r)/N = \sum_{j=r+1}^{N}\widehat{\lambda}_j/N = \|\mathbf{K}\|_r/N$. The variational characterization of the sum of the trailing eigenvalues therefore gives

$$S_{c,Q_r} \leq \frac{\|\mathbf{K}\|_r}{N}, \qquad Q_r\left(\frac{1}{u}+\frac{1}{m}\right) \leq 2r\left(\frac{1}{u}+\frac{1}{m}\right). \tag{6}$$

For completeness, we verify both inequalities in Equation (6). Fix $c \in [C]$ and $r \in [q]$ for this argument. Let $\mathcal{S}_{c,r} \subseteq \mathbb{R}^N$ be the linear span of range$(\mathbf{K}_r)$ and $\mathbf{Y}^{(c)}$, let $\mathbf{P}_{c,r} \in \mathbb{R}^{N\times N}$ be the orthogonal projector onto $\mathcal{S}_{c,r}$, and let $\mathbf{I}_N$ be the identity matrix on $\mathbb{R}^N$. Since rank$(\mathbf{K}_r) \leq r$, we have

$$\begin{aligned}
&\operatorname{rank}(\mathbf{P}_{c,r}) = \dim(\mathcal{S}_{c,r}) \leq \min\{r+1, N\} = Q_r,\\
&(\mathbf{I}_N - \mathbf{P}_{c,r})\mathbf{Y}^{(c)} = \mathbf{0}, \qquad (\mathbf{I}_N - \mathbf{P}_{c,r})\mathbf{K}_r = \mathbf{0}.
\end{aligned}$$

Choose an orthonormal eigenbasis $\mathbf{u}_{c,1},\dots,\mathbf{u}_{c,N} \in \mathbb{R}^N$ of $\mathbf{G}_c/N$ with corresponding eigenvalues $\lambda_{c,1},\dots,\lambda_{c,N}$. For $j \in [N]$, put $\alpha_j := \|\mathbf{P}_{c,r}\mathbf{u}_{c,j}\|_2^2$. Orthogonality gives $0 \leq \alpha_j \leq 1$ and $\sum_{j=1}^{N}\alpha_j = \operatorname{tr}(\mathbf{P}_{c,r}) = \operatorname{rank}(\mathbf{P}_{c,r}) \leq Q_r$. The ordering and nonnegativity of the eigenvalues therefore imply

$$\frac{\operatorname{tr}(\mathbf{P}_{c,r}\mathbf{G}_c)}{N} = \sum_{j=1}^{N}\lambda_{c,j}\alpha_j \leq \lambda_{c,Q_r}\sum_{j=1}^{N}\alpha_j + \sum_{j=1}^{Q_r}(\lambda_{c,j} - \lambda_{c,Q_r}) \leq \sum_{j=1}^{Q_r}\lambda_{c,j}.$$

Indeed, for the first inequality, subtract $\lambda_{c,Q_r}\sum_{j=1}^{N}\alpha_j$ from the preceding sum. The terms with $j > Q_r$ are nonpositive, while $\alpha_j \leq 1$ bounds the terms with $j \leq Q_r$. The second inequality uses $\sum_{j=1}^{N}\alpha_j \leq Q_r$ and $\lambda_{c,Q_r} \geq 0$. Consequently,

$$\begin{aligned}
S_{c,Q_r} &= \frac{\operatorname{tr}(\mathbf{G}_c)}{N} - \sum_{j=1}^{Q_r}\lambda_{c,j} \leq \frac{\operatorname{tr}((\mathbf{I}_N - \mathbf{P}_{c,r})\mathbf{G}_c)}{N}\\
&= \frac{\operatorname{tr}((\mathbf{I}_N - \mathbf{P}_{c,r})(\mathbf{K} - \mathbf{K}_r))}{N} \leq \frac{\operatorname{tr}(\mathbf{K} - \mathbf{K}_r)}{N} = \frac{\|\mathbf{K}\|_r}{N}.
\end{aligned}$$

The equality in the last line follows from $\mathbf{G}_c = \mathbf{K} + \mathbf{Y}^{(c)}\mathbf{Y}^{(c)\top}$ and the two annihilation identities above. The last inequality holds because $\mathbf{K} - \mathbf{K}_r$ is positive semidefinite and

$$\operatorname{tr}(\mathbf{P}_{c,r}(\mathbf{K} - \mathbf{K}_r)) = \operatorname{tr}(\mathbf{P}_{c,r}(\mathbf{K} - \mathbf{K}_r)\mathbf{P}_{c,r}) \geq 0.$$

This argument also applies when $Q_r = N$, in which case $S_{c,Q_r} = 0$. The second inequality holds since $r \geq 1$ gives $Q_r \leq r + 1 \leq 2r$, and it covers the case that $r = N$.

Combining Equations (5) and (6), and minimizing over $r \in [q]$, gives $\rho_c^* \leq 2\kappa \mathrm{KC}(\mathbf{K})$.

STLC and absorption of the full sample loss. Put $H_0 := R^2$ and $K_0 := 2R^2 > 1$. Apply [83, Theorem 4.2] with these constants, the surrogate $\tilde{T}_c$, and the majorant ψ_c. Its constants can be taken as $a_1 := 16R^2(4 + 4\sqrt{219}/7)^2$ and $a_2 := 3536R^2 + \sqrt{219}$. For every $z > 0$, with probability at least $1 - \exp(-z)$, simultaneously for $h \in \mathcal{H}_c$,

$$P_{\mathcal{U}} h \leq P_{\mathcal{L}} h + \tfrac{1}{2} P_N h + a_1 \rho_c^* + \frac{a_2 z}{s}. \tag{7}$$

Equation (7) also holds deterministically for a zero class, with $\rho_c^* = 0$. Using $P_N h = (m/N) P_{\mathcal{L}} h + (u/N) P_{\mathcal{U}} h$ and $h \geq 0$, we obtain

$$\begin{aligned} P_{\mathcal{U}} h &\leq \frac{1 + m/(2N)}{1 - u/(2N)} P_{\mathcal{L}} h + \frac{a_1 \rho_c^* + a_2 z/s}{1 - u/(2N)} \\ &\leq 2 P_{\mathcal{L}} h + 2 a_1 \rho_c^* + \frac{2 a_2 z}{s}. \end{aligned}$$

Here $1 - u/(2N) \geq 1/2$ and $(1 + m/(2N))/(1 - u/(2N)) \leq 2$. Set $z = x + \log C$ and intersect the C classwise events. Their total failure probability is at most $C \exp(-z) = \exp(-x)$. On this single event the inequalities hold for every residual generated by every $\|\mathbf{W}\|_{\mathrm{F}} \leq B$. For $h_{c,\mathbf{W}}(i) := ((\mathbf{Z}\mathbf{W})_{ic} - \mathbf{Y}_{ic})^2$, the restriction convention gives

$$\begin{aligned} \sum_{c=1}^{C} P_{\mathcal{L}} h_{c,\mathbf{W}} &= \frac{\|[\mathbf{Z}\mathbf{W} - \mathbf{Y}]_{\mathcal{L}}\|_{\mathrm{F}}^2}{m} = L_{\mathcal{L}}(\mathbf{W}), \\ \sum_{c=1}^{C} P_{\mathcal{U}} h_{c,\mathbf{W}} &= \frac{\|[\mathbf{Z}\mathbf{W} - \mathbf{Y}]_{\mathcal{U}}\|_{\mathrm{F}}^2}{u} = \mathcal{U}_{\text{test}}(\mathbf{W}). \end{aligned}$$

Equation (2) is then proved by summing over c, using $\rho_c^* \leq 2\kappa \mathrm{KC}(\mathbf{K})$, and choosing

$$c_0 := C \max\{4a_1\kappa, 2a_2\}.$$

Here c_0 depends only on C, B, τ_0. □

Squared loss gradient descent. The linear probe analysis uses the squared training objective $\frac{1}{2} L_{\mathcal{L}}(\mathbf{W})$, not softmax cross entropy. With $\mathbf{W}^{(0)} = \mathbf{0}$ and step size $\eta > 0$, its exact update and residual recurrence are

$$\mathbf{W}^{(t)} = \mathbf{W}^{(t-1)} - \frac{\eta}{m} [\mathbf{Z}]_{\mathcal{L}}^{\top} [\mathbf{Z}\mathbf{W}^{(t-1)} - \mathbf{Y}]_{\mathcal{L}}, \tag{8}$$

$$[\mathbf{Z}\mathbf{W}^{(t)} - \mathbf{Y}]_{\mathcal{L}} = -\left(\mathbf{I}_m - \frac{\eta}{m} \mathbf{K}_{\mathcal{L},\mathcal{L}}\right)^t [\mathbf{Y}]_{\mathcal{L}}. \tag{9}$$

Equation (8) gives Equation (9) by induction. Thus the training term in Theorem 3.1 equals $m^{-1} \|(\mathbf{I}_m - \eta \mathbf{K}_{\mathcal{L},\mathcal{L}}/m)^t [\mathbf{Y}]_{\mathcal{L}}\|_{\mathrm{F}}^2$ at such an iterate. For a nonzero training Gram matrix, $0 < \eta \leq m/\|\mathbf{K}_{\mathcal{L},\mathcal{L}}\|_{\mathrm{OP}}$ ensures nonincreasing residual factors. Components in its null space are not fitted. The theorem still requires a fixed probe norm bound B, and makes no claim that all gradient descent iterates automatically satisfy any chosen bound.

B Further Explanation of Theorem 3.1

Theorem 3.1 concerns the RGAE latent matrix $\mathbf{Z}$, which is the input to conditional diffusion, and an auxiliary bounded linear probe. Its two substantive terms are the empirical label fitting error and a complexity tradeoff between rank and spectral tail. The spectral tail is normalized by N, in accordance with the spectrum of $\mathbf{K}/N$ in [83]. The STLC specialization has neither a $\log N$ multiplier nor an N/u prefactor, and it places no restriction on the relative split sizes beyond $m, u \geq 2$. Its confidence term uses $s = \min\{m, u\}$ and the classwise union term $\log C$. Reducing a spectral

tail at fixed rank lowers that candidate complexity term if the feature scale and probe norm bounds are fixed. It does not ensure a reduction in the empirical fitting error, the scale constants, or the total bound after retraining the representation. In particular, the LFP does not by itself imply that labels are exactly representable in the retained subspace or that their training residual vanishes.

Neither graph homophily nor spectral smoothness of labels with respect to the adjacency matrix is assumed. Strong heterophily may require additional latent directions: a smaller r can then increase T_r or the fitting error. The soft coefficient τ_{T} and cross validated starting rank balance this tradeoff, rather than enforcing one hard rank on every graph. The Chameleon and Actor experiments support the selected settings empirically. They do not guarantee that low rank regularization always improves heterophilic graphs.

The probability statement conditions on a fixed representation independent of the fresh probe split. If representation learning or an attack uses those probe labels to select $\mathbf{Z}$, this theorem cannot simply be applied conditionally to that selected matrix. In the actual RGDM pipeline, the downstream classifier is trained on decoded node attributes and edges, not on $\mathbf{Z}$. The low rank ablation (Table 3) and FND/FED results (Tables 13 to 14) support this second, empirical link. Establishing a joint generalization and robustness theory for data dependent representation learning, class conditional generation, and the downstream GNN is future work.

Algorithm 1 Training RGDM (Training the GAE and the LDM)

Input: The input attribute matrix $\mathbf{X}$, the adjacency matrix $\mathbf{A}$, the training epochs of the autoencoder t_{AE}, the labels Y_L of the labeled nodes $\mathcal{V}_L$, and the training epochs of the LDM t_{LDM}
Output: The parameters of the RGAE $\boldsymbol{\omega}$ and the parameters of the LDM $\boldsymbol{\theta}$

Compute the inter-cluster neighbor map $\mathbf{C}$ and the intra-cluster neighbor map $\mathbf{M}$
Initialize the edge selection mask $\mathbf{S}_{\text{inter}}^{(0)} = \mathbf{C}$ and $\mathbf{S}_{\text{intra}}^{(0)} = \mathbf{M}$
Initialize the parameter $\boldsymbol{\omega}$ of the RGAE
for $t \leftarrow 1$ to t_{AE} **do**
    Update $\boldsymbol{\omega}$ by $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \eta \nabla_{\boldsymbol{\omega}} \mathcal{L}_{\text{RGAE}}^{(t)}(\mathbf{S}_{\text{intra}}^{(t-1)}, \mathbf{S}_{\text{inter}}^{(t-1)})$ with $\mathcal{L}_{\text{RGAE}}^{(t)}(\mathbf{S}_{\text{intra}}^{(t-1)}, \mathbf{S}_{\text{inter}}^{(t-1)})$ from Eq.(1)
    Compute the edge selection mask $\mathbf{S}_{\text{intra}}^{(t)}$ and $\mathbf{S}_{\text{inter}}^{(t)}$
end for
Initialize the parameter $\boldsymbol{\theta}$ of the LDM.
Map the node features $\mathbf{X}$ and the adjacency matrix $\mathbf{A}$ to the latent space using the encoder g_e of the RGAE as $\mathbf{H} = g_e(\mathbf{X}, \mathbf{A})$.
for $t \leftarrow 1$ to t_{LDM} **do**
    Sample a diffusion step s and Gaussian noise $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    Get latent features of $\mathcal{V}_L$ as $\mathbf{H}_L = [\mathbf{H}]_{\mathcal{L}}$
    Form $\mathbf{H}_{L,s} = \sqrt{\bar{\alpha}_s}\mathbf{H}_L + \sqrt{1-\bar{\alpha}_s}\boldsymbol{\varepsilon}$ using the diffusion schedule
    Update $\boldsymbol{\theta}$ on $\left\|\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}(\mathbf{H}_{L,s}, s, Y_L) - \boldsymbol{\varepsilon}\right\|_{\mathrm{F}}^2$ with classifier free label dropout
end for
return The parameters of the RGAE $\boldsymbol{\omega}$ and the parameters of the LDM $\boldsymbol{\theta}$

Algorithm 2 Generation of the Augmented Graph $\mathcal{G}_{\text{aug}}$

Input: The input attribute matrix $\mathbf{X}$, the adjacency matrix $\mathbf{A}$, the number of added nodes N', and the labels of the synthetic data $\{\widehat{y}_i\}_{i=1}^{N'}$
Output: The augmented graph $\mathcal{G}_{\text{aug}} = (\mathcal{V} \cup \mathcal{V}_{\text{syn}}, \mathbf{X}_{\text{aug}}, \mathbf{A}_{\text{aug}})$.

for $i \leftarrow 1$ to N' **do**
    Sample a Gaussian noise $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    Set the class label of the i-th synthetic node to $\widehat{y}_i$
    Generate $\widehat{\mathbf{H}}_i$ from $\boldsymbol{\varepsilon}$ with the LDM for class $\widehat{y}_i$
end for
Decode $\widehat{\mathbf{H}} = \left\{\widehat{\mathbf{H}}_i\right\}_{i=1}^{N'}$ to $\mathbf{X}_{\text{syn}}$ and $\mathbf{A}_{\text{syn}}$ with the decoder of the RGAE as $\mathbf{X}_{\text{syn}}, \mathbf{A}_{\text{syn}} = g_d(\widehat{\mathbf{H}})$
Form $\mathbf{A}_{\text{aug}}$ as in Equation (3) and $\mathbf{X}_{\text{aug}} = [\mathbf{X}; \mathbf{X}_{\text{syn}}]$
return $\mathcal{G}_{\text{aug}} = (\mathcal{V} \cup \mathcal{V}_{\text{syn}}, \mathbf{X}_{\text{aug}}, \mathbf{A}_{\text{aug}})$

C Algorithm for Training the RGDM and Generating the Augmented Graph

Algorithm 1 trains RGDM in two stages: first the RGAE, then the class conditional LDM on the encoded node representations. At epoch t, the RGAE uses the previous masks between and within clusters in Equation (1) to update its parameters, including the mask predictor. The masks are then recomputed for epoch $t+1$, with initial masks $\mathbf{S}_{\text{inter}}^{(0)} = \mathbf{C}$ and $\mathbf{S}_{\text{intra}}^{(0)} = \mathbf{M}$. Algorithm 2 describes the generation process for the augmented graph $\mathcal{G}_{\text{aug}}$.

D Complexity Analysis

D.1 Efficient Computation of the TNN Regularizer

The computation of the TNN regularizer $\|\mathbf{K}\|_{r_0}$ in Eq. (1) does not require explicitly constructing the full kernel matrix $\mathbf{K} = \mathbf{Z}\mathbf{Z}^\top \in \mathbb{R}^{N\times N}$. Let $\widehat{\lambda}_i(\mathbf{K})$ denote the i-th largest eigenvalue of $\mathbf{K}$ and $\sigma_i(\mathbf{Z})$ denote the i-th largest singular value of the latent representation $\mathbf{Z} \in \mathbb{R}^{N\times D'}$ learned by the encoder of the RGAE. Since the nonzero eigenvalues of $\mathbf{K} = \mathbf{Z}\mathbf{Z}^\top$ are determined by the singular values of $\mathbf{Z}$, we have $\widehat{\lambda}_i(\mathbf{K}) = \sigma_i^2(\mathbf{Z})$. As a result, $\|\mathbf{K}\|_{r_0} = \sum_{i=r_0+1}^{N} \widehat{\lambda}_i(\mathbf{K}) = \sum_{i=r_0+1}^{\min\{N,D'\}} \sigma_i^2(\mathbf{Z})$. Therefore, in practice, the low-rank regularization term in RGDM can be computed from the singular values of $\mathbf{Z}$, which only requires $O(ND'^2)$ time when $N > D'$, avoiding the explicit $O(N^2)$ construction of $\mathbf{K}$.

D.2 Detailed Complexity Analysis of RGAE with SHED

In our work, we propose a RGAE with an edge selection mechanism incorporated in the GAT layers for aggregating neighborhood information. In this section, we analyze the inference time complexity and the parameter size of the RGAE with the edge selection mechanism. For comparison, we also analyze the inference time complexity and the parameter size of the GAE that adopts regular GAT layers for aggregating neighborhood information. For ease of comparison, we denote the number of the parameters and the inference cost of all the layers except the GAT layers with the edge selection as S_{MLP} and C_{MLP} respectively. Let E denote the number of edges in the graph. Let L denote the number of GAT layers. The inference time complexity of our RGAE is $\mathcal{O}(LDME + ME + C_{\text{MLP}})$, where $\mathcal{O}(LDME)$ is the computation cost of the GAT layers. $\mathcal{O}(ME)$ is the additional complexity for computing the decision masks on all the edges. In comparison, the inference time complexity of the GAE with regular GAT layers is $\mathcal{O}(DLME + C_{\text{MLP}})$. In our experiments, $D = 512$ and $L = 2$. Therefore, the inference time complexity of the RGAE, that is $\mathcal{O}(1025ME + C_{\text{MLP}})$, is comparable to the inference time complexity of GAE that adopts regular GAT layers whose inference time complexity is $\mathcal{O}(1024ME + C_{\text{MLP}})$.

For the encoder parameter count, use the same GAT layers in both encoders. Put $h_0 \coloneqq D$, and let a_ℓ and h_ℓ be the attention projection width and output width of GAT layer $\ell \in [L]$, respectively. The matrices in Section 3.2 then have dimensions $\mathbf{Q}_\ell \in \mathbb{R}^{h_{\ell-1}\times a_\ell}$, $\mathbf{V}_\ell \in \mathbb{R}^{2a_\ell\times 1}$, and $\mathbf{W}_\ell \in \mathbb{R}^{h_{\ell-1}\times h_\ell}$. For these separately parameterized matrices, let b_{GAT} be the number of GAT bias parameters. Let $b_{\text{mask}} \in \{0,1\}$ be the bias count of the scalar mask predictor, and let S_{MLP} count all remaining parameters outside the GAT layers and the mask predictor. The parameter counts are

$$
\begin{aligned}
S_{\text{GAT}} &\coloneqq \sum_{\ell=1}^{L} (h_{\ell-1}a_\ell + 2a_\ell + h_{\ell-1}h_\ell) + b_{\text{GAT}},\\
S_{\text{RGAE}} &\coloneqq S_{\text{GAT}} + 2D + b_{\text{mask}} + S_{\text{MLP}},\\
S_{\text{GAE}} &\coloneqq S_{\text{GAT}} + S_{\text{MLP}}.
\end{aligned}
$$

The mask predictor takes $\mathbf{X}'_i \| \mathbf{X}'_j \in \mathbb{R}^{2D}$ and has a scalar output, so edge selection adds exactly $2D + b_{\text{mask}}$ parameters when all other layers are fixed.

To thoroughly study the complexity of the RGAE, we also analyze its training time complexity. Let t_{train} be the number of training epochs. The sparse encoder processes all N nodes and E edges in a full graph pass. Its cost must not be multiplied by N a second time. Under the preceding

encoder cost convention, t_{train} passes cost $\mathcal{O}(t_{\text{train}}(DLME + ME + C_{\text{MLP}}))$, before loss evaluation. The latent TNN adds $\mathcal{O}(t_{\text{train}}ND'^2)$ as described in Section D.1. SHED's cost depends on the active cluster count described below. Explicitly materializing every $N \times K \times M$ neighbor map or evaluating a dense pairwise loss still incurs quadratic work or storage. Sparse decoder inference complexity alone does not establish a subquadratic bound for every implementation of RGAE training.

Complexity Analysis of SHED. We propose SHED to reconstruct the edges connected to a node in the graph. To show its efficiency, we analyze the inference time complexity and the parameter size of the GAE with SHED. For comparison, we also analyze the inference time complexity and the parameter size of GAE, where SHED is replaced by a regular edge decoder that directly reconstructs the adjacency matrix $\mathbf{A}$. For ease of comparison, we denote the number of parameters and inference cost of all the MLP and GAT layers except SHED as S_{MLP} and C_{MLP}, respectively. For a node v_i in the graph, let $d_i := |\mathcal{C}(i)| = \sum_{k=1}^{K} \mathbb{1}_{\{\widehat{\mathbf{C}}_{ik} > 1/2\}}$ be the number of active clusters in the set $\mathcal{C}(i)$ defined in Section 3.2. Let D' be the dimension of the input feature for SHED. The inference time complexity of GAE with SHED is $\mathcal{O}(KD' + d_i D'M + C_{\text{MLP}})$, where $\mathcal{O}(KD')$ is the additional complexity for computing the inter-cluster neighbor map and encoding the cluster indices. $\mathcal{O}(d_i D'M)$ is the computation cost for computing the intra-cluster neighbor map. In contrast, the inference time complexity of GAE with a regular edge decoder is $\mathcal{O}(D'KM + C_{\text{MLP}})$. For an observed neighborhood, its number of occupied clusters is at most its degree. This bound need not hold for a predicted cluster map. Let $\bar{d}_{\text{cl}} := N^{-1} \sum_i d_i$ be the average number of active predicted clusters. The operation count below is conditional on this sparsity. Using the reported Pubmed illustration $\bar{d}_{\text{cl}} = 2.25$ gives an average decoder cost. It is not a pointwise bound on d_i or a consequence of the undirected degree statistics. As a result, $D'(K + d_i M) \ll D'KM$. For example, when we set $K = 100$ and $M = 198$ on Pubmed, the illustrative average cost is $\mathcal{O}(545.5D' + C_{\text{MLP}})$, which is much more efficient than the regular edge decoder whose inference time complexity is $\mathcal{O}(19800D' + C_{\text{MLP}})$. In general, the inference time complexity of GAE with SHED is much lower than that of GAE with a regular edge decoder.

In addition, the parameter size of GAE with SHED is $D'^2 + D'K + D'M + S_{\text{MLP}}$, where D'^2 is the number of parameters in the layer for encoding cluster indices. $D'K$ is the number of parameters in the layer for predicting the inter-cluster neighbor map. $D'M$ is the number of parameters in the layer for reconstructing the intra-cluster neighbor map. In contrast, the parameter size of GAE with a regular edge decoder is $D'KM + S_{\text{MLP}}$. When $D' + K + M \ll KM$, the parameter size of GAE with SHED is much smaller than that of GAE with a regular edge decoder. For example, $N = 19717$ on Pubmed, and with $M = 198$ and $K = 100$, we have $KM = 19800 \approx N$.

E Additional Experimental Settings

E.1 Datasets

Following previous works on adversarial attacks and defense of GNNs [57, 51, 56], we evaluate the RGDM on four public benchmark datasets for node classification, including three citation graphs, which are Cora, Citeseer, and Pubmed, and one blog graph, that is, Polblogs. Following previous works on graph adversarial attacks, we evaluate our method and baselines on the largest connected component (LCC) of the graphs. We show the statistics of the datasets in Table 9.

Table 9: Statistics of the datasets.

Datasets	# Node	# Edge	Classes	Features
Cora	2,485	5,069	7	1,433
Citeseer	2,110	3,668	6	3,703
Polblogs	1,222	16,714	2	1,222
Pubmed	19,717	44,338	3	500
ogbn-arxiv	169,343	1,166,243	40	128
Reddit	232,965	11,606,919	41	602

E.2 Training Settings

The training of the RGAE is divided into two phases. In the first phase, we pre-train the RGAE by only optimizing the consistency loss $\mathcal{L}_{\mathrm{C}}^{(t)}$ in Equation (1) for 2000 epochs. In the second phase, we optimize $\mathcal{L}^{(t)}$ for another 1000 epochs. We use the Adam optimizer with a learning rate of 0.001 for the training. The weight decay is set to 1×10^{-5}. We train the LDM in the RGDM after finishing the training of the RGAE. We use the Adam optimizer with a learning rate of 0.0002 to train the LDM for 3000 epochs.

To mitigate the effects of potentially unstable masks $\mathbf{S}_{\text{inter}}$ and $\mathbf{S}_{\text{intra}}$ during early training, we adopt a warm-up strategy widely used in graph structure learning. During the first 100 epochs, the RGAE is trained only with the reconstruction loss and the low-rank regularization term while keeping $\mathbf{S}_{\text{inter}}$ and $\mathbf{S}_{\text{intra}}$ fixed to the original hierarchical edge maps.

E.3 Attack Settings

Non-targeted Adversarial Attacks (Metattack) [51]. We first evaluate the robustness of our method against the non-targeted adversarial attack method Metattack. Metattack treats the graph as a hyperparameter to optimize and uses the meta-gradients to solve the bi-level optimization problem, which minimizes the node classification accuracy. We follow the implementation in [51]. As Metattack has several variants, we follow [57] and adopt the most destructive attack version, Meta-Self, on Cora, Citeseer, and Polblogs datasets. On Pubmed, we adopt the approximate version of Meta-Self, A-Meta-Self, following the settings in [57]. We measure the strength of the attack by the perturbation rate, which is the ratio of perturbed edges among all the edges in the graph. We evaluate the RGDM and all the baselines with the perturbation rates ranging from 0 to 25% with a step of 5%.

Targeted Adversarial Attack (Nettack) [50]. We adopt Nettack as the targeted attack method in evaluating the robustness of our method. Nettack manipulates the edges and node features to degrade the classification accuracy on the target nodes while minimizing the change in the graph's degree distribution and the feature co-occurrences. We use the default attack settings in the original implementation in [50]. The nodes in the test set whose degree is larger than 10 are set as the target nodes. The number of perturbations made on every targeted node is defined as the attack budget. Following [57], we evaluate the RGDM and all the baselines with attack budgets ranging from 1 to 5 with a step size of 1. Following the settings in [57], we only sample 10% of the target nodes for the evaluation on the Nettack.

Gradient-based Attack Method (Topology Attack) [85]. We evaluate the robustness of the RGDM against a gradient-based attack method named Topology Attack [85]. The topology Attack proposes a projected gradient descent (PGD) adversarial attack method that perturbs the edges in a graph to degrade the overall node classification accuracy on the perturbed graph. We adopt the same settings in [85] to attack the graphs in our experiments. Similar to the settings for Metattack, we measure the strength of the attack by the perturbation rate, which is the ratio of the perturbed edges among all the edges in the graph. We evaluate the RGDM and all the baselines with perturbation rates ranging from 0 to 25% with a step of 5%.

E.4 Tuning the Starting Rank r_0 by Cross-Validation

We tune the rank r_0 for the TNN, $\|\mathbf{K}\|_{r_0}$, in Equation (1) by standard cross-validation on each dataset. Let $r_0 = \lceil \gamma \min\{N, D\prime\} \rceil$ where γ is the rank ratio. We select the values of γ by performing 5-fold cross-validation on 20% of the training data in each dataset. The value of γ is selected from $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$.

E.5 Details on Learning the Low-Rank Weight Matrix and Low-Rank Adjacency Matrix

To evaluate the robustness of the proposed low-rank regularization in RGDM, we construct two comparison baselines using weight orthogonality regularization and low rank adjacency reconstruction in the RGAE. Next, we detail the implementation of each ablation model. Following the setting in [76], we regularize the weight matrices of the encoder in the RGAE to approximate orthonormality by adding a penalty term to the training loss of robust RAE, that is $\mathcal{L}_{\text{orth}} = \sum_l \left\| \mathbf{W}_l^\top \mathbf{W}_l - \mathbf{I} \right\|_{\mathrm{F}}^2$,

where $\mathbf{W}_l$ denotes the weight matrix of the l-th layer and $\mathbf{I}$ is the identity matrix. This penalty encourages approximately orthonormal weight columns. By itself, it controls conditioning rather than imposing a low rank. We therefore identify this comparator as weight orthogonality regularization and distinguish it from the spectral tail penalty on the diffusion latent. The strength of this regularization is controlled by the scale factor for $\mathcal{L}_{\text{orth}}$, which is tuned via cross-validation following the setting in [76]. For the ablation model that learns the low-rank adjacency matrix, we replace the original attacked adjacency matrix $\mathbf{A}$ with its low-rank approximation. Following [56], we compute the top-r_A eigenvectors $\mathbf{U}_{r_A}$ of the symmetrically normalized adjacency matrix $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, and reconstruct the low-rank adjacency matrix as $\mathbf{A}_{\text{LR}} = \mathbf{D}^{1/2}\mathbf{U}_{r_A}\mathbf{\Lambda}_{r_A}\mathbf{U}_{r_A}^{\top}\mathbf{D}^{1/2}$, where $\mathbf{\Lambda}_{r_A}$ is the diagonal matrix containing the top-r_A eigenvalues. The rank r_A is selected via cross-validation following [56]. For all datasets, we select r_A from the same candidate sets as described in Section E.4.

E.6 Compared Methods

In our empirical evaluation, we comprehensively benchmark the proposed RGDM against various GNNs, including models that exemplify advances in expressive power, structural modeling, and resilience to adversarial or noisy perturbations. The standard GNN baselines include Graph Convolutional Networks (GCN) [47], Graph Attention Networks (GAT) [66], and Relational GCNs (RGCN) [92], which serve as foundational architectures widely adopted for semi-supervised node classification. Additionally, we incorporate GCN-SVD [56], which leverages truncated singular value decomposition to mitigate the impact of noisy graph spectra. To rigorously assess the robustness of RGDM under adversarial settings, we further compare it with the state-of-the-art robust GNNs explicitly designed to enhance stability and generalization in the presence of graph corruption. These include GraphBel [61], which integrates belief propagation with GNNs for noise-tolerant inference; UAG [93], which unifies adversarial training with graph structure learning; GCORNs [65], a framework that co-regularizes neighborhood consistency and robustness objectives; GADC [64], which employs adaptive denoising through graph diffusion mechanisms; and HANG [62], a hierarchical adversarial defense method. We also include Pro-GNN [57], which jointly optimizes graph structure and model parameters via low-rank and sparsity constraints, and STABLE [94], which stabilizes message passing through spectral smoothing.

F Additional Experimental Results

F.1 Eigen-projection and Concentration Ratio of the Ground Truth Label and the Adversarial Noise

Projection diagnostics. We study how the information of the ground-truth class label and the adversarial noise is distributed on different eigenvectors of the feature gram matrix $\mathbf{K} = \mathbf{Z}\mathbf{Z}^{\top}$ by performing eigen-projection, where $\mathbf{Z}$ is the latent feature learned on the graph with the adversarial noise. We first compute the eigenvectors $\mathbf{U}$ of the feature gram matrix $\mathbf{K}$. Let $\mathbf{U}^{(1:r)} \in \mathbb{R}^{N\times r}$ be the top r-eigenvectors of $\mathbf{K}$ and $\mathbf{U}^{(r)}$ be the r-th eigenvector of $\mathbf{K}$. Then, the eigen-projection value of the ground-truth label $\mathbf{Y}$ on $\mathbf{U}^{(r)}$ is computed by $p_r = \frac{1}{C}\sum_{c=1}^{C} \left\|\mathbf{U}^{(r)\top}\mathbf{Y}^{(c)}\right\|_2^2 / \left\|\mathbf{Y}^{(c)}\right\|_2^2$ for $r \in [N]$, where $\mathbf{Y}^{(c)}$ is the c-th column of $\mathbf{Y}$. We let $\mathbf{p} = [p_1, \ldots, p_N] \in \mathbb{R}^N$. The eigen-projection p_r reflects the amount of the signal projected onto the r-th eigenvector of $\mathbf{K}$, and the signal concentration ratio of a rank r reflects the proportion of signal projected onto the top r eigenvectors of $\mathbf{K}$. The signal concentration ratio for rank r is computed by $\|\mathbf{p}(1:r)\|_1$, where $\mathbf{p}(1:r)$ contains the first r elements of $\mathbf{p}$. It is observed from the red curves in the left of Figure 3 and Figure 4 in Section F.1 of the appendix that the projection of the ground truth labels mostly concentrates on the top eigenvectors of $\mathbf{K}$, known as the Low Frequency Property (LFP) widely studied in deep learning [22, 23, 24, 25, 26, 19]. On the other hand, we study how the information of the adversarial noise is distributed on the eigenvectors of the feature gram matrix $\mathbf{K}$. Let $\mathbf{Z}^{(\text{clean})}$ be the latent features obtained on the clean input graph without adversarial noise. Then $\Delta\mathbf{Z} = \mathbf{Z} - \mathbf{Z}^{(\text{clean})}$ is the adversarial noise in the feature due to the attacks. The low-rank part of $\mathbf{Z}$ characterized by its top-r eigenvectors is $\mathbf{Z}^{(\text{low-rank})} = \mathbf{U}^{(1:r)}\mathbf{U}^{(1:r)\top}\mathbf{Z} = \mathbf{U}^{(1:r)}\mathbf{U}^{(1:r)\top}(\mathbf{Z}^{(\text{clean})} + \Delta\mathbf{Z})$. Then $\mathbf{U}^{(1:r)}\mathbf{U}^{(1:r)\top}\mathbf{Z}^{(\text{clean})}$ is the low-rank part of the clean latent feature $\mathbf{Z}^{(\text{clean})}$, and $\mathbf{U}^{(1:r)}\mathbf{U}^{(1:r)\top}\Delta\mathbf{Z}$ is the adversarial noise induced by the adversarial attack in $\mathbf{Z}^{(\text{low-rank})}$. We compute eigen-projection value of the adversarial

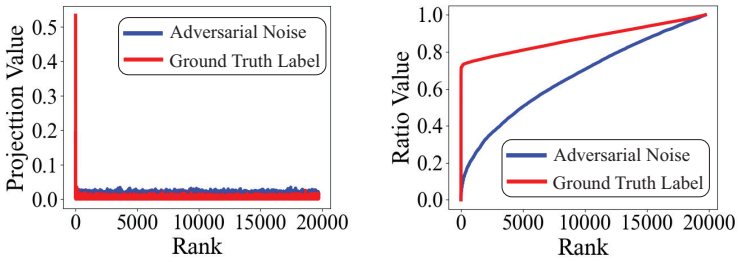
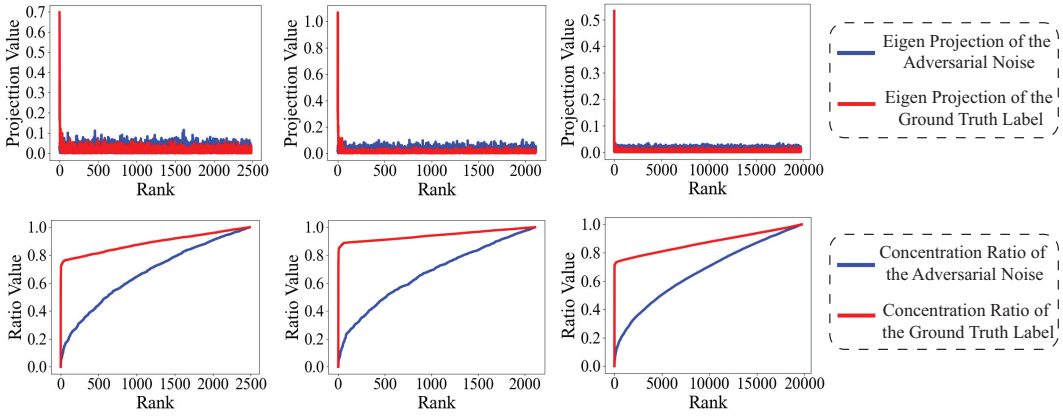


Figure 3: Eigen-projection (left) and concentration ratio (right) of the ground truth label and the adversarial noise on Pubmed. The study is performed for Metattack with a perturbation ratio of 25%. By the rank $r = r_0 = 0.2 \min\{N, D'\}$, the signal concentration ratio of the ground truth label is 0.784. Results on Cora and Citeseer are illustrated in Figure 4 in Section F.1.

noise by $p_r^{(\Delta \mathbf{Z})} = \left\| \mathbf{U}^{(r)\top} \Delta \mathbf{Z} \right\|_{\mathrm{F}} / \|\Delta \mathbf{Z}\|_{\mathrm{F}}$ for $r \in [N]$, which reflects the amount of the adversarial noise projected onto the r-th eigenvector of $\mathbf{K}$. To obtain a ratio in $[0, 1]$, we normalize the cumulative projection amplitudes as $\sum_{j=1}^{r} p_j^{(\Delta \mathbf{Z})} / \sum_{j=1}^{N} p_j^{(\Delta \mathbf{Z})}$ when $\Delta \mathbf{Z} \neq \mathbf{0}$. This is a normalized projection amplitude diagnostic, distinct from the squared energy fraction $\left\| \mathbf{U}^{(1:r)\top} \Delta \mathbf{Z} \right\|_{\mathrm{F}}^2 / \|\Delta \mathbf{Z}\|_{\mathrm{F}}^2$. If $\Delta \mathbf{Z} = \mathbf{0}$, there is no noise concentration ratio to estimate.

The full spectral diagnostics on Cora, Citeseer, and Pubmed are shown in Figure 4. The concentration of label information in leading eigendirections motivates a soft low rank bias in these experiments. It is neither a universal property of attacked graphs nor an assumption of Theorem 3.1.



(a) Cora (b) Citeseer (c) Pubmed

Figure 4: Eigen-projection (first row) and concentration ratio (second row) of the ground truth label and the adversarial noise on Cora, Citeseer, and Pubmed. The study in this figure is performed for Metattack with a perturbation ratio of 25%. By the rank $r = r_0 = 0.2 \min\{N, D'\}$, the signal concentration ratios of the ground truth label for Cora, Citeseer, and Pubmed are 0.844, 0.809, and 0.784 respectively.

F.2 Detailed Semi-Supervised Node Classification Results

The detailed experiment results for the RGDM under the Metattack, Nettack, and Topology Attack are shown in Table 10, Table 11 and Table 12.

F.3 Quality Evaluation of the Augmented Graph

This paper introduces a novel graph diffusion model named RGDM, which synthesizes the graph structures. The synthetic graph structures generated by the RGDM are subsequently combined with the original attacked graph to form an augmented graph. By incorporating an edge selection mecha-

Table 10: Node classification performance (Accuracy±Std) under the non-targeted poisoning attack (Metattack) [51]. The improvements of the RGDM over the corresponding baselines are attached in parentheses after the results of the RGDM.

Dataset	Ptb Rate (%)	GCN	GAT	RGCN	GCN-SVD	GraphBel	UAG	GCORNs	GADC	HANG	Pro-GNN	RGDM (GCN)	RGDM (HANG)	RGDM (Pro-GNN)
Cora	0	83.50±0.44	83.97±0.65	83.09±0.44	80.63±0.45	83.42±0.52	82.05±0.51	82.56±0.48	83.22±0.61	80.07±0.32	82.98±0.23	85.23±0.44 (↑ 1.73)	82.31±0.51 (↑ 2.24)	84.29±0.52 (↑ 1.31)
	5	76.55±0.79	80.44±0.74	77.42±0.39	78.39±0.54	82.78±0.39	79.13±0.59	75.91±1.88	75.68±1.10	78.12±0.72	82.27±0.45	80.19±0.39 (↑ 3.64)	81.04±0.59 (↑ 2.92)	83.19±0.39 (↑ 0.92)
	10	70.39±1.28	75.61±0.59	72.22±0.38	71.47±0.83	77.91±0.86	75.16±0.76	74.63±1.02	73.70±1.17	75.96±0.62	79.03±0.59	76.73±0.38 (↑ 6.34)	78.03±0.76 (↑ 2.07)	81.16±0.86 (↑ 2.13)
	15	65.10±0.71	69.78±1.28	66.82±0.39	66.69±1.18	76.01±1.12	71.03±0.64	72.57±1.24	71.39±1.22	72.89±0.96	76.40±1.27	69.37±0.39 (↑ 4.27)	76.39±0.64 (↑ 3.50)	78.29±1.12 (↑ 1.89)
	20	59.56±1.72	59.94±0.92	59.27±0.37	58.94±1.13	68.78±1.84	65.71±0.89	71.94±1.51	71.20±1.49	70.36±1.09	73.32±1.56	65.97±0.37 (↑ 6.41)	73.25±0.89 (↑ 2.89)	74.85±1.84 (↑ 1.53)
	25	47.53±1.91	54.78±0.74	50.51±0.78	52.06±1.19	56.54±1.58	60.82±1.08	69.48±1.94	68.29±1.36	68.72±1.87	69.72±1.69	61.42±0.78 (↑ 13.89)	70.44±1.08 (↑ 1.72)	72.31±1.58 (↑ 2.59)
Citeseer	0	71.96±0.55	73.26±0.83	71.20±0.83	70.65±0.32	72.26±0.83	72.10±0.63	72.63±0.40	73.05±0.30	73.26±0.38	73.28±0.69	73.66±0.83 (↑ 1.70)	74.21±0.63 (↑ 0.95)	74.47±0.83 (↑ 1.19)
	5	70.88±0.62	72.89±0.83	70.50±0.43	68.84±0.72	71.31±0.43	70.51±0.97	68.18±1.68	68.67±0.89	73.09±0.34	72.93±0.57	73.14±0.43 (↑ 2.26)	74.04±0.97 (↑ 0.95)	74.29±0.43 (↑ 1.36)
	10	67.55±0.89	70.63±0.48	67.71±0.30	68.87±0.62	71.63±0.30	69.54±0.56	66.24±1.52	68.66±1.48	72.43±0.52	72.51±0.75	71.35±0.30 (↑ 3.80)	73.85±0.56 (↑ 1.42)	74.03±0.30 (↑ 1.52)
	15	64.52±1.11	69.02±1.09	65.69±0.37	63.26±0.96	69.82±0.37	65.95±0.94	65.48±1.03	68.29±1.88	70.82±0.87	72.03±1.11	68.72±0.37 (↑ 4.20)	73.21±0.94 (↑ 2.39)	73.65±0.37 (↑ 1.62)
	20	62.03±1.49	61.04±1.52	62.49±1.22	58.55±1.09	64.22±1.22	59.30±1.40	65.55±1.55	67.87±1.57	66.19±1.38	70.02±1.28	66.17±1.22 (↑ 4.14)	69.57±1.40 (↑ 3.38)	71.87±1.22 (↑ 1.85)
	25	56.94±1.09	61.85±1.12	55.35±0.66	57.18±1.87	66.23±0.66	59.89±1.47	65.23±1.26	64.82±1.31	66.40±1.57	68.95±1.78	62.84±0.66 (↑ 5.90)	69.05±1.47 (↑ 2.65)	70.24±0.66 (↑ 1.29)
Polblogs	0	95.69±0.38	95.35±0.20	95.22±0.14	95.31±0.18	85.13±1.22	90.13±1.22	95.31±0.88	95.54±0.17	94.77±1.07	93.20±0.64	96.13±0.14 (↑ 0.44)	94.95±1.22 (↑ 0.18)	95.70±1.22 (↑ 2.50)
	5	73.07±0.80	83.69±1.45	74.34±0.19	89.09±0.22	51.84±1.38	84.84±1.38	89.32±0.33	90.52±0.27	80.19±1.52	93.29±0.18	78.57±0.19 (↑ 5.50)	88.25±1.38 (↑ 8.06)	94.33±1.38 (↑ 1.04)
	10	70.72±1.13	76.32±0.85	71.04±0.34	81.24±0.49	56.54±1.30	75.54±1.30	87.34±1.77	86.30±1.45	74.92±1.32	89.42±1.09	74.33±0.34 (↑ 3.61)	80.47±1.30 (↑ 5.55)	91.35±1.30 (↑ 1.93)
	15	64.96±1.91	68.80±1.14	67.28±0.38	68.10±1.73	53.41±1.08	66.41±1.08	82.07±1.41	83.45±1.75	71.65±1.34	86.04±1.21	69.85±0.38 (↑ 4.89)	78.49±1.08 (↑ 6.84)	88.01±1.08 (↑ 1.97)
	20	51.27±1.23	51.50±1.63	59.89±0.34	57.33±1.15	52.18±0.54	58.18±0.54	69.92±1.26	72.84±1.24	66.27±1.39	79.56±1.68	59.38±0.34 (↑ 8.11)	72.51±0.54 (↑ 6.24)	81.34±0.54 (↑ 1.78)
	25	49.23±1.36	51.19±1.49	56.02±0.56	48.66±1.93	51.39±1.36	55.39±1.36	66.59±1.74	65.52±1.93	65.80±1.33	63.18±1.40	56.47±0.56 (↑ 7.24)	70.79±1.36 (↑ 4.99)	66.92±1.36 (↑ 3.74)
Pubmed	0	87.19±0.09	83.73±0.40	86.16±0.18	83.44±0.21	84.02±0.26	87.06±0.06	86.42±0.71	87.26±0.25	85.08±0.20	87.33±0.18	88.10±0.18 (↑ 0.91)	87.15±0.11 (↑ 2.07)	88.41±0.26 (↑ 1.08)
	5	83.09±0.13	78.00±0.44	81.08±0.20	83.41±0.15	83.91±0.26	86.39±0.06	82.96±0.89	81.64±0.64	85.08±0.18	87.25±0.09	86.99±0.20 (↑ 3.90)	87.02±0.11 (↑ 1.94)	88.26±0.26 (↑ 1.01)
	10	81.21±0.09	74.93±0.38	77.51±0.27	83.27±0.21	84.62±0.26	85.70±0.07	80.76±1.65	80.43±1.64	85.17±0.23	87.25±0.09	85.83±0.27 (↑ 4.62)	87.13±0.14 (↑ 1.96)	88.15±0.26 (↑ 0.90)
	15	78.66±0.12	71.13±0.51	73.91±0.25	83.10±0.18	84.83±0.20	84.76±0.08	76.82±1.83	75.72±1.24	85.00±0.22	87.20±0.09	84.97±0.25 (↑ 6.31)	86.75±0.13 (↑ 1.75)	88.09±0.20 (↑ 0.89)
	20	77.35±0.19	68.21±0.96	71.18±0.31	83.01±0.22	84.89±0.45	83.88±0.05	71.28±1.52	72.80±1.89	85.20±0.19	87.15±0.15	82.45±0.31 (↑ 5.10)	86.13±0.11 (↑ 0.93)	88.00±0.45 (↑ 0.85)
	25	75.50±0.17	65.41±0.77	67.95±0.15	82.72±0.18	85.07±0.15	83.66±0.06	70.75±1.86	71.60±1.50	85.06±0.17	86.76±0.19	81.52±0.15 (↑ 6.02)	86.26±0.12 (↑ 1.20)	87.58±0.15 (↑ 0.82)

Table 11: Node classification performance (Accuracy±Std) under targeted attack (Nettack) [50]. The improvements of RGDM over the corresponding baselines are attached in parentheses after the results of the RGDM.

Dataset	Attack Budget	GCN	GAT	RGCN	GCN-SVD	GraphBel	UAG	GCORNs	GADC	HANG	Pro-GNN	RGDM (GCN)	RGDM (HANG)	RGDM (Pro-GNN)
Cora	0	81.43±1.04	82.14±1.13	81.34±1.55	81.10±1.34	71.55±1.01	82.34±1.08	82.61±0.41	81.36±1.83	80.75±1.27	84.78±0.59	83.43±1.08 (↑ 2.00)	82.10±1.34 (↑ 1.35)	86.00±1.13 (↑ 1.22)
	1	75.06±1.02	76.04±1.08	76.75±1.71	77.23±1.82	63.73±1.25	81.75±1.16	81.05±1.14	80.05±1.53	76.99±1.16	83.85±0.51	78.41±1.16 (↑ 3.35)	80.39±1.82 (↑ 3.40)	84.79±1.08 (↑ 0.94)
	2	70.60±1.10	70.24±1.43	70.96±1.14	72.53±1.60	62.41±1.94	77.96±1.84	78.47±1.32	75.90±1.29	76.51±1.60	78.65±0.66	73.79±1.84 (↑ 3.19)	79.14±1.60 (↑ 2.63)	80.02±1.43 (↑ 1.37)
	3	67.95±1.72	65.54±1.34	66.51±1.60	66.75±1.54	61.20±1.08	72.51±1.14	73.17±0.74	71.94±1.45	73.13±1.85	72.40±0.53	70.46±1.14 (↑ 2.51)	76.31±1.54 (↑ 3.18)	74.31±1.34 (↑ 1.91)
	4	61.57±1.47	61.69±0.90	59.28±1.68	60.72±1.63	56.51±1.72	70.28±1.52	71.26±0.62	70.93±1.19	72.53±1.14	70.12±0.66	64.37±1.52 (↑ 2.80)	75.12±1.63 (↑ 2.59)	72.79±0.90 (↑ 2.67)
	5	55.54±1.66	58.31±1.03	55.30±1.66	57.71±1.82	51.93±1.77	66.30±1.21	66.89±1.08	64.49±1.42	68.80±1.55	66.89±0.73	59.44±1.21 (↑ 3.90)	69.81±1.82 (↑ 1.01)	69.81±1.03 (↑ 2.92)
Citeseer	0	81.02±1.35	82.26±1.04	80.25±1.12	81.08±1.34	76.55±1.23	81.25±1.12	81.94±1.17	80.63±1.04	81.09±1.12	82.12±0.81	82.87±1.12 (↑ 1.85)	82.84±1.34 (↑ 1.75)	83.05±1.04 (↑ 0.93)
	1	78.41±1.62	81.27±1.38	78.25±0.73	80.16±1.04	68.89±1.67	80.25±0.73	80.79±0.89	79.02±1.84	79.05±1.38	81.75±0.79	80.39±0.73 (↑ 1.98)	81.40±1.04 (↑ 2.35)	82.47±1.38 (↑ 0.72)
	2	74.92±1.54	77.43±1.89	75.40±1.04	79.84±0.73	67.62±1.11	79.40±1.04	79.65±1.84	78.22±1.29	77.94±1.29	81.27±0.95	77.15±1.04 (↑ 2.23)	80.11±0.73 (↑ 2.17)	82.30±1.89 (↑ 1.03)
	3	63.97±1.69	60.85±1.99	60.31±1.19	77.14±1.86	60.63±1.87	78.31±1.19	78.10±1.06	78.11±1.39	77.14±1.48	79.68±1.98	68.53±1.19 (↑ 4.56)	78.95±1.86 (↑ 1.81)	80.85±1.99 (↑ 1.17)
	4	55.40±1.60	61.59±1.64	55.49±1.75	69.52±1.31	53.17±1.48	77.49±1.75	77.75±1.66	77.49±1.54	78.41±1.62	77.78±1.84	59.57±1.75 (↑ 4.17)	78.99±1.31 (↑ 0.58)	79.11±1.64 (↑ 1.33)
	5	47.62±1.17	55.56±1.28	47.44±1.01	69.21±1.48	48.73±1.60	71.44±1.01	71.66±1.79	70.54±1.57	73.49±1.48	71.27±1.99	55.71±1.01 (↑ 8.09)	75.92±1.48 (↑ 2.43)	74.60±1.28 (↑ 3.33)
Polblogs	0	97.03±0.24	97.25±0.32	97.04±0.11	97.50±0.22	71.35±1.01	97.11±0.11	97.52±0.26	97.38±0.34	97.41±0.50	97.12±0.15	97.69±0.11 (↑ 0.66)	97.65±0.22 (↑ 0.24)	97.80±0.32 (↑ 0.68)
	1	96.83±0.17	97.22±0.25	97.00±0.07	97.56±0.20	68.17±1.25	96.00±0.07	97.61±0.59	97.18±0.35	97.37±0.37	96.83±0.06	97.67±0.07 (↑ 0.84)	97.63±0.20 (↑ 0.26)	97.69±0.25 (↑ 0.86)
	2	95.61±0.20	96.11±0.65	95.87±0.23	97.12±0.09	65.48±1.85	97.02±0.23	96.08±0.45	95.37±0.59	96.89±0.16	97.17±0.12	97.10±0.23 (↑ 1.49)	97.24±0.09 (↑ 0.35)	97.58±0.65 (↑ 0.41)
	3	95.41±0.18	95.81±0.56	95.59±0.27	96.61±0.14	62.59±1.99	96.59±0.27	95.65±1.01	95.47±0.13	96.65±0.15	96.93±0.12	96.99±0.27 (↑ 1.58)	97.01±0.14 (↑ 0.36)	97.22±0.56 (↑ 0.29)
	4	94.24±0.24	94.80±0.66	94.37±0.26	96.17±0.19	58.68±0.40	96.07±0.26	94.44±0.53	94.97±0.61	96.26±0.53	96.89±0.16	95.44±0.26 (↑ 1.20)	97.03±0.19 (↑ 0.77)	97.12±0.66 (↑ 0.23)
	5	93.00±0.48	93.28±1.43	93.20±0.43	95.13±0.25	59.02±1.19	95.60±0.43	93.42±0.85	93.27±0.46	95.91±0.33	96.13±0.25	94.90±0.43 (↑ 1.90)	96.70±0.25 (↑ 0.79)	96.88±1.43 (↑ 0.75)
Pubmed	0	88.13±1.35	87.03±1.13	84.87±1.35	87.27±1.74	85.47±1.45	87.31±1.35	84.56±1.39	85.49±1.16	85.35±1.20	88.46±1.20	89.21±1.35 (↑ 1.08)	87.28±1.74 (↑ 1.93)	89.58±1.13 (↑ 1.12)
	1	87.04±1.62	84.95±1.08	83.75±1.71	84.15±1.25	84.34±1.33	86.57±1.71	83.88±1.94	84.58±1.28	84.42±1.01	87.58±1.01	88.03±1.71 (↑ 0.99)	86.88±1.25 (↑ 2.46)	88.80±1.08 (↑ 1.22)
	2	84.12±1.54	83.90±1.43	84.21±1.14	83.90±1.94	84.27±1.25	85.21±1.14	84.40±1.73	84.17±1.53	84.38±1.05	85.87±1.05	86.35±1.14 (↑ 1.21)	86.35±1.94 (↑ 1.97)	87.14±1.43 (↑ 1.27)
	3	81.25±1.69	81.27±1.34	82.48±1.60	81.22±1.08	84.11±1.67	83.88±1.60	82.26±1.49	84.35±1.33	84.01±1.12	84.35±1.12	83.37±1.60 (↑ 2.12)	85.21±1.08 (↑ 1.20)	85.79±1.34 (↑ 1.44)
	4	76.35±1.60	78.54±0.90	80.02±1.68	77.02±1.72	79.03±1.22	80.00±1.32	81.06±1.72	78.01±1.20	79.73±1.31	80.04±1.31	78.45±1.32 (↑ 2.10)	81.65±1.23 (↑ 1.92)	81.95±0.90 (↑ 1.91)
	5	68.32±1.17	74.25±1.03	73.14±1.66	71.84±1.77	70.11±1.36	72.20±1.39	72.28±1.13	71.26±1.61	70.59±1.31	72.18±1.31	71.42±1.39 (↑ 3.10)	74.43±1.34 (↑ 3.84)	76.27±1.03 (↑ 4.09)

Table 12: Node classification performance (Accuracy±Std) under Topology Attack [85]. The improvements of the RGDM over the corresponding baselines are attached in parentheses after the results of the RGDM.

Dataset	Ptb Rate (%)	GCN	GAT	RGCN	GCN-SVD	GraphBel	UAG	GCORNs	GADC	HANG	Pro-GNN	RGDM(GCN)	RGDM(HANG)	RGDM(Pro-GNN)
Cora	0	83.50±0.44	83.97±0.65	83.09±0.44	80.63±0.45	83.42±0.52	82.05±0.51	82.63±0.47	83.21±0.72	80.07±0.32	82.98±0.23	85.23±0.44 (↑ 1.73)	82.31±0.51 (↑ 2.24)	84.29±0.52 (↑ 1.31)
	5	75.50±0.44	76.97±0.65	75.04±1.32	75.82±1.24	77.01±1.33	76.22±1.24	75.96±1.81	76.83±1.25	76.99±1.52	77.63±1.85	77.92±1.08 (↑ 2.42)	79.39±1.33 (↑ 2.40)	81.10±1.16 (↑ 3.47)
	10	72.02±1.32	74.10±1.84	73.05±1.25	74.00±1.76	74.00±1.57	76.00±1.76	74.67±1.03	75.90±1.84	76.99±1.52	77.63±1.85	74.22±1.43 (↑ 2.20)	79.44±1.57 (↑ 2.45)	80.36±1.84 (↑ 2.73)
	15	69.42±1.71	70.55±1.69	71.61±1.34	72.90±1.46	72.79±1.26	74.90±1.33	72.58±1.27	74.34±1.52	75.63±1.59	75.40±1.32	72.42±1.34 (↑ 3.00)	76.92±1.26 (↑ 1.29)	76.57±1.14 (↑ 1.17)
	20	67.61±1.02	68.84±1.33	67.87±1.22	70.00±1.83	70.55±1.17	71.85±1.22	71.99±1.54	72.06±1.80	72.15±1.27	71.32±1.79	71.61±0.90 (↑ 4.00)	74.35±1.17 (↑ 2.20)	74.44±1.52 (↑ 3.12)
	25	64.81±1.14	65.51±1.45	66.02±1.39	67.89±1.21	66.32±1.07	68.03±1.39	69.47±1.98	68.92±1.27	69.10±1.33	68.72±1.50	69.53±1.03 (↑ 4.72)	71.67±1.07 (↑ 2.57)	72.07±1.21 (↑ 3.35)
Citeseer	0	71.96±0.55	73.26±0.83	71.20±0.83	70.65±0.32	72.26±0.83	72.10±0.63	72.71±0.53	72.54±0.21	73.26±0.38	73.28±0.69	73.66±0.83 (↑ 1.70)	74.21±0.63 (↑ 0.95)	74.47±0.83 (↑ 1.19)
	5	67.96±0.59	68.15±0.89	68.20±0.83	68.65±1.32	69.76±1.48	71.32±0.83	68.16±1.68	69.57±1.38	69.36±1.20	71.24±1.13	69.76±1.38 (↑ 1.80)	72.10±1.48 (↑ 2.74)	72.77±0.73 (↑ 1.53)
	10	64.10±1.34	66.67±1.35	65.92±1.86	66.57±1.35	66.59±1.98	67.86±1.86	66.26±1.54	68.06±1.46	67.52±1.20	68.19±1.13	67.30±1.89 (↑ 3.20)	68.53±1.98 (↑ 1.01)	69.32±1.04 (↑ 1.13)
	15	61.12±1.37	63.62±1.47	64.64±1.72	65.88±1.16	65.37±1.34	66.43±1.72	65.49±1.08	66.99±1.25	66.89±1.03	66.95±1.85	65.29±1.99 (↑ 4.17)	68.25±1.34 (↑ 1.36)	68.89±1.19 (↑ 1.94)
	20	60.26±1.44	61.85±1.35	62.93±1.01	64.66±1.35	63.99±1.44	64.00±1.01	65.58±1.55	65.35±1.44	66.37±1.04	65.89±1.25	63.55±1.64 (↑ 3.29)	68.17±1.44 (↑ 1.80)	68.44±1.75 (↑ 2.55)
	25	59.02±1.39	59.10±1.24	60.87±1.37	63.22±1.24	63.01±1.26	62.85±1.37	64.28±1.23	64.25±1.23	65.11±1.79	64.93±1.70	63.02±1.28 (↑ 4.00)	65.24±1.26 (↑ 0.13)	66.59±1.01 (↑ 1.66)
Polblogs	0	95.69±0.38	95.35±0.20	95.22±0.14	95.31±0.18	85.13±1.22	90.13±1.22	95.36±0.82	95.13±0.85	94.77±1.07	93.20±0.64	96.13±0.14 (↑ 0.44)	94.95±1.22 (↑ 0.18)	95.70±1.22 (↑ 2.50)
	5	87.69±0.38	88.35±0.20	89.22±0.14	90.15±0.18	85.01±1.22	90.31±0.14	89.35±0.32	90.05±0.20	90.31±1.31	90.85±1.39	89.72±0.25 (↑ 2.03)	92.27±1.22 (↑ 1.96)	92.36±0.07 (↑ 1.51)
	10	84.64±1.69	85.88±1.40	85.85±1.91	86.02±1.33	84.56±1.66	87.00±1.91	87.33±1.74	86.68±1.36	86.71±1.27	87.09±1.41	87.71±0.65 (↑ 3.07)	88.75±1.66 (↑ 2.04)	89.33±0.23 (↑ 2.24)
	15	71.55±1.70	72.01±0.20	72.10±1.80	78.18±1.89	79.89±1.75	81.95±1.80	82.04±1.47	82.12±1.99	82.26±1.55	83.31±1.87	76.58±0.56 (↑ 5.03)	84.11±1.75 (↑ 1.85)	84.27±0.27 (↑ 0.96)
	20	65.00±1.03	67.12±1.21	67.32±1.15	69.35±1.32	71.25±1.82	71.51±1.15	69.98±1.22	69.42±1.39	72.59±1.12	72.45±1.69	69.44±0.66 (↑ 4.44)	73.26±1.82 (↑ 0.67)	73.81±0.26 (↑ 1.36)
	25	64.02±1.35	64.16±1.13	66.10±1.11	67.20±1.44	68.39±1.01	69.02±1.11	66.55±1.79	68.22±1.47	70.80±1.33	69.37±1.01	67.92±1.43 (↑ 3.90)	71.92±1.01 (↑ 1.12)	72.21±0.43 (↑ 2.84)
Pubmed	0	87.19±0.09	83.73±0.40	86.16±0.18	83.44±0.21	84.02±0.26	87.06±0.06	86.43±0.72	86.74±0.15	85.08±0.20	87.33±0.18	88.10±0.18 (↑ 0.91)	87.15±0.11 (↑ 2.07)	88.41±0.26 (↑ 1.08)
	5	79.19±0.09	80.73±0.40	80.16±0.18	81.44±0.21	80.02±0.26	82.44±0.18	82.96±0.86	82.42±0.21	82.17±1.02	83.13±1.67	81.32±1.08 (↑ 2.13)	83.15±0.26 (↑ 0.98)	84.33±1.71 (↑ 1.20)
	10	75.42±1.40	76.85±1.59	77.93±1.40	80.25±1.34	78.40±1.70	80.45±1.40	80.74±1.66	80.81±1.30	80.59±1.84	81.26±1.59	78.22±1.43 (↑ 2.80)	82.10±1.70 (↑ 1.51)	82.74±1.14 (↑ 1.48)
	15	71.96±1.55	72.34±1.55	75.75±1.60	75.21±1.22	73.97±1.76	76.89±1.60	76.87±1.82	76.90±1.25	77.31±1.35	77.09±1.32	75.47±1.34 (↑ 3.51)	78.49±1.76 (↑ 1.18)	78.88±1.60 (↑ 1.79)
	20	68.46±0.65	68.99±1.72	70.03±1.49	72.03±1.52	71.32±1.60	71.62±1.49	71.28±1.58	73.02±1.52	73.20±1.75	72.85±1.14	73.91±0.90 (↑ 5.45)	74.81±1.60 (↑ 1.61)	74.90±1.32 (↑ 2.05)
	25	69.01±1.03	68.16±1.01	69.02±1.18	71.02±1.32	70.01±1.51	70.86±1.18	70.76±1.85	72.85±1.31	72.69±1.12	71.66±1.67	72.43±1.03 (↑ 3.42)	73.96±1.51 (↑ 1.27)	74.24±1.39 (↑ 2.58)

nism in the RGDM, we aim to improve the robustness of the GNNs trained on the augmented graphs against the graph adversarial attacks. In Section 4.1, we have shown that both the vanilla GCN and the robust GNNs trained on the augmented graph achieve significantly better performance for semi-supervised node classification under graph adversarial attacks. In this section, we directly evaluate the data quality of the synthetic graph structures generated by the RGDM. In the visual domain, the Frechet Inception Distance (FID) is a widely used metric to evaluate the quality of the synthetic images generated by the generative models [95, 27]. The FID score measures the similarity between the distribution of the generated images and the distribution of the real images. To compute the FID score, the pre-trained Inception v3 [96] is used to extract the features from both the real images and the generated images, which are then modeled as the multivariate Gaussian distributions. The FID score is then calculated using the Frechet Distance (FD) [95] between the two multivariate Gaussian distributions modeling the real and the generated images [97]. A lower FID score indicates that the generated images are more similar to the real images, suggesting better quality.

Quality Evaluation of the Synthetic Nodes. Although the Inception model cannot be applied to the graph data, we can replace the Inception model in the computation of the FID score with the pre-trained GCN [47] for extracting node features to adapt the metric to evaluate the quality of synthetic nodes generated by the RGDM. To this end, we define the Frechet Node Distance (FND), which is the FD between the multivariate Gaussians modeling the node features extracted by pre-trained GCN. We randomly split the nodes in the original clean graph into two partitions of equal size, which are the base partition and the test partition. To mitigate the influence of the randomness, we compute the FND scores with 10 different random splits and report the mean and the standard deviation of the FND scores across different runs. The FND computed between the nodes in the test partition and the base partition of the original clean graph establishes the baseline of the expected FND score for high-quality nodes. By computing the FND score between the features of nodes in the synthetic graph structures generated by the RGDM and the features of nodes in the base partition of the original clean graph, we evaluate the quality of the synthetic nodes in the synthetic graph structures. For simplicity, we refer to the FND score for the synthetic nodes as the FND between their features and the features of the nodes in the base partition of the original clean graph.

To show the effectiveness of the robustness loss in the training of the RGAE for the RGDM, we also compute the FND for the nodes in the synthetic graph structures generated by the RGDM without the node robustness loss and the edge robustness loss. To demonstrate the advantages of the RGDM over the vanilla diffusion model, the DDPM [27], we train a baseline DDPM model on the input node attributes of the original attacked graph and synthesize the same number of synthetic nodes as the RGDM. The synthetic edges are then generated by connecting each synthetic node to its K-nearest neighbors in the original attacked graph using the K-nearest neighbors (KNN) algorithm with $K = \lceil d_{\text{ave}} \rceil$, where d_{ave} is the average degree of the original clean graph. The synthetic graph structures, including the synthetic nodes and edges generated by the baseline DDPM model, are combined with the original attacked graph to form the augmented graph. Next, we compute the FND score for the nodes in the synthetic graph structures generated by the baseline DDPM model. In addition, we compute FND for nodes generated by the four baseline node augmentation methods, which are the iGraphMix [12], ImGAGN [42], Graphsmote [43], and SNS [45]. We also calculate the FND score for nodes in the original attacked graph. We use the same GCN [47] pre-trained on the original clean graph to extract the node features for computing the FND score. The ablation study is performed on the original attacked graph perturbed by the Metattack with a perturbation rate of 25%.

Table 13: Frechet Node Distance (FND) to the nodes in the base partition of the original clean graph. The mean and standard deviation of the FND scores computed with 10 different random splits of the base partition and the test partition in the original clean graph are reported. The evaluation is performed for Metattack with a perturbation rate of 25%. The FND score for the original clean graph is computed between the nodes in the test partition and the nodes in the base partition of the original clean graph.

Data	Cora	Citeseer	Polblogs	Pubmed
Original Attacked Graph	12.10±0.45	7.53±0.22	8.26±0.37	6.11±0.38
iGraphMix [12]	13.44±0.33	9.02±0.42	10.75±0.44	7.65±0.38
ImGAGN [42]	13.10±0.38	7.96±0.42	9.86±0.43	6.95±0.27
Graphsmote [43]	13.52±0.42	8.95±0.22	10.60±0.27	7.45±0.51
SNS [45]	13.35±0.42	7.88±0.27	9.30±0.85	6.74±0.41
Baseline DDPM	13.28±0.40	8.82±0.26	10.25±0.39	7.25±0.41
RGDM (w/o Robustness Loss)	11.17±0.46	6.15±0.31	8.06±0.44	5.89±0.35
RGDM (w/o Node Robustness Loss)	10.02±0.38	5.89±0.28	6.95±0.35	5.01±0.36
RGDM (w/o Edge Robustness Loss)	9.67±0.27	5.18±0.49	6.65±0.25	5.11±0.38
RGDM	**8.16±0.29**	**4.44±0.32**	**4.35±0.18**	**3.95±0.43**
Original Clean Graph	7.90±0.31	4.28±0.25	4.16±0.27	3.79±0.35

The evaluation results are shown in Table 13, where the lower FND scores indicate the node features are more similar to the features of nodes in the base partition of the original clean graph. The FND score of the nodes in the synthetic graph structures generated by the RGDM is closest to the FND score of the original clean graph, which demonstrates that the RGDM generates faithful synthetic nodes. For example, the FND score for the RGDM on Pubmed is 3.95, which is only 4.2% higher than 3.79, which is the FND score of the original clean graph. In addition, we have several key observations from Table 13. First, the synthetic nodes generated by the baseline DDPM are even worse than the original attacked graph. This is because the DDPM is trained on the original attacked graph

without any robustness adaptations, and the noises in the original attacked graph are propagated to the synthetic nodes in the augmented graph. Second, the robustness loss is critical in generating clean augmented graphs. Without either the node robustness loss or the edge robustness loss, the synthetic nodes generated by the ablation models are worse than the synthetic nodes generated by the RGDM. This observation proves that both the node robustness loss and the edge robustness loss are beneficial for improving the quality of the synthetic nodes in the synthetic graph structures generated by the RGDM, as mentioned in Section 3.2. Third, the synthetic nodes generated by RGDM are significantly more faithful than those generated by the baseline node-level graph augmentation methods.

Quality Evaluation of the Synthetic Edges. Similar to the design of the FND score for evaluating the quality of synthetic nodes, we replace the Inception model in the computation of FID with the pre-trained GNN [98] for edge feature extraction to adapt the metric to evaluate the quality of the synthetic edges generated by the RGDM. To this end, we define Frechet Edge Distance (FED), which is the FD between the multivariate Gaussians modeling the edge features extracted by pre-trained GNN. Similar to the evaluation of the FND, we randomly split the edges in the original clean graph into two partitions of equal size, which are the base partition and the test partition. To mitigate the influence of the randomness, we compute the FED scores with 10 different random splits and report the mean and the standard deviation of the FED across different runs. The FED computed between the edges in the test partition and the edges in the base partition of the original clean graph establishes the baseline of the expected FED score for the high-quality edges. By computing the FED between the features of edges in the synthetic graph structures generated by the RGDM and the features of edges in the base partition of the original clean graph, we evaluate the quality of the edges in the synthetic graph structures. For simplicity, we refer to the FED score for the synthetic edges as the FED between their features and the features of edges in the base partition of the original clean graph. Similar to the evaluation of the synthetic nodes, we also compute the FED score for the edges in the synthetic graph structures generated by the RGDM without the node robustness loss or the edge robustness loss. We compute the FED score for the edges in the synthetic graph structures generated by the baseline DDPM model. Since the edges in the synthetic graph structures are generated by the KNN algorithm, we evaluate the baseline DDPM models using different values of K from $\{\lceil d_{\text{ave}}/4 \rceil, \lceil d_{\text{ave}}/2 \rceil, \lceil d_{\text{ave}} \rceil, \lceil 2 \times d_{\text{ave}} \rceil, \lceil 4 \times d_{\text{ave}} \rceil\}$. In addition, we compute FED for edges generated by the four baseline node augmentation methods, which are the iGraphMix [12], ImGAGN [42], Graphsmote [43], and SNS [45]. We also calculate the FED score for nodes in the original attacked graph. The FED score for edges in the original attacked graph is also computed. We use the same NBFNet [98] pre-trained on the original clean graph to extract the edge features for computing the FED score. The ablation study is performed on the original attacked graph perturbed by the Metattack with a perturbation rate of 25%.

Table 14: Frechet Edge Distance (FED) to the edges in the base partition of the original clean graph. The mean and standard deviation of the FED scores computed with 10 different random splits of the base partition and the test partition in the original clean graph are reported. The evaluation is performed for Metattack with a perturbation rate of 25%. The FED score for the original clean graph is computed between the edges in the test partition and the edges in the base partition of the original clean graph.

Data	Cora	Citeseer	Polblogs	Pubmed
Original Attacked Graph	10.32±0.42	7.83±0.28	8.92±0.39	6.53±0.37
iGraphMix [12]	11.25±0.28	8.42±0.43	9.63±0.44	7.11±0.27
ImGAGN [42]	11.05±0.38	8.04±0.26	8.21±0.51	7.16±0.79
Graphsmote [43]	11.13±0.29	8.16±0.51	9.32±0.41	7.18±0.44
SNS [45]	11.05±0.38	8.04±0.26	8.21±0.51	7.16±0.79
Baseline DDPM ($K = \lceil d_{\text{ave}}/4 \rceil$)	12.03±0.45	8.72±0.31	10.77±0.40	7.93±0.42
Baseline DDPM ($K = \lceil d_{\text{ave}}/2 \rceil$)	11.27±0.38	8.49±0.27	9.32±0.35	7.28±0.34
Baseline DDPM ($K = \lceil d_{\text{ave}} \rceil$)	10.39±0.40	7.96±0.30	8.97±0.37	6.83±0.33
Baseline DDPM ($K = \lceil 2 \times d_{\text{ave}} \rceil$)	10.47±0.36	7.93±0.29	9.21±0.36	6.91±0.35
Baseline DDPM ($K = \lceil 4 \times d_{\text{ave}} \rceil$)	10.88±0.41	8.01±0.32	9.38±0.38	7.08±0.38
RGDM (w/o Robustness Loss)	10.07±0.39	7.64±0.26	8.74±0.33	6.27±0.31
RGDM (w/o Node Robustness Loss)	8.87±0.35	6.73±0.25	7.93±0.31	5.78±0.32
RGDM (w/o Edge Robustness Loss)	8.98±0.34	6.82±0.27	7.72±0.29	5.91±0.30
RGDM	**8.34±0.29**	**5.28±0.22**	**6.61±0.25**	**5.34±0.28**
Original Clean Graph	8.03±0.28	5.10±0.23	6.31±0.24	5.15±0.25

The evaluation results are shown in Table 14, where the lower FED scores indicate the edge features are more similar to the features of edges in the base partition of the original clean graph. The FED score of the edges in the synthetic graph structures generated by the RGDM is closest to the FED score of the original clean graph, which demonstrates that the RGDM generates faithful synthetic edges. For example, the FED score for the RGDM on Pubmed is 5.34, which is only 3.7% higher than 5.15, which is the FED score of the original clean graph. In addition, we have similar observations from Table 14 as those from Table 13. First, the synthetic edges generated by the baseline DDPM are even worse than the original attacked graph. This is because the DDPM is trained on the original attacked graph without any robustness adaptations, and the simple edge generation method using the KNN on top of the synthetic nodes generated by the DDPM is vulnerable to noises in the original attacked graph. Edge generation with smaller K in the KNN algorithm is more vulnerable to the noises in the original attacked graph. Although increasing the value of K improves the quality of the synthetic edges, the edges generated by the KNN on top of the nodes generated by the baseline DDPM are still worse than the original attacked graphs. Second, the robustness loss is critical in generating clean augmented graphs. Without either the node robustness loss or the edge robustness loss, the synthetic edges generated by the ablation models are worse than the synthetic edges generated by the RGDM. This observation proves that both the node robustness loss and the edge robustness loss are beneficial for improving the quality of synthetic edges in the synthetic graph structures generated by the RGDM, as mentioned in Section 3.2.

The synthetic node and edge distributions have FND/FED values close to the clean references in Tables 13 and 14. This is empirical evidence of fidelity, rather than a guarantee that generated structures are free of errors. The role of the normalized perturbation density is specified below.

F.4 Normalized Perturbation Density and the Scope of Augmentation

The main augmentation procedure retains the original attacked graph. Let b denote the number of original edge perturbations, E_{att} its edge count, and E_{syn} the number of added synthetic edges, counting each undirected edge once. The descriptive normalized densities are

$$\rho_{\text{att}} \coloneqq \frac{b}{E_{\text{att}}}, \qquad \rho_{\text{aug}} \coloneqq \frac{b}{E_{\text{att}} + E_{\text{syn}}}. \tag{10}$$

Equation (10) keeps the original perturbation count fixed and changes only the denominator. The reported 25% to 20.6% Citeseer example adds 785 synthetic edges. The Cora example adds 912 and yields approximately 21.1%. These are descriptive normalizations, not estimates of the proportion of all harmful edges after generation. An edge flip budget can include deletions, and synthetic edges can themselves contain errors. In particular, a lower normalized density neither proves that original adversarial edges have been removed nor establishes improved robustness by itself. The proposed mechanism is additional labeled supervision and relatively clean message passing paths, supported empirically by the fidelity and controlled classifier comparisons. Edge selection acts during generator learning and reconstruction, while the original attacked block $\mathbf{A}$ remains unchanged in Equation (3). RGDM (Purification) in Section F.12 is a separate intervention that reconstructs the original graph.

F.5 Evaluation on Training Time and Synthetic Data Generation Time

In this section, we detail the training and synthetic data generation times for the Robust Graph Diffusion Model (RGDM) across various datasets in Table 16. The training time for the RGAE and the LDM in the RGDM using a single NVIDIA A100 GPU is measured in minutes. In addition, we present the time for generating synthetic graph structures by the RGDM in seconds per synthetic node generated in Table 16 as well.

Controlled decoder comparison. Table 15 compares SHED with a traditional global decoder that reconstructs a graph sized neighbor vector, using the reported single A100 setup. RGAE training is measured in minutes. Synthetic node generation is measured in seconds per node and includes both LDM sampling and decoding. Neither quantity is downstream GNN inference time. The training speedups are $480/153 = 3.14$ on Pubmed and $2511/452 = 5.56$ on ogbn-arxiv. Their end to end generation speedups are $0.080/0.073 = 1.10$ and $0.49/0.15 = 3.27$. LDM sampling limits the generation time improvement on smaller graphs, so a decoder operation count reduction does not imply the same wall clock speedup for the entire pipeline.

Table 15: SHED versus a global decoder on one NVIDIA A100. The generation measurement includes LDM sampling and decoding.

Dataset	RGAE training (min)		Generation (s/node)	
	SHED	Global	SHED	Global
Cora	20	29	0.066	0.089
Citeseer	23	30	0.067	0.086
Polblogs	65	88	0.069	0.090
Pubmed	153	480	0.073	0.080
ogbn-arxiv	452	2511	0.15	0.49

Table 16: Time for the training of the RGDM and data generation with the RGDM on different datasets.

Datasets	RGDM Training Time (minutes)		Generation Time (s/sample)
	RGAE	LDM	
Cora	20	39	0.066
Citeseer	23	41	0.067
Polblogs	65	31	0.069
Pubmed	153	154	0.073

Table 17: Time for selecting the optimal number of synthetic nodes N' with cross-validation.

Datasets	Cross-validation Time (minutes)		
	RGDM (GCN)	RGDM (SG-GSR)	RGDM (Pro-GNN)
Cora	1.1	3.5	43.3
Citeseer	1.0	3.4	38.5
Polblogs	1.6	3.9	26.9
Pubmed	9.4	20.5	176.5

Table 18: The number of synthetic nodes N' ($\times |\mathcal{V}_L|$) selected for Metattack, where $\mathcal{V}_L$ is the labeled nodes of the original graph.

Dataset	Ptb Rate (%)	RGDM (GCN)	RGDM (SG-GSR)	RGDM (Pro-GNN)
Cora	0	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	5	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	10	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	15	$3 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	20	$3 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	25	$5 \times \lvert\mathcal{V}_L\rvert$	$5 \times \lvert\mathcal{V}_L\rvert$	$5 \times \lvert\mathcal{V}_L\rvert$
Citeseer	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	5	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	10	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	15	$3 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	20	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	25	$4 \times \lvert\mathcal{V}_L\rvert$	$5 \times \lvert\mathcal{V}_L\rvert$	$4 \times \lvert\mathcal{V}_L\rvert$
Polblogs	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	5	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	10	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	15	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	20	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	25	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
Pubmed	0	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	5	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	10	$3 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	15	$3 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	20	$3 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	25	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$

Table 19: The number of synthetic nodes N' ($\times |\mathcal{V}_L|$) for Nettack, where $\mathcal{V}_L$ is the labeled nodes of the original graph.

Dataset	Attack Budget	RGDM (GCN)	RGDM (SG-GSR)	RGDM (Pro-GNN)
Cora	0	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	1	$3 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	2	$3 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	3	$3 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	4	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	5	$4 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
Citeseer	0	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	1	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	2	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	3	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	4	$3 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	5	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
Polblogs	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	1	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	2	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	3	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	4	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	5	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
Pubmed	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	1	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	2	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	3	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	4	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	5	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$

Table 20: The number of synthetic nodes N' ($\times |\mathcal{V}_L|$) for Topology Attack, where $\mathcal{V}_L$ is the labeled nodes of the original graph.

Dataset	Ptb Rate (%)	RGDM (GCN)	RGDM (SG-GSR)	RGDM (Pro-GNN)
Cora	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	5	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	10	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	15	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	20	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	25	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
Citeseer	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	5	$2 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	10	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	15	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	20	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	25	$3 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
Polblogs	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	5	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	10	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	15	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	20	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$
	25	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
Pubmed	0	$1 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	5	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$
	10	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	15	$3 \times \lvert\mathcal{V}_L\rvert$	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$
	20	$2 \times \lvert\mathcal{V}_L\rvert$	$3 \times \lvert\mathcal{V}_L\rvert$	$4 \times \lvert\mathcal{V}_L\rvert$
	25	$1 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$	$2 \times \lvert\mathcal{V}_L\rvert$

F.6 Cross-validation on the Number of Synthetic Nodes

In our experiments, we generate $10 \times |\mathcal{V}_L|$ synthetic nodes, where $\mathcal{V}_L$ is the training set of the original graph. Then, we search for the optimal size of synthetic nodes N' added in the augmented graph from $1 \times |\mathcal{V}_L|$ to $10 \times |\mathcal{V}_L|$ with a step size of $|\mathcal{V}_L|$. In the cross-validation, we train different GNN classifiers with different numbers of synthetic data and set N' to the one that achieves the best validation accuracy by the end of 40% of their total epoch number. To study the effectiveness of the cross-validation process for selecting the optimal number of synthetic nodes, we calculate the complete cross-validation time of different models on different datasets. All the experiments

are performed on one Nvidia A100 GPU. It is observed from the results in Table 17 that the cross-validation process does not largely increase the computation overhead. The numbers of synthetic nodes added to the augmented graphs for all the experiments are shown in Table 18, Table 19, and Table 20.

F.7 Adversarial Defense on Large Scale Graphs

To evaluate the scalability of the proposed method, we conduct additional experiments on graph adversarial defense for large-scale graph datasets, which are the large-scale social network, Reddit [99], and the large-scale knowledge graph, ogbn-arxiv [100]. The statistics of the Reddit and ogbn-arxiv datasets are detailed in Table 9. Since the Metattack [51], the Nettack [50], and the Topology Attack [85] adopted in Section 4.1 do not scale to large graphs at the size of Reddit and ogbn-arxiv, we use a scalable graph adversarial attack method, the improved DICE [101], in our experiments on Reddit and ogbn-arxiv. The experiments are performed for perturbation rates of 0%, 10%, and 20% following the settings in [101] with GCN as the baseline model. The results are shown in Table 21. It is observed that RGDM consistently improves the performance of the GCN baseline across all perturbation levels on both datasets. For example, RGDM achieves an improvement of 2.25% on ogbn-arxiv at the perturbation rate of 10%, indicating its effectiveness in defending against adversarial perturbations even on large and complex knowledge graphs.

Table 21: Node classification performance (Accuracy$\pm$Std) under the adversarial attack by the improved DICE [101]. The improvements of the RGDM over the baseline are attached in parentheses.

Dataset	Ptb Rate (%)	GCN	RGDM (GCN)
ogbn-arxiv	0	72.82±1.32	**74.48** ±1.15 (↑ 1.66)
	10	62.25±1.23	**64.50** ±1.35 (↑ 2.25)
	20	54.95±1.44	**56.17** ±1.66 (↑ 1.22)
Reddit	0	93.52±1.17	**95.44** ±1.42 (↑ 1.92)
	10	82.62±1.32	**84.73** ±1.12 (↑ 2.11)
	20	74.53±1.57	**75.99** ±1.22 (↑ 1.46)

Edge selection at scale. Table 22 isolates the edge selection module at 20% improved DICE, keeping the GCN backbone fixed. Without the module, the gains over GCN are only 0.35 and 0.47 points. Restoring it adds 0.87 and 0.99 points, respectively. This extends the four dataset sparsification ablation to both large graphs.

Table 22: Edge selection ablation on large graphs under 20% improved DICE, with GCN fixed. Accuracy (%).

Method	ogbn-arxiv	Reddit
GCN	54.95	74.53
RGDM without edge selection	55.30	75.00
RGDM (full)	56.17	75.99

F.8 Sensitivity Analysis on the Hyper-Parameters

In this section, we conduct sensitivity analyses on the key hyper-parameters of RGDM, including the temperature parameter τ for edge selection and the number of clusters K in SHED. The study is carried out using RGDM (Pro-GNN) on the Cora dataset under a 25% Metattack setting. As shown in Table 23, although the highest accuracy is achieved at $\tau = 0.5$, the performance remains stable across different values of τ. In particular, even the lowest performing setting, $\tau = 0.1$, results in only a marginal 0.19% decrease in accuracy compared to the best result.

Table 23: Sensitivity analysis on the temperature for edge selection τ. The study is performed by using RGDM (Pro-GNN) on Cora with 25% of Metattack.

τ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Accuracy	72.12	72.16	72.30	72.29	**72.31**	72.28	72.24	72.22	72.25

We further study the sensitivity of the proposed SHED module to the number of clusters K. The study is conducted using RGDM (Pro-GNN) on the Cora dataset under a 25% Metattack setting. Since SHED partitions the graph into K balanced clusters and sets the maximum cluster size as $M = \lceil N/K \rceil$, changing K affects both the granularity of hierarchical edge decoding and the generation efficiency. As shown in Table 24, RGDM is not sensitive to the choice of K. The best performance is achieved at $K = 100$, which is the default setting used in our experiments. Moreover, the accuracy remains stable when we vary K from 20 to 200. Even the lowest-performing setting only decreases the accuracy by 0.19% compared with the best result. In addition, the generation time remains close to the default setting, indicating that SHED provides stable robustness and efficient generation across a wide range of cluster numbers.

Table 24: Sensitivity analysis on the number of clusters K in SHED. The study is performed by using RGDM (Pro-GNN) on Cora with 25% of Metattack.

K	20	40	60	80	100	120	150	200
$M = \lceil N/K \rceil$	125	63	42	32	25	21	17	13
Generation Time (s/sample)	0.075	0.070	0.068	0.067	0.066	0.067	0.068	0.071
Accuracy	72.12	72.20	72.26	72.29	**72.31**	72.28	72.24	72.20

F.9 Adversarial Defense on Heterophilic Graph

In this section, we study the effectiveness of RGDM for adversarial defense on heterophilic graphs. Following [88], we perform the study on Chameleon [89] and Actor [90] for the defense against the Metattack [51]. The adversarially robust GNN, LHS [88], which is specifically designed for heterophilic graphs, is adopted as the baseline model. The results are shown in Table 25. RGDM improves LHS at each tested perturbation rate on both datasets. At 25% Metattack, the improvements are 2.38 points on Chameleon and 2.53 points on Actor. These results support the cross validated soft penalty in the tested settings and do not imply that stronger low rank regularization always helps. The main text discussion and mean accuracies appear in Table 6.

Table 25: Node classification performance (Accuracy±Std) under the non-targeted poisoning attack (Metattack) [51]. The improvements of the RGDM over the baseline are attached in parentheses.

Dataset	Ptb Rate (%)	LHS	RGDM (LHS)
Chameleon	0	72.31±1.32	**74.02** ±1.13 (↑ 1.71)
	15	71.87±1.61	**72.98** ±1.11 (↑ 1.11)
	25	70.03±1.85	**72.41** ±1.66 (↑ 2.38)
Actor	0	38.87±1.43	**41.80** ±1.42 (↑ 2.93)
	15	36.79±1.11	**39.01** ±1.12 (↑ 2.22)
	25	35.37±1.12	**37.90** ±1.01 (↑ 2.53)

F.10 t-SNE Visualization of the Augmented Graph

Figure 5 illustrates the t-SNE visualization of both the real and synthetic node features within the augmented graph, revealing that the synthetic nodes closely mimic the real nodes.

F.11 Statistical Significance Analysis

We evaluate the paired gain $\Delta_s := a_s^{\text{RGDM}} - a_s^{\text{base}}$ in percentage points for each shared attack seed $s \in [10]$. The sample mean and sample SD are defined by Equation (11):

$$\bar{\Delta} := \frac{1}{10}\sum_{s=1}^{10}\Delta_s, \qquad s_\Delta := \sqrt{\frac{1}{9}\sum_{s=1}^{10}(\Delta_s - \bar{\Delta})^2}, \qquad t_{\text{pair}} := \frac{\sqrt{10}\bar{\Delta}}{s_\Delta}. \tag{11}$$

A two sided paired Student test compares $|t_{\text{pair}}|$ with a Student distribution with 9 degrees of freedom, under the usual assumptions of independent runs and approximately normal paired differences. This tests within backbone augmentation, not differences between unrelated architectures. Table 26 reports the summaries from the separate paired evaluation. Each uncertainty is an SD of differences, not an SD of accuracies or a standard error. **Precision of the significance statements.** The

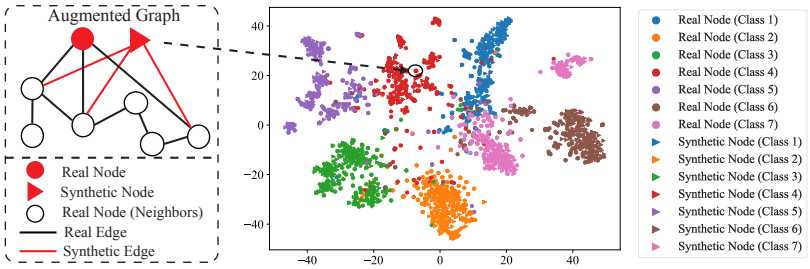


Figure 5: Illustration of the augmented graph after adding the synthetic graph structures into the original graph on Cora. The RGDM is trained on Cora under Metattack with a perturbation rate of 25%. The figure on the right illustrates the t-SNE visualization of the node features for both the real nodes in the original clean graph and the synthetic nodes. The figure on the left illustrates the 2-hop neighborhood of a real node and a synthetic node that have similar node features, evidenced by the t-SNE visualization in the augmented graph. It is observed that the synthetic node shares the same neighbors as the real node.

Table 26: Reported Metattack mean gain $\pm$ paired SD over 10 shared attack seeds (percentage points). Only rows with reported paired SD are included.

Dataset	Ptb (%)	GCN	Pro-GNN	SG-GSR
Cora	0	1.7 ± 0.3	1.3 ± 0.3	0.4 ± 0.1
Cora	15	4.3 ± 0.5	1.9 ± 0.4	1.5 ± 0.3
Cora	25	13.9 ± 0.7	2.6 ± 0.4	1.3 ± 0.3
Citeseer	15	4.2 ± 0.5	1.7 ± 0.4	2.9 ± 0.4
Citeseer	25	5.9 ± 0.5	1.2 ± 0.3	2.6 ± 0.4
Polblogs	15	4.9 ± 0.5	2.0 ± 0.4	1.1 ± 0.3
Polblogs	25	7.3 ± 0.6	3.7 ± 0.5	1.0 ± 0.3
Pubmed	15	6.3 ± 0.5	0.9 ± 0.2	1.4 ± 0.3
Pubmed	25	6.0 ± 0.5	0.8 ± 0.2	1.4 ± 0.3

separately reported significance levels in Table 27 are retained. For Metattack rows with paired summaries, they are stated as conservative upper thresholds, not exact p values reconstructed from rounded means and SD. This distinction is necessary: for example, the Cora 25% SG-GSR summary 1.3 ± 0.3 gives $t_{\text{pair}} \approx 13.70$ in Equation (11), whose nominal two sided p value is smaller than the separately reported 0.008. Reporting $p \leq 0.008$ preserves that significance level without identifying it as the exact p value of those rounded moments. No exact p values or confidence intervals are imputed from unreported runs. The reported significance levels at the strongest budgets are 0.006 to 0.008 for Metattack, 0.005 to 0.006 for Nettack, and 0.006 to 0.007 for Topology Attack. The latter two retain their reported values because paired SD were not supplied for them.

F.12 Study on RGDM for Graph Purification

The RGDM proposed in Section 3 has demonstrated superior performance for generative data augmentation (GDA), which involves generating faithful synthetic graph structures and integrating them into the original attacked graph to obtain an augmented graph. In this section, we study the effectiveness of applying RGDM to purify the attacked graph. The graph purification by RGDM, referred to as RGDM (Purification), is achieved by reconstructing each node and its corresponding edges through the LDM, where Gaussian noise derived from the original node and its edges is progressively denoised by the LDM and subsequently decoded by the GAE to produce clean node features and edges. The RGDM (Purification) variant only purifies the original graph, aiming to explore whether RGDM can be adapted to directly denoise/remove the adversarial perturbations in the original attacked graph. Moreover, the RGDM (Purification) can be combined with RGDM for GDA, denoted as RGDM (GDA) here, leading to a variant that performs both GDA and graph purification, denoted as RGDM (GDA + Purification). The study is performed on Cora under Metattack with a 25% attack rate. Table 28 shows that applying RGDM for purification enhances the performance of Pro-GNN, while GDA with RGDM achieves better performance. Moreover, it is observed that combining GDA and graph purification by RGDM further improves the performance.

Table 27: Reported significance levels with each backbone fixed: Metattack. For rows with paired summaries in Table 26, $\leq$ denotes a conservative upper threshold, not an exact p value reconstructed from rounded moments. Remaining entries are reported p values.

Dataset	Ptb (%)	RGDM (GCN)	RGDM (Pro-GNN)	RGDM (SG-GSR)
Cora	0	≤ 0.042	≤ 0.027	≤ 0.016
	5	0.038	0.024	0.013
	10	0.033	0.021	0.012
	15	≤ 0.029	≤ 0.017	≤ 0.010
	20	0.025	0.015	0.009
	25	≤ 0.021	≤ 0.012	≤ 0.008
Citeseer	0	0.039	0.022	0.018
	5	0.035	0.020	0.015
	10	0.030	0.017	0.013
	15	≤ 0.027	≤ 0.014	≤ 0.011
	20	0.023	0.012	0.009
	25	≤ 0.019	≤ 0.010	≤ 0.007
Polblogs	0	0.041	0.023	0.019
	5	0.036	0.020	0.015
	10	0.031	0.017	0.013
	15	≤ 0.027	≤ 0.014	≤ 0.011
	20	0.024	0.011	0.009
	25	≤ 0.020	≤ 0.009	≤ 0.006
Pubmed	0	0.038	0.021	0.017
	5	0.034	0.018	0.014
	10	0.030	0.015	0.012
	15	≤ 0.027	≤ 0.013	≤ 0.010
	20	0.023	0.010	0.008
	25	≤ 0.019	≤ 0.008	≤ 0.006

Table 27: Reported significance levels with each backbone fixed: Nettack. Continued from the preceding panel. Entries are the reported p values.

Dataset	Budget	RGDM (GCN)	RGDM (Pro-GNN)	RGDM (SG-GSR)
Cora	0	0.041	0.025	0.015
	1	0.036	0.023	0.011
	2	0.032	0.019	0.010
	3	0.027	0.016	0.008
	4	0.022	0.013	0.007
	5	0.019	0.011	0.006
Citeseer	0	0.037	0.021	0.016
	1	0.033	0.019	0.013
	2	0.028	0.016	0.011
	3	0.024	0.013	0.009
	4	0.021	0.011	0.007
	5	0.018	0.009	0.006
Polblogs	0	0.040	0.021	0.017
	1	0.034	0.018	0.013
	2	0.029	0.016	0.011
	3	0.025	0.013	0.009
	4	0.022	0.010	0.007
	5	0.019	0.008	0.005
Pubmed	0	0.037	0.020	0.015
	1	0.033	0.017	0.012
	2	0.029	0.014	0.010
	3	0.026	0.012	0.008
	4	0.021	0.009	0.006
	5	0.018	0.007	0.005

Table 27: Reported significance levels with each backbone fixed: Topology Attack. Continued from the preceding panel. Entries are the reported p values.

Dataset	Ptb (%)	RGDM (GCN)	RGDM (Pro-GNN)	RGDM (SG-GSR)
Cora	0	0.043	0.029	0.017
	5	0.039	0.025	0.012
	10	0.035	0.020	0.011
	15	0.030	0.018	0.009
	20	0.026	0.014	0.008
	25	0.022	0.012	0.007
Citeseer	0	0.038	0.023	0.017
	5	0.034	0.020	0.014
	10	0.029	0.017	0.012
	15	0.026	0.014	0.010
	20	0.022	0.012	0.008
	25	0.020	0.010	0.007
Polblogs	0	0.042	0.022	0.018
	5	0.035	0.019	0.014
	10	0.030	0.017	0.012
	15	0.026	0.015	0.010
	20	0.023	0.011	0.008
	25	0.021	0.009	0.006
Pubmed	0	0.039	0.022	0.016
	5	0.034	0.018	0.013
	10	0.030	0.015	0.011
	15	0.027	0.013	0.009
	20	0.022	0.010	0.007
	25	0.019	0.008	0.006

Table 28: Study on the effectiveness of graph purification and GDA by RGDM. The evaluation is conducted on Cora under Metattack with 25% perturbation rate.

Method	Accuracy (%)
Pro-GNN	69.7
RGDM (Purification)	71.0
RGDM (GDA)	72.3
RGDM (GDA + Purification)	**72.9**

F.13 Comparison with Diffusion-Based Graph Purification Methods

To demonstrate the superiority of RGDM over existing diffusion-based graph purification methods [28, 29], we have performed experiments comparing RGDM with GDDM [28] and DiffSP [29] on Cora, Citeseer, Polblogs, and Pubmed under Metattack with a perturbation rate of 25%. We run each experiment 10 times with different random seeds and report the mean and the standard deviation of the node classification accuracy. We use SG-GSR [33], which is the state-of-the-art robust GNN model, as the baseline model for RGDM following the settings in Section 4.2 of the main paper. The GDDM and DiffSP are used to purify the input graph to SG-GSR, and our proposed RGDM is used to augment the input graph to SG-GSR with synthetic graph structures. It is observed in Table 29 that RGDM (SG-GSR) outperforms DiffSP and GDDM on all the datasets. To validate the significance of the improvements, we calculate the p-values of the t-tests between RGDM (SG-GSR) and the best-performing baseline method on each of the datasets. As observed in the table below, the p-values for RGDM (SG-GSR) against the best baseline methods across all datasets are consistently smaller than 0.05, indicating statistically significant improvements of RGDM (SG-GSR) over the top baseline methods among DiffSP and GDDM.

Table 29: Comparison of classification accuracy (±Std) and p-values across different datasets between RGDM and diffusion-based graph purification methods.

Datasets	GDDM [28]	DiffSP [29]	RGDM (SG-GSR)	p-value
Cora	77.2 ± 0.3	76.8 ± 0.5	$\mathbf{79.1 \pm 1.6}$	0.0022
Citeseer	71.6 ± 0.5	71.8 ± 0.2	$\mathbf{75.3 \pm 1.0}$	0.0012
Polblogs	84.8 ± 0.8	85.6 ± 0.5	$\mathbf{88.8 \pm 1.8}$	0.0004
Pubmed	85.9 ± 0.6	86.8 ± 0.4	$\mathbf{88.6 \pm 1.3}$	0.0017

F.14 Ablation on the Number of Synthetic Nodes

To assess the effect of the number of synthetic nodes on robustness, we conduct an ablation study by varying the number of synthetic nodes from 0 to $10 \times |\mathcal{V}_L|$. We evaluate the performance of RGDM with Pro-GNN as the base GNN for node classification under Metattack with a perturbation rate of 25%. As shown in Table 30, increasing the number of synthetic nodes initially improves performance, with the best result achieved at $5 \times |\mathcal{V}_L|$. However, further increasing the number of synthetic nodes leads to slightly degraded performance.

Table 30: Performance of RGDM (Pro-GNN) under Metattack with varying numbers of synthetic nodes. The perturbation rate is 25%.

# Synthetic Nodes ($\times |\mathcal{V}_L|$)	1	2	3	4	5	6	7	8	9	10
Accuracy (%)	70.5	71.0	71.9	72.2	**72.3**	72.1	72.0	71.7	71.4	71.2

F.15 Performance Comparison with Vanilla DDPM-Augmented Graphs

To demonstrate the superiority of the synthetic graph structures generated by RGDM, we compare RGDM (Pro-GNN) with two baselines, which are Pro-GNN trained on the original attacked graph and Pro-GNN trained on a graph augmented with synthetic nodes generated by a vanilla DDPM. The vanilla DDPM baseline synthesizes labeled synthetic nodes via conditional DDPM and connects them using k-nearest neighbors (KNN), as detailed in Section F.3. The study is performed on Cora under Metattack with a perturbation rate of 25%. As shown in Table 31, the vanilla DDPM even degrades performance, confirming that RGDM's gains stem from the generated faithful synthetic graph structures, rather than simply increasing the number of training nodes/edges.

Table 31: Performance comparison of Pro-GNN on augmented graph with synthetic graph structures generated by DDPM and RGDM. The study is performed on Cora under Metattack with a perturbation rate of 25%.

Method	Accuracy (%)
Pro-GNN	69.7
DDPM (Pro-GNN)	68.2
RGDM (Pro-GNN)	**72.3**

F.16 Study on the Impact of Label Ratio

To assess the robustness of RGDM (Pro-GNN) under varying levels of supervision, we evaluate performance on the Cora dataset under a 25% Metattack perturbation rate while varying the labeled training ratio. In this study, we consider both sparse label regimes (2.5% to 10%) and denser settings (20% to 60%), with 10% of nodes fixed for validation and the remainder used for testing. As shown in Table 32, RGDM consistently outperforms the Pro-GNN baseline across all label ratios. Under extremely limited supervision (2.5%–10%), RGDM maintains strong performance, indicating its utility as a data augmentation strategy in low-resource scenarios. As the label ratio increases, RGDM continues to achieve better performance, demonstrating its robustness and effectiveness even with more abundant training data. These results confirm that the improvements of RGDM are not confined to a specific supervision regime and generalize well across both low- and high-label density settings.

Table 32: Performance comparison of Pro-GNN and RGDM (Pro-GNN) under different label ratios on Cora under 25% Metattack.

Label Ratio	2.5%	5%	7.5%	10%	20%	40%	60%
Pro-GNN	62.5	65.6	67.5	69.7	73.8	75.2	76.6
RGDM (Pro-GNN)	**64.8**	**68.0**	**69.3**	**72.3**	**75.9**	**77.6**	**78.4**

F.17 Ablation Study on the Sparse Hierarchical Edge Decoder

To further validate the effectiveness of the proposed Sparse Hierarchical Edge Decoder (SHED) in generating robust synthetic edges, we conduct an ablation study comparing SHED against the Bi-Level Neighborhood Decoder (BLND) used in DoG [19]. In particular, we replace SHED in RGDM with the BLND module and denote the resulting model as RGDM (BLND). Both models are evaluated under the Metattack [51] with a perturbation rate of 25%, using SG-GSR [33] as the backbone GNN. The experiments are conducted on Cora, Citeseer, Polblogs, and Pubmed. As shown in Table 33, RGDM equipped with SHED consistently outperforms RGDM (BLND) across all benchmarks. We also report the hierarchical reconstruction variant from LR-FGDM [20], denoted RGDM (HGAE). With SG-GSR fixed, replacing HGAE by SHED improves accuracy by $2.3/2.8/2.1/2.6$ points. Replacing BLND by SHED improves it by $1.7/2.2/1.6/2.0$ points. This reconstruction comparison complements the full system GCN baselines in Table 4 and the sequential component experiment in Table 5.

Table 33: Hierarchical reconstruction comparison: SHED, BLND from DoG [19], and the HGAE variant from LR-FGDM [20]. SG-GSR is used as the base GNN under Metattack with 25% perturbation.

Dataset	Cora	Citeseer	Polblogs	Pubmed
RGDM (HGAE from LR-FGDM)	76.8	72.5	86.7	86.0
RGDM (BLND)	77.4	73.1	87.2	86.6
RGDM	**79.1**	**75.3**	**88.8**	**88.6**

F.18 Adaptive Attack Evaluation

The experiments in Section 4.1 follow the commonly used attack-first-then-defend protocol, where the graph is first perturbed by adversarial attacks and the defense is then applied to the attacked graph. Although this protocol is widely adopted in graph adversarial robustness, it may not fully reflect a defense-aware adversary. In particular, an adaptive adversary may be aware of the RGDM pipeline and attack the augmented graph after the synthetic graph structures have been generated. To address this concern, we further evaluate RGDM under an adaptive attack setting, following the adaptive evaluation protocol in prior graph robustness studies [91].

Settings. We consider an adaptive version of Metattack [51] with a perturbation rate of 25%. Different from the standard Metattack evaluation in Section 4.1, where the attack is performed before RGDM augmentation, the adaptive attacker perturbs the final augmented graph $\mathcal{G}_{\text{aug}}$ used for training the downstream GNN. The attacker is aware of the full RGDM pipeline, including the generation of synthetic graph structures, edge selection, low-rank regularization, and the downstream SG-GSR classifier. For a fair comparison, the perturbation budget is fixed to the same number of edge perturbations as the 25% Metattack on the original graph. We repeat each experiment 10 times with different random seeds and report the mean and standard deviation of node classification accuracy.

Let $n_{\text{aug}} := N + N'$ and let $\mathbf{P} \in \{0,1\}^{n_{\text{aug}} \times n_{\text{aug}}}$ be a symmetric zero diagonal flip matrix. The defense aware attack searches over the admissible set in Equation (12):

$$\mathcal{P}_b := \left\{ \mathbf{P} : \mathbf{P} = \mathbf{P}^\top,\ \mathbf{P}_{ii} = 0,\ \sum_{1 \le i < j \le n_{\text{aug}}} \mathbf{P}_{ij} \le b \right\}, \qquad \mathbf{A}_{\text{aug}}^{\text{adv}} := \mathbf{A}_{\text{aug}} \oplus \mathbf{P}. \tag{12}$$

Here $\oplus$ denotes entrywise binary edge flipping. In Equation (12), b is the same absolute number of flips used by the 25% attack on the corresponding original graph, not 25% of the larger augmented edge count. Candidate flips may involve original or synthetic nodes. The attacker knows the RGAE, edge selection mechanism, latent regularizer, LDM, and downstream classifier, and optimizes the

Metattack surrogate against downstream retraining on the perturbed augmented graph. The first variant attacks the realized augmented graph. The second additionally differentiates its meta gradient through the relaxed edge selection mask: the forward pass thresholds $\widehat{\mathbf{S}}_{ij} = \sigma(\boldsymbol{\theta}_{ij}/\tau)$ at $1/2$, and the backward pass uses its straight through gradient with $\tau = 0.5$. This is a gradient approximation to the discrete selection operation, not a proof of globally optimal adversarial perturbations. Knowledge of the LDM does not by itself establish differentiation through its entire stochastic sampling trajectory. The reported differentiable variant specifically tests the edge selection mechanism.

Table 34: Adaptive attack evaluation under Metattack [51] with 25% perturbation rate. The adaptive attacker is aware of the defense and attacks the final augmented graph used for training SG-GSR. The p-values are computed between SG-GSR and RGDM (SG-GSR).

Dataset	SG-GSR	RGDM (SG-GSR)	p-value
Cora	74.8 ± 2.4	$\mathbf{77.8 \pm 1.8}$	0.006
Citeseer	69.5 ± 2.6	$\mathbf{73.4 \pm 2.1}$	0.003
Polblogs	84.7 ± 2.5	$\mathbf{87.6 \pm 2.2}$	0.008
Pubmed	84.9 ± 2.7	$\mathbf{87.2 \pm 2.4}$	0.012

As shown in Table 34, RGDM consistently improves the robustness of SG-GSR even when the attacker is aware of the defense pipeline and attacks the final augmented graph. For example, RGDM (SG-GSR) significantly outperforms SG-GSR by 3.9% on Citeseer under adaptive Metattack with a 25% perturbation rate. The p-values are smaller than 0.05 on all datasets, demonstrating that the improvements of RGDM over SG-GSR remain statistically significant under adaptive attacks. These results indicate that the robustness gains of RGDM are not merely caused by the attack-first-then-defend workflow order. Instead, the generated synthetic graph structures, together with edge selection and low-rank regularization, provide additional robust supervision that improves the downstream GNN even under defense-aware perturbations.

Differentiating through edge selection. Table 35 reports the adaptive experiment that differentiates through edge selection. Its SG-GSR / RGDM gains are $2.4/3.2/2.2/1.7$ points on Cora/Citeseer/Polblogs/Pubmed, compared with $3.0/3.9/2.9/2.3$ points for the final graph attack in Table 34. The second variant reports mean accuracies. It does not report standard deviations or p values. Both variants are summarized in the main text (Table 7). The positive gains support robustness to these tested attacks at this fixed budget, not certification against arbitrary white box attacks.

Table 35: Adaptive Metattack differentiating through the relaxed edge selection mask with the same absolute edge flip budget and $\tau = 0.5$. Mean accuracy (%).

Dataset	SG-GSR	RGDM (SG-GSR)	Gain (points)
Cora	74.5	76.9	2.4
Citeseer	69.2	72.4	3.2
Polblogs	84.4	86.6	2.2
Pubmed	84.6	86.3	1.7

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions and scope of the paper are discussed in Section 1 of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the limitation that the proposed method only applies to node-level data generation on graph datasets in the introduction, Section 1, of this paper.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The theoretical result is detailed in Theorem 3.1 and proved in Section A of the supplementary.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The anonymous link to the code is attached at the end of the abstract. Implementation details for the experiments are stated in Section E.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The anonymous link to the code is attached at the end of the abstract. Implementation details for the experiments are stated in Section E. The datasets used in the experiments are public benchmarks. Detailed information on the datasets is provided in Section E.1.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental settings are specified in Section E. The settings for adversarial attacks adopted in our experiments are provided in Section E.3. The datasets used in the experiments are public benchmarks. Detailed information on the datasets is provided in Section E.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Standard errors for the results of our method and baseline methods are provided in the tables presenting the experimental results in Section 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The computing resources used in the experiments of this paper are specified in Section E of our paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have read NeurIPS Code of Ethics in https://neurips.cc/public/EthicsGuidelines and confirm that the research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

Justification: The experiments performed do not have societal impacts, as we did not perform any experiments on real-world data collected from society. The experiments are performed by using simulated attack methods.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This paper poses no such risks for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper performs experiments on widely used public benchmarks, which are properly cited in Section E.1.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We do not use LLM in this paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.