Certified Robustness of Graph Neural Networks against Adversarial Structural Perturbation

Binghui Wang, Jinyuan Jia, Xiaoyu Cao, Neil Zhenqiang Gong
ECE Department, Duke University
{binghui.wang, jinyuan.jia, xiaoyu.cao, neil.gong}@duke.edu

Abstract

Graph neural networks (GNNs) have recently gained much attention for node and graph classification tasks on graph-structured data. However, multiple recent works showed that an attacker can easily make GNNs predict incorrectly via perturbing the graph structure, i.e., adding or deleting edges in the graph. We aim to defend against such attacks via developing certifiably robust GNNs. Specifically, we prove the *first* certified robustness guarantee of any GNN for both node and graph classifications against structural perturbation. Moreover, we show that our certified robustness guarantee is *tight*. Our results are based on a recently proposed technique called *randomized smoothing*, which we extend to graph data. We also empirically evaluate our method for both node and graph classifications on multiple GNNs and multiple benchmark datasets. For instance, on the Cora dataset, Graph Convolutional Network with our randomized smoothing can achieve a certified accuracy of 0.49 when the attacker can arbitrarily add/delete at most 15 edges in the graph.

1 Introduction

Graphs are a powerful tool to represent data from diverse domains such as social networks, biology, finance, security, etc.. *Node classification* and *graph classification* are two basic tasks on graphs. Specifically, given a graph, node classification aims to classify nodes in the graph in a semi-supervised fashion, while graph classification aims to assign a label to the entire graph instead of individual nodes. Node and graph classifications have many applications, including but not limited to, profiling users in social networks [57, 20], classifying proteins [17, 43], and fraud detection [32, 41, 45, 46]. Various methods such as label propagation [59], belief propagation [33], iterative classification [38], and graph neural networks (GNNs) [22, 17, 16, 43, 53] have been proposed for node/graph classifications. Among them, GNNs have attracted much attention recently because of their expressiveness and superior performance.

However, several recent works [60, 9, 61, 1, 44] have demonstrated that an attacker can easily fool GNNs to make incorrect predictions via perturbing 1) the node features and/or 2) the structure of the graph (i.e., adding and/or deleting edges in the graph). Thus, it is of great importance to develop certifiably robust GNNs against such attacks. To the best of our knowledge, [62], [2], and [63] are the only existing works to study certified robustness of GNNs for node classification. Specifically, Zügner et al. [62] formulate the certified robustness of GNNs as a linear programming problem. Moreover, they derive the certified robustness guarantee based on the dual form of the optimization problem, which is motivated by [48].

However, their work assumes the attacker perturbs the node features. Bojchevski et al. [2] and Zügner et al. [63] study certified robustness of GNNs against structural perturbation. However, their works are customized to a specific GNN [23].

We aim to bridge this gap in this work. Specifically, we aim to provide certified robustness guarantees of *any* GNN classifier against structural perturbations for both node and graph classification. Towards this goal, we leverage a recently developed technique called *randomized smoothing* [4, 29, 24, 28, 8, 18], which can turn any *base classifier* (e.g., a GNN classifier in our problem) to a robust one via adding random noise to a testing example (i.e., a graph in our problem). A classifier is certifiably robust if it provably predicts the same label when the attacker adds/deletes at most K edges in the graph, where we call K *certified perturbation size*.

However, existing randomized smoothing methods are insufficient to certify robustness of GNNs. Specifically, they all assume testing examples are continuous data and add Gaussian/Laplacian noise to them. However, graph structures are essentially *binary* data, i.e., a pair of nodes can be either connected or unconnected; and Gaussian/Laplacian noise is not semantically meaningful for such binary data. We develop randomized smoothing for binary data and leverage it to obtain certified robustness of GNNs against structural perturbation.

First, we theoretically derive a certified perturbation size for any GNN classifier with randomized smoothing via addressing several technical challenges. For instance, we divide the graph structure space into regions in a novel way such that we can apply the Neyman-Pearson Lemma [31] to certify robustness. We also prove that our derived certified perturbation size is *tight* if no assumptions on the GNN classifier are made.

Second, we design a method to compute our certified perturbation size in practice. It is challenging to compute our certified perturbation size as it involves estimating probability bounds *simultaneously* and solving an optimization problem. To address the challenge, we first adopt the simultaneous confidence interval estimation method [18] to estimate the probability bounds with probabilistic guarantees. Then, we design an algorithm to solve the optimization problem to obtain our certified perturbation size with the estimated probability bounds.

We also empirically evaluate our method. Specifically, for node classification, we consider Graph Convolutional Network (GCN) [22] and Graph Attention Network (GAT) [43] on several benchmark datasets including Cora, Citeseer, and Pubmed [38]. For graph classification, we consider Graph Isomorphism Network (GIN) [52] on benchmark datasets including MUTAG, PROTEINS, and IMDB [54]. For instance, on the Cora dataset, GCN with our randomized smoothing can achieve certified accuracies of 0.55, 0.50, and 0.49 when the attacker arbitrarily adds/deletes at most 5, 10, and 15 edges, respectively. On the MUTAG dataset, GIN with our randomized smoothing can achieve certified accuracies of 0.45, 0.45, and 0.40 when the attacker arbitrarily adds/deletes at most 5, 10, and 15 edges, respectively.

Our major contributions can be summarized as follows:

- We prove the first certified robustness guarantee of any GNN against structural perturbation. Moreover, we show that our certified robustness guarantee is tight.
- Our certified perturbation size is the solution to an optimization problem and we design an algorithm to solve the optimization problem.
- We empirically evaluate our method for both node and graph classifications on multiple benchmark datasets.

2 Background on Graph Neural Networks

2.1 Node Classification vs. Graph Classification

We consider GNNs for both *node classification* [22, 17, 43] and *graph classification* [17, 52]. Suppose we are given an undirected graph G with node features.

- **Node classification.** A node classifier predicts labels for nodes in the graph in a semi-supervised fashion. Specifically, a subset of nodes in G are already labeled, which are called *training nodes* and denoted as V_T. A node classifier f_n takes the graph G and the training nodes V_T as an input and predicts the label for each remaining node, i.e., $f_n(G, V_T, u)$ is the predicted label for a node u.
- **Graph classification.** A graph classifier aims to predict a label for the entire graph instead of individual nodes, i.e., $f_g(G)$ is the predicted label for the graph G. Such a graph classifier can be trained using a set of graphs with ground truth labels.

2.2 Adversarial Structural Perturbation

An attacker aims to fool a node classifier or graph classifier to make predictions as the attacker desires via perturbing the graph structure, i.e., deleting some edges and/or adding some edges in the graph [60, 9, 61, 1, 44]. Since our work focuses on structural perturbation, we treat the node feature matrix and the training nodes as constants. Moreover, we simply write a node classifier $f_n(G, V_T, u)$ or a graph classifier $f_g(G)$ as $f(\mathbf{s})$, where the binary vector $\mathbf{s}$ represents the graph structure. Note that a node classifier should also take a node u as input and predict its label. However, we omit the explicit dependency on a node u for simplicity. We call $\mathbf{s}$ *structure vector*. For instance, $\mathbf{s}$ can be the concatenation of the upper triangular part of the adjacency matrix of the graph (excluding the diagonals) when the attacker can modify the connection status of any pair of nodes, i.e., $\mathbf{s}$ includes the connection status for each pair of nodes in the graph. When the attacker can only modify the connection status between u and each remaining node, $\mathbf{s}$ can also be the uth row of the adjacency matrix of the graph (excluding the self-loop (u, u)th entry). We assume the structure vector $\mathbf{s}$ has n entries. As we will see, such simplification makes it easier to present our certified robustness against structural perturbation.

We denote by vector $\delta \in \{0, 1\}^n$ the attacker's perturbation to the graph structure. Specifically, $\delta_i = 1$ if and only if the attacker changes the ith entry in the structure vector $\mathbf{s}$, i.e., the attacker changes the connection status of the corresponding pair of nodes. Moreover, $\mathbf{s} \oplus \delta$ is the perturbed structure vector, which represents the perturbed graph structure, where $\oplus$ is the XOR operator between two binary variables. We use $||\delta||_0$ to measure the magnitude of the adversarial perturbation as it has semantic meanings. Specifically, $||\delta||_0$ is the number of node pairs whose connection statuses are modified by the attacker.

3 Certified Robustness

In this section, we derive the certified robustness of arbitrary GNN against structural perturbations.

3.1 Randomized Smoothing with Binary Noise

We first define a noise distribution in the discrete structure vector space $\{0,1\}^n$. Then, we define a *smoothed classifier* based on the noise distribution and a node/graph classifier (called *base classifier*). Specifically, we consider the noise vector ϵ has the following probability distribution in the discrete space $\{0,1\}^n$:

$$\Pr(\epsilon_i = 0) = \beta,\ \Pr(\epsilon_i = 1) = 1 - \beta, \tag{1}$$

where $i = 1, 2, \cdots, n$. When we add a random noise vector ϵ to the structure vector $\mathbf{s}$, the ith entry of $\mathbf{s}$ is preserved with probability β and changed with probability $1 - \beta$. In other words, our random noise means that, for each pair of nodes in the graph, we keep its connection status with probability β and change its connection status with probability $1 - \beta$. Based on the noise distribution and a base node/graph classifier $f(\mathbf{s})$, we define a smoothed classifier $g(\mathbf{s})$ as follows:

$$g(\mathbf{s}) = \arg\max_{c \in \mathcal{C}} \Pr(f(\mathbf{s} \oplus \epsilon) = c), \tag{2}$$

where $\mathcal{C}$ is the set of labels, $\oplus$ is the XOR operator between two binary variables, $\Pr(f(\mathbf{s} \oplus \epsilon) = c)$ is the probability that the base classifier f predicts label c when we add random noise ϵ to the structure vector $\mathbf{s}$, and $g(\mathbf{s})$ is the label predicted for $\mathbf{s}$ by the smoothed classifier. Moreover, we note that $g(\mathbf{s} \oplus \delta)$ is the label predicted for the perturbed structure vector $\mathbf{s} \oplus \delta$. Existing randomized smoothing methods [4, 29, 24, 28, 8, 18] add random continuous noise (e.g., Gaussian noise, Laplacian noise) to a testing example (i.e., the structure vector $\mathbf{s}$ in our case). However, such continuous noise is not meaningful for the binary structure vector.

Our goal is to show that a label is provably the predicted label by the smoothed classifier g for the perturbed structure vector $\mathbf{s} \oplus \delta$ when the ℓ_0-norm of the adversarial perturbation δ, i.e., $\|\delta\|_0$, is bounded. Next, we theoretically derive the certified perturbation size K, prove the tightness of the certified perturbation size, and discuss how to compute the certified perturbation size in practice.

3.2 Theoretical Certification

3.2.1 Overview

Let $X = \mathbf{s} \oplus \epsilon$ and $Y = \mathbf{s} \oplus \delta \oplus \epsilon$ be two random variables, where ϵ is the random binary noise drawn from the distribution defined in Equation 1. X and Y represent random structure vectors obtained by adding random binary noise ϵ to the structure vector $\mathbf{s}$ and its perturbed version $\mathbf{s} \oplus \delta$, respectively.

Suppose, when taking X as an input, the base GNN classifier f correctly predicts the label c_A with the largest probability. Then, our key idea is to guarantee that, when taking Y as an input, f still predicts c_A with the largest probability. Moreover, we denote c_B as the predicted label by f with the second largest probability. Then, our goal is to find the maximum perturbation size such that the following inequality holds:

$$\Pr(f(Y) = c_A) > \Pr(f(Y) = c_B). \tag{3}$$

Note that it is challenging to compute the probabilities $\Pr(f(Y) = c_A)$ and $\Pr(f(Y) = c_B)$ exactly because f is highly nonlinear in practice. To address the challenge, we first derive

a lower bound of $\Pr(f(Y) = c_A)$ and an upper bound of $\Pr(f(Y) = c_B)$. Then, we require that the lower bound of $\Pr(f(Y) = c_A)$ is larger than the upper bound of $\Pr(f(Y) = c_B)$. Specifically, we derive the lower bound and upper bound by constructing certain regions in the graph structure space $\{0,1\}^n$ such that the probabilities Y is in these regions can be efficiently computed for any $||\delta||_0 = k$. Then, we iteratively search the maximum k under the condition that the lower bound is larger than the upper bound. Finally, we treat the maximum k as the certified perturbation size K.

3.2.2 Deriving the lower and upper bounds

Our idea is to divide the graph structure space $\{0,1\}^n$ into regions in a novel way such that we can apply the Neyman-Pearson Lemma [31] to derive the lower bound and upper bound. First, for any data point $\mathbf{z} \in \{0,1\}^n$, we have the density ratio $\frac{\Pr(X=\mathbf{z})}{\Pr(Y=\mathbf{z})} = \left(\frac{\beta}{1-\beta}\right)^{w-v}$ based on the noise distribution defined in Equation 1, where $w = \|\mathbf{s}-\mathbf{z}\|_0$ and $v = \|\mathbf{s}\oplus\delta-\mathbf{z}\|_0$ (please refer to Section B in for details). Therefore, we have the density ratio $\frac{\Pr(X=\mathbf{z})}{\Pr(Y=\mathbf{z})} = \left(\frac{\beta}{1-\beta}\right)^{m}$ for any $\mathbf{z} \in \{0,1\}^n$, where $m = -n, -n+1, \cdots, n-1, n$. Furthermore, we define a region $\mathcal{R}(m)$ as the set of data points whose density ratios are $\left(\frac{\beta}{1-\beta}\right)^m$, i.e., $\mathcal{R}(m) = \{\mathbf{z} \in \{0,1\}^n : \frac{\Pr(X=\mathbf{z})}{\Pr(Y=\mathbf{z})} = \left(\frac{\beta}{1-\beta}\right)^m\}$, and we denote by $r(m)$ the corresponding density ratio, i.e., $r(m) = \left(\frac{\beta}{1-\beta}\right)^m$. Moreover, we rank the $2n+1$ regions in a descending order with respect to their density ratios, and denote the ranked regions as $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_{2n+1}$.

Suppose we are given a lower bound of the largest label probability $\Pr(f(X) = c_A)$ for c_A and denote it as $\underline{p_A}$, and an upper bound of the remaining label probability $\Pr(f(X) = c)$ for $c \neq c_A$ and denote it as $\overline{p_B}$. Assuming there exist c_A and $\underline{p_A}, \overline{p_B} \in [0,1]$ such that

$$\Pr(f(X) = c_A) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c \neq c_A} \Pr(f(X) = c). \tag{4}$$

Next, we construct two regions $\mathcal{A}$ and $\mathcal{B}$ such that $\Pr(X \in \mathcal{A}) = \underline{p_A}$ and $\Pr(X \in \mathcal{B}) = \overline{p_B}$, respectively. Specifically, we gradually add the regions $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_{2n+1}$ to $\mathcal{A}$ until $\Pr(X \in \mathcal{A}) = \underline{p_A}$. Moreover, we gradually add the regions $\mathcal{R}_{2n+1}, \mathcal{R}_{2n}, \cdots, \mathcal{R}_1$ to $\mathcal{B}$ until $\Pr(X \in \mathcal{B}) = \overline{p_B}$. We construct the regions $\mathcal{A}$ and $\mathcal{B}$ in this way such that we can apply the Neyman-Pearson Lemma [31] for them. Formally, we define the regions $\mathcal{A}$ and $\mathcal{B}$ as follows:

$$\mathcal{A} = \bigcup_{j=1}^{a^\star - 1} \mathcal{R}_j \cup \underline{\mathcal{R}_{a^\star}} \tag{5}$$

$$\mathcal{B} = \bigcup_{j=b^\star+1}^{2n+1} \mathcal{R}_j \cup \underline{\mathcal{R}_{b^\star}}, \tag{6}$$

where

$$a^\star = \operatorname*{arg\,min}_{a \in \{1,2,\cdots,2n+1\}} a, \ s.t. \ \sum_{j=1}^{a} \Pr(X \in \mathcal{R}_j) \geq \underline{p_A},$$

$$b^\star = \operatorname*{arg\,max}_{b \in \{1,2,\cdots,2n+1\}} b, \ s.t. \ \sum_{j=b}^{2n+1} \Pr(X \in \mathcal{R}_j) \geq \overline{p_B}.$$

$\underline{\mathcal{R}_{a^\star}}$ is any subregion of $\mathcal{R}_{a^\star}$ such that $\Pr(X \in \underline{\mathcal{R}_{a^\star}}) = \underline{p_A} - \sum_{j=1}^{a^*-1} \Pr(X \in \mathcal{R}_j)$, and $\underline{\mathcal{R}_{b^\star}}$ is any subregion of $\mathcal{R}_{b^\star}$ such that $\Pr(X \in \underline{\mathcal{R}_{b^\star}}) = \overline{p_B} - \sum_{j=b^*+1}^{2n+1} \Pr(X \in \mathcal{R}_j)$.

Finally, based on the Neyman-Pearson Lemma, we can derive a lower bound of $\Pr(f(Y) = c_A)$ and an upper bound of $\Pr(f(Y) = c_B)$. Formally, we have the following lemma:

Lemma 1. *We have the following bounds:*

$$Pr(f(Y) = c_A) \geq Pr(Y \in \mathcal{A}), \tag{7}$$
$$Pr(f(Y) = c_B) \leq Pr(Y \in \mathcal{B}). \tag{8}$$

Proof. See Section A.1. □

3.2.3 Deriving the certified perturbation size

Given Lemma 1, we can derive the certified perturbation size K as the maximum k such that the following inequality holds for $\forall ||\delta||_0 = k$:

$$\Pr(Y \in \mathcal{A}) > \Pr(Y \in \mathcal{B}). \tag{9}$$

Formally, we have the following theorem:

Theorem 1 (Certified Perturbation Size)**.** *Let f be any base node/graph classifier. The random noise vector ϵ is defined in Equation 1. The smoothed classifier g is defined in Equation 2. Given a structure vector $\mathbf{s} \in \{0,1\}^n$, suppose there exist c_A and $\underline{p_A}, \overline{p_B} \in [0,1]$ that satisfy Equation 4. Then, we have*

$$g(\mathbf{s} \oplus \delta) = c_A, \forall ||\delta||_0 \leq K, \tag{10}$$

where the certified perturbation size K is the solution to the following optimization problem:

$$K = \arg\max k, \tag{11}$$
$$s.t.\ Pr(Y \in \mathcal{A}) > Pr(Y \in \mathcal{B}), \forall ||\delta||_0 = k. \tag{12}$$

Proof. See Section A.2. □

Next, we show that our certified perturbation size is tight in the following theorem.

Theorem 2 (Tightness of the Certified Perturbation Size)**.** *Assume $\underline{p_A} \geq \overline{p_B}$, $\underline{p_A} + \overline{p_B} \leq 1$, and $\underline{p_A} + (|\mathcal{C}| - 1) \cdot \overline{p_B} \geq 1$, where $|\mathcal{C}|$ is the number of labels. For any perturbation δ with $||\delta||_0 > K$, there exists a base classifier f^* consistent with Equation 4 such that $g(\mathbf{s} \oplus \delta) \neq c_A$ or there exist ties.*

Proof. See Section A.3. □

We have several observations on our major theorems.

- Our theorems are applicable to any base node/graph classifier. Moreover, although we focus on classifiers on graphs, our theorems are applicable to any base classifier that takes binary features as input.

- Our certified perturbation size K depends on $\underline{p_A}$, $\overline{p_B}$, and β. In particular, when the probability bounds $\underline{p_A}$ and $\overline{p_B}$ are tighter, our certified perturbation size K is larger. We use the probability bounds $\underline{p_A}$ and $\overline{p_B}$ instead of their exact values, because it is challenging to compute p_A and p_B exactly.
- When no assumptions on the base classifier are made and randomized smoothing with the noise distribution defined in Equation 1 is used, it is impossible to certify a perturbation size larger than K.

3.3 Certification in Practice

Computing our certified perturbation size K in practice faces two challenges. The first challenge is to estimate the probability bounds $\underline{p_A}$ and $\overline{p_B}$. The second challenge is to solve the optimization problem in Equation 11 to get K with the given $\underline{p_A}$ and $\overline{p_B}$. We leverage the method developed in [18] to address the first challenge, and we develop an efficient algorithm to address the second challenge.

Estimating $\underline{p_A}$ and $\overline{p_B}$: We view the probabilities $p_1, p_2, \cdots, p_{|\mathcal{C}|}$ as a multinomial distribution over the label set $\mathcal{C}$. If we sample a noise ϵ from our noise distribution uniformly at random, then $f(\mathbf{s} \oplus \epsilon)$ can be viewed as a sample from the multinomial distribution. Then, estimating $\underline{p_A}$ and $\overline{p_c}$ for $c \in \mathcal{C} \setminus \{c_A\}$ is essentially a one-sided simultaneous confidence interval estimation problem. In particular, we leverage the simultaneous confidence interval estimation method developed in [18] to estimate these bounds with a confidence level at least $1 - \alpha$. Specifically, we sample d random noise, i.e., $\epsilon^1, \epsilon^2, \cdots, \epsilon^d$, from the noise distribution defined in Equation 1. We denote by d_c the frequency of the label c predicted by the base classifier for the d noisy examples. Formally, we have $d_c = \sum_{j=1}^{d} \mathbb{I}(f(\mathbf{s} \oplus \epsilon^j) = c)$ for each $c \in \mathcal{C}$ and $\mathbb{I}$ is the indicator function. Moreover, we assume c_A has the largest frequency and the smoothed classifier predicts c_A as the label, i.e., $g(\mathbf{s}) = c_A$. According to [18], we have the following probability bounds with a confidence level at least $1 - \alpha$:

$$\underline{p_A} = B\left(\frac{\alpha}{|\mathcal{C}|}; d_{c_A}, d - d_{c_A} + 1\right) \tag{13}$$

$$\overline{p_c} = B\left(1 - \frac{\alpha}{|\mathcal{C}|}; d_c + 1, d - d_c\right), \ \forall c \neq c_A, \tag{14}$$

where $B(q; u, w)$ is the qth quantile of a beta distribution with shape parameters u and w. Then, we estimate $\overline{p_B}$ as follows:

$$\overline{p_B} = \min\left(\max_{c \neq c_A} \overline{p_c}, 1 - \underline{p_A}\right). \tag{15}$$

Computing K: After estimating $\underline{p_A}$ and $\overline{p_B}$, we solve the optimization problem in Equation 11 to obtain K. First, we have:

$$\Pr(Y \in \mathcal{A}) = \sum_{j=1}^{a^\star - 1} \Pr(Y \in \mathcal{R}_j) + (\underline{p_A} - \sum_{j=1}^{a^\star - 1} \Pr(X \in \mathcal{R}_j))/r_{a^\star}, \tag{16}$$

$$\Pr(Y \in \mathcal{B}) = \sum_{j=b^\star + 1}^{2n+1} \Pr(Y \in \mathcal{R}_j) + (\overline{p_B} - \sum_{j=b^\star + 1}^{2n+1} \Pr(X \in \mathcal{R}_j))/r_{b^\star}, \tag{17}$$

where $r_{a^\star}$ and $r_{b^\star}$ are the density ratios in the regions $\mathcal{R}_{a^\star}$ and $\mathcal{R}_{b^\star}$, respectively. Therefore, the key to solve the optimization problem is to compute $\Pr(X \in \mathcal{R}(m))$ and $\Pr(Y \in \mathcal{R}(m))$ for each $m \in \{-n, -n+1, \cdots, n\}$ when $||\delta||_0 = k$. Specifically, we have:

$$\Pr(X \in \mathcal{R}(m)) = \sum_{j=\max\{0,m\}}^{\min\{n,n+m\}} \beta^{n-(j-m)}(1-\beta)^{(j-m)} \cdot t(m,j) \tag{18}$$

$$\Pr(Y \in \mathcal{R}(m)) = \sum_{j=\max\{0,m\}}^{\min\{n,n+m\}} \beta^{n-j}(1-\beta)^{j} \cdot t(m,j), \tag{19}$$

where $t(m,j)$ is defined as follows:

$$t(m,j) = \begin{cases} 0, & \text{if } (m+k) \bmod 2 \neq 0, \\ 0, & \text{if } 2j - m < k, \\ \binom{n-k}{\frac{2j-m-k}{2}}\binom{k}{\frac{k-m}{2}}, & \text{otherwise.} \end{cases} \tag{20}$$

See Section B for the details on obtaining Equation 18 and 19. Then, we iteratively find the largest k such that the constraint in Equation 12 is satisfied.

4 Evaluation

4.1 Experimental Setup

We evaluate our method on multiple GNNs and benchmark datasets for both node classification and graph classification.

Benchmark datasets and GNNs: We use benchmark graphs and GNNs for both node and graph classification. Table 1 shows the statistics of our graphs.

- **Node classification**: We consider Graph Convolutional Network (GCN) [22] and Graph Attention Network (GAT) [43] for node classification. Moreover, we use the Cora, Citeseer, and Pubmed datasets [38]. They are citation graphs, where nodes are documents and edges indicate citations between them. In particular, an undirected edge between two documents is created if one cites the other. The bag-of-words feature of a document is treated as the node feature. Each document also has a label.

- **Graph classification**: We consider Graph Isomorphism Network (GIN) [52] for graph classification. Moreover, we use the MUTAG, PROTEINS, and IMDB datasets [54]. MUTAG and PROTEINS are bioinformatics datasets. MUTAG contains 188 mutagenic aromatic and heteroaromatic nitro compounds, where each compound represents a graph and each label means whether or not the compound has a mutagenic effect on the Gram-negative bacterium *Salmonella typhimurium*. PROTEINS is a dataset where nodes are secondary structure elements and there is an edge between two nodes if they are neighbors in the amino-acid sequence or in three-dimensional space. Each protein is represented as a graph and is labeled as enzyme or non-enzyme. IMDB is a movie collaboration dataset. Each graph is an ego-network of an actor/actress, where nodes are actors/actresses and an edge between two actors/actresses indicates that they appear in the same movie. Each graph is obtained from a certain genre of movies, and the task is to classify the genre of a graph.

Table 1: Dataset statistics.

Dataset		#Nodes	#Edges	#Classes
Node Classification	**Cora**	2,708	5,429	7
	Citeseer	3,327	4,732	6
	Pubmed	19,717	44,338	3
Dataset		#Graphs	Ave.#Nodes	#Classes
Graph Classification	**MUTAG**	188	17.9	2
	PROTEINS	1,113	39.1	2
	IMDB	1,500	13.0	3

Training and testing: For each node classification dataset, following previous works [22, 43], we sample 20 nodes from each class uniformly at random as the training dataset. Moreover, we randomly sample 100 nodes for testing. For each graph classification dataset, we use 90% of the graphs for training and the remaining 10% for testing, similar to [52].

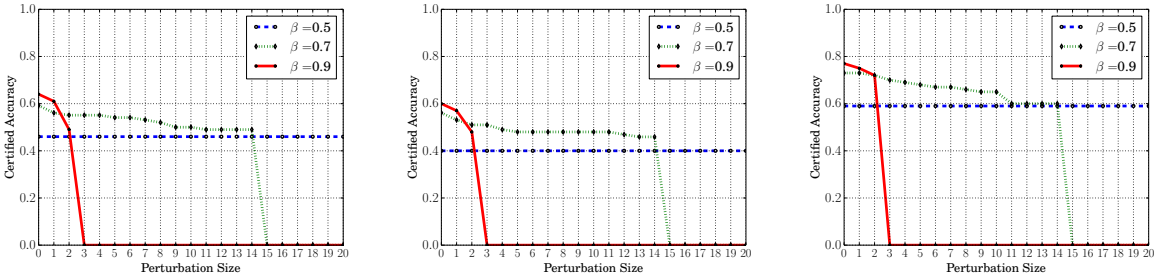
Implementation and parameter setting: We implement our method in pyTorch. To compute the certified perturbation size, our method needs to specify the noise parameter β, the confidence level $1-\alpha$, and the number of samples d. Unless otherwise mentioned, we set $\beta = 0.7$, $1-\alpha = 99.9\%$, and $d = 10,000$. We also explore the impact of each parameter while fixing the other parameters to the default settings in our experiments. When computing the certified perturbation size for a node u in node classification, we consider an attacker perturbs the connection status between u and the remaining nodes in the graph. We use the publicly available source code for GCN, GAT, and GIN. Moreover, we use the default settings in the source code to train classifiers. We use our randomized smoothing to smooth each classifier.

4.2 Experimental Results

Like Cohen et al. [8], we use *certified accuracy* as the metric to evaluate our method. Specifically, for a smoothed GNN classifier and a given perturbation size, certified accuracy is the fraction of testing nodes (for node classification) or testing graphs (for graph classification), whose labels are correctly predicted by the smoothed classifier and whose certified perturbation size is no smaller than the given perturbation size.

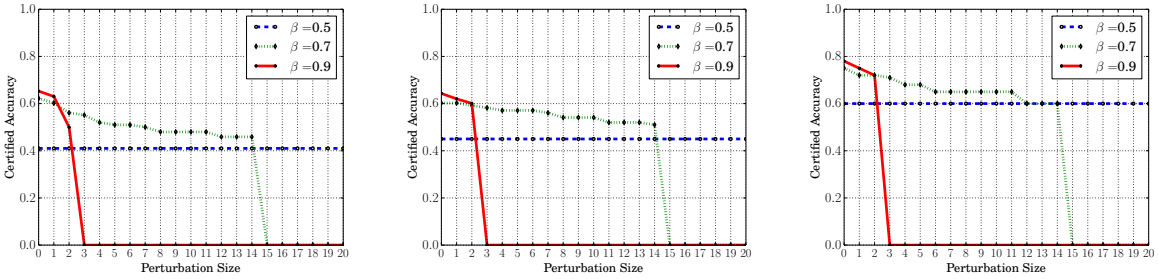
Impact of the noise parameter β: Figure 1 and Figure 2 respectively show the certified accuracy of the smoothed GCN and smoothed GAT vs. perturbation size for different β on the three node classification datasets. Figure 3 shows the certified accuracy of GIN with randomized smoothing vs. perturbation size for different β on the three graph classification datasets.

We have two observations. First, when $\beta = 0.5$, the certified accuracy is independent with the perturbation size, which means that an attacker can not attack a smoothed GNN classifier via perturbing the graph structure. This is because $\beta = 0.5$ essentially means that the graph is sampled from the space $\{0,1\}^n$ uniformly at random. In other words, the graph structure does not contain information about the node labels and a smoothed GNN classifier is reduced to only using the node features. Second, β controls a tradeoff between accuracy under no attacks and robustness. Specifically, when β is larger, the accuracy under no attacks



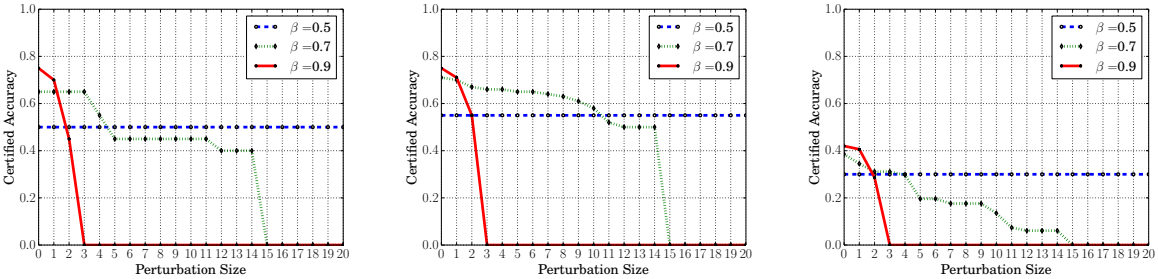
(a) Cora (b) Citeseer (c) Pubmed

Figure 1: Impact of β on the certified accuracy of the smoothed GCN.



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

Figure 2: Impact of β on the certified accuracy of the smoothed GAT.

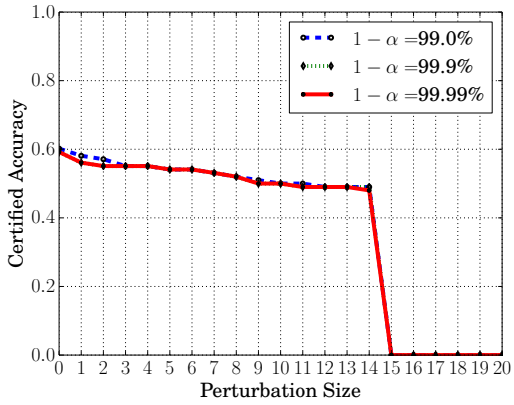


(a) MUTAG (b) PROTEINS (c) IMDB

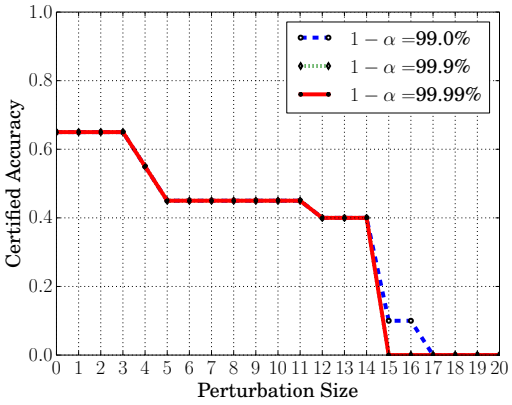
Figure 3: Impact of β on the certified accuracy of the smoothed GIN.

(i.e., perturbation size is 0) is larger, but the certified accuracy drops more quickly as the perturbation size increases.

Impact of the confidence level $1-\alpha$: Figure 4a and 4b show the certified accuracy of the smoothed GCN and smoothed GIN vs. perturbation size for different confidence levels, respectively. We observe that as the confidence level $1-\alpha$ increases, the certified accuracy curve becomes slightly lower. This is because a higher confidence level leads to a looser estimation of the probability bound $\underline{p_A}$ and $\overline{p_B}$, which means a smaller certified perturbation size for a testing node/graph. However, the differences of the certified accuracies between different

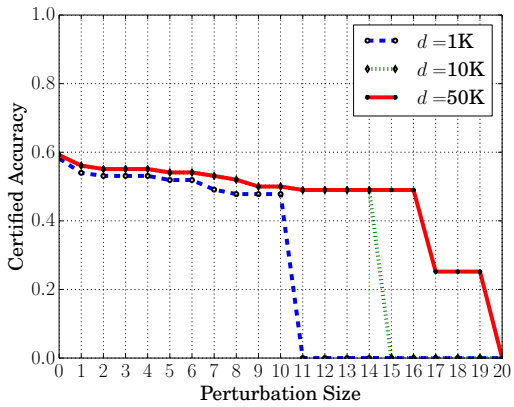


(a) Smoothed GCN on Cora

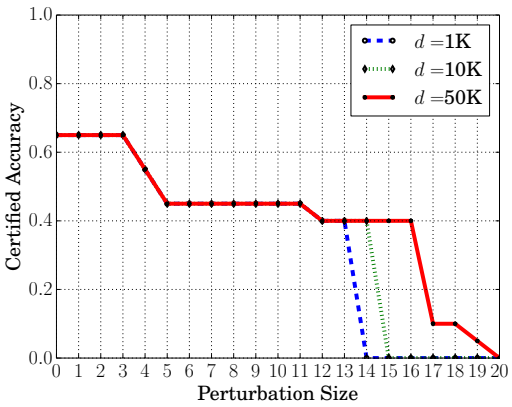


(b) Smoothed GIN on MUTAG

Figure 4: Impact of $1-\alpha$ on the certified accuracy of (a) the smoothed GCN and (b) the smoothed GIN.



(a) Smoothed GCN on Cora



(b) Smoothed GIN on MUTAG

Figure 5: Impact of d on the certified accuracy of (a) the smoothed GCN and (b) the smoothed GIN.

confidence levels are negligible when the confidence levels are large enough.

Impact of the number of samples d: Figure 5a and 5b show the certified accuracy of the smoothed GCN and smoothed GIN vs. perturbation size for different numbers of samples d, respectively. We observe that as the number of samples increases, the certified accuracy curve becomes higher. This is because a larger number of samples makes the estimated probability bound $\underline{p_A}$ and $\overline{p_B}$ tighter, which means a larger certified perturbation size for a testing node/graph. On Cora, the smoothed GCN achieves certified accuracies of 0.55, 0.50, and 0.49 when $d = 50K$ and the attacker arbitrarily adds/deletes at most 5, 10, and 15 edges, respectively. On MUTAG, GIN with our randomized smoothing achieves certified accuracies of 0.45, 0.45, and 0.40 when $d = 50K$ and the attacker arbitrarily adds/deletes at most 5, 10, and 15 edges, respectively.

5 Related Work

We review studies on certified robustness for classifiers on both non-graph data and graphs.

5.1 Non-graph Data

Various methods have been tried to certify robustness of classifiers. A classifier is certifiably robust if it predicts the same label for data points in a certain region around an input example. Existing certification methods leverage Satisfiability Modulo Theories [37, 5, 12, 21], mixed integer-linear programming [7, 14, 3], linear programming [48, 49], semidefinite programming [34, 35], dual optimization [10, 11], abstract interpretation [15, 30, 39], and layer-wise relaxtion [47, 56]. However, these methods are not scalable to large neural networks and/or are only applicable to specific neural network architectures.

Randomized smoothing [4, 29, 24, 28, 8, 36, 25, 55, 27, 18] was originally developed to defend against adversarial examples. It was first proposed as an empirical defense [4, 29]. For instance, Cao and Gong [4] proposed to use uniform random noise from a hypercube centered at an input example to smooth a base classifier. Lecuyer et al. [24] derived a certified robustness guarantee for randomized smoothing with Gaussian noise or Laplacian noise via differential privacy techniques. Li et al. [28] leveraged information theory to derive a tighter certified robustness guarantee. Cohen et al. [8] leveraged the Neyman-Pearson Lemma [31] to obtain a tight certified robustness guarantee under L_2-norm for randomized smoothing with Gaussian noise. Specifically, they showed that a smoothed classifier verifiably predicts the same label when the L_2-norm of the adversarial perturbation to an input example is less than a threshold. Salman et al. [36] designed an adaptive attack against smoothed classifiers and used the attack to train classifiers in an adversarial training paradigm to enlarge the certified robuseness under L_2-norm. Zhai et al. [55] proposed to explicitly maximize the certified robustness via a new training strategy. All these certification methods focused on top-1 predictions. Jia et al. [18] extended [8] to derive the first certification of top-k predictions against adversarial examples. Compared to other methods for certified robustness, randomized smoothing has two key advantages: 1) it is scalable to large neural networks, and 2) it is applicable to any base classifier.

However, existing randomized smoothing methods are insufficient to certify robustness of GNNs against structural perturbations. Specifically, they focus on continuous input examples and add Gaussian or Laplacian noise to the input examples. However, graph structure is essentially binary data, and Gaussian or Laplacian noise is not semantically meaningful for the graph structure.

5.2 Graph Data

Several studies [9, 60, 61, 1, 44, 50, 51, 40, 6] have shown that attackers can fool GNNs via manipulating the node features and/or graph structure. [50, 51, 58, 42, 13] proposed empirical defenses without certified robustness guarantees. To the best of our knowledge, [62] and [2] are the only existing works to study certified robustness of GNNs. Zugner et al. [62] consider node feature perturbation. In particular, Zugner et al. [62] derived certified robustness for a particular type of GNN called graph convolutional network [22] against node feature perturbation. More specifically, they formulated the certified robustness as a linear programming problem, where the objective is to require that a node's prediction is constant within an allowable node

feature perturbation. Then, they derived the robustness guarantee based on the dual form of the optimization problem, which is motivated by [48]. Bojchevski et al. [2] and Zügner et al. [63] consider structural perturbation. However, both the two works are only applicable to a specific GNN. Specifically, Bojchevski et al. [2] require that GNN prediction is a linear function of (personalized) PageRank [23] and Zügner et al. [63] can only certify the robustness of GCNs. We note that a recent work [19] leveraged randomized smoothing to certify the robustness of community detection against adversarial structural perturbation. They essentially model the problem as binary classification, i.e., whether a set of nodes are in the same community or not, and then leverage randomized smoothing to certify its robustness. In contrast, our work focuses on certified robustness of multi-class classification of GNN against structural perturbation. Furthermore, our method can be applied to any GNN and we show the tightness of our certified robustness guarantee.

6 Conclusion and Future Work

In this work, we develop the certified robustness guarantee of GNNs for both node and graph classifications against structural perturbation. Our results are applicable to any GNN classifier. Our certification is based on randomized smoothing for binary data which we develop in this work. Moreover, we prove that our certified robustness guarantee is tight when randomized smoothing is used and no assumptions on the GNNs are made. Interesting future work includes 1) extending our method to certify robustness of GNNs against both node-feature and structural perturbations, and 2) incorporating the information of a given GNN to further improve the certified robustness guarantee.

References

[1] Aleksandar Bojchevski and Stephan Günnemann. Adversarial attacks on node embeddings via graph poisoning. In *ICML*, 2019.

[2] Aleksandar Bojchevski and Stephan Günnemann. Certifiable robustness to graph perturbations. In *NeurIPS*, 2019.

[3] Rudy R Bunel, Ilker Turkaslan, Philip Torr, Pushmeet Kohli, and Pawan K Mudigonda. A unified view of piecewise linear neural network verification. In *NeurIPS*, 2018.

[4] Xiaoyu Cao and Neil Zhenqiang Gong. Mitigating evasion attacks to deep neural networks via region-based classification. In *ACSAC*, 2017.

[5] Nicholas Carlini, Guy Katz, Clark Barrett, and David L Dill. Provably minimally-distorted adversarial examples. *arXiv*, 2017.

[6] Heng Chang, Yu Rong, Tingyang Xu, Wenbing Huang, Honglei Zhang, Peng Cui, Wenwu Zhu, and Junzhou Huang. A restricted black-box adversarial framework towards attacking graph embedding models. In *AAAI*, 2020.

[7] Chih-Hong Cheng, Georg Nührenberg, and Harald Ruess. Maximum resilience of artificial neural networks. In *ATVA*, 2017.

[8] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

[9] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. Adversarial attack on graph structured data. In *ICML*, 2018.

[10] Krishnamurthy Dvijotham, Sven Gowal, Robert Stanforth, Relja Arandjelovic, Brendan O'Donoghue, Jonathan Uesato, and Pushmeet Kohli. Training verified learners with learned verifiers. *arXiv*, 2018.

[11] Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. In *UAI*, 2018.

[12] Ruediger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *ATVA*, 2017.

[13] 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 *WSDM*, 2020.

[14] Matteo Fischetti and Jason Jo. Deep neural networks and mixed integer linear optimization. *Constraints*, 2018.

[15] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *IEEE S & P*, 2018.

[16] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *ICML*, 2017.

[17] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017.

[18] Jinyuan Jia, Xiaoyu Cao, Binghui Wang, and Neil Zhenqiang Gong. Certified robustness for top-k predictions against adversarial perturbations via randomized smoothing. In *ICLR*, 2020.

[19] Jinyuan Jia, Binghui Wang, Xiaoyu Cao, and Neil Zhenqiang Gong. Certified robustness of community detection against adversarial structural perturbation via randomized smoothing. In *The Web Conference*, 2020.

[20] Jinyuan Jia, Binghui Wang, Le Zhang, and Neil Zhenqiang Gong. Attriinfer: Inferring user attributes in online social networks using markov random fields. In *WWW*, 2017.

[21] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *CAV*, 2017.

[22] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[23] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *ICLR*, 2019.

[24] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE S & P*, 2019.

[25] Guang-He Lee, Yang Yuan, Shiyu Chang, and Tommi S Jaakkola. Tight certificates of adversarial robustness for randomly smoothed classifiers. In *NeurIPS*, 2019.

[26] Erich L Lehmann and Joseph P Romano. *Testing statistical hypotheses*. Springer Science & Business Media, 2006.

[27] Alexander Levine and Soheil Feizi. Robustness certificates for sparse adversarial attacks by randomized ablation. In *AAAI*, 2020.

[28] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. 2019.

[29] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *ECCV*, 2018.

[30] Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, 2018.

[31] Jerzy Neyman and Egon Sharpe Pearson. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706):289–337, 1933.

[32] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *WWW*, 2007.

[33] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[34] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *ICLR*, 2018.

[35] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *NeurIPS*, 2018.

[36] Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, 2019.

[37] Karsten Scheibler, Leonore Winterer, Ralf Wimmer, and Bernd Becker. Towards verification of artificial neural networks. In *MBMV*, 2015.

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

[39] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *NeurIPS*, 2018.

[40] 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 *The Web Conference*, 2020.

[41] Acar Tamersoy, Kevin Roundy, and Duen Horng Chau. Guilt by association: large scale malware detection by mining file-relation graphs. In *KDD*, 2014.

[42] Xianfeng Tang, Yandong Li, Yiwei Sun, Huaxiu Yao, Prasenjit Mitra, and Suhang Wang. Transferring robustness for graph neural network against poisoning attacks. In *WSDM*, 2020.

[43] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[44] Binghui Wang and Neil Zhenqiang Gong. Attacking graph-based classification via manipulating the graph structure. In *CCS*, 2019.

[45] Binghui Wang, Neil Zhenqiang Gong, and Hao Fu. Gang: Detecting fraudulent users in online social networks via guilt-by-association on directed graphs. In *ICDM*, 2017.

[46] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. In *KDD Workshop*, 2019.

[47] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks. In *ICML*, 2018.

[48] Eric Wong and J Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

[49] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. In *NeurIPS*, 2018.

[50] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. Adversarial examples on graph data: Deep insights into attack and defense. In *IJCAI*, 2019.

[51] 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 *IJCAI*, 2019.

[52] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[53] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *ICML*, 2018.

[54] Pinar Yanardag and SVN Vishwanathan. Deep graph kernels. In *KDD*, 2015.

[55] Runtian Zhai, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang. Macer: Attack-free and scalable robust training via maximizing certified radius. In *ICLR*, 2020.

[56] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *NeurIPS*, 2018.

[57] Elena Zheleva and Lise Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *WWW*, 2009.

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

[59] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.

[60] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *KDD*, 2018.

[61] Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *ICLR*, 2019.

[62] Daniel Zügner and Stephan Günnemann. Certifiable robustness and robust training for graph convolutional networks. In *KDD*, 2019.

[63] Daniel Zügner and Stephan Günnemann. Certifiable robustness of graph convolutional networks under structure perturbations. In *KDD*, 2020.

A Proofs

A.1 Proof of Lemma 1

We first describe the Neyman-Pearson Lemma, which we use to prove Lemma 1.

Lemma 2 (Neyman-Pearson Lemma for Binary Random Variables)**.** *Let X and Y be two random variables in the discrete space $\{0,1\}^n$ with probability distributions $Pr(X)$ and $Pr(Y)$, respectively. Let $h : \{0,1\}^n \to \{0,1\}$ be a random or deterministic function.*

- *Let $S_1 = \{\mathbf{z} \in \{0,1\}^n : \frac{Pr(X=\mathbf{z})}{Pr(Y=\mathbf{z})} > r\}$ and $S_2 = \{\mathbf{z} \in \{0,1\}^n : \frac{Pr(X=\mathbf{z})}{Pr(Y=\mathbf{z})} = r\}$ for some $r > 0$. Assume $S_3 \subseteq S_2$ and $S = S_1 \bigcup S_3$. If $Pr(h(X) = 1) \geq Pr(X \in S)$, then $Pr(h(Y) = 1) \geq Pr(Y \in S)$.*
- *Let $S_1 = \{\mathbf{z} \in \{0,1\}^n : \frac{Pr(X=\mathbf{z})}{Pr(Y=\mathbf{z})} < r\}$ and $S_2 = \{\mathbf{z} \in \{0,1\}^n : \frac{Pr(X=\mathbf{z})}{Pr(Y=\mathbf{z})} = r\}$ for some $r > 0$. Assume $S_3 \subseteq S_2$ and $S = S_1 \bigcup S_3$. If $Pr(h(X) = 1) \leq Pr(X \in S)$, then $Pr(h(Y) = 1) \leq Pr(Y \in S)$.*

Proof. The proof can be found in a standard statistics textbook, e.g., [26]. For completeness, we include the proof here. Without loss of generality, we assume that h is random and denote $h(1|z)$ (resp. $h(0|z)$) as the probability that $h(z) = 1$ (resp. $h(z) = 0$)). We denote S^c as the

complement of S, i.e., $S^c = \{0,1\}^n \setminus S$. For any $\mathbf{z} \in S$, we have $\frac{\Pr(X=\mathbf{z})}{\Pr(Y=\mathbf{z})} \geq r$, and for any $\mathbf{z} \in S^c$, we have $\frac{\Pr(X=\mathbf{z})}{\Pr(Y=\mathbf{z})} \leq r$. We prove the first part, and the second part can be proved similarly.

$$
\begin{aligned}
&\Pr(h(Y)=1) - \Pr(Y \in S) = \\
&\sum_{\mathbf{z}\in\{0,1\}^n} h(1|\mathbf{z})\Pr(Y=\mathbf{z}) - \sum_{\mathbf{z}\in S}\Pr(Y=\mathbf{z}) \\
&= \Big(\sum_{\mathbf{z}\in S^c} h(1|\mathbf{z})\Pr(Y=\mathbf{z}) + \sum_{\mathbf{z}\in S} h(1|\mathbf{z})\Pr(Y=\mathbf{z})\Big) \\
&\quad - \Big(\sum_{\mathbf{z}\in S} h(1|\mathbf{z})\Pr(Y=\mathbf{z}) + \sum_{\mathbf{z}\in S} h(0|\mathbf{z})\Pr(Y=\mathbf{z})\Big) \\
&= \sum_{\mathbf{z}\in S^c} h(1|\mathbf{z})\Pr(Y=\mathbf{z}) - \sum_{\mathbf{z}\in S} h(0|\mathbf{z})\Pr(Y=\mathbf{z}) \\
&\geq \frac{1}{r}\Big(\sum_{\mathbf{z}\in S^c} h(1|\mathbf{z})\Pr(X=\mathbf{z}) - \sum_{\mathbf{z}\in S} h(0|\mathbf{z})\Pr(X=\mathbf{z})\Big) \\
&= \frac{1}{r}\Bigg(\Big(\sum_{\mathbf{z}\in S^c} h(1|\mathbf{z})\Pr(X=\mathbf{z}) + \sum_{\mathbf{z}\in S} h(1|\mathbf{z})\Pr(X=\mathbf{z})\Big) \\
&\quad - \Big(\sum_{\mathbf{z}\in S} h(1|\mathbf{z})\Pr(X=\mathbf{z}) + \sum_{\mathbf{z}\in S} h(0|\mathbf{z})\Pr(X=\mathbf{z})\Big)\Bigg) \\
&= \frac{1}{r}\Big(\sum_{\mathbf{z}\in\{0,1\}^n} h(1|\mathbf{z})\Pr(X=\mathbf{z}) - \sum_{\mathbf{z}\in S}\Pr(X=\mathbf{z})\Big) \\
&= \frac{1}{r}\Big(\Pr(h(X)=1) - \Pr(X\in S)\Big) \\
&\geq 0.
\end{aligned}
$$

□

Next, we restate Lemma 1 and show our proof.

Lemma 1. *We have the following bounds:*

$$Pr(f(Y)=c_A) \geq Pr(Y\in\mathcal{A}), \tag{7}$$

$$Pr(f(Y)=c_B) \leq Pr(Y\in\mathcal{B}). \tag{8}$$

Proof. Based on the regions $\mathcal{A}$ and $\mathcal{B}$ defined in Equation 5 and Equation 6, we have $\Pr(X \in \mathcal{A}) = \underline{p_A}$ and $\Pr(X \in \mathcal{B}) = \overline{p_B}$.

Moreover, based on the conditions in Equation 4, we have:

$$\Pr(f(X)=c_A) \geq \underline{p_A} = \Pr(X\in\mathcal{A});$$

$$\Pr(f(X)=c_B) \leq \overline{p_B} = \Pr(X\in\mathcal{B}).$$

Next, we leverage Lemma 2 to derive the condition for $\Pr(f(Y)=c_A) > \Pr(f(Y)=c_B)$. Specifically, we define $h(\mathbf{z}) = \mathbb{I}(f(\mathbf{z})=c_A)$. Then, we have:

$$\Pr(h(X)=1) = \Pr(f(X)=c_A) \geq \Pr(X\in\mathcal{A}).$$

Moreover, we have $\frac{\Pr(X=\mathbf{z})}{\Pr(Y=\mathbf{z})} > r_{a^\star}$ for any $\mathbf{z} \in \bigcup_{j=1}^{a^\star-1} \mathcal{R}_j$ and $\frac{\Pr(X=\mathbf{z})}{\Pr(Y=\mathbf{z})} = r_{a^\star}$ for any $\mathbf{z} \in \underline{\mathcal{R}_{a^\star}}$, where $r_{a^\star}$ is the probability density ratio in the region $\mathcal{R}_{a^\star}$. Therefore, according to the first part of Lemma 2, we have:

$$\Pr(f(Y) = c_A) \geq \Pr(Y \in \mathcal{A}).$$

Similarly, based on the second part of Lemma 2, we have:

$$\Pr(f(Y) = c_B) \leq \Pr(Y \in \mathcal{B}).$$

□

A.2 Proof of Theorem 1

Recall that our goal is to make $\Pr(f(Y) = c_A) > \Pr(f(Y) = c_B)$. Based on Lemma 1, it is sufficient to require:

$$\Pr(Y \in \mathcal{A}) > \Pr(Y \in \mathcal{B}). \tag{21}$$

Therefore, we derive the certified perturbation size K as the maximum k such that the above inequality holds for $\forall ||\delta||_0 = R$.

A.3 Proof of Theorem 2

Our idea is to construct a base classifier f^* consistent with the conditions in Equation 4, but the smoothed classifier is not guaranteed to predict c_A. Let disjoint regions $\mathcal{A}$ and $\mathcal{B}$ be defined as in Equation 5 and Equation 6. We denote $\Gamma = \{1, 2, \cdots, c\} \setminus \{c_A, c_B\}$. Then, for each label c_i in Γ, we can find a region $\mathcal{C}_i \subseteq \{0,1\}^n \setminus (\mathcal{A} \cup \mathcal{B})$ such that $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset, \forall i, j \in \Gamma, i \neq j$ and $\Pr(X \in \mathcal{C}_i) \leq \overline{p_B}$. We can construct such regions because $\underline{p_A} + (|\mathcal{C}| - 1) \cdot \overline{p_B} \geq 1$. Given those regions, we construct the following base classifier:

$$f^*(\mathbf{z}) = \begin{cases} c_A & \text{if } \mathbf{z} \in \mathcal{A} \\ c_B & \text{if } \mathbf{z} \in \mathcal{B} \\ c_i & \text{if } \mathbf{z} \in \mathcal{C}_i \end{cases}$$

By construction, we have $\Pr(f^*(X) = c_A) = \underline{p_A}$, $\Pr(f^*(X) = c_B) = \overline{p_B}$, and $\Pr(f^*(X) = c_i) \leq \overline{p_B}$ for any $c_i \in \Gamma$, which are consistent with Equation 4. From our proof of Theorem 1, we know that when $||\delta||_0 > K$, we have:

$$\Pr(Y \in \mathcal{A}) \leq \Pr(Y \in \mathcal{B}),$$

or equivalently we have:

$$\Pr(f^*(Y) = c_A) \leq \Pr(f^*(Y) = c_B)$$

Therefore, we have either $g(\mathbf{s} \oplus \delta) \neq c_A$ or there exist ties when $||\delta||_0 > K$.

B Computing $\Pr(X \in \mathcal{R}(m))$ and $\Pr(Y \in \mathcal{R}(m))$

We first define the following regions:

$$\mathcal{R}(w,v) = \{\mathbf{z} \in \{0,1\}^n : ||\mathbf{z}-\mathbf{s}||_0 = w \text{ and } ||\mathbf{z}-\mathbf{s}\oplus\delta||_0 = v\},$$

for $w, v \in \{0, 1, \cdots, n\}$. Intuitively, $\mathcal{R}(w,v)$ includes the binary vectors that are w bits different from $\mathbf{s}$ and v bits different from $\mathbf{s}\oplus\delta$. Next, we compute the size of the region $\mathcal{R}(w,v)$ when $||\delta||_0 = k$. Without loss of generality, we assume $\mathbf{s} = [0, 0, \cdots, 0]$ as a zero vector and $\mathbf{s}\oplus\delta = [1, 1, \cdots, 1, 0, 0, \cdots, 0]$, where the first k entries are 1 and the remaining $n-k$ entries are 0. We construct a binary vector $\mathbf{z} \in \mathcal{R}(w,v)$. Specifically, suppose we flip i zeros in the last $n-k$ zeros in both $\mathbf{s}$ and $\mathbf{s}\oplus\delta$. Then, we flip $w-i$ of the first k bits of $\mathbf{s}$ and flip the rest $k-w+i$ bits of the first k bits of $\mathbf{s}\oplus\delta$. In order to have $\mathbf{z} \in \mathcal{R}(w,v)$, we need $k-w+i+i = v$, i.e., $i = (w+v-k)/2$. Therefore, we have the size $|\mathcal{R}(w,v)|$ of the region $\mathcal{R}(w,v)$ as follows:

$$|\mathcal{R}(w,v)| = \begin{cases} 0, & \text{if } (w+v-k) \bmod 2 \neq 0, \\ 0, & \text{if } w+v < k, \\ \binom{n-k}{\frac{w+v-k}{2}}\binom{k}{\frac{w-v+k}{2}}, & \text{otherwise} \end{cases}$$

Moreover, for each $\mathbf{z} \in \mathcal{R}(w,v)$, we have $\Pr(X = \mathbf{z}) = \beta^{n-w}(1-\beta)^w$ and $\Pr(Y = \mathbf{z}) = \beta^{n-v}(1-\beta)^v$. Therefore, we have:

$$\begin{aligned} \Pr(X \in \mathcal{R}(w,v)) &= \beta^{n-w}(1-\beta)^w \cdot |\mathcal{R}(w,v)|, \\ \Pr(Y \in \mathcal{R}(w,v)) &= \beta^{n-v}(1-\beta)^v \cdot |\mathcal{R}(w,v)|. \end{aligned}$$

Note that $\mathcal{R}(m) = \cup_{v-w=m}\mathcal{R}(w,v)$. Therefore, we have:

$$\begin{aligned} &\Pr(X \in \mathcal{R}(m)) \\ =&\Pr(X \in \cup_{v-w=m}\mathcal{R}(w,v)) \\ =&\Pr(X \in \cup_{j=\max(0,m)}^{\min(n,n+m)}\mathcal{R}(j-m,j)) \\ =&\sum_{j=\max(0,m)}^{\min(n,n+m)} \Pr(X \in \mathcal{R}(j-m,j)) \\ =&\sum_{j=\max(0,m)}^{\min(n,n+m)} \beta^{n-(j-m)}(1-\beta)^{j-m} \cdot |\mathcal{R}(j-m,j)| \\ =&\sum_{j=\max(0,m)}^{\min(n,n+m)} \beta^{n-(j-m)}(1-\beta)^{j-m} \cdot t(m,j), \end{aligned}$$

where $t(m,j) = |\mathcal{R}(j-m,j)|$.

Similarly, we have

$$\Pr(Y \in \mathcal{R}(m)) = \sum_{j=\max(0,m)}^{\min(n,n+m)} \beta^{n-j}(1-\beta)^j \cdot t(m,j).$$