CiDer: A Black-box Approach to Classify Node with Certified Robustness Guarantees

Xiaoyu Liang*, Haohua Du*, Wen Ma*, Ye Tian†, and Xiaoya Xu‡
*Beihang University, China
†University of Science and Technology of China, China
‡Institute 706 The Second Academy of China Aerospace Science & Industry Corp., China
Email: {xiaoyuliang, duhaohua, marvin2022}@buaa.edu.cn, tianye@mail.ustc.edu.cn, xuxiaoya@buaa.edu.cn

***Abstract*—Due to the outstanding performance of graph node classification in tasks such as detecting illegal nodes in transaction networks, adversarial attacks aiming to perturb classification results have proliferated. Although current defenses based on randomized smoothing have shown some effectiveness, these approaches still require modifications to the classification model to ensure accuracy.**

Here, we propose a novel approach – CiDer, that theoretically guarantees the robustness of graph node classification results in a black-box setting, which means no assumptions on the form of attack and the classification model. The key idea behind our approach is to leverage the denoise capability of diffusion models on features to perform adversarial purification on the data. We then prove this stochastic purification method can ensure certified robustness under certain attack budgets. Our extensive experiments corroborate our theory and demonstrate that node classifiers worked with CiDer achieve significantly superior performance compared to state-of-the-art, e.g., the accuracy improves by 7% on Cora and the optimal result improves by 30% on PubMed.

***Index Terms*—graph node classification, adversarial attack, diffusion model**

I. Introduction

Graph node classifiers based on deep learning are susceptible to adversarial attacks: adding imperceptible perturbations to the graph input can mislead a well-trained classifier, causing it to produce incorrect classification results. For example, malicious adversaries may cause graph node classifiers to misclassify illegal nodes in transaction networks [1].

There have been many works on defending node classifiers like Graph Neural Network (GNN) against such adversarial attacks. The empirical defense methods are effective for some attacks, but fall behind adversarial training methods, leading to an unstop game [2]–[4]. To solve the attack-and-defense dilemma, Randomized Smoothing (RS)-based methods [5]–[7] no longer focus on the attacking method but on the effect, and they achieve defense against unknown attacks by tuning any classifier into a new one that is certifiably robust to arbitrary adversarial perturbations. Specifically, RS boils down to adding random noise to the input and smoothing the decision boundary over multiple noisy samples. Nevertheless, since RS-based methods operate the classifier, they have two major drawbacks. First, the defense requires fine-tuning the classifier under different attack budgets, leading to inefficiencies. Second, the classifier fine-tuned on randomized data or adversarial examples suffers from accuracy degradation on benign examples.

*Haohua Du is the corresponding author. This research is partially supported by National Key R&D Program of China under Grant No. 2022YFB3104400, China National Natural Science Foundation with No. 62102016.

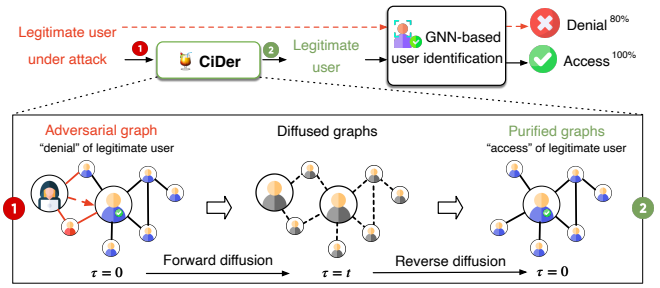

Fig. 1. **CiDer illustration**: A legitimate user under attack (malicious modification on graph structure and attribute) may be denied by GNN-based identification systems. CiDer gives a certified robust identification label by removing adversarial perturbation with a discrete diffusion model.

Here, we propose a novel graph node classification defense approach – **CiDer** (Certified robustness via Diffusion model on graph), which provides node classification with certifiable robustness, without modifying the classifier or obtaining any attack information. The key idea of CiDer is to carefully select the core features of different node types, ensuring that as long as the attacker's perturbation does not disrupt these core features, the classifier can identify the node types with certified robustness. To find such core features, we rely on the powerful purification capabilities of diffusion models [8]. Diffusion models consist of two processes: 1) a forward diffusion process, which gradually adds noise to the input, transforming the data into noise; 2) a reverse generative process, which starts from the noise and generates data step-by-step by denoising. During the generative process, diffusion models purify adversarial perturbed features by refining the data. The high quality and diversity of the generated outputs ensure that the purified features closely follow the original distribution of clean data.

However, realizing CiDer is not trivial due to the following challenges.

TABLE I
GRAPH ADVERSARIAL DEFENSES

Strategy	Method	Model-agnostic	Attack-agnostic	Budget-agnostic
Adversarial training	[3] [9] [10] [11] [12] [13] [14]	◐	○	○
Attack detection	[15] [16] [4] [17] [18]	●	○	●
Specific certification	[19] [20] [21]	○	●	○
Randomized smoothing	[7] [22] [23] [24] [25]	◐	●	○
Denoised smoothing	CiDer	●	●	●

1) Although the stochasticity in diffusion models can defend against stochastic attacks, this randomness poses challenges for proving certified robustness. It is hard to reasonably constrain the forward and reverse processes of the diffusion model while ensuring that the outcomes of such stochastic processes meet the requirements of deterministic robustness.
2) The features in graph node classification are not inherently intuitive and typically require embedding to capture the diversity of connectivity patterns. Generally, different classifiers employ various embedding methods, meaning the contributions of graph structure and node attributes to the features are different. Consequently, it is hard to purify implicit node features by solely processing the explicit graph structure and node attributes.
3) Diffusion models are computationally inefficient on large graphs due to the quadratic growth of potential edges and exponential growth of node attribute combination w.r.t. the number of nodes.

Solutions. 1) We constrain the stochasticity by minimizing the maximal perturbation at each step of the diffusion process, which can be regarded as a special instance of RS. It thus can certify the robustness under different attack budgets. 2) Leveraging the relatively independent characteristics of graph structure and node attributes, we designed a stepwise diffusion and denoising algorithm based on the multiplication theorem, which independently analyzes the impact of graph structure and node attributes on the features. 3) We use subgraph extraction as a pre-processing step to downsize the input graph, which can effectively enhance computational efficiency.

The main contributions in **CiDer** summarize as follows:

- We design CiDer, which can defend against any malicious attacker on node classification with robustness certificates for any black-box GNN models without retraining or fine-tuning.
- Unlike traditional RS-based approaches, we defend via the perspective of data by *Graph Joint Diffuser* based on diffusion model [8], which extracts core features to mitigate the impact of unknown attacks. We also introduce a subgraph-based method to reduce the computation cost and further enhance CiDer's applicability.
- We implemented and evaluated our approaches on multiple real-world graph datasets. The extensive experimental results show that CiDer can provide robustness certificates for a wide range of GNNs. Compared to the state-of-the-art, the accuracy improves by 7% on Cora and the optimal result improves by 30% on PubMed.

II. MOTIVATION AND RELATED WORKS

Graph-structured data has been widely used in domains like social networks [26], transaction networks [1], recommendation systems [27]. In these fields, GNNs serve as essential tools for tasks such as node classification [28]–[30], graph classification [31]. However, GNN is proved to be vulnerable to adversarial attack. [32]. This paper focuses on graph modification attack where unnoticeable perturbation may lead to incorrect predicted label [33]–[36]. Existing defenses can be categorized into *heuristic defense* and *certified defense*. Heuristic defenses typically improve GNN robustness by analyzing attack patterns. At the model level, adversarial training [10] improves model robustness by designing a specialized training process. At the data level, attack detection methods [4] remove adversarial perturbations to revert the malicious changes and obtain a 'cleaner' graph.

Though heuristic defense proved to be effective in defending against some specific attacks, numerous new attacks may be developed to invalidate them. In contrast to their *unprovable* robustness, certified defense generate a robustness certificate to provide *provable* robust defense. Robustness certificates for GNN can be mainly categorized into two principles: model-specific and model-agnostic.

Model-specific [21] certificates are designed for a specific class of GNN models (e.g., 2-layer Graph Convolutional Network) and a specific task (e.g. node-level classification). A common theme is to phrase certification as a constrained optimization problem [20], [21], [37].

To provide a certificate for GNN with arbitrary architecture, randomized smoothing methods [7], [22], [23] apply randomization on input sample multiple times, then obtain label by majority vote and obtain certificate by classification margin. Putting RS into practice usually involves conducting robust training. Even though RS is so-called *black-box*, the luxury training process implies that the model parameters are accessible. Additionally, since the samples used for robust training follow specific patterns, making RS budget-aware.

Table I summarizes adversarial defenses on graphs. Model-agnostic refers to applicability to black-box models, including architectures and parameters. Attack-agnostic ensures defense against any admissible attack with certifiable robustness. Budget-agnostic adapts to varying attack budgets (perturbation magnitude). Our goal is to develop a method to defend GNNs against unknown attacks, models, and budgets.

III. PROBLEM FORMULATION

In this section, we first introduce the task of semi-supervised node classification, followed by our threat model and defense

goal. Finally, we demonstrate the key idea we employed - Discrete Denoising Diffusion Probabilistic model (D3PM) [8].

A. Node Classification with GNN

We consider the task of node classification in a large graph $G = (X, A)$ with N vertices, where $X \in \{0,1\}^{N\times D}$ represents binary attribute matrix and $A \in \{0,1\}^{N\times N}$ is undirected adjacent matrix. The objective of the node classification GNN is to learn a function $f_\theta : \mathcal{V} \to \{1, 2 \dots K\}$, mapping each node v to a class label y.

The majority of node classification GNN is based on the message-passing framework. The basic principle of message passing is that the representation of a node is a (recursive) local *aggregation* of information from its neighboring nodes. Specifically, one layer of Graph Convolutional Network (GCN) [28] and Graph Attention Network (GAT) [29] only aggregates the message from the 1-hop neighbor of the target node.

B. Security Model

CiDer does not consider specific attack strategies used by adversaries but focuses solely on the adversary's attack budget. Under the given threat model, it provides certified defense through robustness certificates.

Threat Model. We consider the situation of graph modification evasion attacks. Given a classifier $f_\theta(\cdot)$, the malicious attacker can carefully craft an adversarial example $\tilde{G} = (\tilde{X}, \tilde{A})$ with perturbation δ to change the predicted class for target node v. Assuming a graph modification attacker that can craft an adversarial graph $\tilde{G}$ by modifying the graph structure and node attribute, we now define the attack budget.

We assume the attacker can modify the benign sample with $\delta = (\delta_X, \delta_A)$. The threat model constraints are to limit the change on node attribute and graph structure:

$$\Delta = \{(\delta_X, \delta_A) : ||\delta_X^+||_0 \le \Delta_X^+, ||\delta_X^-||_0 \le \Delta_X^-, \\ ||\delta_A^+||_0 \le \Delta_A^+, ||\delta_A^-||_0 \le \Delta_A^-\}. \quad (1)$$

$\Delta = (\Delta_X, \Delta_A)$ specifies the attack budget, which includes all the perturbations that flip at most Δ_A^+ 0-bit to 1-bit and at most Δ_A^- 1-bit to 0-bit in the adjacent matrix and similarly for the attribute matrix. For the graph with binary node attributes and undirected edges, the graph perturbed by $\delta \in \Delta$ is equivalent to having at most Δ_A^+ additional edges, Δ_A^- fewer edges, Δ_X^+ additional attributes and Δ_X^- fewer attributes.

Robustness Certificate. To build a provable defense, our goal is to derive a robustness certificate for any black-box model. Given a perturbation constraint (attack budget Δ), the *robustness certificate* provides a lower bound of accuracy regardless of the concrete attack algorithms. Considering the node classification task, we now define the robustness certificate for target node v following [7], [24].

$$\rho_{G,\tilde{G}}(p, y^*) = \min_{\bar{f}_\theta : \Pr(\bar{f}_\theta(v|G)=y^*)=p} \Pr(\bar{f}_\theta(v|\tilde{G}) = y^*), \quad (2)$$

where y^* is the predicted label for v on benign graph sample G and $\tilde{G}$ is a neighbouring graph point of G. We have that $\rho_{G,\tilde{G}}(p, y^*) \le \Pr(f_\theta(v|\tilde{G}) = y^*)$ is a lower bound on the probability that a v in the perturbed graph $\tilde{G}$ is classified as y^*. Now, given an attack budget Δ, if it holds that:

$$\min_{\tilde{G}} \rho_G(p, y^*) > \max_{\tilde{G}} \Pr(\bar{f}_\theta(v|\tilde{G}) = y \ne y^*), \quad (3)$$

where $\tilde{G} = G + \delta, \delta \in \Delta$, then we can guarantee that the probability of v in $\tilde{G}$ classified as y^* is higher than any other labels. This shows that the predicted label for the target node is certifiably robust.

Since calculating the probability $p_y(G)$ of predicted class y for sample G is often intractable, the common approach is to compute the lower bound $\underline{p_{y^*}}(G)$ and upper bound $\overline{p_{y\ne y^*}}(G)$ based on the Clopper-Pearson Bernoulli confidence interval using Monte-Carlo samples from $\tilde{G} \in G + \Delta$ [6], [7]. The certificate guarantees that the prediction class for v is certifiably robust under the perturbation set Δ with a certain confidence level.

C. Discrete Denoising Diffusion Probabilistic Model

In this paper, we propose a diffusion model based on the D3PM [8] to improve the robustness of the GNN.

D3PM [8] is a denoising diffusion probabilistic model for discrete data (e.g. graph), which consists of two main components: a discrete noise model and a denoising neural network. Let $\text{Cat}(\cdot)$ denote the categorical distribution. The diffusion process on discrete categorical variable Z is defined as:

$$q(Z_t|Z_{t-1}) = \text{Cat}(Z_t; p = Z_{t-1}Q_t), \quad (4)$$

$$q(Z_t|Z_0) = \text{Cat}(Z_t; p = Z_0\bar{Q}_t), \quad (5)$$

where Q_t is the categorical transition matrix and $\bar{Q}_t = Q_1Q_2 \cdots Q_t$. The denoising neural network $p_\theta(\bar{Z}_0|Z_t)$ is trained to predict the clean data from timestamp t to 0 step by step, and the reverse generative process is denoted as:

$$p_\theta(Z_{t-1}|Z_t) = \sum_{\bar{Z}_0} q(Z_{t-1}|Z_t, \bar{Z}_0)\bar{p}_\theta(\bar{Z}_0|Z_t). \quad (6)$$

There are existing methods applying D3PM to graph data [38]–[40]. However, they are either not applicable to large-scale graph data, or cannot efficiently applied to node classification tasks.

IV. CiDer, Certified Robustness for Node Classification

In this section, we introduce **CiDer**, an approach that can provide certified adversarial robustness to graph modification attacks for any off-the-shelf classifiers.

A. Overview

As illustrated in Fig. 2, CiDer has three major components – the subgraph extractor, the denoising graph joint diffuser, and the certificate generator. The subgraph extractor preprocesses the data to reduce CiDer's unnecessary computational load, thereby improving overall system efficiency. The diffuser purifies the features of the processed data, mitigating the impact of adversarial perturbations and ensuring the classifier's

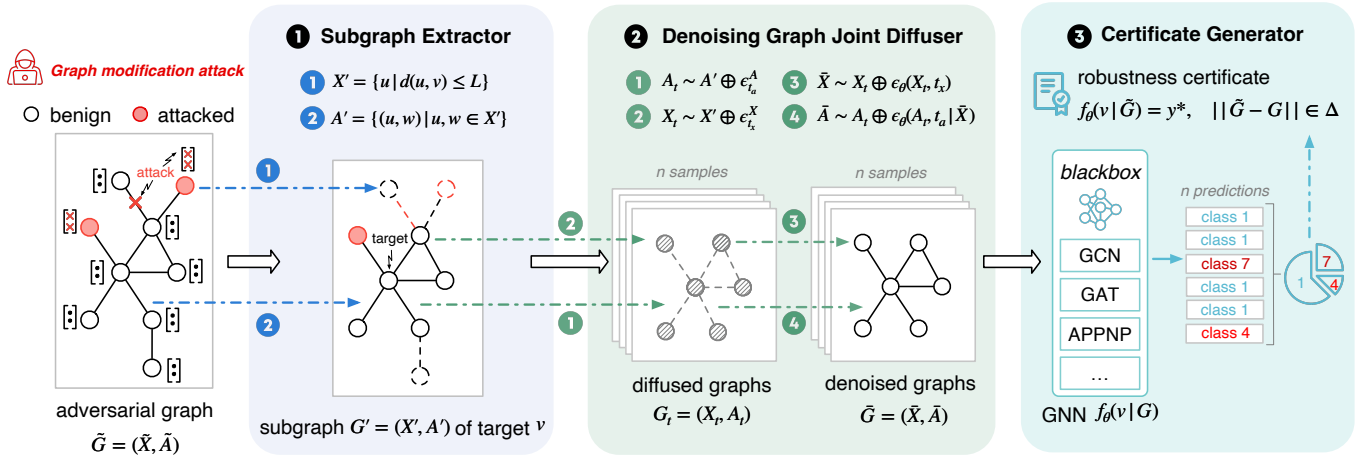

Fig. 2. **CiDer**: The adversarial graph $\tilde{G}$ go through three steps: L-hop subgraph is extracted by subgraph extractor. Graph joint diffuser purifies perturbation to get a denoised graph. The robust label and certificate are reported according to the majority vote of the denoised subgraphs through arbitrary GNN.

performance remains unaffected. Finally, the generator provides robustness certificates and provable robust labels through theoretical proof and parameter analysis.

B. Subgraph Extractor

Given a large graph $\tilde{G}$ and target node v, CiDer first extracts subgraphs around v to downsize the input of the diffusion model.

Specifically, we extract the L-hop subgraph around v. Here L represents the number of layers of the GNN. This is due to the fact that under the message-passing mechanism, the prediction of the node is only affected by the direct neighbor [28], [29]. Let $d(\cdot)$ denote the hop distance between nodes. The extracted L-hop subgraph is $G' = (X', A')$ with X' and A' as follows:

$$X' = \{u \in X | d(u,v) \le L\}, \tag{7}$$

$$A' = \{(u,w) \in A | u,w \in X'\}. \tag{8}$$

C. Denoising Graph Joint Diffuser

Considering a subgraph that may contain graph modification perturbation δ, we aim to make a robust prediction for target node v with any classifier $f_\theta(\cdot)$. To do so, we introduce Graph Joint Diffuser as a graph denoiser. The core idea is to randomize the adversarial graph with sparsity-aware noise and denoise it before making a prediction.

Randomized Smoothing. We obtain a robustness certificate by the Randomized Smoothing (RS) framework, which can certify the robustness of arbitrary classifiers against any admissible adversarial examples.

Given an input graph and the target node, RS obtains categorical prediction with a smooth version of base classifier $f(\cdot)$:

$$g_\theta(v|\tilde{G}) := \arg_y \max \Pr_{\tilde{G} \sim \phi(G)} [f_{\theta^*}(v|\tilde{G}) = y]. \tag{9}$$

$\phi(G)$ is the RS scheme on a graph. $g_\theta(\cdot)$ returns the most likely class (the majority vote) with base classifier $f(\cdot)$ when the input is randomized with $\phi(\cdot)$.

Ref. [7] builds an efficient robustness certificate for discrete data by sparsity-aware RS called sparse smoothing. We adopt their randomization scheme $\phi(\cdot)$:

$$\Pr(\phi(Z) \neq Z) = p_+^{(1-z)} p_-^z, \tag{10}$$

where $Z \in \{X, A\}$, representing a joint randomization on node attribute and graph structure. $\phi(\cdot)$ flips the bit $z = 1$ to 0 with probability p_- and flips the bit $z = 0$ to 1 with probability p_+. The flipping represents an edge/attribute addition/deletion. Using this randomization scheme, sparse smoothing obtains a joint certification for the structure and the attributes.

The scheme $\phi(\cdot)$ is equivalent to first drawing a random noise sample $\epsilon^Z \sim \text{Ber}(p = p_+^{(1-z)} p_-^z)$ from a Bernoulli distribution with $p = p_-$ if $z = 1$ and $p = p_+$ if $z = 0$. Then $\phi(\cdot)$ randomizes the variable as $\phi(Z) = Z \oplus \epsilon^Z$, where $\oplus$ is the 'exclusive or' binary operation.

To retain accuracy, they train the base classifier with $\phi(\cdot)$ randomized input which is akin to data augmentation with Bernoulli noise. The robust training process makes sparse smoothing model-parameter-aware and budget-aware. The problem remains whether we can obtain state-of-art certified accuracy without white-box access to the base classifier.

Denoised Smoothing. We avoid redundant retraining or fine-tuning by discrete denoised smoothing. The $f_{\theta^*}(\cdot)$ in Eq. 9 is composed of a custom-trained denoiser combined with an *off-the-shelf* base classifier:

$$f_{\theta^*}(v|\tilde{G}) := f_\theta(v|\mathcal{D}(\tilde{G})). \tag{11}$$

$\mathcal{D}(\cdot)$ is a denoiser that modifies the noisy input graph to the benign data distribution. Our approach avoids using noisy data or adversarial examples to train the base classifier, but instead, uses a denoising step before passing samples through $f_\theta(\cdot)$ trained on clean data. The decoupling of graph randomization and graph prediction makes CiDer a model-agnostic and budget-agnostic approach.

Joint Discrete Denoise Diffusion. Our intantiation of the denoiser $\mathcal{D}(\cdot)$ is based on D3PM [8], which can model the

joint distribution of node attribute and graph structure. We present *Graph Joint Diffuser*, a single model to capture the complex interdependent correlation between X and A.

Formally, we have graph data sampled from the distribution $q(G_0) = q(X_0, A_0)$, where $G_0 = G'$ is the output from the subgraph extractor. Let scalar β_t denote the noise schedule and $\alpha_t = 1 - \beta_t$ like general diffusion model notations. For attribute matrix X and adjacent matrix A with $\{0, 1\}$ entries, we propose to use the transition matrix Q_t in Eq. 4 as:

$$Q_t^X = \alpha_t \mathbf{I} + \beta_t \mathbf{1}\mathbf{m}^{\mathbf{X}}, \tag{12}$$

$$Q_t^A = \alpha_t \mathbf{I} + \beta_t \mathbf{1}\mathbf{m}^{\mathbf{A}}, \tag{13}$$

where $\mathbf{I}^{2\times 2}$ is the identity matrix, $\mathbf{1}^{2\times 1}$ is the one-valued vector of dimension, $\mathbf{m}^{\mathbf{Z}}$ is the row vector representing the empirical marginal distribution of of variable Z. Since $(\mathbf{1}\mathbf{m})^2 = \mathbf{1}\mathbf{m}$, we still have $\bar{Q}_t = \bar{\alpha}_t \mathbf{I} + \bar{\beta}_t \mathbf{1}\mathbf{m}$ for $\bar{\alpha}_t = \prod_{\tau=1}^{t} \alpha_\tau$ and $\bar{\beta}_t = 1 - \bar{\alpha}_t$. With this model, the probability of 1-bit flipping to 0-bit is proportional to the probability of 0-bit in the training data, and vice versa. In the context of sparse graphs, this means that jumping to the state of non-edge and non-feature will be very likely, for the 0-bit is far more than the 1-bit in attribute and adjacent matrix. Hence, this marginal transition matrix can preserve the sparsity pattern in the graph.

The noise schedule β_t and the diffusion steps T can be set differently for X and A to enable an asynchronous diffusion process. The heuristic reason behind this is that attributes and structure contribute differently when it comes to node classification tasks. Using an asynchronous noise schedule, different noise granularities can be set for X and A.

This discrete diffusion process is equivalent to adding a random noise $\epsilon_t \sim \text{Ber}(p = (\beta_t \mathbf{m_1})^{1-z}(\beta_t \mathbf{m_0})^z)$ from a Bernoulli distribution, which is identical to the randomization scheme in [7]. Here, $\mathbf{m_1}$ and $\mathbf{m_0}$ represent the marginal distribution of 1-bit and 0-bit respectively. Following the classic Gaussian noise diffusion notation, the discrete diffusion process can be presented as:

$$\begin{aligned} &q(X_{t_x}, A_{t_a} | X_0, A_0) \\ &= \text{Cat}(X_{t_x}, A_{t_a}; p_X = X_0 \oplus \epsilon_{t_x}^X, p_A = A_0 \oplus \epsilon_{t_a}^A), \end{aligned} \tag{14}$$

where $\epsilon_{t_z}^Z \sim \text{Ber}(p = (\bar{\beta}_{t_z} \mathbf{m_1}^Z)^{1-z}(\bar{\beta}_{t_z} \mathbf{m_0}^Z)^z)$ for $Z \in \{X, A\}$. The asynchronous noise $\epsilon_{t_x}^X$ is the discrete noise added to the attribute matrix at timestamp t_x and $\epsilon_{t_a}^A$ is the discrete noise added to the adjacent matrix at timestamp t_a.

The reverse process in Eq. 6 is approximated by a denoise neural network, which is intended to estimate the injected Bernoulli noise (diffusion model actually predicts Z_0, and the two terms are equivalent). Typically, the noise prediction network $\epsilon_\theta^Z(Z_{t_z}, t_z)$ is adopted to predict the noise $\epsilon_{t_z}^Z$ added to Z at time step t_z, where t_z is uniformly sampled from $\{1, \cdots, T_z\}$.

As for modeling the joint distribution of X and A, we aims at estimating the conditional expectation over the joint noise $\mathbb{E}[\epsilon_{t_x}^X, \epsilon_{t_a}^A | X_{t_x}, A_{t_a}]$. We employ a graph joint noise prediction network $\epsilon_\theta(X_{t_x}, A_{t_a}, t_x, t_a)$ to predict the Bernoulli noise injected to X and A at timestamp t_x and t_a. The noise predictor is trained by minimizing the regression loss:

$$\mathbb{E}_{X_0, A_0, \epsilon_{t_x}^X, \epsilon_{t_a}^A, t_x, t_a} || \epsilon_\theta(X_{t_x}, A_{t_a}, t_x, t_a) - [\epsilon_{t_x}^X, \epsilon_{t_a}^A] ||. \tag{15}$$

To model the complex relationship, we implemented the joint noise predictor by asynchronous denoising X and A. The denoising network $\epsilon_\theta(X_{t_x}, A_{t_a}, t_x, t_a)$ consists of two sub-noise predictor: $\epsilon_\theta^X(X_{t_x}, t_x)$ and $\epsilon_\theta^A(A_{t_a}, t_a | \bar{X}_0)$. Specifically, we first use a multi-layer perceptron to predict attribute noise and then use a message-passing neural network to predict structure noise conditioned on denoised attribute $\bar{X}_0$.

In practice, we train joint noise predictor by minimizing cross-entropy loss between the predicted attribute and structure probabilities $\bar{p}^G = (\bar{p}^X, \bar{p}^A)$ satisfying $\bar{p}^X = X_{t_x} \oplus \Pr(\epsilon_\theta(X_{t_x}, t_x))$ and $\bar{p}^A = A_{t_a} \oplus \Pr(\epsilon_\theta(A_{t_a}, t_a | \bar{X}_0))$ where $\Pr(\epsilon_\theta(\cdot))$ is the predicted probability of noise. The joint cross-entropy loss is:

$$\text{Loss}(\bar{p}^G, G) = \ell_{\text{CE}}(\bar{p}^X, X) + \lambda \ell_{\text{CE}}(\bar{p}^A, A), \tag{16}$$

where λ controls the relative importance of structure over attributes. The Graph Joint Diffuser model is visualized in Fig. 3.

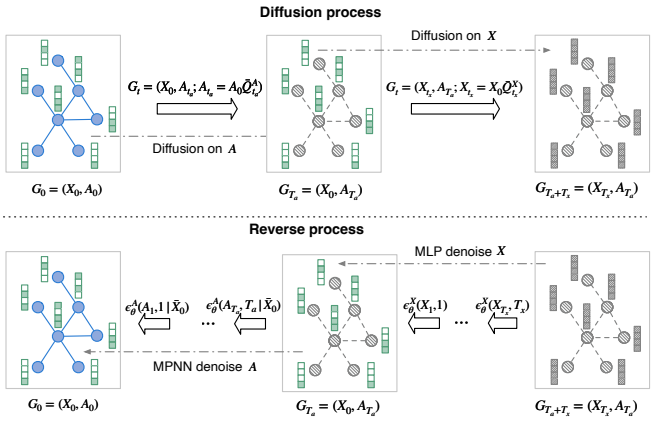

Fig. 3. **Graph joint diffuser**: We apply asynchronous noise on node attribute and graph structure and use a joint noise predictor to reconstruct the clean sample. We use MLP to predict attribute noise and MPNN to predict structure noise conditioned on reconstructed attributes.

Algorithm. CiDer makes use of the Graph Joint Diffuser not in the standard training and sampling way, instead it's sufficient to only focus on the training process where the "denoising" happens. Given timestamps t_x and t_a, the model is trained to reconstruct the clean sample from noisy input X_{t_x} and A_{t_a}. The training algorithm is illustrated in Algotithm 1.

Given a Graph Joint Diffuser described above, the randomized noise can be purified by simply passing the sample and corresponding timestamp through the diffusion model. It is important to note that we do not follow the standard sampling method that reconstructs the sample from timestamp T to 0 step by step, instead, we use *one-shot* denoising like [41]. We reconstruct sample G_0 by running the reverse step once and using the one-step purified graph for classification.

Algorithm 1: Training Graph Joint Diffuser

Input : Graph $G = (X, A)$

$t_x \sim \text{Uniform}(1, \dots, T_x)$
$t_a \sim \text{Uniform}(1, \dots, T_a)$
Sample $G_t \sim (X_0 \oplus \epsilon^X_{t_x}) \times (A_0 \oplus \epsilon^A_{t_a})$
/* Get $\bar{p}^X$,$\bar{p}^A$ with joint noise predictor. */
$\bar{p}^X = X_{t_x} \oplus \Pr(\epsilon_\theta(X_{t_x}, t_x))$
$\bar{p}^A = A_{t_a} \oplus \Pr(\epsilon_\theta(A_{t_a}, t_a | \bar{X}_0))$
Take gradient descent step on Eq. 16

After denoising with the diffuser, the purified graph can be fed into GNN. Decoupling the denoising process from the GNN can create a *one-model-fits-all* denoiser. This approach can denoise any noise scale, any noise scale combination of attribute and structure, and be applicable to any GNN.

D. Certificate Generator

Following [6], [7], we generate a robustness certificate using Monte-Carlo sampling. Our key idea is to divide the graph space into regions such that we can apply the Neyman-Pearson Lemma [42] to derive the lower bound of the most-likely class.

Given randomized graph $\phi(G)$, the base node classifier f predicts y^*. The key idea of certification is to guarantee that when taking $\phi(\tilde{G}), \tilde{G} = G + \delta$ as input, f gives y^* as a prediction as well. The goal is to find the maximum perturbation Δ such that:

$$\Pr(f(\phi(\tilde{G})) = y^*) > \Pr(f(\phi(\tilde{G})) = y \neq y^*). \tag{17}$$

Since neural network f is non-linear, the probability calculation is often intractable. To address this difficulty, the common approach is to derive the lower bound for $\Pr(f(\phi(\tilde{G})) = y^*)$ and the upper bound for $\Pr(f(\phi(\tilde{G})) = y)$ [6], [25].

Specifically, if the graph space can be partitioned into regions $\mathcal{R}_i$ and the probability of $\phi(\tilde{G}) \in \mathcal{R}_i$ can be computed efficiently for any Δ, then by searching Δ under condition Eq. 17, we can give a certificate under perturbation size Δ. We can partition graph space into disjoint regions $\mathcal{G} = \bigcup_i \mathcal{R}_i, \text{s.t.} \mathcal{R}_i \cap \mathcal{R}_j = \emptyset$. The partition of regions satisfies the constant likelihood ratio, which means for every $Y \in \mathcal{R}_i$ and some constant c_i, it holds that:

$$\Pr(\phi(G) = Y) / \Pr(\phi(\tilde{G}) = Y) = c_i. \tag{18}$$

Suppose the lower bound of the most-likely label's probability $\Pr(f_\theta(\phi(G)) = y^*)$ is denoted as $\underline{p_{y^*}}$, and upper bound of the rest of the labels' probability $\Pr(f_\theta(\phi(G)) = y \neq y^*)$ is denoted as $\overline{p_y}$. To derive certificate is to demonstrate that there exist y^* and $\underline{p_{y^*}}$, $\overline{p_y}$ such that:

$$\Pr(f_\theta(\phi(G)) = y^*) \geq \underline{p_{y^*}} \geq \overline{p_y} \geq \max_{y \neq y^*} \Pr(f_\theta(\phi(G)) = y). \tag{19}$$

We can construct two union regions $\mathcal{R}^*$ and $\mathcal{R}$ such that $\Pr(\phi(G) \in \mathcal{R}^*) = \underline{p_{y^*}}$ and $\Pr(\phi(G) \in \mathcal{R}) = \overline{p_y}$, respectively. To do this, we gradually add regions $\mathcal{R}_i$ in likelihood c_i descending order to $\mathcal{R}^*$ until $\Pr(\phi(G) \in \mathcal{R}^*) = \underline{p_{y^*}}$ and similarly for $\mathcal{R}$ in likelihood c_i in ascending order. By gradually adding regions we can apply Neyman-Pearson Lemma to $\mathcal{R}^*$ and $\mathcal{R}$. Based on this, we can derive a lower bound of $\Pr(f_\theta(\tilde{G}) = y^*)$ and an upper bound of $\Pr(f_\theta(\tilde{G}) = y)$:

$$\Pr(f_\theta(\phi(\tilde{G})) = y^*) \geq \Pr(\phi(\tilde{G}) \in \mathcal{R}^*), \tag{20}$$

$$\Pr(f_\theta(\phi(\tilde{G})) = y) \leq \Pr(\phi(\tilde{G}) \in \mathcal{R}). \tag{21}$$

Given the proof above, the key to solving the certificate question is to add sub-regions $\mathcal{R}_i$ to $\mathcal{R}^*$ in likelihood c_i descending order. The Eq. 2 is equivalent to the following Linear Problem (LP) [24]:

$$\min_f f^T \tilde{r} \quad \text{s.t.} f^T r = p, \quad 0 \leq f \leq 1, \tag{22}$$

where f is the vector we are optimizing over corresponding to the classifier, and for each region, probability r is a vector where $r_i = \Pr(\phi(\tilde{G}) \in \mathcal{R}_i)$. The solution for Eq. 22 can be obtained by a greedy algorithm: sort the $\mathcal{R}_i$ such that $c_1 \geq c_2 \geq \cdots \geq c_I$, then iteratively assign $f_i = 1$ until the budget is met.

Now the challenge lies in how to efficiently partition regions and compute $\Pr(\phi(G) \in \mathcal{R}_i)$. Let $Z \in \{X, A\}$, we define the sphere $\mathcal{S}_{\Delta^+,\Delta^-}(Z)$ as:

$$\mathcal{S}_{\Delta^+,\Delta^-}(Z) = \{\tilde{Z} = Z + \delta : ||\delta^+|| = \Delta^+, ||\delta^-|| = \Delta^-\}. \tag{23}$$

That is the minimum in Eq. 2 is always attained at some $\tilde{G} \in \mathcal{S}_{\Delta^+,\Delta^-}(G)$. We define the region $\mathcal{R}_q^{\Delta^+,\Delta^-}$ containing all the vectors Z that can be obtained by flipping exactly q bits in the set of dimensions where Z and $\tilde{Z}$ disagree and the remaining dimensions are the same as Z. The partition defined above can divide the attribute and structure spaces into constant likelihood disjoint regions. *Proposition 2* in [7] gives the probability of $\phi(Z) \in \mathcal{R}_q^{\Delta^+,\Delta^-}$.

Proposition. *Given any $Z, \tilde{Z} \in \mathcal{S}_{\Delta^+,\Delta^-}(Z)$ and any region $\mathcal{R}_q^{\Delta^+,\Delta^-}$, $\Pr(\phi(Z) \in \mathcal{R}_q^{\Delta^+,\Delta^-}) = \Pr(Q = q)$ where $Q \sim \text{PB}([p_+, \Delta^+], [p_-, \Delta^-]) = \text{PB}(\underbrace{p_+, \dots, p_+}_{\Delta^+ \text{times}}, \underbrace{p_- \dots, p_-}_{\Delta^+ \text{times}})$ is a Poisson-Binomial random variable on $\{0, \dots, \Delta^+ + \Delta^-\}$.*

Constructing $\mathcal{R}_q^{\Delta^+,\Delta^-}$ is equivalent to conducting an experiment where the bits of variable Z are successfully flipped q times with probability p_+ and p_-. It's also equivalent to retaining q bits in $\tilde{Z}$ with probability $1 - p_+$ and $1 - p_-$. So Eq. 2 can be seen as performing likelihood testing where the two hypotheses correspond to two Poisson-Binomial distributions with different parameters. For all $Z \in \mathcal{R}_q^{\Delta^+,\Delta^-}$, the likelihood ratio is:

$$\mu_q^{\Delta^+,\Delta^-} = \frac{\Pr(\phi(Z) = Y)}{\Pr(\phi(\tilde{Z}) = Y)} = \left[\frac{p_+}{1 - p_-}\right]^{q - \Delta^-} \left[\frac{p_-}{1 - p_+}\right]^{q - \Delta^+} \tag{24}$$

and is constant in the region $\mathcal{R}_q^{\Delta^+,\Delta^-}$ satisfying the condition in Eq. 18.

Given the region partition and likelihood estimation above, we can efficiently solve the problem Eq. 22 and generate the robustness certificate. Since we use the exact same threat

model as in sparse smoothing [7], to derive the certificate, the minor technicality required for our approach is to map the diffusion noise timestamp to the randomization scheme parameter $p = (p_+, p_-)$. To be specific, the randomization scheme in sparse smoothing is to add Bernoulli noise $\epsilon^Z \sim \text{Ber}(p = p_+^{(1-z)} p_-^z)$ to variable Z. The diffusion process in Graph Joint Diffuser is to add Bernoulli noise $\epsilon_{t_z}^Z \sim \text{Ber}(p = (\bar{\beta}_{t_z} \mathbf{m_1}^Z)^{1-z} (\bar{\beta}_{t_z} \mathbf{m_0}^Z)^z)$ to variable Z where $Z = \{X, A\}$. It is obvious that $p_+ = \bar{\beta}_{t_z} \mathbf{m_1}^Z$ and $p_- = \bar{\beta}_{t_z} \mathbf{m_0}^Z$.

Algorithm Given the noise mapping described above and *certify* function in sparse smoothing [7], the certificate generator algorithm is illustrated in Algorithm 2.

Algorithm 2: Certificate Generator

Input : Sample G, target node v, diffuser $\mathcal{D}$, noise timestamp $t = (t_x, t_a)$, sample count n, node classifier f, significance level η

Output: Certificate

Initialize predicted label list Y
for $i \in \{1, \ldots, n\}$ **do**
  $\tilde{G} \leftarrow$ `ApplyNoise` (G, t_x, t_a)
  $p_+ = \bar{\beta}_{t_z} \mathbf{m_1}^Z$
  $p_- = \bar{\beta}_{t_z} \mathbf{m_0}^Z$
  /* Denoise with Graph Joint Diffuser $\mathcal{D}$. */
  $\overline{G} \leftarrow \mathcal{D}(\tilde{G})$
  Append $f(v|\overline{G})$ to Y
end
Certificate $\leftarrow$ `Certify` (n, Y, η, p_+, p_-)

V. Evaluations

Datasets and tested classifiers. We utilize four large citation networks with binary node attributes and undirected edges for evaluation. Cora ($N = 2708, |\mathcal{E}| = 10556, D = 1433, K = 7$), Cora-ML($N = 2995, |\mathcal{E}| = 16316, D = 2879, K = 7$), Citeseer($N = 3312, |\mathcal{E}| = 9072, D = 3703, K = 6$) and PubMed($N = 19717, |\mathcal{E}| = 88648, D = 500, K = 3$) [43] are citation networks with attributes indicating presence/absence of keywords and node labels representing paper categories. Edges $\mathcal{E}$ between nodes represent citation relationship between papers.

Following [7], [22], [23], we utilise popular node classifier GCN [28], GAT [29] and APPNP [30] for evaluation. We instantiate 1-layer and 2-layer models respectively. The GIN [31] model, which is more suitable for graph classification tasks, was not experimented with due to page limits.

Baselines. There is a line of works focusing on certified defense for graph node classification with RS. Sparse smoothing [7] first presented a certificate on sparse data with a data-dependent randomized smoothing scheme. Interception smoothing [22] provided a grey-box certificate for message-passing GNN by randomly intercepting messages from nodes. Hierarchical smoothing [23] generated the robustness certificate by applying sparse smoothing on randomly selected nodes. Among these works, interception smoothing and hierarchical smoothing used different threat models than we do and can only provide defense against node attribute perturbation. Hence, we use sparse smoothing [7] as our baseline. Specifically, we treat l_0 constraint threat model RS [24] and Bernoulli RS [25] as special cases of sparse smoothing.

Settings. We extract the 1-hop subgraph of each node to build a dataset to train the Graph Joint Diffuser. We utilized 20% of the data for testing, while the remaining portion was split into training and validation sets in an 80% to 20% ratio. For all datasets, we set maximum diffusion step $T_x = T_a = 500$ and use popular cosine diffusion schedule [44]. The relative importance level of structure in Eq. 16 is set to $\lambda = 5$.

For GNN training, both standard training and sparse data augmentation training, we employed the classic transductive learning approach. For each class, 20 nodes and 30 nodes were selected for the training and validation datasets respectively, with the remaining nodes serving as the test set.

At test time, we use significance level $\eta = 0.01$, $n_0 = 10$ samples to estimate the majority class and $n_1 = 1000$ samples for certification. We report the *clean accuracy* and the *certified accuracy* (the number of test samples that are correctly classified and certifiably robust for a given budget $\Delta = (\Delta_X, \Delta_A)$).

Experimental results. Table II shows the clean accuracy under joint perturbation on attribute and adjacency. 1-GCN refers to a GCN with one convolutional layer, and similar notations apply for others. *Naïve*$^\phi$, *Sparse*, and *CiDer* represent models trained on clean samples and tested with perturbed data, trained and tested with sparse smoothing, and trained on clean samples but tested with CiDer, respectively.

TABLE II
Clean accuracy for joint perturbation

Dataset	Model	Clean Accuracy				
		1-GCN	2-GCN	1-GAT	2-GAT	APPNP
Cora	Naïve$^\phi$	0.17	0.18	0.19	0.20	0.24
	Sparse	0.75	0.76	0.71	0.71	0.67
	CiDer	**0.82**	**0.79**	**0.77**	**0.77**	**0.73**
Cora-ML	Naïve$^\phi$	0.24	0.27	0.17	0.18	0.27
	Sparse	**0.79**	0.81	0.74	0.73	0.77
	CiDer	**0.79**	**0.82**	**0.83**	**0.83**	**0.85**
Citeseer	Naïve$^\phi$	0.20	0.19	0.16	0.19	0.19
	Sparse	0.67	0.68	0.67	0.66	0.71
	CiDer	**0.72**	**0.71**	**0.73**	**0.73**	**0.72**
PubMed	Naïve$^\phi$	0.43	0.45	0.40	0.39	0.39
	Sparse	0.63	0.72	0.48	0.54	0.61
	CiDer	**0.77**	**0.78**	**0.78**	**0.78**	**0.79**

Note: Test samples on all dataset are randomized by $\epsilon^Z \sim \text{Ber}(p = p_+^{(1-z)} p_-^z)$, abbreviated as $p^Z = (p_+, p_-)$. The randomization setting for each dataset are as follows: for Cora, Cora-ML and Citeseer, $p^X = (0.01, 0.65), p^A = (0.00, 0.66)$, for PubMed $p^X = (0.07, 0.59), p^A = (0.00, 0.66)$ corresponding to diffusion timestamp $t = 300$.

Naïve$^\phi$ results show that clean-data-trained GNNs fail to distinguish randomized samples, with randomized graph accuracy near random guessing. This highlights that i) CiDer's

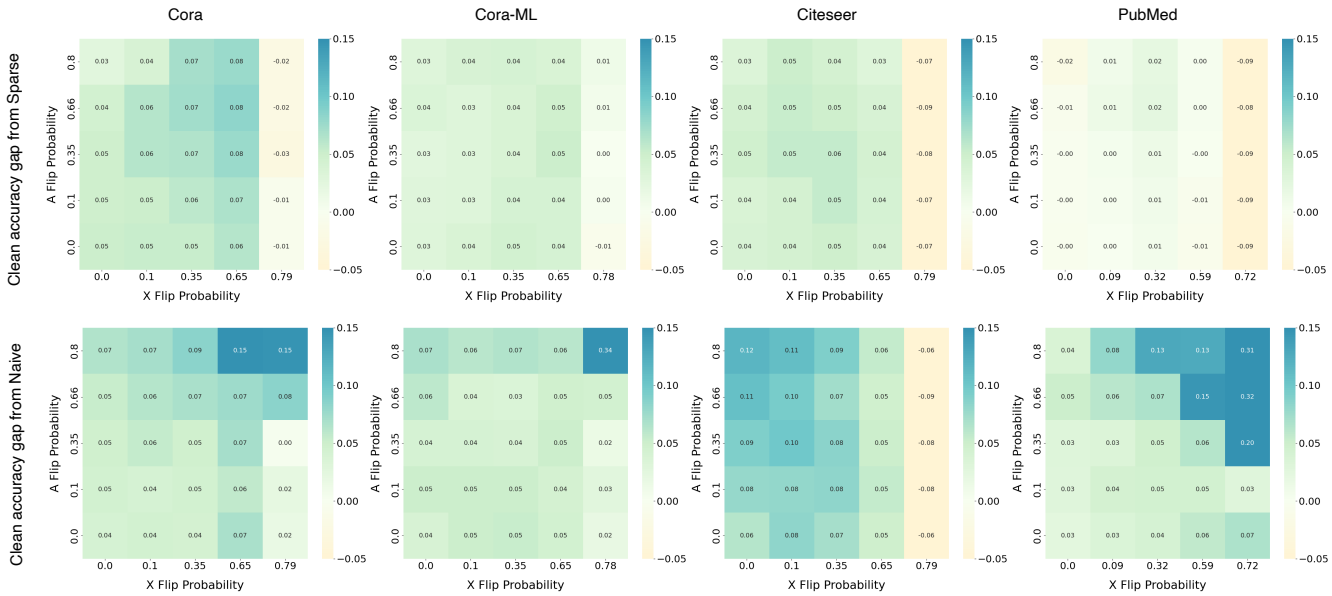

Fig. 4. **Clean accuracy gap**: Each column represents a dataset. The first row shows the accuracy gap between CiDer and standard 1-GCN, while the second row displays the accuracy gap between CiDer and sparse smoothing. Blue cells indicate a larger gap (CiDer significantly outperforms the baselines), while yellow cells indicate a negative gap.

Graph Joint Diffuser effectively denoises, enabling high Naïve GNN accuracy, and ii) randomized smoothing requires augmentation, as standard GNNs struggle with noisy samples.

CiDer achieves higher or comparable accuracy to sparse smoothing across various datasets, model architectures, and GNN layers. On the three smaller datasets, CiDer improves accuracy by around 5%. On the PubMed dataset, the average clean accuracy increased by 18.5%. This demonstrates that GNNs with sparse smoothing struggle to learn from large-scale graphs with complex noise, leading to significant accuracy loss. CiDer mitigates this issue by reducing the problem size through subgraph extraction, offering certified defense while maintaining clean accuracy.

We examined CiDer's performance across different GNN architectures and layer counts. CiDer extracts a subgraph of the target node for training and testing, yielding similar results for 1-layer and 2-layer GNNs. In contrast, sparse smoothing shows significant accuracy differences across GNNs with varying layer counts. CiDer still outperforms the 2-layer GNN sparse method. Furthermore, the sparse method struggles with GAT, while CiDer maintains high accuracy across architectures, as it purifies the data, making it model-agnostic.

Fig. 4 shows the clean accuracy gap between CiDer and sparse smoothing, as well as between CiDer and the Naïve model (tested with clean sample), evaluated using 1-GCN. In each subplot, the horizontal and vertical axes represent the noise scale for the attribute and adjacent matrix, respectively. The numbers on the axis represent the flipping probabilities p_- in $\text{Ber}(p = p_+^{(1-z)} p_-^z)$ with p_+ omitted. From low to high, the noise scales correspond to the diffusion timestamps $\{0, 100, 200, 300, 350\}$.

The figure shows that models with CiDer defense generally show an accuracy improvement compared to Naïve models. When the noise in node attributes is small, CiDer can improve accuracy. This is because the process of reconstructing clean samples effectively moves the randomly shifted sample points back to their original distribution. During this process, CiDer also removes real-world noises presented in the original data distribution. When node noise is large, the loss of attribute information makes it difficult for CiDer to accurately recover the data, indicating that higher robustness leads to a certain degree of accuracy degradation.

From the figure, it can be seen that across all datasets, all noise scales, and all noise scale combinations, CiDer achieves higher accuracy than the sparse smoothing method.

Specifically, CiDer shows a significant accuracy improvement on three datasets under large noise. We speculate that this is because CiDer can effectively reconstruct noisy samples with high noise scales, while sparse smoothing fails to learn useful information from such noisy samples during training. On the Citeseer dataset, CiDer performs well when the attribute noise is low. We speculate that the slight accuracy drop under large noise is due to the node attribute dimension being larger in Citeseer, leading to more complexity.

Fig. 5 shows the Cora-ML certified accuracy of CiDer and sparse smoothing under the multi-class classification scenario, tested with 1-GCN. The singular certificate of attribute is derived using randomization scheme $\epsilon^X = (0.01, 0.65)$ and singular certificate of adjacent is derived from randomization scheme $\epsilon^A = (0.00, 0.66)$. The joint certificate is derived

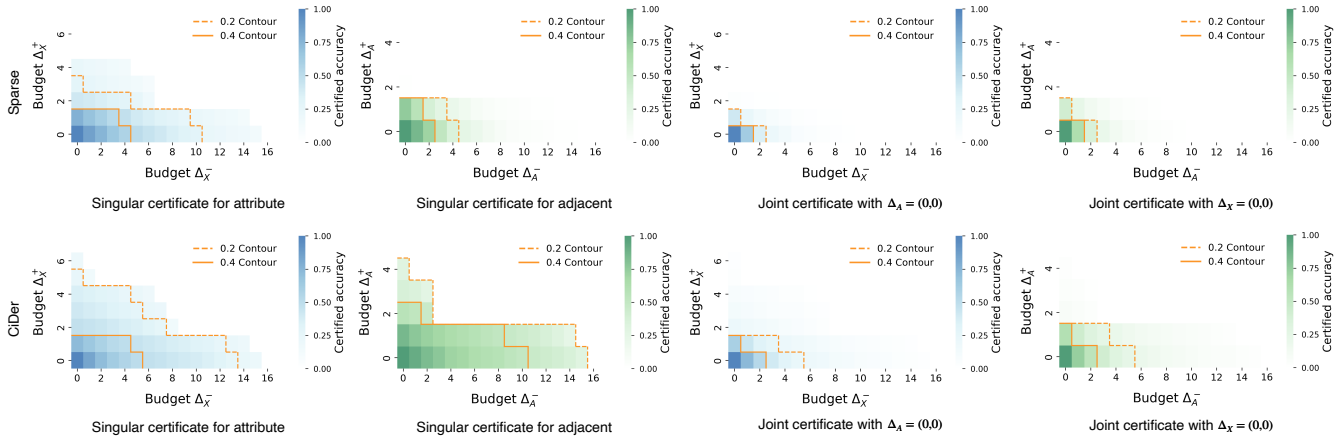

Fig. 5. **Certificate**: The top row shows the certified accuracy using sparse smoothing, while the bottom row uses CiDer. The left two columns display singular certificates of X and A, and the right two columns present joint certificates of $X(A)$ under a fixed $A(X)$ budget. The orange and yellow lines in the figure represent the contour line where the accuracy reaches 0.4 and 0.2.

using joint randomization scheme of the same setting. In the certified accuracy experiments, we sample $n_0 = 10$ graphs to estimate the most-likely and second-likely class, while sampling $n_1 = 10000$ graphs to derive robustness certificates. Sample number can be set larger for higher robustness.

The blue heatmaps represent the certificates for node attributes, and the green heatmaps represent the certificates for graph structure. The horizontal axis of the heatmap indicates the attack deletion budget Δ^- for attributes/edges, while the vertical axis represents the attack addition budget Δ^+. Since p_- is larger than p^+, the certified attack Δ^- is larger than the attack Δ^+.

Under the same randomization configuration and sampling counts, CiDer achieves higher certified accuracy in both singular and joint certificate scenarios. Specifically, when randomizing only the attributes, though the certified budget is similar, CiDer can maintain higher accuracy. When randomizing only the structure, CiDer significantly outperforms sparse smoothing both in the certifiable budget and in accuracy under the same budget.

Discussion

CiDer leverages the powerful denoise capability of the discrete diffusion denoising model to derive a robustness certificate for arbitrary GNN trained on clean data.

Our work focuses on node classification on unweighted and undirected graphs. We have not addressed other types of graphs, such as directed graphs (e.g. Texas [45]), and other graph learning tasks, such as graph classification. Dealing with more complex graphs requires complicated encoding and a different threat model, which presents unique challenges. Graph learning on more complex graphs is beyond the scope of this paper and will be explored in future research. CiDer seems to be intuitively applicable for other graph learning models, but we did not conduct the exploration due to the page limits.

Additionally, using the discrete diffusion model entails high computational complexity. To achieve high certified accuracy under a large attack budget, CiDer requires a large number of Monte Carlo samples, just like RS. Due to the expensive computation required for repeated sampling and denoising processes, the inference time increases with larger graph scales or higher robustness requirements. Our proposed model has a computational complexity of $O(D)+O((|\mathcal{E}|'+N')D)$, where $|\mathcal{E}|'$ and N' represent subgraph's node and edge count. We also profiled the computational cost: for the Cora dataset, the average computation cost per node per sample is 7915 FLOPs. With the *Subgraph Extractor* we presented in IV-B, the scale of diffused graphs can be reduced to some extent. Improving computational efficiency will be a key area of focus in our future research efforts.

Conclusion

In recent years, there has been an increasing focus on defending against adversarial attacks targeting graph node classification. Although these methods have been proven effective against specific attack techniques or models, some are limited by the need for prior knowledge of the attacker and others are costly due to the need for retraining the classifier.

We present a novel approach - CiDer, which provides robustness guarantees for arbitrary graph node classifiers, including non-deep learning methods and graph neural network, since it only deals with the graph data. Our provable defense can be effectively applied to arbitrary black-box model in a plug-and-play manner with no assumptions about the attacker. CiDer exclusively makes use of a discrete denoise diffusion model to purify the core features of graph input. Experiments demonstrate the effectiveness of our approach, achieving higher clean accuracy and certified accuracy compared to the state-of-the-art on multiple real-world datasets.

References

[1] H. H. X. Nguyen, T. K. Dang, and P. T. Tran-Truong, "Money laundering detection using A transaction-based graph learning approach," in *Proc. Int. Conf. Ubiquitous Informat. Manage. Commun.*, Jan. 2024, pp. 1–8.

[2] K. Li, Y. Liu, X. Ao, and Q. He, "Revisiting graph adversarial attack and defense from a data distribution perspective," in *Proc. Int. Conf. Learn. Representations*, May 2023.

[3] K. Xu, H. Chen, S. Liu, P. Chen, T. Weng, M. Hong, and X. Lin, "Topology attack and defense for graph neural networks: An optimization perspective," in *Proc. Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 3961–3967.

[4] H. Wu, C. Wang, Y. Tyshetskiy, A. Docherty, K. Lu, and L. Zhu, "Adversarial examples for graph data: Deep insights into attack and defense," in *Proc. Int. Joint Conf. Artif. Intell.*, July 2019, pp. 4816–4823.

[5] M. Lécuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *Proc. IEEE Symp. Secur. Privacy*, May 2019, pp. 656–672.

[6] J. Cohen, E. Rosenfeld, and J. Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *Proc. Int. Conf. Mach. Learn.*, June 2019, pp. 1310–1320.

[7] A. Bojchevski, J. Klicpera, and S. Günnemann, "Efficient robustness certificates for discrete data: Sparsity-aware randomized smoothing for graphs, images and more," in *Proc. Int. Conf. Mach. Learn.*, July 2020, pp. 1003–1013.

[8] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg, "Structured denoising diffusion models in discrete state-spaces," in *Proc. Adv. Neural Informat. Process. Syst*, Dec. 2021, pp. 17 981–17 993.

[9] F. Feng, X. He, J. Tang, and T. Chua, "Graph adversarial training: Dynamically regularizing based on graph structure," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2493–2504, 2021.

[10] L. Gosch, S. Geisler, D. Sturm, B. Charpentier, D. Zügner, and S. Günnemann, "Adversarial training for graph neural networks: Pitfalls, solutions, and new directions," in *Proc. Adv. Neural Informat. Process. Syst*, Dec. 2023, pp. 58 088–58 112.

[11] Q. Dai, X. Shen, L. Zhang, Q. Li, and D. Wang, "Adversarial training methods for network embedding," in *Proc. Int. World Wide Web Conf.*, May 2019, pp. 329–339.

[12] K. Sun, Z. Lin, H. Guo, and Z. Zhu, "Virtual adversarial training on graph convolutional networks in node classification," in *Proc. Chin. Conf. Pattern Recognit. Comput. Vis.*, Nov. 2019, pp. 431–443.

[13] K. Zhao, Q. Kang, Y. Song, R. She, S. Wang, and W. P. Tay, "Adversarial robustness in graph neural networks: A hamiltonian approach," in *Proc. Adv. Neural Informat. Process. Syst*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., Dec. 2023.

[14] A. Liu, W. Li, T. Li, B. Li, H. Huang, and P. Zhou, "Towards inductive robustness: Distilling and fostering wave-induced resonance in transductive gcns against graph adversarial attacks," *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 12, pp. 13 855–13 863, Mar. 2024.

[15] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, Aug. 2020, pp. 66–74.

[16] N. Entezari, S. A. Al-Sayouri, A. Darvishzadeh, and E. E. Papalexakis, "All you need is low (rank): Defending against adversarial attacks on graphs," in *Proc. 30th Int. Conf. Web Search Data Mining*, Feb. 2020, pp. 169–177.

[17] X. Xu, H. Wang, A. Lal, C. A. Gunter, and B. Li, "Edog: Adversarial edge detection for graph neural networks," in *Proc. IEEE Conf. Secure Trustworthy Mach. Learn.*, Feb. 2023, pp. 291–305.

[18] X. Tang, Y. Li, Y. Sun, H. Yao, P. Mitra, and S. Wang, "Transferring robustness for graph neural network against poisoning attacks," in *Proc. 13th Int. Conf. Web Search Data Mining*, Feb. 2020, pp. 600–608.

[19] A. Bojchevski and S. Günnemann, "Certifiable robustness to graph perturbations," in *Proc. Adv. Neural Informat. Process. Syst*, Dec. 2019, pp. 8317–8328.

[20] D. Zügner and S. Günnemann, "Certifiable robustness of graph convolutional networks under structure perturbations," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, Aug. 2020, pp. 1656–1665.

[21] ——, "Certifiable robustness and robust training for graph convolutional networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, Aug. 2019, pp. 246–256.

[22] Y. Scholten, J. Schuchardt, S. Geisler, A. Bojchevski, and S. Günnemann, "Randomized message-interception smoothing: Gray-box certificates for graph neural networks," in *Proc. Adv. Neural Informat. Process. Syst*, Nov./Dec. 2022.

[23] Y. Scholten, J. Schuchardt, A. Bojchevski, and S. Günnemann, "Hierarchical randomized smoothing," in *Proc. Adv. Neural Informat. Process. Syst*, Dec. 2023.

[24] G. Lee, Y. Yuan, S. Chang, and T. S. Jaakkola, "Tight certificates of adversarial robustness for randomly smoothed classifiers," in *Proc. Adv. Neural Informat. Process. Syst*, Dec. 2019, pp. 4911–4922.

[25] B. Wang, J. Jia, X. Cao, and N. Z. Gong, "Certified robustness of graph neural networks against adversarial structural perturbation," in *Proc. 27th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, Aug. 2021, pp. 1645–1653.

[26] Y. Dou, G. Ma, P. S. Yu, and S. Xie, "Robust spammer detection by nash reinforcement learning," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, Aug. 2020, pp. 924–933.

[27] S. Zhang, H. Yin, T. Chen, Q. V. H. Nguyen, Z. Huang, and L. Cui, "Gcn-based user representation learning for unifying robust recommendation and fraudster detection," in *Proc. 43rd ACM SIGIR Int. Conf. Res. Develop. Informat. Retrieval*, July 2020, pp. 689–698.

[28] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, Apr. 2017.

[29] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, Apr./May 2018.

[30] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *Proc. Int. Conf. Learn. Representations*, May 2019.

[31] X. Wang and M. Zhang, "How powerful are spectral graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, July 2022, pp. 23 341–23 362.

[32] S. Wang, Y. Dong, B. Zhang, Z. Chen, X. Fu, Y. He, C. Shen, C. Zhang, N. V. Chawla, and J. Li, "Safety in graph machine learning: Threats and safeguards," *arXiv:2405.11034*, 2024.

[33] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, Aug. 2018, pp. 2847–2856.

[34] S. Geisler, T. Schmidt, H. Sirin, D. Zügner, A. Bojchevski, and S. Günnemann, "Robustness of graph neural networks at scale," in *Proc. Adv. Neural Informat. Process. Syst*, Dec. 2021, pp. 7637–7649.

[35] G. Zhu, M. Chen, C. Yuan, and Y. Huang, "Simple and efficient partial graph adversarial attack: A new perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 8, pp. 4245–4259, 2024.

[36] L. Wen, J. Liang, K. Yao, and Z. Wang, "Black-box adversarial attack on graph neural networks with node voting mechanism," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 10, pp. 5025–5038, 2024.

[37] S. Günnemann, "Graph neural networks: Adversarial robustness," *Graph neural networks: foundations, frontiers, and applications*, pp. 149–176, 2022.

[38] C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard, "Digress: Discrete denoising diffusion for graph generation," in *Proc. Int. Conf. Learn. Representations*, May 2023.

[39] M. Li, E. Kreacic, V. K. Potluru, and P. Li, "Graphmaker: Can diffusion models generate large attributed graphs?" *arXiv:310.13833*, 2023.

[40] X. Chen, J. He, X. Han, and L. Liu, "Efficient and degree-guided graph generation via discrete diffusion modeling," in *Proc. Int. Conf. Mach. Learn.*, July 2023, pp. 4585–4610.

[41] N. Carlini, F. Tramèr, K. D. Dvijotham, L. Rice, M. Sun, and J. Z. Kolter, "(certified!!) adversarial robustness for free!" in *Proc. Int. Conf. Learn. Representations*, May 2023.

[42] J. Neyman and E. S. Pearson, "Ix. on the problem of the most efficient tests of statistical hypotheses," *Philos. Trans. R. Soc. Lond. Ser. A-Math. Phys. Eng. Sci.*, vol. 231, no. 694-706, pp. 289–337, 1933.

[43] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–106, 2008.

[44] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proc. Int. Conf. Mach. Learn.*, July 2021, pp. 8162–8171.

[45] H. Pei, B. Wei, K. C. Chang, Y. Lei, and B. Yang, "Geom-gcn: Geometric graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, Apr. 2020.