Certified Robustness of Community Detection against Adversarial Structural Perturbation via Randomized Smoothing

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

ABSTRACT

Community detection plays a key role in understanding graph structure. However, several recent studies showed that community detection is vulnerable to adversarial structural perturbation. In particular, via adding or removing a small number of carefully selected edges in a graph, an attacker can manipulate the detected communities. However, to the best of our knowledge, there are no studies on certifying robustness of community detection against such adversarial structural perturbation. In this work, we aim to bridge this gap. Specifically, we develop the first certified robustness guarantee of community detection against adversarial structural perturbation. Given an arbitrary community detection method, we build a new *smoothed* community detection method via randomly perturbing the graph structure. We theoretically show that the smoothed community detection method provably groups a given arbitrary set of nodes into the same community (or different communities) when the number of edges added/removed by an attacker is bounded. Moreover, we show that our certified robustness is *tight*. We also empirically evaluate our method on multiple real-world graphs with ground truth communities.

KEYWORDS

Community Detection; Certified Robustness

ACM Reference Format:
Jinyuan Jia, Binghui Wang, Xiaoyu Cao, Neil Zhenqiang Gong. 2020. Certified Robustness of Community Detection against Adversarial Structural Perturbation via Randomized Smoothing. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3366423.3380029

1 INTRODUCTION

Graph is a powerful tool to represent many complex systems. For example, an online social network can be viewed as a graph, where nodes are users and edges represent friendships or interactions between users. Community detection is a basic tool to understand the structure of a graph and has many applications. For instance, communities in a social graph may represent users with common interests, locations, occupations, etc.. Therefore, many community detection methods (e.g., [2, 14, 16, 23–25, 30, 41]) have been proposed by various fields such as network science, applied physics, and bioinformatics. Roughly speaking, a community detection method divides the nodes in a graph into groups such that nodes in the same group are densely connected and nodes in different groups are sparsely connected.

However, multiple recent studies showed that community detection is vulnerable to adversarial structural perturbations [8, 9, 13, 29, 37]. Specifically, via adding or removing a small number of carefully selected edges in a graph, an attacker can manipulate the detected communities. For example, an attacker can spoof a community detection method to split a set of nodes, which are originally detected as in the same community, into different communities. An attacker can also spoof a community detection method to merge a set of nodes, which are originally detected as in different communities, into the same community. We call these two attacks *splitting attack* and *merging attack*, respectively. However, to the best of our knowledge, there are no studies to certify robustness of community detection against such adversarial structural perturbation. We note that several heuristic defenses [9, 29] were proposed to enhance the robustness of community detection against structural perturbation. However, these defenses lack formal guarantees and can often be defeated by strategic attacks that adapt to them.

In this work, we aim to bridge this gap. In particular, we aim to develop certified robustness of community detection against structural perturbation. Given an arbitrary community detection method, our techniques transform the method to a robust community detection method that provably groups a given arbitrary set of nodes into the same community (against splitting attacks) or into different communities (against merging attacks) when the number of edges added/removed by the attacker is no larger than a threshold. We call the threshold *certified perturbation size*.

Our robustness guarantees are based on a recently proposed technique called *randomized smoothing* [11, 21, 26], which is the state-of-the-art method to build provably robust machine learning methods. Specifically, given an arbitrary function f, which takes $\mathbf{x}$ as an input and outputs a categorial value. Randomized smoothing constructs a smoothed function g via adding random noise to the input $\mathbf{x}$. Moreover, the output of the smoothed function is the function f's output that has the largest probability when adding random noise to the input $\mathbf{x}$. Suppose an attacker can add a perturbation to the input $\mathbf{x}$. The smoothed function provably has the same output once the perturbation added to the input $\mathbf{x}$ is bounded.

We propose to certify robustness of community detection using randomized smoothing. Specifically, given a graph, an arbitrary community detection method, and an arbitrarily set of nodes in the graph, we construct a function f, which takes the graph as an input and outputs 1 if the community detection method groups the set of nodes into the same community, otherwise the function f outputs 0. Then, we build a smoothed function g via adding random noise

The first two authors made equal contributions.

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380029

to the graph structure, i.e., randomly adding or removing edges in the graph. Finally, we certify the robustness of the smoothed function g against adversarial structural perturbation.

However, existing randomized smoothing methods are insufficient to certify robustness of community detection. Specifically, they assume the input $\mathbf{x}$ is continuous and add Gaussian or Laplacian noise to it. However, graph structure is binary data, i.e., a pair of nodes can be connected or unconnected; and Gaussian or Laplacian noise is not semantically meaningful for such binary data. To address the challenge, we develop randomized smoothing for binary data. We theoretically derive a certified perturbation size via addressing several technical challenges. For instance, we prove a variant of the Neyman-Pearson Lemma [31] for binary data; and we divide the graph structure space into regions in a novel way such that we can apply the variant of the Neyman-Pearson Lemma to certify robustness of community detection. Moreover, we prove that our certified perturbation size is *tight* if no assumptions on the community detection method are made. Our certified perturbation size is the solution to an optimization problem. Therefore, we further design an algorithm to solve the optimization problem.

We empirically evaluate our method using multiple real-world graph datasets with ground-truth communities including Email, DBLP, and Amazon datasets. We choose the efficient community detection method called Louvain's method [2]. We study the impact of various parameters on the certified robustness.

In summary, our key contributions are as follows:

- We develop the first certified robustness of community detection against adversarial structural perturbation. Moreover, we show that our certified robustness is tight.
- Our certified perturbation size is the solution to an optimization problem and we develop an algorithm to solve the optimization problem.
- We evaluate our method on multiple real-world datasets.

2 BACKGROUND

2.1 Community Detection

Suppose we are given an undirected graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. A community detection method divides the nodes in the graph into groups, which are called communities. In non-overlapping community detection, a node only belongs to one community, while in overlapping community detection, a node may belong to multiple communities. Formally, a community detection algorithm $\mathcal{A}$ takes a graph as an input and produces a set of communities $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$, where $V = \cup_{i=1}^{k} C_i$ and C_i is the set of nodes that are in the ith community. For simplicity, we represent the graph structure as a binary vector $\mathbf{x}$, where an entry of the vector represents the connection status of the corresponding pair of nodes. Specifically, an entry $x_i = 1$ if the corresponding pair of nodes are connected, otherwise $x_i = 0$. Moreover, we denote by n the length of the binary vector $\mathbf{x}$. Therefore, we can represent community detection as $\mathcal{C} = \mathcal{A}(\mathbf{x})$.

2.2 Attacks to Community Detection

Adversarial structural perturbation: We consider an attacker can manipulate the graph structure, i.e., adding or removing some edges in the graph. In particular, an attacker may have control of some nodes in the graph and can add or remove edges among them. For instance, in a social graph, the attacker-controlled nodes may be fake users created by the attacker or normal users compromised by the attacker. We denote by a binary vector $\boldsymbol{\delta}$ the attacker's perturbation to the graph, where $\delta_i = 1$ if and only if the attacker changes the connection status of the corresponding pair of nodes. $\mathbf{x} \oplus \boldsymbol{\delta}$ is the perturbed graph structure, where the operator $\oplus$ is the XOR between two binary variables. Moreover, we use $||\boldsymbol{\delta}||_0$ to measure the magnitude of the perturbation because $||\boldsymbol{\delta}||_0$ has semantic interpretations. In particular, $||\boldsymbol{\delta}||_0$ is the number of edges added or removed by the attacker.

Two attacks: An attacker can manipulate the detected communities via adversarial structural perturbation [8, 9, 13, 29, 37]. Specifically, there are two types of attacks to community detection:

- **Splitting attack.** Given a set of nodes (called *victim nodes*) $\Gamma = \{u_1, u_2, \cdots, u_c\}$ that are in the same community. A splitting attack aims to perturb the graph structure such that a community detection method divides the nodes in Γ into different communities. Formally, we have communities $\mathcal{C}' = \{C_1, C_2, \cdots, C_{k'}\} = \mathcal{A}(\mathbf{x} \oplus \boldsymbol{\delta}')$ after the attacker adds perturbation $\boldsymbol{\delta}'$ to the graph structure, but there does not exist a community C_i such that $\Gamma \subset C_i$.
- **Merging attack.** Given a set of victim nodes Γ that are in different communities. A merging attack aims to perturb the graph structure such that a community detection method groups the nodes in Γ into the same community. Formally, we have communities $\mathcal{C}'' = \{C_1, C_2, \cdots, C_{k''}\} = \mathcal{A}(\mathbf{x} \oplus \boldsymbol{\delta}'')$ after the attacker adds perturbation $\boldsymbol{\delta}''$ to the graph structure, and there exists a community C_i such that $\Gamma \subset C_i$.

We aim to develop certified robustness of community detection against the splitting and merging attacks.

2.3 Randomized Smoothing

Randomized smoothing is state-of-the-art method to build provably secure machine learning methods [6, 11]. Suppose we are given a function f, which takes $\hat{\mathbf{x}}$ as an input and outputs a categorical value in a domain $\{1, 2, \cdots, d\}$. Randomized smoothing aims to construct a smoothed function g via adding random noise $\hat{\boldsymbol{\epsilon}}$ to the input $\hat{\mathbf{x}}$. Moreover, the output of the smoothed function is the output of the function f that has the largest probability when adding random noise to the input $\hat{\mathbf{x}}$. Formally, we have:

$$g(\hat{\mathbf{x}}) = \underset{\hat{y} \in \{1,2,\cdots,d\}}{\operatorname{argmax}} \Pr(f(\hat{\mathbf{x}} + \hat{\boldsymbol{\epsilon}}) = \hat{y}), \tag{1}$$

where $\hat{\boldsymbol{\epsilon}}$ is random noise drawn from a certain distribution. Suppose an attacker can add perturbation $\hat{\boldsymbol{\delta}}$ to the input $\hat{\mathbf{x}}$. Existing studies [11, 21, 26] assume $\hat{\mathbf{x}}$ is continuous data. Moreover, they showed that, when the random noise is drawn from a Gaussian distribution or Laplacian distribution, the smoothed function provably has the same output when the L_2-norm or L_1-norm of the perturbation $\hat{\boldsymbol{\delta}}$ is bounded. However, in our problem, the graph structure is binary data. Gaussian or Laplacian noise is not semantically meaningful for such binary data. To address the challenge, we will develop randomized smoothing for binary data and apply it to certify robustness against the splitting and merging attacks.

3 CERTIFIED ROBUSTNESS

3.1 Randomized Smoothing on Binary Data

We first construct a function f to model the splitting and merging attacks. Specifically, given a graph whose structure we represent as a binary vector $\mathbf{x}$, a community detection algorithm $\mathcal{A}$, and a set of victim nodes denoted as Γ, the function f outputs 1 if the nodes in Γ are grouped into the same community detected by $\mathcal{A}$ and outputs 0 otherwise. Formally, we define f as follows:

$$f(\mathbf{x}) = \begin{cases} 1, \text{ if } \exists i, \Gamma \subset C_i, \textit{where } C_i \in \mathcal{A}(\mathbf{x}) \\ 0, \text{ otherwise.} \end{cases} \tag{2}$$

We simply use $\mathbf{x}$ as an input for the function f because we study structural perturbation and other parameters–such as the community detection algorithm $\mathcal{A}$ and the set of victim nodes Γ–can be assumed to be constants. An attacker adds a perturbation vector $\boldsymbol{\delta}$ to the graph structure, i.e., $\mathbf{x} \oplus \boldsymbol{\delta}$ is the perturbed structure. When the nodes in Γ are in the same community before attack (i.e., $f(\mathbf{x}) = 1$) and $f(\mathbf{x} \oplus \boldsymbol{\delta})$ produces 0, a splitting attack succeeds. When the nodes in Γ are in different communities before attack (i.e., $f(\mathbf{x}) = 0$) and $f(\mathbf{x} \oplus \boldsymbol{\delta})$ produces 1, a merging attack succeeds.

We construct a smoothed function g via adding random noise to the graph structure $\mathbf{x}$. Specifically, we define a noise distribution in the discrete space $\{0, 1\}^n$ as follows:

$$\Pr(\epsilon_i = 0) = \beta, \Pr(\epsilon_i = 1) = 1 - \beta, \forall i \in \{1, 2, \cdots, n\}, \tag{3}$$

where n is the length of $\mathbf{x}$ and ϵ_i is the random binary noise added to the ith entry of $\mathbf{x}$. Formally, $\mathbf{x} \oplus \boldsymbol{\epsilon}$ is the noisy graph structure. Our random noise means that the connection status (connected or unconnected) of a pair of nodes is preserved with a probability β and changed with a probability $1 - \beta$.

We note that the detected communities $C = \mathcal{A}(\mathbf{x} \oplus \boldsymbol{\epsilon})$ are random since $\boldsymbol{\epsilon}$ is random. Therefore, the output $f(\mathbf{x} \oplus \boldsymbol{\epsilon})$ is also random. The smoothed function g outputs the value that has a larger probability. Formally, we have:

$$\begin{aligned} g(\mathbf{x}) &= \underset{y \in \{0,1\}}{\operatorname{argmax}} \Pr(f(\mathbf{x} \oplus \boldsymbol{\epsilon}) = y) \\ &= \begin{cases} 1, \text{ if } \Pr(f(\mathbf{x} \oplus \boldsymbol{\epsilon}) = 1) > 0.5 \\ 0, \text{ otherwise.} \end{cases} \end{aligned} \tag{4}$$

Certifying robustness against a splitting attack is to certify that $g(\mathbf{x} \oplus \boldsymbol{\delta}) = 1$ for all $||\boldsymbol{\delta}||_0 \leq L_1$, while certifying robustness against a merging attack is to certify that $g(\mathbf{x} \oplus \boldsymbol{\delta}) = 0$ for all $||\boldsymbol{\delta}||_0 \leq L_2$. In other words, we aim to certify that $g(\mathbf{x} \oplus \boldsymbol{\delta}) = y$ for all $||\boldsymbol{\delta}||_0 \leq L$, where $y \in \{0, 1\}$ and L is called *certified perturbation size*.

3.2 Deriving Certified Perturbation Size

In this section, we derive the certified perturbation size of the smoothed function g theoretically for a given graph, community detection algorithm, and a set of victim nodes. In the next section, we will design algorithms to compute the certified perturbation size in practice. Our results can be summarized in the following two theorems.

Theorem 1 (Certified Perturbation Size). *Given a graph-structure binary vector $\mathbf{x}$, a community detection algorithm $\mathcal{A}$, and a set of victim nodes Γ. The function f, random noise $\boldsymbol{\epsilon}$, and smoothed function g are defined in Equation 2, 3, and 4, respectively. Assume there exists $\underline{p} \in [0, 1]$ such that:*

$$Pr(f(\mathbf{x} \oplus \boldsymbol{\epsilon}) = y) \geq \underline{p} > 0.5, \tag{5}$$

where $\underline{p}$ is a lower bound of the probability $p = Pr(f(\mathbf{x} \oplus \boldsymbol{\epsilon}) = y)$ that f outputs y under the random noise $\boldsymbol{\epsilon}$. Then, we have:

$$g(\mathbf{x} \oplus \boldsymbol{\delta}) = y, \forall ||\boldsymbol{\delta}||_0 \leq L, \tag{6}$$

where L is called certified perturbation size and is the solution to the following optimization problem:

$$L = \operatorname{argmax} l, \tag{7}$$

$$s.t.\ ||\boldsymbol{\delta}||_0 = l, \tag{8}$$

$$\sum_{i=1}^{\mu-1} Pr(\mathbf{x} \oplus \boldsymbol{\delta} \oplus \boldsymbol{\epsilon} \in \mathcal{H}_i) + (\underline{p} - \sum_{i=1}^{\mu-1} Pr(\mathbf{x} \oplus \boldsymbol{\epsilon} \in \mathcal{H}_i)) \cdot \frac{Pr(\mathbf{x} \oplus \boldsymbol{\delta} \oplus \boldsymbol{\epsilon} \in \mathcal{H}_\mu)}{Pr(\mathbf{x} \oplus \boldsymbol{\epsilon} \in \mathcal{H}_\mu)} > 0.5, \tag{9}$$

where we define region $\mathcal{H}(e) = \{\mathbf{z} \in \{0, 1\}^n : \frac{Pr(\mathbf{x} \oplus \boldsymbol{\epsilon} = \mathbf{z})}{Pr(\mathbf{x} \oplus \boldsymbol{\delta} \oplus \boldsymbol{\epsilon} = \mathbf{z})} = \left(\frac{\beta}{1-\beta}\right)^e\}$ and density ratio $h(e) = \left(\frac{\beta}{1-\beta}\right)^e$, where $e = -n, -n+1, \cdots, n-1, n$. We rank the regions $\mathcal{H}(-n), \mathcal{H}(-n+1), \cdots, \mathcal{H}(n)$ in a descending order with respect to the density ratios $h(-n), h(-n+1), \cdots, h(n)$. Moreover, we denote the ranked regions as $\mathcal{H}_1, \mathcal{H}_2, \cdots, \mathcal{H}_{2n+1}$. Furthermore, μ is defined as follows:

$$\mu = \underset{\mu' \in \{1, 2, \cdots, 2n+1\}}{\operatorname{argmin}} \mu',\ s.t.\ \sum_{i=1}^{\mu'} Pr(\mathbf{x} \oplus \boldsymbol{\epsilon} \in \mathcal{H}_i) \geq \underline{p}$$

Proof. See our technical report [19]. □

Next, we show that our certified perturbation size is tight.

Theorem 2 (Tightness of the Certified Perturbation Size). *For any perturbation $\boldsymbol{\delta}$ with $||\boldsymbol{\delta}||_0 > L$, there exists a community detection algorithm $\mathcal{A}^*$ (and thus a function f^*) consistent with Equation 5 such that $g(\mathbf{x} \oplus \boldsymbol{\delta}) \neq y$ or there exists ties.*

Proof. See our technical report [19]. □

We have the following observations from our two theorems:

- Our certified perturbation size can be applied to any community detection method.
- Our certified perturbation size depends on $\underline{p}$ and β. When the probability lower bound $\underline{p}$ is tighter, our certified perturbation size is larger. We use the probability lower bound $\underline{p}$ instead of its exact value p because it is challenging to compute the exact value.
- When using the noise distribution defined in Equation 3 and no further assumptions are made on the community detection algorithm, it is impossible to certify a perturbation size that is larger than L.

3.3 Computing Certified Perturbation Size

Given a graph-structure binary vector $\mathbf{x}$, a community detection algorithm $\mathcal{A}$, and a set of victim nodes Γ, we aim to compute the certified perturbation size in practice. We face two challenges. The first challenge is to estimate y and obtain the probability lower bound $\underline{p}$. The second challenge is how to solve the optimization problem in Equation 7. To address the first challenge, we first estimate a value of y, and then use the one-sided Clopper-Pearson method [5] to estimate the probability bound with probabilistic guarantees. To address the second challenge, we develop an efficient algorithm to solve the optimization problem.

Estimating y and $\underline{p}$: We leverage a Monte-Carlo method to estimate y and $\underline{p}$ with probabilistic guarantees. Specifically, we first randomly sample N noise, and we use $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, \cdots, \boldsymbol{\epsilon}_N$ to denote them. Then, we compute the frequency of the output 0 and 1 for the function f, i.e., $m_0 = \sum_{i=1}^{N} \mathbb{I}(f(\mathbf{x} \oplus \boldsymbol{\epsilon}_i) = 0)$ and $m_1 = \sum_{i=1}^{N} \mathbb{I}(f(\mathbf{x} \oplus \boldsymbol{\epsilon}_i) = 1)$, where $\mathbb{I}$ is an indicator function. We estimate $\hat{y} = \text{argmax}_{i \in \{0,1\}} m_i$. Then, we estimate $\underline{p}$ by leveraging the one-sided Clopper-Pearson method. Estimating $\underline{p}$ can be viewed as estimating the parameter of a Binomial distribution. In particular, $m_{\hat{y}}$ can be viewed as a sample from a Binomial distribution $Bin(N, p)$, where $m_{\hat{y}}$ is the frequency of the value $\hat{y}$ and $Bin(N, p)$ denotes a Binomial distribution with parameters N and p. Therefore, we can estimate $\underline{p}$ by leveraging the one-sided Clopper-Pearson method. Specifically, we have:

$$\underline{p} = B(\alpha; m_{\hat{y}}, N - m_{\hat{y}} + 1), \tag{10}$$

where $1-\alpha$ represents the confidence level and $B(\alpha; m_{\hat{y}}, N-m_{\hat{y}}+1)$ denotes the αth quantile of the beta distribution with parameters $m_{\hat{y}}$ and $N - m_{\hat{y}} + 1$.

Solving the optimization problem: After obtaining the probability bound $\underline{p}$, we solve the optimization problem in Equation 7 to obtain L. The key to solve the optimization problem is to compute $\Pr(\mathbf{x} \oplus \boldsymbol{\epsilon} \in \mathcal{H}(e))$ and $\Pr(\mathbf{x} \oplus \boldsymbol{\delta} \oplus \boldsymbol{\epsilon} \in \mathcal{H}(e))$ for each $e \in \{-n, -n+1, \cdots, n\}$ when $||\boldsymbol{\delta}||_0 = l$. Specifically, we have:

$$\Pr(\mathbf{x} \oplus \boldsymbol{\epsilon} \in \mathcal{H}(e)) = \sum_{i=\max\{0,e\}}^{\min\{n, n+e\}} \beta^{n-(i-e)}(1-\beta)^{(i-e)} \cdot \theta(e,i) \tag{11}$$

$$\Pr(\mathbf{x} \oplus \boldsymbol{\delta} \oplus \boldsymbol{\epsilon} \in \mathcal{H}(e)) = \sum_{i=\max\{0,e\}}^{\min\{n, n+e\}} \beta^{n-i}(1-\beta)^{i} \cdot \theta(e,i), \tag{12}$$

where $\theta(e, i)$ is defined as follows:

$$\theta(e,\, i) = \begin{cases} 0, & \text{if } (e+l) \bmod 2 \neq 0, \\ 0, & \text{if } 2i - e < l, \\ \binom{n-l}{\frac{2i-e-l}{2}}\binom{l}{\frac{l-e}{2}}, & \text{otherwise} \end{cases} \tag{13}$$

The calculation details can be found in our technical report [19]. Once we can compute the probabilities $\Pr(\mathbf{x} \oplus \boldsymbol{\epsilon} \in \mathcal{H}(e))$ and $\Pr(\mathbf{x} \oplus \boldsymbol{\delta} \oplus \boldsymbol{\epsilon} \in \mathcal{H}(e))$, we can iteratively find the largest l such that the constraint in Equation 9 is satisfied. Such largest l is our certified perturbation size L.

Complete certification algorithm: Algorithm 1 shows our complete certification algorithm. The function SampleUnderNoise randomly samples N noise from the noise distribution defined in Equation 3, adds each noise to the graph structure, and computes the frequency of the function f's output 0 and 1. Then, our algorithm estimates $\hat{y}$ and $\underline{p}$. Based on $\underline{p}$, the function CertifiedPerturbationSize computes the certified perturbation size by solving the optimization problem in Equation 7. Our algorithm returns $(\hat{y}, L)$ if $\underline{p} > 0.5$ and ABSTAIN otherwise. The following proposition shows the probabilistic guarantee of our certification algorithm.

Algorithm 1: Certify

Input: f, β, $\mathbf{x}$, N, α.
Output: ABSTAIN or $(\hat{y}, L)$.

```
m_0, m_1 = SampleUnderNoise(f, β, x, N)
ŷ = argmax_{i∈{0,1}} m_i
p = B(α; m_ŷ, N − m_ŷ + 1)
if p > 0.5 then
    L = CertifiedPerturbationSize(p)
    return (ŷ, L)
else
    return ABSTAIN
```

Proposition 1. *With probability at least $1-\alpha$ over the randomness in Algorithm 1, if the algorithm returns an output value $\hat{y}$ and a certified perturbation size L (i.e., does not ABSTAIN), then we have $g(\mathbf{x} \oplus \boldsymbol{\delta}) = \hat{y}, \forall ||\boldsymbol{\delta}||_0 \leq L$.*

Proof. See our technical report [19]. □

4 EVALUATION

4.1 Experimental Setup

Datasets: We consider three undirected graph datasets with "ground-truth" communities, i.e., Email, DBLP, and Amazon. We obtained the datasets from SNAP (http://snap.stanford.edu/). Due to limited space, we only show the results on the Email dataset. Note that we have similar observations on the other two datasets, and the experimental results can be found in our technical report [19]. The Email dataset describes the communications between members in a research institution. The graph consists of 1,005 nodes, each of which represents a member; and 25,571 edges, indicating the email communications between members. The 42 departments in the institution are considered as the ground-truth communities and each node belongs to exactly one of them.

Community detection algorithm: We use the popular Louvain's method [2] to detect communities. The method optimizes modularity in a heuristic and iterative way. We note that the method produces communities in multiple hierarchical levels, and we take the last level since in which the maximum of the modularity is attained. We use a publicly available implementation.[1]

Evaluation metric: We use *certified accuracy* as the metric to evaluate our certification method. We take defending against the splitting attack as an example to illustrate certified accuracy. Suppose we are given M sets of victim nodes $\Gamma_1, \Gamma_2, \cdots, \Gamma_M$. The nodes in each victim set Γ_i are in the same ground-truth community. The goal of a splitting attack is to perturb the graph structure such that the Louvain's method groups the victim nodes in a set Γ_i into at

[1] https://sites.google.com/site/findcommunities/

least two communities. Our certification algorithm in Algorithm 1 produces an output y_i and a certified perturbation size L_i for each victim set Γ_i. $y_i = 1$ means that we can provably guarantee that the nodes in Γ_i are grouped into the same community. Given a perturbation size l, we define the certified accuracy $CK(l)$ at the perturbation size l as the fraction of sets of victim nodes whose output $y_i = 1$ and certified perturbation size is at least l. Our certified accuracy $CK(l)$ is the fraction of sets of victim nodes that our method can provably detect as in the same community when an attacker adds or removes at most l edges in the graph. Formally, we have:

Certified Accuracy for Defending against Splitting Attacks:

$$CK(l) = \frac{\sum_{i=1}^{M} \mathbb{I}(y_i = 1)\mathbb{I}(L_i \geq l)}{M}, \tag{14}$$

where $\mathbb{I}$ is an indicator function. For merging attacks, the nodes in a victim set Γ_i are in different ground-truth communities. The goal of a merging attack is to perturb the graph structure such that the Louvain's method groups the victim nodes in a set Γ_i into the same community. Given a perturbation size l, we define the certified accuracy $CK(l)$ at the perturbation size l as the fraction of sets of victim nodes whose output $y_i = 0$ and certified perturbation size is at least l. Our certified accuracy $CK(l)$ is the fraction of sets of victim nodes that our method can provably detect as in more than one communities when an attacker adds or removes at most l edges in the graph. Formally, we have:

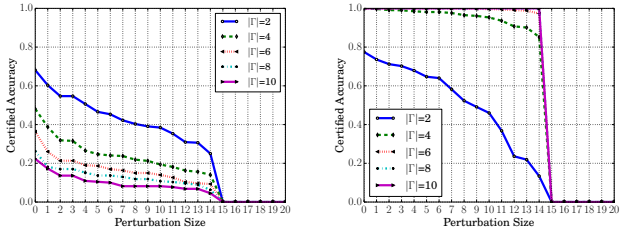
Certified Accuracy for Defending against Merging Attacks:

$$CK(l) = \frac{\sum_{i=1}^{M} \mathbb{I}(y_i = 0)\mathbb{I}(L_i \geq l)}{M}. \tag{15}$$

Parameter setting: Our method has the following parameters: the noise parameter β, the confidence level $1 - \alpha$, and the number of samples N. Unless otherwise mentioned, we use the following default parameters: $\beta = 0.7$, $1 - \alpha = 0.999$, and $N = 10,000$. To estimate certified accuracy for defending against splitting attacks, we randomly sample two sets of $|\Gamma|$ nodes from each ground-truth community whose size is larger than $|\Gamma|$, and we treat them as victim sets. To estimate certified accuracy for defending against merging attacks, we randomly sample 1,000 victim sets, each of which includes nodes randomly sampled from $|\Gamma|$ different communities. By default, we assume each set of victim nodes includes 2 nodes, i.e., $|\Gamma| = 2$. We also study the impact of each parameter, including β, $1 - \alpha$, N, and $|\Gamma|$. When studying the impact of one parameter, we fix the remaining parameters to be their default values. We randomly pick 100 nodes as attacker-controlled nodes for each dataset, and the attacker perturbs the edges between them.

4.2 Experimental Results

Impact of the number of victim nodes $|\Gamma|$: Figure 1a shows the certified accuracy vs. perturbation size for defending against splitting attacks with different number of victim nodes on the Email dataset, while Figure 1b shows the results for defending against merging attacks. We observe that as the number of victim nodes increases, the curve of the certified accuracy becomes lower for splitting attacks and higher for merging attacks. This is because it is harder to provably guarantee that a larger set of nodes are detected as in the same community (defending against splitting attacks); and it is easier to provably guarantee that a larger set of nodes are detected as in more than one communities (defending against merging attacks).



(a) Splitting attacks (b) Merging attacks

Figure 1: Impact of the number of victim nodes $|\Gamma|$ on defending against splitting attacks and merging attacks on Email.

Impact of the noise parameter β: Figure 2a shows the certified accuracy vs. perturbation size for defending against splitting attacks with different noise parameter β on Email. We observe that β provides a tradeoff between normal accuracy without attacks and robustness. Specifically, when β is larger, the normal accuracy, i.e., certified accuracy at perturbation size 0, is larger, while the certified accuracy decreases more quickly as the perturbation size increases. We also have similar observations for defending against merging attacks, and thus we omit the results for simplicity.

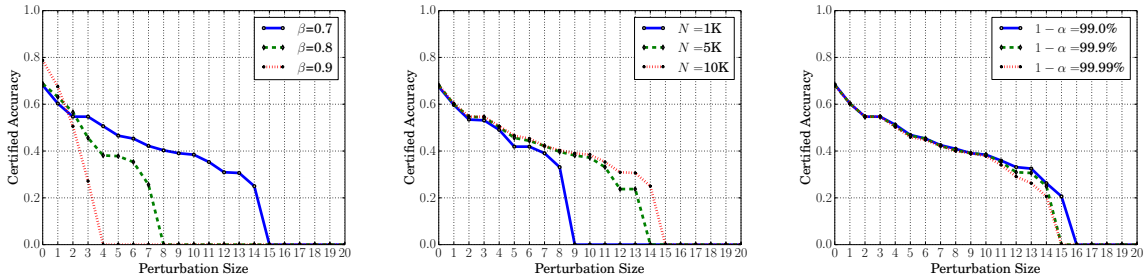
Impact of the number of sampled noise N: Figure 2b shows the certified accuracy vs. perturbation size for defending against splitting attacks with different numbers of sampled noise N on the Email dataset. We observe that the curve is higher as N increases. This is because a larger N makes the estimated probability bound $\underline{p}$ tighter and thus the certified perturbation size is also larger.

Impact of the confidence level $1-\alpha$: Figure 2c shows the certified accuracy vs. perturbation size for defending against splitting attacks with different confidence levels $1 - \alpha$ on the Email dataset. We observe that as the confidence level $1 - \alpha$ increases, the curve of the certified accuracy becomes lower. The reason is that a higher confidence level causes a looser estimated probability bound $\underline{p}$ and thus the certified perturbation size is smaller. However, we note that the differences between different confidence levels are negligible when the confidence levels are large enough.

5 RELATED WORK

Adversarial attacks to non-graph data and their defenses: For non-graph data, *adversarial example* is a well-known adversarial attack. Specifically, an attacker adds a carefully crafted perturbation to an input example such that a machine learning classifier makes predictions for the perturbed example as the attacker desires. The input example with carefully crafted perturbation is called adversarial example [17, 35]. Various empirical defenses (e.g., [17, 28, 32]) have been proposed to defend against adversarial examples. However, these defenses were often soon broken by adaptive attacks [1, 7].

In response, various certified defenses (e.g., [10, 15, 33, 34, 38]) against adversarial examples have been developed. Among these methods, randomized smoothing [6, 11, 18, 21, 26, 27] is state-of-the-art. Randomized smoothing turns an arbitrary classifier/function into a robust one via adding random noise to the input. Our work



(a) Impact of β (b) Impact of N (c) Impact of $1-\alpha$

Figure 2: Impact of the parameters β, N, and $1-\alpha$ on defending against splitting attacks on Email.

uses randomized smoothing. However, different from the existing randomized smoothing methods, which assume continuous input and add continuous noise, we propose randomized smoothing on binary data and leverage it to certify robustness of community detection against splitting and merging attacks. We note that a concurrent work [22] generalized randomized smoothing to discrete data. The major difference between our approach and [22] is that we leverage a variant of the Neyman-Pearson Lemma to derive the certified perturbation size.

Adversarial attacks to graph data and their defenses: Compared to non-graph data, adversarial attacks to graph data and their defenses are much less studied. *Adversarial structural perturbation* is a recently proposed attack to graph data. For instance, several recent studies [3, 12, 36, 43, 44] showed that Graph Neural Networks (GNNs) are vulnerable to adversarial structural perturbations. Specifically, an attacker can slightly perturb the graph structure and/or node features to mislead the predictions made by GNNs. Some empirical defenses [39, 40, 42] were proposed to defend against such attacks. However, these methods do not have certified robustness guarantees. Zügner & Günnemann [45] developed the first certified robustness guarantee against node-feature perturbations for graph convolutional network [20]. Bojchevski & Günnemann [4] proposed the first method for verifying certifiable (non-)robustness of graph convolutional network against structural perturbations. These work is different from ours as we focus on certifying robustness of community detection.

Multiple studies [8, 9, 13, 29, 37] have shown that community detection is vulnerable to adversarial structural perturbation. Several heuristic defenses [9, 29] were proposed to enhance the robustness of community detection against adversarial structural perturbations. However, these defenses lack formal guarantees. Our work is the first certified robustness guarantee of community detection against adversarial structural perturbations.

6 DISCUSSION AND LIMITATIONS

Given a set of nodes that are in the same ground-truth community (or in different ground-truth communities), our certified robustness guarantees that the nodes are provably detected as in the same community (or in different communities) when the number of added or removed edges in the graph is at most a certain threshold (called certified perturbation size). We note that when we add or remove enough edges in a graph, the "ground-truth" communities may change, and thus we may expect the set of nodes to be detected as in different communities (or in the same community). Therefore, the certified perturbation size should not be too large. We believe it is an interesting future work to explore what certified perturbation size should be expected for a particular application scenario.

Our work shows that we can provably guarantee that a set of nodes are or are not in the same community when an attacker adds or deletes a bounded number of edges in the graph. However, it is still an open question on how to obtain communities from our smoothed community detection method. One possible way to obtain communities is as follows: we first randomly pick a node as the initial community C. For each remaining node, we compute the probability of the node being clustered into the same community with each node in C under randomized smoothing. Then, we compute the average probability and if it is larger than a threshold, we add the node to C. When no more nodes can be added to C, we randomly pick another node from the remaining nodes and repeat the above process until all nodes are clustered into certain communities. We believe it is an interesting future work to explore how to derive communities from the smoothed method. We note that the communities derived from the smoothed community detection method may be less accurate than those derived from the base community detection method. In other words, there may be a tradeoff between accuracy and robustness.

7 CONCLUSION

In this work, we develop the first certified robustness guarantee of community detection against adversarial structural perturbations. Specifically, our results show that a set of nodes can be provably detected as in the same community (against splitting attacks) or in different communities (against merging attacks) when the number of edges added or removed by an attacker is no larger than a threshold. Moreover, we show that our derived threshold is tight when randomized smoothing with our discrete noise is used. Our method can turn any community detection method to be provably robust against adversarial structural perturbation to defend against splitting and merging attacks. We also empirically demonstrate the effectiveness of our method using three real-world graph datasets with ground-truth communities. Interesting future work includes leveraging the information of the community detection algorithm to further improve the certified robustness guarantees and exploring what certified perturbation size should be expected for a particular application scenario.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for insightful reviews. This work was supported by NSF grant No. 1937787 and No. 1937786.

REFERENCES

[1] Anish Athalye, Nicholas Carlini, and David Wagner. 2018. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *International Conference on Machine Learning*. 274–283.

[2] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[3] Aleksandar Bojchevski and Stephan Günnemann. 2019. Adversarial Attacks on Node Embeddings via Graph Poisoning. In *ICML*.

[4] Aleksandar Bojchevski and Stephan Günnemann. 2019. Certifiable Robustness to Graph Perturbations. In *Advances in Neural Information Processing Systems*. 8317–8328.

[5] Lawrence D Brown, T Tony Cai, and Anirban DasGupta. 2001. Interval estimation for a binomial proportion. *Statistical science* (2001), 101–117.

[6] Xiaoyu Cao and Neil Zhenqiang Gong. 2017. Mitigating evasion attacks to deep neural networks via region-based classification. In *Proceedings of the 33rd Annual Computer Security Applications Conference*. ACM, 278–287.

[7] Nicholas Carlini and David Wagner. 2017. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 3–14.

[8] Jinyin Chen, Lihong Chen, Yixian Chen, Minghao Zhao, Shanqing Yu, Qi Xuan, and Xiaoniu Yang. 2019. GA-Based Q-Attack on Community Detection. *IEEE Transactions on Computational Social Systems* 6, 3 (2019), 491–503.

[9] Yizheng Chen, Yacin Nadji, Athanasios Kountouras, Fabian Monrose, Roberto Perdisci, Manos Antonakakis, and Nikolaos Vasiloglou. 2017. Practical attacks against graph-based clustering. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1125–1142.

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

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

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

[13] Valeria Fionda and Giuseppe Pirro. 2017. Community deception or: How to stop fearing community detection algorithms. *IEEE Transactions on Knowledge and Data Engineering* 30, 4 (2017), 660–673.

[14] Santo Fortunato and Marc Barthelemy. 2007. Resolution limit in community detection. *Proceedings of the national academy of sciences* 104, 1 (2007), 36–41.

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

[16] Michelle Girvan and Mark EJ Newman. 2002. Community structure in social and biological networks. *Proceedings of the national academy of sciences* 99, 12 (2002), 7821–7826.

[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

[18] Jinyuan Jia, Xiaoyu Cao, Binghui Wang, and Neil Zhenqiang Gong. 2020. Certified Robustness for Top-k Predictions against Adversarial Perturbations via Randomized Smoothing. In *International Conference on Learning Representations*.

[19] Jinyuan Jia, Binghui Wang, Xiaoyu Cao, and Neil Zhenqiang Gong. 2020. Certified Robustness of Community Detection against Adversarial Structural Perturbation via Randomized Smoothing. *arXiv* (2020).

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

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

[22] Guang-He Lee, Yang Yuan, Shiyu Chang, and Tommi Jaakkola. 2019. Tight certificates of adversarial robustness for randomly smoothed classifiers. In *Advances in Neural Information Processing Systems*. 4911–4922.

[23] Jure Leskovec, Kevin J Lang, Anirban Dasgupta, and Michael W Mahoney. 2008. Statistical properties of community structure in large social and information networks. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 695–704.

[24] Jure Leskovec, Kevin J Lang, Anirban Dasgupta, and Michael W Mahoney. 2009. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics* 6, 1 (2009), 29–123.

[25] Jure Leskovec, Kevin J Lang, and Michael Mahoney. 2010. Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th international conference on World wide web*. ACM, 631–640.

[26] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. 2019. Second-order adversarial attack and certifiable robustness. *NeurIPS* (2019).

[27] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. 2018. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 369–385.

[28] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

[29] Shishir Nagaraja. 2010. The impact of unlinkability on adversarial community detection: effects and countermeasures. In *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 253–272.

[30] Mark EJ Newman. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103, 23 (2006), 8577–8582.

[31] Jerzy Neyman and Egon Sharpe Pearson. 1933. 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 (1933), 289–337.

[32] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 582–597.

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

[34] Karsten Scheibler, Leonore Winterer, Ralf Wimmer, and Bernd Becker. 2015. Towards Verification of Artificial Neural Networks.. In *MBMV*.

[35] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *ICLR*.

[36] Binghui Wang and Neil Zhenqiang Gong. 2019. Attacking Graph-based Classification via Manipulating the Graph Structure. In *CCS*.

[37] Marcin Waniek, Tomasz P Michalak, Michael J Wooldridge, and Talal Rahwan. 2018. Hiding individuals and communities in a social network. *Nature Human Behaviour* 2, 2 (2018), 139.

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

[39] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. 2019. Adversarial Examples on Graph Data: Deep Insights into Attack and Defense. In *IJCAI*.

[40] Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. 2019. Topology Attack and Defense for Graph Neural Networks: An Optimization Perspective. In *IJCAI*.

[41] Jaewon Yang and Jure Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42, 1 (2015), 181–213.

[42] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2019. Robust Graph Convolutional Networks Against Adversarial Attacks. In *KDD*.

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

[44] Daniel Zügner and Stephan Günnemann. 2019. Adversarial attacks on graph neural networks via meta learning. *ICLR* (2019).

[45] Daniel Zügner and Stephan Günnemann. 2019. Certifiable Robustness and Robust Training for Graph Convolutional Networks. In *KDD*.