AuditVotes: a Framework towards Deployable Certified Robustness for GNNs

Yuni Lai
The Hong Kong Polytechnic University
Hong Kong, China
csylai@comp.polyu.edu.hk

Kai Zhou
The Hong Kong Polytechnic University
Hong Kong, China
kaizhou@polyu.edu.hk

Abstract

Graph Neural Networks (GNNs) are powerful but vulnerable to adversarial attacks, necessitating the research on certified robustness that can provide GNNs with robustness guarantees. Existing randomized smoothing methods struggle with a trade-off between utility and robustness due to high noise levels. We introduce **AuditVotes**, which integrates randomized smoothing with two components, augmentation and conditional smoothing, aiming to improve data and vote quality. We instantiated **AuditVotes** with simple strategies, and preliminary results demonstrate its significant promise in enhancing certified robustness, representing a substantial step toward deploying certifiably robust GNNs in real-world applications.

1 Introduction

Graph Neural Networks (GNNs) [8] have emerged as a powerful tool for learning and inference on graph-structured data. However, GNNs are vulnerable to adversarial attacks, which can significantly degrade their performance by introducing small, carefully crafted perturbations to the input data [6]. This vulnerability has urged extensive research into developing robust GNN models. One promising direction is certified robustness [5], which aims to provide provable guarantees that a model's predictions will remain stable under any possible adversarial perturbation within a specified range.

The most representative approach for achieving certified robustness is randomized smoothing [1], including our latest works [3, 4]. The basic idea is a majority voting scheme, where each vote is the prediction result of the base classifier over a *randomized* graph, produced by adding carefully calibrated random noise to the original graph. However, despite the tremendous advancements achieved on certified robustness for GNNs in recent years, the certified accuracy of existing models remains suboptimal. This hinders the practical deployment of GNNs in security-sensitive applications where robustness is paramount.

Thus, we are motivated to further improve the certification performance of certifiably robust GNNs to facilitate their deployment in real-world applications. The inefficacy of previous schemes are mainly attributed to the high degree of random noise applied to the input graph. For example, in order to guarantee a larger certified radius, a representative scheme [1] will apply a higher probability p_+ of adding edge. But, even with a small value of p_+ such as 0.01, the clean accuracy of a smoothed classifier will drop significantly. That is, there is an inherent trade-off between model utility and certification performance, controlled by the degree of randomization.

To mitigate this trade-off, we propose **AuditVotes** as a general enhancement to the randomized smoothing framework. Our general strategy is to keep a larger degree of randomization (for better certification), and instead improve the quality of the graph data as well as the quality of votes (to restore model utility). To this end, **AuditVotes** (shown in Figure 1) features two components, augmentation and conditional smoothing, that are seamlessly integrated to the randomized smoothing pipeline. Specifically, we introduce augmentation to pre-process the randomized graphs, after which they are sent to the base classifier for classification. After that, we apply conditional smoothing to the prediction results of the base classifier, through which we aim to filter out some low-quality predictions to improve the quality of votes. Combining these pre-processing of graph data and post-processing of votes, we can significantly improve the overall certification performance.

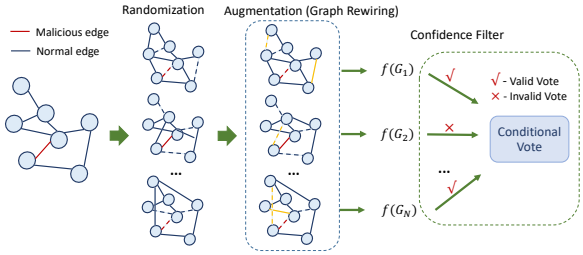


Figure 1: The Framework of AuditVotes, featuring two added components: augmentation and conditional smoothing.

In this proposal, we have demonstrated the effectiveness of **AuditVotes** by instantiating it with simple strategies, as detailed in Section 4. Specifically, we employed Jaccard similarity to enhance the randomized graphs as the augmentation and utilized prediction confidence as a filter condition to implement conditional smoothing. The evaluation results presented in Section 5 indicate that **AuditVotes** shows significant promise in improving certified robustness, marking a substantial step towards deploying certifiably robust GNNs in real-world applications. For future work, we plan to explore more advanced techniques to enhance these components further and improve the efficacy of certified robustness.

2 Background: Certified Robustness for GNNs

We represent a graph as $\mathcal{G} = (\mathbf{V}, \mathbf{E}, \mathbf{X})$, where $\mathbf{V}$ and $\mathbf{E}$ denote the set of nodes and edges, respectively, and $\mathbf{X}$ is the node feature matrix. We use $\mathbf{A}$ to denote the adjacency matrix of the graph $\mathcal{G}$. We consider the semi-supervised node classification tasks, where a GNN model $f(\cdot)$ is trained on a subset of labeled nodes $\mathbf{V}_L \subset \mathbf{V}$ and predicts the labels for the remaining nodes in $\mathbf{V}/\mathbf{V}_L$. Each node is associated with a label among $\mathcal{Y} = \{1, 2, \cdots, C\}$. It is known that GNN models are vulnerable to adversarial attacks [6] such as Graph Modification Attacks (GMA) and Graph Injection Attacks (GIA), where the attacker can modify the input graph $\mathcal{G}$ (e.g., structure attack) to mislead node classification.

Certified robustness [5] is proposed as a countermeasure that can provide *provable defense* in the sense that as long as the attacker's power is constrained, the predictions of the established classifier are theoretically guaranteed to be stable. The mainstream approach to realize certified robustness is *randomized smoothing* with representative works [1, 3, 4] calibrated for GNNs. Specifically, given an input graph $\mathcal{G}$ and *any* base classifier $f(\cdot)$ (such as a GNN), they will add random noise to $\mathcal{G}$, resulting in a collection of randomized graphs denoted by $\phi(\mathcal{G})$, where $\phi(\cdot)$ denotes the randomization process. Then, the base classifier f is used to make predictions over the random graphs, and the final prediction is obtained through majority voting. Equivalently, randomized smoothing can convert any base classifier into a *smoothed classifier* $g(\cdot)$, defined as

$$g_v(\mathcal{G}) := \arg\max_{y \in \mathcal{Y}} p_{v,y}(\mathcal{G}) := \mathbb{P}(f_v(\phi(\mathcal{G})) = y), \tag{1}$$

where $\mathbb{P}(f_v(\phi(\mathcal{G})) = y)$ denotes the probability of predicting a node v as class y. Then, it can be proved that the smoothed classifier g is provably robust with respect to a certain perturbation space $\mathcal{B}$, which defines the set of perturbations introduced by the attacker. In this paper, we take SparseSmooth [1] as an example, and certify against GMA perturbation: $B_{r_a,r_d}(\mathcal{G}) := \{\mathcal{G}' | \sum_{ij} \mathbb{I}(\mathbf{A}'_{ij} = \mathbf{A}_{ij} - 1) \leq r_d, \sum_{ij} \mathbb{I}((\mathbf{A}'_{ij} = \mathbf{A}_{ij} + 1) \leq r_a\}$, where the attacker can add at most r_a edges and delete at most r_d edges among existing nodes. The $\phi(\mathcal{G})$ is defined as adding edges with probability p_+ and removing edges with probability p_-.

3 The AuditVotes Framework

We now present **AuditVotes** as a general framework (shown in Figure 1) to improve the performance of certified robustness for GNNs. The term **AuditVotes** derives from its two essential components, **Au**gmentation and Con**dit**ional Smoothing, encapsulating our main idea of auditing or enhancing the quality of votes to bolster the majority-vote-based certification.

3.1 Noise-dependent Augmentation

The augmentation component operates on the randomized graph $\phi(\mathcal{G})$ before it goes through the base classifier f, thus serving as a *pre-processing* step. Intuitively, randomizing will inject undesirable noise into the graph, which is the fundamental cause of the trade-off between clean accuracy and the certified radius. It is observed that a lightly larger p_+ will significantly decrease the clean accuracy of the model (i.e., model utility); however, larger p_+ is essential to ensure a larger certification radius against adding-edge attacks (note that most attacks against GNNs will tend to add edges).

To address the above trade-off, our strategy is to *keep good randomization parameters* that will lead to superior certification performance, and process the randomized graph to *restore model utility*. Our choice for processing the randomized graphs is graph augmentation [2, 9], which is a widely used technique of crafting training data to improve model performance.

Effectively fitting augmentation into the randomized smoothing pipeline has several requirements. First, the augmentation process should be efficient, as it needs to process a large number of randomized graphs. Second, the augmentation should be customized to the specific randomization scheme, more specifically, the type of noise added to the graph. In Section 4, we realized a simple augmentation scheme based on node similarities to demonstrate the effectiveness of this idea. As future work, we propose a *noise-dependent inductive* graph augmenter to fulfill this task. Specifically, we use the noise as guidance to pre-train a graph generator (i.e., augmenter $\mathcal{A}$) with the objective of maximizing model utility. Then, we create a function composition $f(\mathcal{A}(\cdot))$ to be the new base classifier. With this scheme, the existing randomized-smoothing-based certificates such as [1, 3, 4] can be easily applied to our augmented model. Notably, it also offers us the freedom to design the augmenter $\mathcal{A}$.

3.2 Conditional Smoothing

Despite the effectiveness of augmentation, the base classifier could still make low-quality predictions over the processed graph. We thus further propose a *conditional smoothing* framework to *post-process* the prediction made by the base classifier before it goes into the voting procedure.

The main idea is to filter out some low-quality predictions based on certain conditions. We use $h(\cdot)$ to denote a filtering function with output $\{0, 1\}$, where 0 means *keeping* the prediction in voting and 1 otherwise. Then, we can construct a **conditional smoothed classifier** $g^c(\cdot)$ as follows:

$$g^c(\mathcal{G}) := \arg\max_{y \in \mathcal{Y}} p_{v,y}(\mathcal{G}) := \mathbb{P}(f(\phi(\mathcal{G})) = y | h(\phi(\mathcal{G})) = 0), \tag{2}$$

where $p_{v,y}(\mathcal{G})$ denotes the probability of predicting an input sample $\mathcal{G}$ as class y *conditioned on the criteria that* $h(\phi(\mathcal{G})) = 0$.

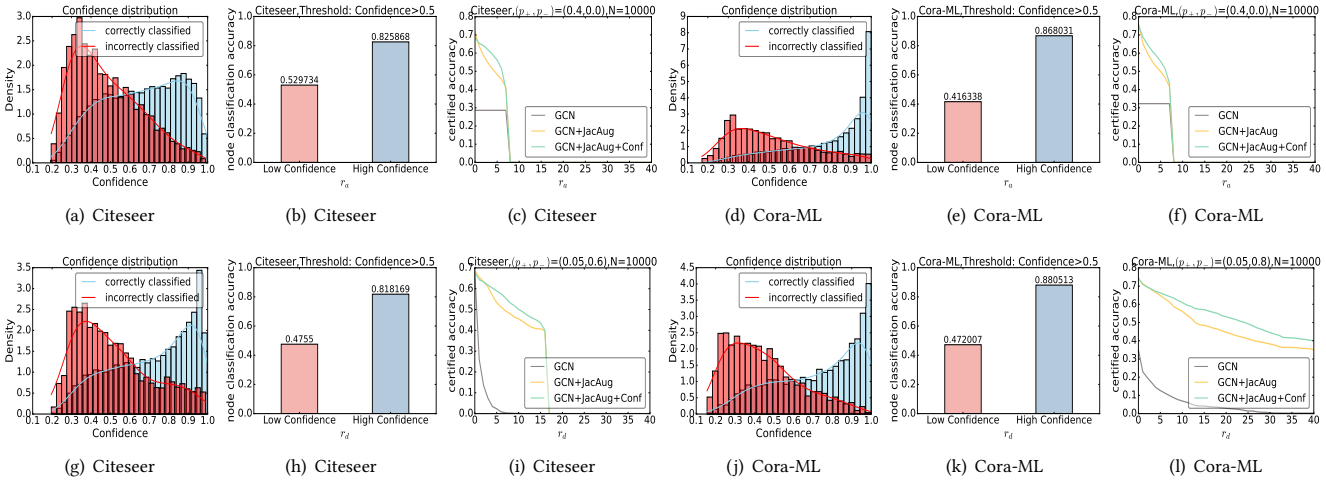
Next, we establish the condition for the conditional smoothed classifier to be certifiably robust, which is a non-trivial adaption from the existing certification method [1]. Specifically, given a node v, let y_A and y_B denote the top predicted class and the runner-up class, respectively. Let $\underline{p_A}$ and $\overline{p_B}$ denote the lower bound of p_{v,y_A} and the upper bound of $\overline{p_{v,y_B}}$, respectively.

Theorem 3.1. *Let the conditional smoothed classifier $g^c(\cdot)$ be as defined in Eq. (2). We divide $\mathbb{G}$ into disjoint regions $\mathbb{G} = \bigcup_i^I \mathcal{R}_i$, where $\mathcal{R}_i$ denote the consent likelihood ratio region that $\frac{\mathbb{P}(\phi(\mathcal{G})=Z)}{\mathbb{P}(\phi(\mathcal{G}')=Z)} = c_i$ (a constant). Let $r_i = \mathbb{P}(\phi(\mathcal{G}) \in \mathcal{R}_i)$, $r'_i = \mathbb{P}(\phi(\mathcal{G}') \in \mathcal{R}_i)$ denote the probability that the random sample fall in the partitioned region $\mathcal{R}_i$. We define μ as follows:*

$$\begin{aligned} \mu := \min_{\mathbf{s},\mathbf{t}} \quad & \mathbf{s}^T \mathbf{r}' - \mathbf{t}^T \mathbf{r}', \\ s.t. \quad & \mathbf{s}^T \mathbf{r} = \underline{p_A}, \ \mathbf{t}^T r = \overline{p_B}, \quad 0 \leq \mathbf{s} \leq 1, \ 0 \leq \mathbf{t} \leq 1. \end{aligned} \tag{3}$$

Then for $\forall \mathcal{G}' \in \mathcal{B}_{r_a,r_b}(\mathcal{G})$, we have $g_v(\mathcal{G}') = g_v(\mathcal{G})$, if $\mu > 0$.

We provide the proof for Theorem 3.1 in Appendix A. Given the condition to determine certified robustness, Theorem 3.1 lays the theoretical foundation for integrating the filter function h into the randomized smoothing framework. Notably, it offers us the freedom to design the function h to improve the quality of votes. In Section 4, we implement a simple filter function based on the confidence of the predictions made by the base classifier. In future work, we will explore the more advanced designs of h. One promising direction is to cast it as a supervised classification problem since we can generate a lot of randomized graphs offline, from which we can learn which graphs tend to give false directions.



(a) Citeseer (b) Citeseer (c) Citeseer (d) Cora-ML (e) Cora-ML (f) Cora-ML

(g) Citeseer (h) Citeseer (i) Citeseer (j) Cora-ML (k) Cora-ML (l) Cora-ML

Figure 2: Certified accuracy and result analysis of AuditVotes model.

4 Instantiation

In this section, we give simple examples to instantiate the two components, augmentation and conditional smoothing, in our **AuditVotes** framework. We explore the design of these two components in our future work.

Augmentation based on Similarity. Inspired by the simple but effective defense approach proposed for GNNs, the GCNJaccard [7], we propose a simple graph rewiring augmentation. We set a threshold τ to prune the edge with $J_{u,v} > \tau$, and we add the edge if the Jaccard similarity exceeds a threshold ξ. To reduce computation overload and avoid repeated similarity computation, we only need to pre-calculate Jaccard similarity for all the potential edges ($O(n^2)$), which we denote by a Jaccard matrix $\mathbf{J}$. Then, at the time of inference, we only need to apply the mask to conduct the graph rewiring augmentation efficiently:

$$\mathbf{A}' = \mathbf{A} \circ (\mathbf{J} > \tau) + (\mathbf{J} > \xi) \circ (\mathbf{A} = 0), \tag{4}$$

where $\circ$ denotes element-wise matrix multiplication, and $(\mathbf{J} > \tau)$ denotes the matrix with each element assigned with 1 if $J_{u,v} > \tau$, otherwise 0. Similarly, $(\mathbf{A} = 0)$ denotes a matrix with element assigned with 1 if $\mathbf{A}_{u,v} = 0$, otherwise 0.

Filtering based on confidence. In this paper, we find that simply employing the prediction confidence of the base classification can effectively improve the certified accuracy. For any random sample $\forall \mathcal{G} \in \mathbb{G}$, the node classifier f will also return the confidence for the prediction, and we denote the confidence as $conf(\mathcal{G})$. Intuitively, the classifier has higher accuracy for those samples with high confidence. For this reason, we investigate the impact and effectiveness of confidence in filtering high-quality samples. We set the sample with a high confidence as a filtering criterion. Specifically, we set a threshold θ to filter the sample: We only keep the random sample with confidence $conf(\phi(\mathcal{G})) > \theta$. That is, we can set $h(\phi(\mathcal{G})) = 0$ if $conf(\phi(\mathcal{G})) > \theta$, otherwise 0.

5 Preliminary Results

Experimental Setup: Our evaluation involves experiments on the GCN model using the Citeseer and Cora-ML datasets, following [1]. In both datasets, we maintain thresholds at $\tau = 0.1$, $\xi = 0.2$, and $\theta = 0.5$. We consider three models: 1) GCN: the original SparseSmooth model [1], 2) GCN+JacAug: the GCN model augmented with our Jaccard augmenter (JacAug), and 3) GCN+JacAug+Conf: the complete **AuditVotes** framework integrating both the Jaccard augmenter (JacAug) and the confidence filter (Conf).

We begin by evaluating the effectiveness of the augmentation in improving certified accuracy by comparing models with and without JacAug (Figure 2(c,f)). We obverse that the GCN model with JacAug augmenter greatly improves the clean accuracy of the smoothed model (from 0.29 to 0.71 on the Citeseer and from 0.32 to 0.76 on Cora-ML).

Next, we investigate the impact of our conditional smoothing approach by applying the confidence filter to the GCN+JacAug model. Firstly, we visualize the distribution of confidence levels based on the correctness of node classification (Figure 2(a,d)), highlighting distinct confidence distributions for correctly and incorrectly predicted samples. Notably, the high-confidence group exhibits significantly higher node classification accuracy (Figure 2(b,e)). By leveraging our confidence filter, we achieve a notable enhancement in certified accuracy. For instance, with an attack budget involving adding 5 edges, the GCN+JacAug+Conf model improves certified accuracy by 16.7% on Citeseer and 10.0% on Cora-ML compared to GCN+JacAug.

In conclusion, the JacAug augmentation significantly boosts model utility, while the conditional smoothing model with the confidence filter further enhances certified accuracy. Our **AuditVotes** model, encompassing both JacAug and Conf, demonstrates the most substantial enhancement in certified accuracy. Specifically, when the attack budget involves adding 5 edges, the certified accuracy for GCN, GCN+JacAug, and GCN+JacAug+Conf are 0.29, 0.48, and 0.56 on the Citeseer dataset, and 0.32, 0.50, and 0.55 on the Cora-ML dataset (Figure 2(c,f)). Compared to SparseSmooth [1], our

AuditVotes model enhances certified accuracy by 93.1% on Citeseer and 71.9% on Cora-ML, highlighting its potential in optimizing the trade-off between utility and robustness.

References

[1] Aleksandar Bojchevski, Johannes Gasteiger, and Stephan Günnemann. 2020. Efficient robustness certificates for discrete data: Sparsity-aware randomized smoothing for graphs, images and more. In *International Conference on Machine Learning*. PMLR, 1003–1013.

[2] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. 2022. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter* 24, 2 (2022), 61–77.

[3] Yuni Lai, PAN Bailin, Kaihuang Chen, Yancheng Yuan, and Kai Zhou. 2024. Collective Certified Robustness against Graph Injection Attacks. In *Forty-first International Conference on Machine Learning (ICML)*.

[4] Yuni Lai, Yulin Zhu, Bailin Pan, and Kai Zhou. 2024. Node-aware Bi-smoothing: Certified Robustness against Graph Injection Attacks. In *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 215–215.

[5] Linyi Li, Tao Xie, and Bo Li. 2023. Sok: Certified robustness for deep neural networks. In *2023 IEEE symposium on security and privacy (SP)*. IEEE, 1289–1310.

[6] Lichao Sun, Yingtong Dou, Carl Yang, Kai Zhang, Ji Wang, S Yu Philip, Lifang He, and Bo Li. 2022. Adversarial attack and defense on graph data: A survey. *IEEE Transactions on Knowledge and Data Engineering* 35, 8 (2022), 7693–7711.

[7] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. 2019. Adversarial examples for graph data: deep insights into attack and defense. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 4816–4823.

[8] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

[9] Tong Zhao, Wei Jin, Yozen Liu, Yingheng Wang, Gang Liu, Stephan Günnemann, Neil Shah, and Meng Jiang. 2022. Graph data augmentation for graph machine learning: A survey. *arXiv preprint arXiv:2202.08871* (2022).

A Proof for Theorem 3.1

Theorem 3.1. *(Restate) Let the conditional smoothed classifier $g^c(\cdot)$ be as defined in Eq. (2). We divide $\mathbb{G}$ into disjoint regions $\mathbb{G} = \bigcup_i^I \mathcal{R}_i$, where $\mathcal{R}_i$ denote the consent likelihood region that $\frac{\mathbb{P}(\phi(\mathcal{G})=Z)}{\mathbb{P}(\phi(\mathcal{G}')=Z)} = c_i$ (a constant). Let $r_i = \mathbb{P}(\phi(\mathcal{G}) \in \mathcal{R}_i)$, $r_i' = \mathbb{P}(\phi(\mathcal{G}') \in \mathcal{R}_i)$ denote the probability that the random sample fall in the partitioned region $\mathcal{R}_i$. We define μ as follows:*

$$\begin{aligned} \mu := \min_{\mathbf{s},\mathbf{t}} \quad & \mathbf{s}^T\mathbf{r}' - \mathbf{t}^T\mathbf{r}', \\ s.t. \quad & \mathbf{s}^T\mathbf{r} = \underline{p_A},\ t^T r = \overline{p_B}, \quad 0 \le s \le 1,\ 0 \le t \le 1. \end{aligned} \tag{5}$$

Then for $\forall \mathcal{G}' \in \mathcal{B}_{r_a,r_b}(\mathcal{G})$, we have $g_v(\mathcal{G}') = g_v(\mathcal{G})$, if $\mu > 0$.

Proof. We employ a similar proof scheme as SparseSmooth [1]. Specifically, the problem of certified robustness can be formulated as a linear programming problem in (5). Intuitively, the linear programming problem is to find a worst-case classifier that tends to assign the highest probability of class y_A to $\phi(\mathcal{G})$ and assign the lowest probability of class y_A to $\phi(\mathcal{G}')$. Therefore, the worst-case classifier will assign class y_A in decreasing order of the constant likelihood regions until $\mathbb{P}(f(\phi(\mathcal{G})) = y_A | h(\phi(\mathcal{G})) = 0) = \underline{p_A}$, and assign class y_B in increasing order of the constant likelihood regions until $\mathbb{P}(f(\phi(\mathcal{G})) = y_B | h(\phi(\mathcal{G})) = 0) = \overline{p_B}$.

Let $\mathcal{G}'$ denote the perturbed sample from $\mathcal{G}$, and $z \in \mathbb{G}$ be any possible graph obtained by $\phi(\mathcal{G})$ or $\phi(\mathcal{G}')$. We now need to compute the likelihood ratio with condition $h(z) = 0$. Notably, the likelihood ratio only depends on the difference between $\mathcal{G}$ and $\mathcal{G}'$, and does not depend on the filter $h(z)$, so we have the likelihood ratio $\Lambda(z)$ as follows:

$$\Lambda(z) = \frac{\mathbb{P}(\phi(\mathcal{G}) = z | h(z) = 0)}{\mathbb{P}(\phi(\mathcal{G}') = z | h(z) = 0)} = \frac{\mathbb{P}(\phi(\mathcal{G}) = z)}{\mathbb{P}(\phi(\mathcal{G}') = z)}. \tag{6}$$

That is, our conditional smoothing model (with arbitrary $h(\cdot)$) does not affect the likelihood ratio, allowing us to utilize the same constant likelihood ratio region as in SparseSmooth [1]. This underscores the seamless adaptability of the certifying condition to our conditional smoothing model. The only difference lies in the definition of $\underline{p_A}$ and $\overline{p_B}$. In SparseSmooth [1], the $\underline{p_A}$ and $\overline{p_B}$ are estimated among all the votes, while in our model, the probabilities $\underline{p_A}$ and $\overline{p_B}$ are estimated among the valid votes ($h(z) = 0$).

Assuming that there are two constant likelihood regions $\mathcal{R}_1$ and $\mathcal{R}_2$, and $\Lambda(z \in \mathcal{R}_1) > \Lambda(z \in \mathcal{R}_2)$, then we can decompose the conditional probabilities $\underline{p_A}$ and $\overline{p_B}$ as follows:

$$\begin{aligned} \underline{p_A} &= \mathbb{P}(f(\phi(\mathcal{G})) = y_A | h(\phi(\mathcal{G})) = 0) \\ &= \mathbb{P}(f(z) = y_A, \phi(\mathcal{G}) = z \in \mathcal{R}_1 | h(z) = 0) \\ &\quad + \mathbb{P}(f(z) = y_A, \phi(\mathcal{G}) = z \in \mathcal{R}_2 | h(z) = 0) \\ &= \mathbb{P}(f(z) = y_A | \phi(\mathcal{G}) = z \in \mathcal{R}_1, h(z) = 0) \\ &\quad \times \mathbb{P}(\phi(\mathcal{G}) = z \in \mathcal{R}_1 | h(z) = 0) \\ &\quad + \mathbb{P}(f(z) = y_A | \phi(\mathcal{G}) = z \in \mathcal{R}_2, h(z) = 0) \\ &\quad \times \mathbb{P}(\phi(\mathcal{G}) = z \in \mathcal{R}_2 | h(z) = 0) \\ &= \mathbb{P}(f(z) = y_A | \phi(\mathcal{G}) = z \in \mathcal{R}_1, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}) = z \in \mathcal{R}_1) \\ &\quad + \mathbb{P}(f(z) = y_A | \phi(\mathcal{G}) = z \in \mathcal{R}_2, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}) = z \in \mathcal{R}_2). \end{aligned}$$

$$\begin{aligned} \overline{p_B} &= \mathbb{P}(f(\phi(\mathcal{G})) = y_B | h(\phi(\mathcal{G})) = 0) \\ &= \mathbb{P}(f(z) = y_B, \phi(\mathcal{G}) = z \in \mathcal{R}_1 | h(z) = 0) \\ &\quad + \mathbb{P}(f(z) = y_B, \phi(\mathcal{G}) = z \in \mathcal{R}_2 | h(z) = 0) \\ &= \mathbb{P}(f(z) = y_B | \phi(\mathcal{G}) = z \in \mathcal{R}_1, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}) = z \in \mathcal{R}_1) \\ &\quad + \mathbb{P}(f(z) = y_B | \phi(\mathcal{G}) = z \in \mathcal{R}_2, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}) = z \in \mathcal{R}_2). \end{aligned}$$

The worst-case classifier is:

$$\mathbb{P}(f(z) = y_A | h(z) = 0) = \begin{cases} \underline{p_A}, & z \in \mathcal{R}_1, h(z) = 0 \\ 0, & z \in \mathcal{R}_2, h(z) = 0 \end{cases}$$

$$\mathbb{P}(f(z) = y_B | h(z) = 0) = \begin{cases} \overline{p_B}, & z \in \mathcal{R}_1, h(z) = 0 \\ 1, & z \in \mathcal{R}_2, h(z) = 0. \end{cases}$$

Next, our goal is to estimate the prediction probabilities given arbitrary perturbed graph $\mathcal{G}' \in \mathcal{B}_{r_a,r_b}$.

$$\begin{aligned} p_A' &:= \mathbb{P}(f(\phi(\mathcal{G}')) = y_A | h(\phi(\mathcal{G}')) = 0) \\ &= \mathbb{P}(f(z) = y_A, \phi(\mathcal{G}') = z \in \mathcal{R}_1 | h(z) = 0) \\ &\quad + \mathbb{P}(f(z) = y_A, \phi(\mathcal{G}') = z \in \mathcal{R}_2 | h(z) = 0) \\ &= \mathbb{P}(f(z) = y_A | \phi(\mathcal{G}') = z \in \mathcal{R}_1, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_1) \\ &\quad + \mathbb{P}(f(z) = y_A | \phi(\mathcal{G}') = z \in \mathcal{R}_2, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_2) \\ &= \underline{p_A} \cdot \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_1). \end{aligned} \tag{7}$$

$$
\begin{aligned}
p'_B &:= \mathbb{P}(f(\phi(\mathcal{G}')) = y_B | h(\phi(\mathcal{G}')) = 0) \\
&= \mathbb{P}(f(z) = y_B, \phi(\mathcal{G}') = z \in \mathcal{R}_1 | h(z) = 0) \\
&\quad + \mathbb{P}(f(z) = y_B, \phi(\mathcal{G}') = z \in \mathcal{R}_2 | h(z) = 0) \\
&= \mathbb{P}(f(z) = y_B | \phi(\mathcal{G}') = z \in \mathcal{R}_1, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_1) \\
&\quad + \mathbb{P}(f(z) = y_B | \phi(\mathcal{G}') = z \in \mathcal{R}_2, h(z) = 0) \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_2) \\
&= \overline{p_B} \cdot \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_1) + \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_2). \qquad (8)
\end{aligned}
$$

Finally, the prediction of sample $\mathcal{G}$ by the smoothed classifier defined in (2) can be certified if:

$$
\begin{aligned}
\mu &= p'_A - p'_B \\
&= \mathbb{P}(f(\phi(\mathcal{G}')) = y_A | h(\phi(\mathcal{G}')) = 0) \\
&\quad - \mathbb{P}(f(\phi(\mathcal{G}')) = y_B | h(\phi(\mathcal{G}')) = 0) \\
&= \underline{p_A} \cdot \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_1) \\
&\quad - \overline{p_B} \cdot \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_1) - \mathbb{P}(\phi(\mathcal{G}') = z \in \mathcal{R}_2) \\
&> 0.
\end{aligned}
$$

□