Unsupervised Adversarially Robust Representation Learning on Graphs

Jiarong Xu[1], **Yang Yang**[1*], **Junru Chen**[1], **Xin Jiang**[3], **Chunping Wang**[2], **Jiangang Lu**[1], **Yizhou Sun**[3]

[1]Zhejiang University
[2]FinVolution Group
[3]University of California, Los Angeles
{xujr, yangya, jrchen_cali, lujg}@zju.edu.cn, jiangxjames@ucla.edu, wangchunping02@xinye.com, yzsun@cs.ucla.edu

Abstract

Unsupervised/self-supervised pre-training methods for graph representation learning have recently attracted increasing research interests, and they can be generalized to various downstream applications. Yet, the adversarial robustness of such pre-trained graph learning models remains largely unexplored. More importantly, most existing defense techniques for end-to-end graph representation learning methods require pre-specified label definitions, and thus cannot be directly applied to the pre-training methods. In this paper, we propose an unsupervised defense technique to robustify pre-trained deep graph models, so that the perturbations on the input graph can be successfully identified and blocked before the model is applied to different downstream tasks. Specifically, we introduce a mutual information-based measure, *graph representation vulnerability (GRV)*, to quantify the robustness of graph encoders on the representation space. We then formulate an optimization problem to learn the graph representation by carefully balancing the trade-off between the expressive power and the robustness (*i.e.*, GRV) of the graph encoder. The discrete nature of graph topology and the joint space of graph data make the optimization problem intractable to solve. To handle the above difficulty and to reduce computational expense, we further relax the problem and thus provide an approximate solution. Additionally, we explore a provable connection between the robustness of the unsupervised graph encoder and that of models on downstream tasks. Extensive experiments demonstrate that even without access to labels and tasks, our model is still able to enhance robustness against adversarial attacks on three downstream tasks (*i.e.*, node classification, link prediction, and community detection) by an average of +16.5% compared with existing methods.

1 Introduction

Graphs, a common mathematical abstraction for modeling pairwise interactions between objects, are widely applied in numerous domains, including bioinformatics, social networks, and finance. Owing to their prevalence, deep learning on graphs, such as graph neural networks (GNNs) (Kipf et al. 2017; Hamilton et al. 2017), have recently undergone rapid development, and made major progress in various analytical tasks, including node classification (Kipf et al. 2017; Hamilton et al. 2017), link prediction (Kipf et al. 2016), and graph classification (Xu et al. 2019b). However, most deep learning models on graphs are trained with task-specific labels in an end-to-end manner for a particular task. This motivates some recent efforts to pre-train an expressive graph encoder on unlabeled data and further feed the learned representations to (supervised/unsupervised) off-the-shelf machine learning models for relevant downstream tasks (Hu et al. 2019, 2020; Qiu et al. 2020). The pre-training models on graphs enable the learned representations to be directly applicable to different applications with a simple and inexpensive machine learning model attached after the encoded representations.

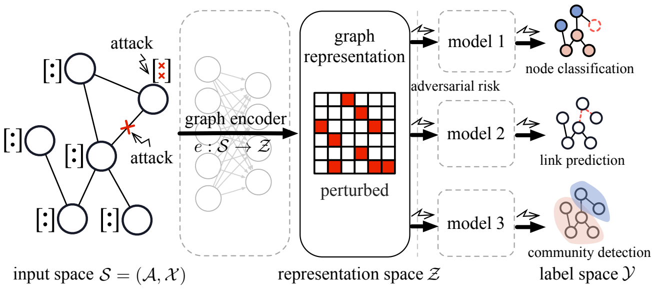

attack
attack
graph encoder
$e : \mathcal{S} \rightarrow \mathcal{Z}$
graph representation
perturbed
adversarial risk
model 1
model 2
model 3
node classification
link prediction
community detection
input space $\mathcal{S} = (\mathcal{A}, \mathcal{X})$
representation space $\mathcal{Z}$
label space $\mathcal{Y}$

Figure 1: Overview of a graph pre-training pipeline under adversarial attacks. If the graph encoder is vulnerable to the attacks, the adversarial risk would propagate to every downstream task via the perturbed graph representation.

Despite the promising results achieved by deep learning models on graphs, recent studies have shown that these models are vulnerable to adversarial attacks (Dai et al. 2018; Zügner et al. 2019a; Bojchevski et al. 2019a; Xu et al. 2020b; Zheng et al. 2021). In other words, even imperceptible perturbations on graph topology and node attributes can significantly affect the learned graph representation, thereby degrading the performance of downstream tasks (Chen et al. 2020; Xu et al. 2020c). This so-called adversarial vulnerability has given rise to tremendous concerns regarding the utilization of deep learning models on graphs, especially in security-critical applications such as drug discovery (Gilmer et al. 2017) and financial surveillance (Yang et al. 2019, 2021). However, the adversarial vulnerability of pre-training models on graphs is far overlooked. In this work, we show that graph pre-training models also suffer from the adversarial vulnerability problem. Actually, owing to the complicated and deep structure, the graph encoder is more vulner-

*Corresponding author.

able to adversarial attacks than the simple machine learning models used for downstream tasks in a graph pre-training pipeline (Tanay et al. 2016). As Figure 1 shows, once the graph encoder is vulnerable to adversarial attacks, the adversarial risk would propagate to every task via the perturbed representations.

Most efforts targeted at this adversarial vulnerability problem focus on supervised, end-to-end models designed for a particular application scenario (Zügner et al. 2019b; Bojchevski et al. 2019b; Wang et al. 2019; Jin et al. 2020; Wang et al. 2021; Xu et al. 2021). However, the dependency on the supervised information largely limits the scope of their application and usefulness. For example, these models do not perform well on downstream tasks in which training labels are missing, *e.g.*, community detection in social networks. In addition, training multiple models for different downstream tasks is both costly and insecure (Feurer et al. 2015). In contrast, robust unsupervised pre-training models can easily handle the above issues because adversarial attacks are identified and blocked before propagating to downstream tasks. Moreover, these models are applicable to a more diverse group of applications, including node classification, link prediction, and community detection. Unfortunately, however, these robust pre-training models under the unsupervised setting still remain largely unexplored.

There are many interesting yet challenging questions in this new field of research. Conventionally, the robustness of a model is defined based on the label space (Xu et al. 2020a; Zügner et al. 2019b; Bojchevski et al. 2019b), which is not the case in our setting. Thus the first difficulty we meet is to quantify the robustness of an unsupervised model (without the knowledge of the true or predicted labels).

To overcome the above challenge, in this paper, we first introduce the *graph representation vulnerability* (GRV), an information theoretic-based measure used to quantify the robustness of a graph encoder. We then formulate an optimization problem to study the trade-off between the expressive power of a graph encoder and its robustness to adversarial attacks, measured in GRV. However, how to efficiently compute or approximate the objective of the optimization problem becomes the next issue. First, it remains a big problem on how to describe the ability of the attack strategies or the boundary of perturbations, because adversarial attacks on graphs perturb both the discrete graph topology and the continuous node attributes. Second, the rigorous definition of the objective is intractable.

To handle the above issues, we first quantify the ability of adversarial attacks using Wasserstein distance between probability distributions, and provide a computationally efficient approximation for it. We then adopt a variant of projected gradient descent method to solve the proposed optimization problem efficiently. A sub-optimal solution for the problem gives us a well-qualified and robust graph encoder.

Last but not least, we explore several interesting theoretical connections between the proposed measure of robustness (GRV) and the classifier robustness based on the label space. To show the practical usefulness of our model, we apply the learned representations to three different downstream tasks. Experimental results reveal that under adversarial attacks, our model beats the best baseline by an average of +1.8%, +1.8%, and +45.8% on node classification, link prediction, and community detection task, respectively.

2 Preliminaries and Notations

In most cases, we use upper-case letters (*e.g.*, X and Y) to denote random variables and calligraphic letters (*e.g.*, $\mathcal{X}$ and $\mathcal{Y}$) to denote their support, while the corresponding lower-case letters (*e.g.*, $\boldsymbol{x}$ and $\boldsymbol{y}$) indicate the realizations of these variables. We denote the probability distributions of the random variables using subscripts (*e.g.*, μ_X and μ_Y) and the corresponding empirical distributions with hat accents (*e.g.*, $\hat{\mu}_X$ and $\hat{\mu}_Y$). We use bold upper-case letters to represent matrices (*e.g.*, $\mathbf{A}$). When indexing the matrices, $\mathbf{A}_{ij}$ denotes the element at the i-th row and the j-th column, while $\mathbf{A}_i$ represents the vector at the i-th row. Let $(\mathcal{X}, d)$ denote the metric space, where $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a distance function on $\mathcal{X}$. The set of all probability measures on $\mathcal{X}$ is $\mathcal{M}(\mathcal{X})$.

We assume a generic unsupervised graph representation learning setup. In brief, we are provided with an undirected and unweighted graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ with the node set $\mathbf{V} = \{v_1, v_2, ..., v_{|\mathbf{V}|}\}$ and edge set $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V} = \{e_1, e_2, ..., e_{|\mathbf{E}|}\}$. We are also provided with the adjacency matrix $\mathbf{A} \in \{0,1\}^{|\mathbf{V}|\times|\mathbf{V}|}$ of the graph $\mathbf{G}$, a symmetric matrix with elements $\mathbf{A}_{ij} = 1$ if $(v_i, v_j) \in \mathbf{E}$ or $i = j$, and $\mathbf{A}_{ij} = 0$ otherwise. We augment $\mathbf{G}$ with the node attribute matrix $\mathbf{X} \in \mathbb{R}^{|\mathbf{V}|\times c}$ if nodes have attributes. Accordingly, we define our input as $\boldsymbol{s} = (\boldsymbol{a}, \boldsymbol{x}) \in \mathcal{S}$; thus, we can conceive of $\boldsymbol{x}$ as the attribute matrix and $\boldsymbol{a}$ as the adjacency matrix of $\mathbf{G}$ under a transductive learning setting, while $\boldsymbol{a}$ and $\boldsymbol{x}$ are the adjacency matrix and attribute matrix respectively of a node's subgraph under an inductive learning setting. We define an encoder $e : \mathcal{S} \to \mathcal{Z}$, which maps an input $\boldsymbol{s} = (\boldsymbol{a}, \boldsymbol{x}) \in \mathcal{S}$ to a representation $e(\boldsymbol{a}, \boldsymbol{x}) \in \mathcal{Z}$, and a simple machine learning model $f : \mathcal{Z} \to \mathcal{Y}$ that maps a representation $z \in \mathcal{Z}$ to a label $f(z) \in \mathcal{Y}$. We go on to define $g = f \circ e$ as their composition, such that $(f \circ e)(\boldsymbol{a}, \boldsymbol{x}) = f(e(\boldsymbol{a}, \boldsymbol{x}))$. The mutual information between two random variables X and Y is denoted by $\mathrm{I}(X; Y)$. Specifically, it is defined as the Kullback–Leibler divergence between the joint distribution $p(\boldsymbol{x}, \boldsymbol{y})$ and the product of the marginal distributions $p(\boldsymbol{x})p(\boldsymbol{y})$.

Admissible perturbations on graphs. The Wasserstein distance can be conceptualized as an optimal transport problem: we wish to transport the mass with probability distribution μ_S into another distribution $\mu_{S'}$ at the minimum cost. Formally, the p-th Wasserstein distance between μ_S and $\mu_{S'}$ is

$$W_p = (\mu_S, \mu_{S'}) = \Big(\inf_{\pi \in \Pi(\mu_S, \mu_{S'})} \int_{\mathcal{S}^2} d(\boldsymbol{s}, \boldsymbol{s}')\, d\pi(\boldsymbol{s}, \boldsymbol{s}') \Big)^{1/p},$$

where $\Pi(\mu_S, \mu_{S'})$ denotes the collection of all measures on $\mathcal{S} \times \mathcal{S}$ with marginal μ_S and $\mu_{S'}$, respectively. The choice of ∞-Wasserstein distance (*i.e.*, $p = \infty$) is conventional in learning graph representations (Champion et al. 2008).

Based on ∞-Wasserstein distance, we can quantify the ability of the adversarial attacks. An attack strategy is modeled as a probability distribution close to that of $S = (A, X)$, and all possible attack strategies stay in a ball around the genuine distribution μ_S, with a pre-defined budget $\tau > 0$:

$$\mathcal{B}_\infty(\mu_S, \tau) = \{\mu_{S'} \in \mathcal{M}(S) \colon W_\infty(\mu_S, \mu_{S'}) \leq \tau\}.$$

3 Graphs Representations Robust to Adversarial Attacks

In a widely adopted two-phase graph learning pipeline, the first step is to pre-train a graph encoder e (without the knowledge of any labels), which maps the joint input space $\mathcal{S}$ (*i.e.*, the graph topology $\mathcal{A}$ and node attributes $\mathcal{X}$) into some, usually lower-dimensional, representation space $\mathcal{Z}$. Then the encoded representation is used to solve some target tasks.

In this section, we explain how to obtain a well-qualified graph representation robust to adversarial attacks. We first propose a measure to quantify the robustness without label information in §3.1. In §3.2, we formulate an optimization problem to explore the trade-off between the expressive power and the robustness of the graph encoder. We then describe every component in the proposed optimization problem, and explain how we obtain a sub-optimal solution efficiently in §3.3.

3.1 Quantifying the Robustness of Graph Representations

In this section, we propose the *graph representation vulnerability* (GRV) to quantify the robustness of an encoded graph representation. Intuitively, the learned graph representation is robust if its quality does not deteriorate too much under adversarial attacks. Now we introduce in detail how to measure the quality of representations using MI, and how to describe the difference of representation quality before and after adversarial attacks.

The use of mutual information. A fundamental challenge to achieving a qualified graph representation is the need to find a suitable objective that guides the learning process of the graph encoder. In the case of unsupervised graph representation learning, the commonly used objectives are random walk-based (Perozzi et al. 2014; Grover et al. 2016) or reconstruction-based (Kipf et al. 2016). These objectives impose an inductive bias that neighboring nodes have similar representations. However, the inductive bias is easy to break under adversarial attacks (Jin et al. 2020; Entezari et al. 2020), because the connections among local neighborhoods are prone to be broken under adversarial attacks. As an alternative solution, we turn to maximize the MI between the input attributed graph and the representation output by the encoder, *i.e.*, $\mathrm{I}(S; e(S))$, from a more global view. In our case, maximizing the $\mathrm{I}(S; e(S))$ encourages the representations to be maximally informative about the input graph and to avoid the above-mentioned inductive bias.

Graph representation vulnerability. In addition to the measure of the quality of a graph representation, we also need to describe the robustness of a representation. Intuitively, an encoder is called robust if the value of the objective stays relatively stable after tiny perturbations on the input. Therefore, the encoder is robust if the MI before and after attack are close to each other. Thus, we propose the *graph representation vulnerability (GRV)* to quantify this difference:

$$\mathrm{GRV}_\tau(e) = \mathrm{I}(S; e(S)) - \inf_{\mu_{S'} \in \mathcal{B}_\infty(\mu_S, \tau)} \mathrm{I}(S'; e(S')), \tag{1}$$

where $S = (A, X)$ is the random variable following the benign data distribution, and $S' = (A', X')$ follows the adversarial distribution. The first term $\mathrm{I}(S; e(S))$ in (1) is the MI between the benign graph data and the encoded representation, while the term $\mathrm{I}(S', e(S'))$ uses the graph data after attack. The attack strategy $\mu_{S^\star}$ that results in the minimum MI is called the *worst-case* attack, and is defined as

$$\mu_{S^\star} = \operatorname*{argmin}_{\mu_{S'} \in \mathcal{B}_\infty(\mu_S, \tau)} \mathrm{I}(S'; e(S')).$$

Hence by definition, the graph representation vulnerability (GRV) describes the difference of the encoder's behavior using benign data and under the worst-case adversarial attack. A lower value of $\mathrm{GRV}_\tau(e)$ implies a more robust encoder to adversarial attacks. Formally, an encoder is called (τ, γ)-robust if $\mathrm{GRV}_\tau(e) \leq \gamma$.

An analogy to the graph representation vulnerability (GRV) has been studied in the image domain (Zhu et al. 2020). However, the extension of (Zhu et al. 2020) to the graph domain requires nontrivial effort. An image is considered to be a single continuous space while a graph is a joint space $\mathcal{S} = (\mathcal{A}, \mathcal{X})$, consisting of a discrete graph-structure space $\mathcal{A}$ and a continuous feature space $\mathcal{X}$. Moreover, the perturbation on the joint space $(\mathcal{A}, \mathcal{X})$ is difficult to track because a minor change in the graph topology or node attributes will propagate to other parts of the graph via edges. This is different in the image domain, where the distributions of all the pixels are assumed to be i.i.d.. Therefore, the discrete nature of graph topology and joint space $(\mathcal{A}, \mathcal{X})$ make the worst-case adversarial attack extremely difficult to estimate. Thus, the optimization method we apply is substantially different from that in (Zhu et al. 2020); see §3.2 and §3.3. Furthermore, more complicated analysis is needed to verify our approach in theory; see §4 for details.

3.2 Optimization Problem

The trade-off between model robustness and the expressive power of encoder has been well-studied (Tsipras et al. 2019; Zhang et al. 2019). In our case, this trade-off can be readily explored by the following optimization problem

$$\text{maximize} \quad \ell_1(\Theta) = \mathrm{I}(S; e(S)) - \beta \mathrm{GRV}_\tau(e), \tag{2}$$

where the optimization variable is the learnable parameters Θ of the encoder e, and $\beta > 0$ is a pre-defined parameter.

However, in practice, the "most robust" encoder is usually not the desired one (as it sacrifices too much in the encoder's expressive power). An intuitive example for the "most robust" encoder is the constant map, which always outputs the same representation whatever the input is. Hence, a "robust enough" encoder would be sufficient, or even better. To this end, we add a soft-margin γ to GRV, and obtain the following optimization problem

$$\text{maximize} \quad \ell_2(\Theta) = \mathrm{I}(S; e(S)) - \beta \max\{\mathrm{GRV}_\tau(e), \gamma\}. \tag{3}$$

The second term is positive if $\mathrm{GRV}_\tau > \gamma$ and constant otherwise. As a result, when the encoder is sufficiently robust, the second term in ℓ_2 does no contribution, and thus Problem (3) turns to the standard MI maximization using benign data. Furthermore, when $\beta = 1$, Problem (3) can be divided into two simple sub-problems, depending on the value of $\mathrm{GRV}_\tau(e)$:

$$\begin{cases} \max\limits_{\Theta} \inf\limits_{\mu_{S'} \in \mathcal{B}_\infty(\mu_S, \tau)} \mathrm{I}(S'; e(S')), & \text{if } \mathrm{GRV}_\tau > \gamma \\ \max\limits_{\Theta} \ \mathrm{I}(S; e(S)), & \text{otherwise.} \end{cases} \tag{4}$$

In this case ($\beta = 1$), when $\mathrm{GRV}_\tau(e) > \gamma$, the problem maximizes the MI under the worst-case adversarial attack. In other words, the robust encoder tries to maintain the mutual dependence between the graph data and the encoded representation, under all kinds of adversarial attacks. When the encoder is sufficiently robust (*i.e.*, $\mathrm{GRV}_\tau(e) \le \gamma$), the problem turns to maximize the encoder's expressive power.

GNN as the parameterized encoder. GNN has been extensively used as an expressive function for parameterizing the graph encoder (Kipf et al. 2016, 2017). In this paper, we adopt a one-layer GNN: $e(\mathbf{A}, \mathbf{X}) = \sigma(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{X}\Theta)$, where $\hat{\mathbf{A}}$ is the adjacency matrix with self-loops, $\hat{\mathbf{D}}$ is the corresponding degree matrix, σ is the ReLU function, and Θ is the learnable parameters.

3.3 Approximate Solution

Although we formulate robust graph learning as an optimization problem (4), this problem is still difficult to solve for several reasons. First of all, the mutual information $\mathrm{I}(S, e(S))$ is extremely hard to compute, mainly because $S = (A, X)$ is a joint random variable involving a high-dimensional discrete variable A. In addition, the search space of the adversarial attacks, $\mathcal{B}_\infty(\mu_S, \tau)$, is intractable to quantify: There is no conventional or well-behaved choice for the distance metric d in such a complicated joint space. Even when we know the metric, the distance between two random variables is difficult to calculate. Apart from the above challenges, the classical, well-known projected gradient descent algorithm does not work in the joint space $S = (A, X)$, and thus the worst-case adversarial attack $\mu_{S^\star}$ is no way to find. Therefore, we further address the above issues in detail.

MI estimation. Directly computing $\mathrm{I}(S; e(S))$ in Problem (4) is intractable, especially for a joint distribution $S = (A, X)$ which includes a high-dimensional discrete random variable A. Some authors propose to maximize the average MI between a high-level "global" representation and local input regions, and show significant improvement in the quality of representations (Hjelm et al. 2019; Veličković et al. 2019). Inspired by recent work Deep Graph Infomax (Veličković et al. 2019), we use a noise-contrastive type objective as an approximation of $\mathrm{I}(S; e(S))$:

$$\ell_{\text{enc}}(S, e) = \mathbb{E}_S\left[\log \mathcal{D}\left(z, z_\mathbf{G}\right)\right] + \mathbb{E}_{\tilde{S}}\left[\log\left(1 - \mathcal{D}\left(\tilde{z}, z_\mathbf{G}\right)\right)\right], \tag{5}$$

where z denotes the local/node representation obtained from a GNN encoder; $z_\mathbf{G} = \text{sigmoid}\left(\mathbb{E}_S(z)\right)$ denotes the global/graph representation; $\tilde{S}$ is the random variable of negative examples, and $\tilde{z}$ is the realization of $e(\tilde{S})$. The critic function $\mathcal{D}(z, z_\mathbf{G})$ represents the score assigned to a pair of local and global representations obtained from the natural samples (*i.e.*, the original graph), while $\mathcal{D}(\tilde{z}, z_\mathbf{G})$ is that obtained from negative samples. We adopt $\mathcal{D}_\Phi = \text{sigmoid}(z^T \Phi z_\mathbf{G})$, where Φ is a learnable scoring matrix. Finally, in practice, the expectation over an underlying distribution is typically approximated by the mean of n independent samples $\{(\boldsymbol{a}^i, \boldsymbol{x}^i)\}_{i \in [n]}$.

Adversarial distribution estimation. Besides the estimation of MI, another challenge in solving Problem (4) is how to find the worst-case adversarial distribution $\mu_{S^\star} \in \mathcal{B}_\infty(\mu_S, \tau)$. Here, we elaborate the three difficulties in finding $\mu_{S^\star}$, and explain in detail how we solve them one by one.

First, it is difficult to choose an appropriate metric d on the joint space $\mathcal{S} = (\mathcal{A}, \mathcal{X})$ that faithfully measures the distance between each pair of point elements. An intuitive choice for the distance between any pair of $\boldsymbol{s}_1 = (\boldsymbol{a}_1, \boldsymbol{x}_1)$ and $\boldsymbol{s}_2 = (\boldsymbol{a}_2, \boldsymbol{x}_2)$ in the joint metric space $(\mathcal{A}, d_\mathcal{A})$ and $(\mathcal{X}, d_\mathcal{X})$ would be the L_p-norm $\|\left(d_\mathcal{A}(\boldsymbol{a}_1, \boldsymbol{a}_2), d_\mathcal{X}(\boldsymbol{x}_1, \boldsymbol{x}_2)\right)\|_p$. However, this intuition fails in our case because the changes in both the graph topology and the node attributes are not in the same order of magnitude. Thereby, we have to consider the perturbations in $\mathcal{A}$ and $\mathcal{X}$ separately. With a little abuse of notation, we redefine the perturbation bound as follows:

$$\begin{aligned} \mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon) = \{(\mu_{A'}, \mu_{X'}) \in \mathcal{M}(\mathcal{A}) \times \mathcal{M}(\mathcal{X}) \mid \\ W_\infty(\mu_A, \mu_{A'}) \le \delta, W_\infty(\mu_X, \mu_{X'}) \le \epsilon\}, \end{aligned}$$

where the small positive numbers δ and ϵ play the role of perturbation budget now. This is indeed a subset of the previous search space $\mathcal{B}_\infty(\mu_S, \tau)$.

Moreover, although the search space has been restricted, the ∞-Wasserstein constrained optimization problem remains intractable: We still have no clue about the underlying probability distribution. Similar to what we did to estimate MI, we turn to replace the real data distribution with an empirical one. Suppose we have a set of i.i.d. samples $\{(\boldsymbol{a}^i, \boldsymbol{x}^i)\}_{i \in [n]}$ (note that $n = 1$ under a transductive learning setting), based on which we can compute the empirical distribution $(\hat{\mu}_A, \hat{\mu}_X)$. The empirical search space is

$$\begin{aligned} &\hat{\mathcal{B}}\left(\{\boldsymbol{a}^i\}_{i=1}^n, \{\boldsymbol{x}^i\}_{i=1}^n, \delta, \epsilon\right) \\ &= \left\{(\hat{\mu}_{A'}, \hat{\mu}_{X'}) \,\middle|\, \|\boldsymbol{a}^{i'} - \boldsymbol{a}^i\|_0 \le \delta, \|\boldsymbol{x}^{i'} - \boldsymbol{x}^i\|_\infty \le \epsilon, i \in [n]\right\}, \end{aligned}$$

where $\hat{\mu}_{A'}$ and $\hat{\mu}_{X'}$ are the empirical distributions computed from the perturbed samples $\{(\boldsymbol{a}^{i'}, \boldsymbol{x}^{i'})\}_{i \in [n]}$. Here we use the cardinality (*i.e.*, L_0-norm) to measure the change in graph topology, and the L_∞-norm to measure the change in continuous node attributes (when node attributes are discrete, or even binary, we can also use L_0-norm for them). Finally, we notice that the empirical space $\hat{\mathcal{B}}\left(\{\boldsymbol{a}^i\}_{i=1}^n, \{\boldsymbol{x}^i\}_{i=1}^n, \delta, \epsilon\right)$ is again a subset of $\mathcal{B}_\infty(\hat{\mu}_A, \hat{\mu}_X, \delta, \epsilon)$.

The remaining question is how to efficiently find the worst-case adversarial attack. The classical choice for the image domain, *i.e.*, the projected gradient descent (PGD) method (Madry et al. 2018), is no longer applicable in our case, as the graph topology is a Boolean random matrix. As a remedy for the discrete case, we adopt a projected gradient descent topology attack for graph topology (Xu et al. 2019a). More specifically, we first find a convex hull of the discrete feasible set, and apply the projected gradient method. A binary sub-optimal solution of worst-case $\boldsymbol{a}'$ is then recovered using random sampling. This projected gradient descent topology attack helps us identify the worst-case adversarial example efficiently.

4 Theoretical Connection to Label Space

In this section, we examine the ability of the proposed robust graph encoder of blocking perturbation and benefiting downstream tasks. To better understand the power of our robust model, we establish a theoretical connection between the robustness of representations (measured by our proposed GRV) and the robustness of the potential model built upon the representations. We take node classification as an example task, and the result can be easily generalized to other classical graph learning tasks. We first introduce the concept of *adversarial gap (AG)* to measure the robustness of the downstream node classifier, and then explore some interesting theoretical connections between GRV and AG.

Adversarial gap. Adversarial gap (AG) is a classical measure of robustness for node classification in inductive learning. Let $\boldsymbol{a}$ and $\boldsymbol{x}$ be the adjacency matrix and the attribute matrix of an induced subgraph. Denote by $(\mathcal{S}, d)$ the input metric space and $\mathcal{Y}$ the space of labels. For a node classifier $g\colon \mathcal{S} \to \mathcal{Y}$, we define the *adversarial risk* of g with the budget $\tau \geq 0$ as

$$\begin{aligned}&\mathrm{AdvRisk}_\tau(g)\\&\quad= \mathbb{P}\big[\exists \boldsymbol{s}' = (\boldsymbol{a}', \boldsymbol{x}') \in \mathcal{B}(\boldsymbol{s}, \tau), \text{s.t. } g(\boldsymbol{a}', \boldsymbol{x}') \neq y\big],\end{aligned}$$

where $\mathcal{B}(\boldsymbol{x}, \tau) = \{\boldsymbol{x}' \in \mathcal{X} \mid d(\boldsymbol{x}', \boldsymbol{x}) \leq \tau\}$. The *adversarial gap* (AG) is then defined as

$$\mathrm{AG}_\tau(g) = \mathrm{AdvRisk}_\tau(g) - \mathrm{AdvRisk}_0(g),$$

which measures the relative vulnerability of the given model g. Apparently from the definition, a smaller value of AG (or AdvRisk) implies a more robust node classifier g.

Table 1 briefly summarizes the robustness measures, including AG, RV and GRV. The traditional model robustness, adversarial gap (*i.e.*, $\mathrm{AG}_\epsilon(g)$ and $\mathrm{AG}_\tau(g)$), is based on the label space $\mathcal{Y}$, while the MI-based robustness measures (*i.e.*, $\mathrm{RV}_*(e)$ and $\mathrm{GRV}_*(e)$) are built upon the representation space $\mathcal{Z}$. The prior work (Zhu et al. 2020), which defines $\mathrm{RV}_\epsilon(e)$ on a single input space $\mathcal{X}$ in the image domain, has shown that $\mathrm{RV}_\epsilon(e)$ has a clear connection with classifier robustness. Comparatively, the graph representation venerability $\mathrm{GRV}_\tau(e)$ is defined on a joint input space $(\mathcal{A}, \mathcal{X})$

Robustness measure	Domain	Input space	Output space
$\mathrm{AG}_\epsilon(g)$	Image	Single $\mathcal{X}$	$\mathcal{Y}$
$\mathrm{AG}_\tau(g)$	Graph	Joint $(\mathcal{A}, \mathcal{X})$	$\mathcal{Y}$
$\mathrm{RV}_\epsilon(e)$	Image	Single $\mathcal{X}$	$\mathcal{Z}$
$\mathrm{GRV}_\tau(e)$	Graph	Joint $(\mathcal{A}, \mathcal{X})$	$\mathcal{Z}$

Table 1: Summary of robustness measures. Adversarial gap (AG) is built on the label space $\mathcal{Y}$, while representation vulnerability (RV) and graph representation vulnerability (GRV) are MI-based measures built on the representation space $\mathcal{Z}$. The subscript ϵ denotes the perturbation budget of $\boldsymbol{x}$ (*i.e.*, the image) on the image domain, while the subscript τ denotes the perturbation budget of $(\boldsymbol{a}, \boldsymbol{x})$ on the graph domain.

in the graph domain. Thus the new definition is essentially different from the one on the image domain due to the existence of both discrete and continuous input data structures. In what follows, some interesting theoretic are presented to show inherent relationship between the graph representation vulnerability $\mathrm{GRV}_\tau(e)$ and the adversarial gap $\mathrm{AG}_\tau(g)$. We first work on two special cases under each of the following assumptions. Both assumptions are imposed on the statistical independence between the input random variables (*i.e.*, A or X) and the output label Y.

- Topology-aware: given $X \perp Y$, $p(Y|A, X) = p(Y|A)$
- Attribute-aware: given $A \perp Y$, $p(Y|A, X) = p(Y|X)$

Special cases. To obtain a tractable surrogate model, we consider a simplified GNN-based encode architecture $\boldsymbol{z} = \boldsymbol{a}^T\boldsymbol{x}\boldsymbol{\Theta}$ (Wu et al. 2019a). Thus, the representation of each node depends only on its one-hop neighbors, and then the corresponding column of $\mathbf{A}$ can be used directly to compute the representation for each node. Additionally, inspired by (Miyato et al. 2017; Dai et al. 2019), in which perturbation on intermediate representations is defined, we opt to define the adversarial distribution w.r.t μ_{A^TX} instead of that w.r.t μ_A and μ_X respectively. This assumption is reasonable owing to our focus on the robustness of our model rather than the real attack strategies. Accordingly, we assume that the set of adversarial distributions is

$$\begin{aligned}&\mathcal{B}_\infty(\mu_{A^TX}, \rho)\\&\quad= \quad \{\mu_{A'^TX'} \in \mathcal{M}(\mathcal{H}) : W_\infty(\mu_{A^TX}, \mu_{A'^TX'}) \leq \rho\},\end{aligned}$$

where $\mathcal{H} = \{\boldsymbol{a}^T\boldsymbol{x} : \forall \boldsymbol{a} \in \mathcal{A}, \boldsymbol{x} \in \mathcal{X}\}$.

In Theorems 4.1 and 4.2, we denote by $\boldsymbol{a} \in \{0, 1\}^{|\mathbf{V}|}$ one column in $\mathbf{A}$ and $\boldsymbol{x} = \mathbf{X}$. The subscript ρ of GRV, AdvRisk and AG represents that they are defined via $\mathcal{B}_\infty(\mu_{A^TX}, \rho)$, while $\mathcal{F} = \{f : z \mapsto y\}$ denotes the set of non-trivial downstream classifiers, $f^* = \arg\min_{f\in\mathcal{F}} \mathrm{AdvRisk}_\rho(f \circ e)$ is the optimal classifier built upon e, and H_b is the binary entropy function. Moreover, when indexing $\boldsymbol{a}$ and $\boldsymbol{x}$, $\boldsymbol{a}_i$ denotes the i-th entry of $\boldsymbol{a}$ and $\boldsymbol{x}_i$ denotes the i-th row of $\boldsymbol{x}$.

Theorem 4.1 (Topology-aware) *Let $(\mathcal{A}, \|\cdot\|_0)$ and $(\mathcal{X}, \|\cdot\|_p)$ be the input metric spaces, $\mathcal{Y} = \{-1, +1\}$ be the label space and $\mathcal{Z} = \{-1, +1\}$ be the representation space. The set of encoders with $\Theta \in \mathbb{R}^{|\mathbf{V}|}$ is as follows:*

$$\mathcal{E} = \{e : (\boldsymbol{a}, \boldsymbol{x}) \in \mathcal{S} \mapsto \mathrm{sgn}[\boldsymbol{a}^T\boldsymbol{x}\boldsymbol{\Theta}] \mid \|\boldsymbol{\Theta}\|_2 = 1\}. \quad (6)$$

Assume that all samples $(\boldsymbol{s}, \boldsymbol{y}) \sim \mu_{SY}$ are generated from $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1, +1\}$, $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim} \mathrm{Bernoulli}(0.5 + \boldsymbol{y} \cdot (p - 0.5))$ and $\boldsymbol{x}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}_c)$ where $i = 1, 2, \ldots, |V|$ and $0 < p < 1$. Then, given $\rho \geq 0$, for any $e \in \mathcal{E}$, we obtain

$$\mathrm{GRV}_\rho(e) = 1 - H_b(0.5 + \mathrm{AG}_\rho(f^* \circ e)).$$

Next, consider a simpler case in which $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1, +1\}$ and $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim} \mathrm{Bernoulli}(0.5 + \boldsymbol{y} \cdot (p - 0.5))$ hold, but $\boldsymbol{x}_i = \mathbf{1}_c$, $i = 1, \ldots, |V|$ and the set of encoders follows such that $\mathcal{E} = \{e : (\boldsymbol{a}, \boldsymbol{x}) \mapsto \mathrm{sgn}[(\boldsymbol{a}^T\boldsymbol{x} - 0.5|\boldsymbol{V}|\mathbf{1}_c^T)\boldsymbol{\Theta}] \mid \|\boldsymbol{\Theta}\|_2 = 1\}$, which can be regarded as the non-attribute case. Then, given $\rho \geq 0$, for any $e \in \mathcal{E}$, we have

$$\mathrm{GRV}_\rho(e) \geq 1 - H_b\big(0.5 - 0.5\mathrm{AG}_\rho(f^* \circ e)\big) \quad (7a)$$

$$\mathrm{GRV}_\rho(e) \leq 1 - H_b\big(0.5 - \mathrm{AG}_\rho(f^* \circ e)\big) \quad (7b)$$

Theorem 4.1 reveals an explicit connection between $\mathrm{GRV}_\rho(e)$ and $\mathrm{AG}_\rho(f^* \circ e)$ achieved by the best classifier in the topology-aware case. We note that $H_b(\theta)$ is concave on $(0, 1)$ and that the maximum of H_b is attained uniquely at $\theta = 0.5$. Thus, a smaller GRV implies a smaller AG, and vice versa.

Theorem 4.2 (Attribute-aware) *Let $(\mathcal{A}, \|\cdot\|_0)$ and $(\mathcal{X}, \|\cdot\|_p)$ be the input metric spaces, $\mathcal{Y} = \{-1, +1\}$ be the label space and $\mathcal{Z} = \{-1, +1\}$ be the representation space. Suppose that the set of encoders is as in* (6). *Assume that the samples $(\boldsymbol{s}, \boldsymbol{y}) \sim \mu_{SY}$ are generated from $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1, +1\}$, $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim}$* Bernoulli(0.5) *and $\boldsymbol{x}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{y} \cdot \boldsymbol{\mu}, \sigma^2 I_c)$ where $i = 1, 2, \ldots, |\boldsymbol{V}|$. Then, given $\rho \geq 0$, for any $e \in \mathcal{E}$, we have:*

$$\mathrm{GRV}_\rho(e) = 1 - H_b\left(\tfrac{1}{2} - \mathrm{AG}_\rho(f^* \circ e)\right). \tag{8}$$

Next, consider a simpler case in which $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1, +1\}$, $\boldsymbol{x}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{y} \cdot \boldsymbol{\mu}, \sigma^2 I_c)$ but $\boldsymbol{a} \in \{0, 1\}^{|\boldsymbol{V}|}$, $\sum_{i=1}^{|\boldsymbol{V}|} \boldsymbol{a}_i = n_0 + n_1$, where $n_0 = |\boldsymbol{V}|/4 + \boldsymbol{y} \cdot (p - |\boldsymbol{V}|/4)$, $n_1 = |\boldsymbol{V}|/4 + \boldsymbol{y} \cdot (q - |\boldsymbol{V}|/4)$ and $p + q = |\boldsymbol{V}|/2$, $0 \leq p, q \leq |\boldsymbol{V}|/2$, $p, q \in \mathbb{Z}$; that is, $\boldsymbol{a}^T \boldsymbol{x}$ will aggregate n_0 samples with $\boldsymbol{y} = +1$ and n_1 samples with $\boldsymbol{y} = -1$. Further suppose that the set of encoders is as presented in (6). *Then, given $\rho \geq 0$,* (7) *also holds for any $e \in \mathcal{E}$.*

Similarly, we have $\mathrm{GRV}_\rho \propto \mathrm{AG}_\rho$ in Theorem 4.2. Note that Theorems 4.1 and 4.2 still hold when $\boldsymbol{a}$ contains self-loops.

General case. We illustrate a more general case in which Y is dependent on both A and X. In the general case, we can extend (Zhu et al. 2020, Theorem 3.4) to the graph domain. Regardless of the encoder, the theorem below provides a general lower bound of adversarial risk over any downstream classifiers that involves both MI and GRV.

Theorem 4.3 *(Zhu et al. 2020). Let $(\mathcal{S}, d)$ be the input metric space, $\mathcal{Z}$ be the representation space and $\mathcal{Y}$ be the label space. Assume that the distribution of labels μ_Y over $\mathcal{Y}$ is uniform and S is the random variable following the joint distribution of inputs μ_{AX}. Further suppose that $\mathcal{F}$ is the set of downstream classifiers. Given $\tau \geq 0$,*

$$\inf_{f \in \mathcal{F}} \mathrm{AdvRisk}_\tau(f \circ e) \geq 1 - \frac{I(S; e(S)) - \mathrm{GRV}_\tau(e) + \log 2}{\log |\mathcal{Y}|}$$

holds for any encoder e.

Theorem 4.3 suggests that lower adversarial risk over all downstream classifiers cannot be achieved without either lower GRV or higher MI between S and $e(S)$. It turns out that jointly maximizing $I(S; e(S))$ and minimizing $\mathrm{GRV}_\tau(e)$ enables the learning of robust representations. Note that Theorem 4.3 also holds in the graph classification task.

5 Experiments

In the experiments, we train our model in a fully unsupervised manner, and then apply the output representations to three graph learning tasks. Compared with non-robust and other robust graph representation models, the proposed model produces more robust representations to defend adversarial attacks. Furthermore, the superiority of our model still holds under different strengths of attacks and under various attack strategies.

5.1 Experimental Setup

For evaluation, we use three datasets, Cora, Citeseer and Polblogs, and compare our model with the following baselines.

- **Non-robust graph representation learning**: 1) Raw: concatenating graph topology and node attributes (only graph topology for Polblogs); 2) DeepWalk (Perozzi et al. 2014): a random walk-based unsupervised graph model; 3) DeepWalk+X: concatenating the Deepwalk embedding and the node attributes; 4) GAE (Kipf et al. 2016): variational graph auto-encoder and 5) DGI (Veličković et al. 2019): another unsupervised graph model based on MI.
- **Defense models**: 1) Dwns_AdvT (Dai et al. 2019): a defense model designed for Deepwalk; 2) RSC (Bojchevski et al. 2017): a robust unsupervised graph model via spectral clustering; 3) DGI-EdgeDrop (Rong et al. 2020): a defense model that works by dropping 10% of edges during training DGI; 4) DGI-Jaccard (Wu et al. 2019b): DGI applied to a pruned adjacency matrix in which nodes with low Jaccard similarity are forced to be disconnected; and 5) DGI-SVD (Entezari et al. 2020): DGI applied to a low-rank approximation of the adjacency matrix obtained by truncated SVD.

We also include Ours-soft, an variant of our model which removes soft margin on GRV.

Implementation details. In the training phase, we adopt the projected gradient descent topology attack (Xu et al. 2019a) and PGD attack (Madry et al. 2018) to construct adversarial examples of $\boldsymbol{a}$ and $\boldsymbol{x}$, respectively. We set γ = 5e-3, $\delta = 0.4|\mathbf{E}|$, and $\epsilon = 0.1$. For Polblogs, we do not perform attacks on the pseudo node attributes. In evaluation, we use the same attack strategy as in training, but set $\delta = 0.2|\mathbf{E}|$ to satisfy imperceptible constraint. Considering the training efficiency and the real attack during evaluation, the step size and iteration number are set different. Note that DeepWalk and RSC both require the entire graph, and thus we retrain them using polluted data. The evaluation is performed on node classification, link prediction, and community detection. We run 10 trials for all the experiments and report their average performance and standard deviation. Codes are available at: https://github.com/galina0217/robustgraph.

5.2 Results

Performance on downstream tasks. Table 2 summarizes the performance of different models in three tasks. We see that our model beats the best baseline by an average of +1.8% on the node classification, +1.8% on the link prediction and +45.8% on the community detection. It's worth noting that, in community detection, adversarial attacks can cause dramatic influence on model performance because the task itself is very sensitive to the global graph topology. The difference between the performance of our model and that of those non-robust graph learning models indicates the importance of defense. Moreover, our model still stands out with huge lead when compared with existing defense models. Besides, the ablation study, *i.e.*, comparing the last two rows in Table 2, shows the superiority of the soft margin on GRV.

	Node classification (Acc%)			Link prediction (AUC%)			Community detection (NMI%)		
Model \ Dataset	Cora	Citeseer	Polblogs	Cora	Citeseer	Polblogs	Cora	Citeseer	Polblogs
Raw	57.4±3.0	49.7±1.6	73.9±0.9	60.5±0.1	50.2±0.5	89.0±0.4	9.7±7.5	1.0±0.5	0.2±0.1
DeepWalk	56.2±1.1	16.5±0.9	80.4±0.5	55.4±0.8	50.3±0.3	89.2±0.7	34.6±0.6	11.1±1.0	0.4±0.5
DeepWalk + **X**	59.3±0.4	26.5±0.5	-	55.9±0.6	50.9±0.3	-	34.2±3.7	11.1±1.3	-
GAE	14.0±1.2	16.2±1.1	49.9±1.2	52.4±1.4	50.9±1.8	50.5±1.3	10.9±2.1	1.4±1.7	9.2±1.0
DGI	69.3±2.8	53.2±2.2	75.2±2.4	68.6±0.4	57.6±2.1	91.2±1.1	30.3±3.5	8.5±3.8	6.0±5.6
Dwns_AdvT	59.2±1.2	25.0±1.0	80.7±0.5	56.0±0.7	50.7±0.4	89.5±0.8	35.0±0.7	11.5±1.0	0.9±0.7
RSC	46.9±3.5	34.0±2.2	58.9±1.7	52.5±0.4	57.2±0.2	61.5±0.4	4.9±0.7	1.8±0.4	4.4±4.3
DGI-EdgeDrop	56.0±4.3	49.0±4.5	79.8±1.7	66.2±0.8	**61.3**±0.9	89.3±1.6	30.1±6.8	7.34±0.8	9.0±7.8
DGI-Jaccard	69.4±2.8	57.1±1.3	79.3±0.8	63.8±0.8	57.6±1.0	84.7±0.9	16.4±1.1	6.1±0.6	12.9±0.0
DGI-SVD	68.1±8.0	56.1±16.4	81.6±0.7	60.1±0.8	54.7±1.3	85.2±0.7	16.2±0.9	6.5±0.8	13.0±0.0
Ours-soft	69.4±0.7	57.5±2.0	79.7±2.1	68.1±0.3	58.2±1.3	90.3±0.5	39.2±8.8	23.5±1.9	12.6±9.6
Ours	**70.7±0.9**	**58.4**±1.4	**82.7**±2.2	**69.2**±0.4	59.8±1.3	**91.8**±0.4	**41.4**±4.7	**23.6**±2.8	**14.8**±2.7

Table 2: Summary of results for the node classification, link prediction and community detection tasks using polluted data.

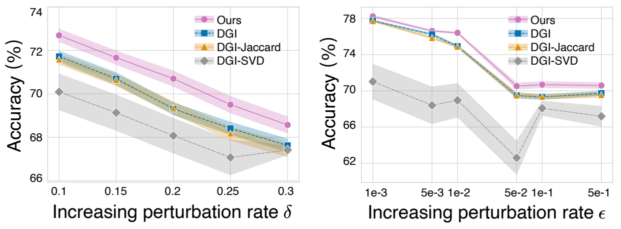


Figure 2: Accuracy of different models under various perturbation rates δ and ϵ. The downstream task is node classification and we use the Cora dataset for illustration. The shaded area indicates the standard deviation (×0.1) over 10 runs.

Model \ Attacker	Degree	Betw	Eigen	DW
Raw	87.4±0.3	84.1±0.8	86.4±0.6	87.9±0.4
DeepWalk	87.8±0.9	83.5±1.2	84.3±1.0	87.7±0.9
DeepWalk + **X**	85.8±2.7	82.7±2.1	85.0±1.1	88.3±0.9
GAE	83.7±0.9	81.0±1.6	81.5±1.4	85.4±1.1
DGI	86.6±1.1	84.8±1.2	84.8±1.0	86.4±1.1
Dwns_AdvT	88.0±1.0	84.1±1.3	84.6±1.0	88.0±0.8
RSC	52.1±1.3	51.9±0.7	51.4±0.5	52.6±1.1
DGI-EdgeDrop	87.1±0.3	**87.0**±0.6	80.5±0.5	86.3±0.3
DGI-Jaccard	82.1±0.3	80.7±0.4	80.6±0.3	82.2±0.2
DGI-SVD	86.5±0.2	85.6±0.2	86.1±0.2	85.3±0.3
Ours-soft	88.5±0.7	85.7±1.5	86.2±0.4	88.7±0.7
Ours	**89.3**±0.7	86.3±1.2	**86.7**±0.4	**89.0**±0.8

Table 3: Defense against different attackers on Polblogs for the node classification task.

Performance under different rates of perturbation. We further compare our model with several strong competitors under various strength of adversarial attacks on the graph topology and the node attributes, by choosing different perburation rates δ and ϵ, respectively. We use the node classification task and the Cora dataset as an illustrative example. As shown in Figure 2, the performance of our model is consistently superior to other competitors, both on average and in worst-case. Note that the strong competitor DGI generates negative samples in the training phase, and this might explain the robustness of the DGI model. Comparably, the high standard deviation of DGI-SVD might be attributed to the continuous low-rank approximation of the adjacency matrix: the output of truncated SVD is no longer a 0–1 matrix, which violates the discrete nature of graph topology.

Performance under other attack strategies. In practice, we do not know which kind of attack strategies the malicious users are going to use. Thus it is interesting and important to know the performance of our model across different types of adversarial attacks. We adapt some common attack strategies to the unsupervised setting and use them as baselines. 1) Degree/Betw/Eigen: flip edges based on the sum of the degree/betweenness/eigenvector centrality of two end nodes; 2) DW (Bojchevski et al. 2019a): a black-box attack method designed for DeepWalk. We set the size of the sampled candidate set to 20K, as suggested in (Bojchevski et al. 2019a). This time we consider the node classification task on Polblogs for illustration. This choice is convincing because all the above attack strategies only vary the graph topology, which is the only information we know about Polblogs. Results in Table 3 show that our model's superiority persists in three attack strategies out of four. Comparison between Table 2 and Table 3 shows that the projected gradient descent topology attack via MI is the most effective attack strategy used here, which verifies that our model learns the worst adversarial example that deteriorates the performance most.

6 Conclusion

In this paper, we study unsupervised robust representation learning on graphs. We introduce the graph representation vulnerability to quantify the robustness of an unsupervised graph encoder. After that we propose a robust unsupervised graph model that can enhance robustness as well as improve expressive power. We further build sound theoretical connections between GRV and one example task, node classification. Extensive experimental results demonstrate the effectiveness of our method on blocking perturbations on input graphs, regardless of the downstream tasks.

Acknowledgements. This work is supported by NSFC (62176233), the National Key Research and Development Project of China (2018AAA0101900), NSF III-1705169, Okawa Foundation Grant, Amazon Research Awards, Picsart gift, NSFC (71531006), the Major Scientific Project of Zhejiang Laboratory (2020MC0AE01) and the Fundamental Research Funds for the Central Universities (Zhejiang University New Generation Industrial Control System (NGICS) Platform).

References

Adamic, L. A.; and Glance, N. 2005. The Political Blogosphere and the 2004 U.S. Election: Divided They Blog. In *LinkKDD*, 36–43.

Bojchevski, A.; et al. 2017. Robust spectral clustering for noisy data: Modeling sparse corruptions improves latent embeddings. In *SIGKDD*, 737–746.

Bojchevski, A.; et al. 2019a. Adversarial attacks on node embeddings via graph poisoning. In *ICML*, 695–704.

Bojchevski, A.; et al. 2019b. Certifiable robustness to graph perturbations. In *NeurIPS*, 1656–1665.

Champion, T.; et al. 2008. The ∞-Wasserstein Distance: Local Solutions and Existence of Optimal Transport Maps. *SIAM Journal on Mathematical Analysis*, 40(1): 1–20.

Chen, L.; Li, J.; Peng, J.; Xie, T.; Cao, Z.; Xu, K.; He, X.; and Zheng, Z. 2020. A survey of adversarial learning on graph. *arXiv preprint arXiv:2003.05730*.

Dai, H.; Li, H.; Tian, T.; Huang, X.; Wang, L.; Zhu, J.; and Song, L. 2018. Adversarial attack on graph structured data. In *ICML*, 1115–1124.

Dai, Q.; Shen, X.; Zhang, L.; Li, Q.; and Wang, D. 2019. Adversarial Training Methods for Network Embedding. In *WWW*, 329–339.

Entezari, N.; Al-Sayouri, S. A.; Darvishzadeh, A.; and Papalexakis, E. E. 2020. All you need is low (rank) defending against adversarial attacks on graphs. In *WSDM*, 169–177.

Feurer, M.; Klein, A.; Eggensperger, K.; Springenberg, J.; Blum, M.; and Hutter, F. 2015. Efficient and Robust Automated Machine Learning. In *NeurIPS*, 2755–2763.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *ICML*, 1263–1272.

Grover, A.; et al. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*, 855–864.

Hamilton, W. L.; et al. 2017. Inductive representation learning on large graphs. In *NeurIPS*, 1025–1035.

Hjelm, R. D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Bachman, P.; Trischler, A.; and Bengio, Y. 2019. Learning deep representations by mutual information estimation and maximization. In *ICLR*.

Hu, W.; Liu, B.; Gomes, J.; Zitnik, M.; Liang, P.; Pande, V.; and Leskovec, J. 2019. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*.

Hu, Z.; Dong, Y.; Wang, K.; Chang, K.-W.; and Sun, Y. 2020. Gpt-gnn: Generative pre-training of graph neural networks. In *SIGKDD*, 1857–1867.

Jin, W.; Ma, Y.; Liu, X.; Tang, X.; Wang, S.; and Tang, J. 2020. Graph structure learning for robust graph neural networks. In *SIGKDD*, 66–74.

Kipf, T. N.; et al. 2016. Variational graph auto-Encoders. *NIPS Workshop on Bayesian Deep Learning*.

Kipf, T. N.; et al. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards deep learning models resistant to adversarial attacks. In *ICLR*.

Miyato, T.; et al. 2017. Adversarial training methods for semi-supervised text classification. In *ICLR*.

Perozzi, B.; et al. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*, 701–710.

Prechelt, L. 1998. Early stopping-but when? In *Neural Networks: Tricks of the trade*, 55–69. Springer.

Qiu, J.; Chen, Q.; Dong, Y.; Zhang, J.; Yang, H.; Ding, M.; Wang, K.; and Tang, J. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *SIGKDD*, 1150–1160.

Rong, Y.; Huang, W.; Xu, T.; and Huang, J. 2020. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. In *ICLR*.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine*, 29(3): 93–93.

Tanay, T.; et al. 2016. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, abs/1608.07690.

Tsipras, D.; Santurkar, S.; Engstrom, L.; Turner, A.; and Madry, A. 2019. Robustness may be at odds with accuracy. In *ICLR*.

Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep graph infomax. In *ICLR*.

Wang, B.; Jia, J.; Cao, X.; and Gong, N. Z. 2021. Certified robustness of graph neural networks against adversarial structural perturbation. In *SIGKDD*, 1645–1653.

Wang, X.; et al. 2019. GraphDefense: Towards robust graph convolutional networks. *arXiv preprint arXiv:1911.04429*.

Wu, F.; Zhang, T.; Souza Jr, A. H. d.; Fifty, C.; Yu, T.; and Weinberger, K. Q. 2019a. Simplifying graph convolutional networks. In *ICML*, 6861–6871.

Wu, H.; Wang, C.; Tyshetskiy, Y.; Docherty, A.; Lu, K.; and Zhu, L. 2019b. Adversarial examples for graph data: Deep insights into attack and defense. In *IJCAI*, 4816–4823.

Xu, H.; Ma, Y.; Liu, H.-C.; Deb, D.; Liu, H.; Tang, J.-L.; and Jain, A. K. 2020a. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, 17(2): 151–178.

Xu, J.; Sun, Y.; Jiang, X.; Wang, Y.; Yang, Y.; Wang, C.; and Lu, J. 2020b. Query-free Black-box Adversarial Attacks on Graphs. *arXiv preprint arXiv:2012.06757*.

Xu, J.; Yang, Y.; Pu, S.; Fu, Y.; Feng, J.; Jiang, W.; Lu, J.; and Wang, C. 2021. NetRL: Task-aware Network Denoising via Deep Reinforcement Learning. *IEEE Transactions on Knowledge and Data Engineering*, 1–1.

Xu, J.; Yang, Y.; Wang, C.; Liu, Z.; Zhang, J.; Chen, L.; and Lu, J. 2020c. Robust Network Enhancement from Flawed Networks. *IEEE Transactions on Knowledge and Data Engineering*, 1–1.

Xu, K.; Chen, H.; Liu, S.; Chen, P.-Y.; Weng, T.-W.; Hong, M.; and Lin, X. 2019a. Topology attack and defense for graph neural networks: An optimization perspective. In *IJCAI*, 3961–3967.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019b. How Powerful are Graph Neural Networks? In *ICLR*.

Yang, Y.; Xu, Y.; Sun, Y.; Dong, Y.; Wu, F.; and Zhuang, Y. 2021. Mining Fraudsters and Fraudulent Strategies in Large-Scale Mobile Social Networks. *IEEE Transactions on Knowledge and Data Engineering*, 33(1): 169–179.

Yang, Y.; Xu, Y.; Wang, C.; Sun, Y.; Wu, F.; Zhuang, Y.; and Gu, M. 2019. Understanding Default Behavior in Online Lending. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, 2043–2052. Association for Computing Machinery.

Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.; Ghaoui, L. E.; and Jordan, M. 2019. Theoretically Principled Trade-off between Robustness and Accuracy. In *ICML*, 7472–7482.

Zheng, Q.; Zou, X.; Dong, Y.; Cen, Y.; Yin, D.; Xu, J.; Yang, Y.; and Tang, J. 2021. Graph Robustness Benchmark: Benchmarking the Adversarial Robustness of Graph Machine Learning. *NeurIPS D&B*.

Zhu, S.; et al. 2020. Learning adversarially robust representations via Worst-Case mutual information maximization. In *ICML*, 11609–11618.

Zügner, D.; et al. 2019a. Adversarial attacks on graph neural networks via meta learning. In *ICLR*.

Zügner, D.; et al. 2019b. Certifiable robustness and robust training for graph convolutional networks. In *SIGKDD*, 246–256.

A Appendix

A.1 Notations

The main notations can be found in the following table.

Notation	Description
$\mathbf{G}, \mathbf{A}, \mathbf{X}$	The input graph, the adjacency matrix and the node attribute matrix of $\mathbf{G}$
$A, \boldsymbol{a}$	The random variable representing structural information and its realization
$X, \boldsymbol{x}$	The random variable representing attributes and its realization
$S, \boldsymbol{s}$	The random variable (A, X) and its realization $(\boldsymbol{a}, \boldsymbol{x})$
$\mathcal{A}, \mathcal{X}, \mathcal{S}, \mathcal{Z}, \mathcal{Y}$	The input space w.r.t graph topology, node attributes, their joint, representations and labels.
μ_X	The probability distribution of X
$\mu_{X'}$	The adversarial probability distribution of X'
$\hat{\mu}_X^{(n)}$	The empirical distribution of X
$\hat{\mu}_{X'}^{(n)}$	The adversarial empirical distribution of X'
e, f, g	The encoder function, the classifier function and their composition

Table 4: Description of major notations.

A.2 Algorithm

Overall, the algorithm we use is a variant of the classical gradient-based methods, as presented in Algorithm 1. In every iteration, we first find out the distribution of the worst-case adversarial attack, and thus we can calculate the value of GRV. If GRV is larger than γ, we try to enhance the robustness in this iteration, and apply one gradient descent step for the first sub-problem in Problem (4). Otherwise when $\text{GRV}_\tau(e) < \gamma$, the encoder is considered robust enough, and thus we focus on improving the expressive power by one gradient descent move in the second sub-problem in (4). As a small clarification, the stopping criterion in Algorithm 1 follows the famous early-stopping technique (Prechelt 1998).

The time complexity of our algorithm is $O(|\mathbf{V}|^2 + |\mathbf{V}| * c)$ (where $|\mathbf{V}|$ and c are the number of nodes and the dimension of node attributes), which mainly comes from the worst-case attack estimation. The mutual information solution only costs $O(|\mathbf{V}|)$. If a light version is desired, you can directly switch to another handy but more efficient attack strategy.

Algorithm 1: Optimization algorithm.

```
Input: Graph $\mathbf{G} = (\mathbf{A}, \mathbf{X})$, learning rate $\alpha$.
Output: Graph encoder parameters $\Theta$.
1:  Randomly initialize $\Theta$.
2:  while Stopping condition is not met do
3:      $\mu_{S^\star} \leftarrow \text{argmin}_{\mu_{S'} \in \mathcal{B}_\infty(\mu_S, \tau)} \mathrm{I}(S'; e(S'))$.
4:      $\text{GRV}_\tau(e) \leftarrow \mathrm{I}(S; e(S)) - \mathrm{I}(S^\star; e(S^\star))$.
5:      if $\text{GRV}_\tau(e) > \gamma$ then
6:          $\Theta \leftarrow \Theta - \alpha \nabla_\Theta \mathrm{I}(S^\star; e(S^\star))$.
7:      else
8:          $\Theta \leftarrow \Theta - \alpha \nabla_\Theta \mathrm{I}(S; e(S))$.
9:      end if
10: end while
    Return: $\Theta$.
```

A.3 Dataset Details

we use three benchmark datasets for evaluation: specifically, Cora, Citeseer (Sen et al. 2008) and Polblogs (Adamic and Glance 2005). The first two are citation networks commonly used for node classification, where nodes represent documents and edges represent the citation links between two documents. Each node has a human-annotated topic as the class label as well as a feature vector. The feature vector is a sparse bag-of-words representation of the document. All nodes are labeled to enable differentiation between their topic categories. Polblogs is a network of weblogs on the topic of US politics. Links between blogs are extracted from crawls of the blog's homepage. The blogs are labelled to identify their political persuasion (liberal or conservative). As Polblogs is a dataset without node attributes, we construct an identity matrix as its node attribute matrix. The detailed dataset statistics can be found in the following table.

Dataset	Type	# vertices	# edges	# classes	# attributes
Cora	Citation	2,810	7,981	7	1,433
Citeseer	Citation	2,110	3,757	6	3,703
Polblogs	Web	1,222	16,714	2	-

Table 5: Dataset statistics.

A.4 Implementation Details

Our evaluation is performed on three downstream tasks, and we explain the detailed settings below.

- Node classification: logistic regression is used for evaluation, and only accuracy score is reported as the test sets are almost balanced. For Cora and Citeseer, we use the same dataset splits as in (Kipf et al. 2017), but do not utilize the labels in the validation set. For Polblogs, we allocate 10% of the data for training and 80% for testing.
- Link prediction: logistic regression is used to predict whether a link exists or not. Following conventions, we generate the positive test set by randomly removing 10% of existing links and form the negative test set by randomly sampling the same number of nonexistent links. The training set consists of the remaining 90% of existing links and the same number of additionally sampled nonexistent links. We use the area under the curve (AUC) as the evaluation metric on the link prediction task.
- Community detection: following the basic schemes for community detection based on graph representation learning, we apply the learned representations to the K-means algorithm. The normalized mutual information (NMI) is used as the evaluation metric here.

We conduct all experiments on a single machine of Linux system with an Intel Xeon E5 (252GB memory) and a

NVIDIA TITAN GPU (12GB memory). All models are implemented in PyTorch [1] version 1.4.0 with CUDA version 10.0 and Python 3.7.

Implementations of our model. We train our proposed model using the Adam optimizer with a learning rate of 1e-3 and adopt early stopping with a patience of 20 epochs. We choose the one-layer GNN as our encoder and set the dimension of its last layer as 512. The weights are initialized via Xavier initialization. When evaluating the learned representations via the logistic regression classifier, we set its learning rate as 1e-2 and train 100 epochs.

In the training phase, the step size of the projected gradient descent topology attack is set to be 20 and the step size of PGD attack is set to be 1e-5. The iteration numbers of both attackers are set to be 10. In the testing phase, the step size of projected gradient descent topology attack is set to 1e-3. The iteration numbers are set to 50 for both attacks. Others attacker parameters are the same as that in the training phase.

Implementations of baselines. For all the baselines, we directly adopt their implementations and keep all the hyperparameters as the default values in most cases.

A.5 Additional Results

Empirical Connection between GRV and AG In §4, we established a theoretical connection between the graph representation vulnerability (GRV) and adversarial gap (AG) under different assumptions. Here, we also conduct experiments to corroborate whether a similar connection still holds in more complicated scenarios. Again, we take the node classification task on the Cora dataset as our illustrative example. We compare some metrics of three kinds of encoders: GNNs of which the last layers have dimensions 512, 384, and 256, respectively. The left of Figure 3 presents a positive correlation between the adversarial gap and the value of GRV. This numerical results shows that GRV is indeed a good indicator for the robustness of graph representations. Finally, as a supplementary experiment, the right of Figure 3 plots the prediction accuracy under polluted data versus our approximation of the objective function $\ell_2(\Theta) = \mathrm{I}(S; e(S)) - \mathrm{GRV}_\tau(e)$ (with $\beta = 1$). The figure shows a positive correlation between these two quantities, which verify the use of $\ell_2(\Theta)$ to enhance the adversarial accuracy.

[1] https://github.com/pytorch/pytorch

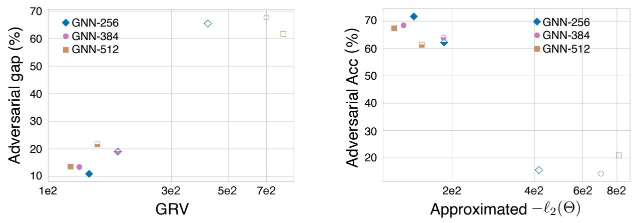
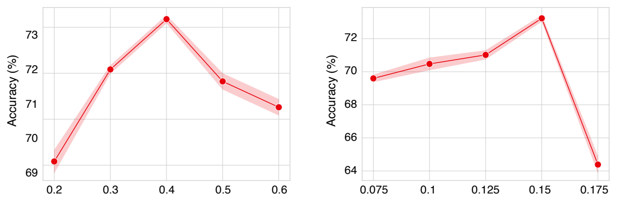
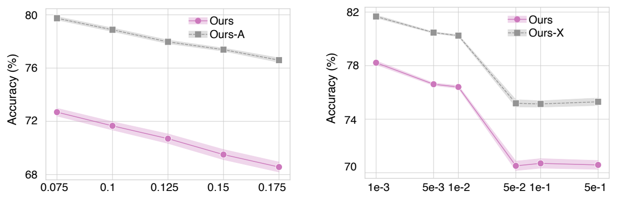


Figure 3: *Left.* Connection between GRV and AG. *Right.* Connection between adversarial accuracy and our approximation of $\ell_2(\Theta)$. Filled points, half-filled points, and unfilled points indicate our models with $\delta = 0.4$, $\epsilon = 0.1$, our model with $\delta = 0.1$, $\epsilon = 0.025$, and the DGI model, respectively.



(a) Budget of graph topology α (b) Budget of node attributes ϵ

Figure 4: Hyperparameter analysis.



(a) Increasing perturbation rate on graph topology (b) Increasing perturbation rate on node attributes

Figure 5: The performance of our model against the perturbations performed on a single input space $\mathcal{A}$ or $\mathcal{X}$ compared with that on the joint input space under increasing attack rate.

Sensitivity of the budget hyperparameters. The budget hyperparameters δ and ϵ determine the number of changes made to the original graph topology and node attributes respectively when finding the worst-case adversarial distribution, and are thus important hyperparameters in our proposed model. We use grid search to find their suitable values in our model through numerical experiments on Cora in Figure 4. Better performance can be obtained when $\alpha = 0.3$ and $\epsilon = 0.15$. We further observe that when α and ϵ are small, the budgets are not sufficient enough to find the worst-case adversarial distribution; while when α and ϵ are big, introducing too much adversarial attack will also lead to a decrease in the performance to some extent.

Defending perturbations on the joint input space is more challenging. In all the above-mentioned experiments, we further evaluate our model's robustness against the perturbations performed on the joint space $(\mathcal{A}, \mathcal{X})$. Here, we consider the perturbations performed on a single space $\mathcal{A}$ or $\mathcal{X}$ (*i.e.*, Ours-A/X: our model against perturbations performed on $\mathcal{A}$ or $\mathcal{X}$) under increasing attack rate as the setting of § 5.1, and present their results in Figure 5. We can see that when facing with the perturbations on the joint input space, the performance drops even more. This indicates that defending perturbations on the joint input space is more challenging, that is why we focus on the model robustness against perturbations on the joint input space.

A.6 Proofs

Proof of Theorem 4.1. For simplicity, we denote $n := |V|$. Let R^y be the random variable following the Binomial distribution according to A (*i.e.*, $R^{\boldsymbol{y}} = \sum_{i=1}^{n} A_i \sim B(n, 0.5 + \boldsymbol{y}(p-0.5)))$ and its realization $\boldsymbol{r}^{\boldsymbol{y}} = \sum_{i=1}^{n} \boldsymbol{a}_i$. Let H be the random variable following the Gaussian distribution according to A and X (*i.e.*, $H = A^T X \sim \mathcal{N}(\mathbf{0}, R^{\boldsymbol{y}}\sigma^2\boldsymbol{I})$), then its realization $\boldsymbol{h} = \boldsymbol{a}^T\boldsymbol{x}$ is exactly the aggregation operator of GNNs. We first compute the explicit formulation of the representation vulnerability $\mathrm{GRV}_\rho(e)$. Note that $\mathrm{I}(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$. For any given $e \in \mathcal{E}$, we have

$$\begin{aligned}\mathrm{GRV}_\rho(e) &= \mathrm{I}(S; e(S)) - \inf_{\mu_{A'^T X'} \sim \mathcal{B}_\infty(\mu_{A^T X}, \rho)} \mathrm{I}(S'; e(S')) \\ &= H_b(0.5) - \inf_{\mu_{H'} \sim \mathcal{B}_\infty(\mu_H, \rho)} H_b(e(S')),\end{aligned}$$

because $H(e(S)|S) = 0$, and the distribution of $\boldsymbol{h} = \boldsymbol{a}^T\boldsymbol{x}$, informally defined as $\boldsymbol{h} \sim 0.5\mathcal{N}(\mathbf{0}, \boldsymbol{r}^{+1}\sigma^2\boldsymbol{I}) + 0.5\mathcal{N}(\mathbf{0}, \boldsymbol{r}^{-1}\sigma^2\boldsymbol{I}))$, is symmetric w.r.t. 0. We thus have, $H_b(e(S)) = -P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta \geq 0)\log P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta \geq 0) - P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta < 0)\log P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta < 0) = H_b(0.5)$. We note that the binary entropy function $H_b(\theta) = -\theta\log(\theta) - (1-\theta)\log(1-\theta)$ is concave on $(0,1)$ and that the maximum of H_b is attained uniquely at $\theta = 0.5$. To obtain the infimum of $H_b(e(S'))$, we should either maximize or minimize $P_{\boldsymbol{s}'\sim\mu_{S'}}(\boldsymbol{h}'\Theta \geq 0)$.

Bound of $P_{\boldsymbol{s}'\sim\mu_{S'}}(\boldsymbol{h}'\Theta \geq 0)$. To achieve the bound of $P_{\boldsymbol{s}'\sim\mu_{S'}}(\boldsymbol{h}'\Theta \geq 0)$, we first consider the bound of $|\Delta\boldsymbol{h}\Theta|$ where $\Delta\boldsymbol{h} = \boldsymbol{h}' - \boldsymbol{h} = (\boldsymbol{a} + \Delta\boldsymbol{a})^T(\boldsymbol{x} + \Delta\boldsymbol{x}) - \boldsymbol{a}^T\boldsymbol{x}$. According to $\mu_{H'} \sim \mathcal{B}_\infty(\mu_H, \rho)$, we can get $\|\Delta H\|_p \leq \rho$ holds almost surely *w.r.t.* the randomness of H and the transport map defined by ∞-Wasserstein distance. Then, according to the Hölder's inequality, we have $|\Delta\boldsymbol{h}\Theta| \leq \|\Delta\boldsymbol{h}\|_p\|\Theta\|_q \leq \rho\|\Theta\|_q$, which indicates $P_{\boldsymbol{s}\sim\mu_S}(|\Delta\boldsymbol{h}\Theta| \leq \rho\|\Theta\|_q) \approx 1$. We have,

$$\begin{aligned}\underbrace{P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta - \rho\|\Theta\|_q \geq 0)}_{①} &\leq P_{\boldsymbol{s}'\sim\mu_{S'}}(\boldsymbol{h}'\Theta \geq 0) \\ &\leq \underbrace{P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta + \rho\|\Theta\|_q \geq 0)}_{②}.\end{aligned}$$

Compute GRV_ρ. Next, we will induce the more detailed formulations of the two bounds above. The lower bound is

$$\begin{aligned}① &= P_{\boldsymbol{h}\sim\mathcal{N}(\mathbf{0},\boldsymbol{r}\sigma^2\boldsymbol{I}),\boldsymbol{r}\sim\mu_R}(\boldsymbol{h}\Theta - \rho\|\Theta\|_q \geq 0) \\ &= P_{Z\sim\mathcal{N}(0,1)}P_{\boldsymbol{r}\sim\mu_R}(Z \geq \rho\|\Theta\|_q/\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2).\end{aligned}$$

Then according to De Moivre-Laplace Central Limit Theorem, we use Gaussian distribution to approximate Binomial distribution $\boldsymbol{r}^{\boldsymbol{y}}$ (*e.g.*, $\boldsymbol{r}^+ \to \mathcal{N}(np, npq)$ where $q = 1-p$). We have,

$$\begin{aligned}① &= \frac{1}{2}P_{Z\sim\mathcal{N}(0,1)}[P_{\boldsymbol{r}^+\sim B(n,p)}(Z\sqrt{\boldsymbol{r}^+}\sigma\|\Theta\|_2 \geq \rho\|\Theta\|_q) \\ &\qquad + P_{\boldsymbol{r}^-\sim B(n,1-p)}(Z\sqrt{\boldsymbol{r}^-}\sigma\|\Theta\|_2 \geq \rho\|\Theta\|_q)] \\ &\approx \frac{1}{2}P_{Z\sim\mathcal{N}(0,1),Z>0}[P_{Y\sim\mathcal{N}(0,1)}(Y \geq M\text{-}\sqrt{\frac{np}{q}}) \\ &\qquad + P_{Y\sim\mathcal{N}(0,1)}(Y \geq M\text{-}\sqrt{\frac{nq}{p}}) \\ &=: P_1/2 \quad (0 \leq P_1 \leq 1).\end{aligned}$$

where $M = (\rho^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2)/\sqrt{npq}$. Similarly, we have $② \approx 1/2 + P_2/2 \quad (0 \leq P_2 \leq 1)$. Thus, $\mathrm{GRV}_\rho(e) = H_b(1/2) - H_b(\max\{|P_1/2 - 1/2|, |P_2/2|\} + 1/2)$.

Compute AG_ρ. Given the formulation of GRV_ρ, we further aim to establish its connection to AG_ρ. Here we induce the detailed formulation of AG_ρ. In our case, the only two non-trivial classifiers to be discussed are $f_1(z) = z$ and $f_2(z) = -z$.

For given $e \in \mathcal{E}$, we have $\mathrm{AdvRisk}_\rho(f_1 \circ e)$ as:

$$\begin{aligned}③ :=& \mathrm{AdvRisk}_\rho(f_1 \circ e) \\ =& P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\exists\, \boldsymbol{s}' \in \mathcal{B}(\boldsymbol{h},\rho),\ \text{s.t. } \mathrm{sgn}(\boldsymbol{h}'\Theta) \neq \boldsymbol{y}] \\ =& P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\min_{\boldsymbol{s}'\in\mathcal{B}(\boldsymbol{h},\rho)} \boldsymbol{y}\cdot\boldsymbol{h}'\Theta \leq 0] \\ =& P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\boldsymbol{y}\cdot\boldsymbol{h}\Theta \leq -\min_{\Delta\boldsymbol{s}\in\mathcal{B}(0,\rho)} \boldsymbol{y}\cdot\Delta\boldsymbol{h}\Theta].\end{aligned}$$

Given that $|\Delta\boldsymbol{h}\Theta| \leq \rho\|\Theta\|_q$, we have $-\min_{\Delta\boldsymbol{s}\in\mathcal{B}(0,\rho)} \boldsymbol{y}\cdot\Delta\boldsymbol{h}\Theta = \rho\|\Theta\|_q$ holds for any $\boldsymbol{y}$ and

$$\begin{aligned}③ =& \frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}(\mathbf{0},\boldsymbol{r}^+\sigma^2\boldsymbol{I}),\boldsymbol{r}^+\sim B(n,p)}(\boldsymbol{h}\Theta \leq \rho\|\Theta\|_q) \\ &+ \frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}(\mathbf{0},\boldsymbol{r}^-\sigma^2\boldsymbol{I}),\boldsymbol{r}^-\sim B(n,q)}(\boldsymbol{h}\Theta \geq -\rho\|\Theta\|_q) \\ \approx& 1/2 + P_2/2.\end{aligned}$$

We also have $\mathrm{AdvRisk}_{\rho=0}(f_1 \circ e)$ as $④ := \mathrm{AdvRisk}_{\rho=0}(f_1 \circ e) = P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\boldsymbol{y}\cdot\boldsymbol{h}\Theta \leq 0] = 1/2$. Thus, $\mathrm{AG}_\rho(f_1 \circ e) = ③ - ④ = P_2/2$.

Similarly, for given $e \in \mathcal{E}$, we have $\mathrm{AdvRisk}_\rho(f_2 \circ e)$ as $⑤ := \mathrm{AdvRisk}_\rho(f_2 \circ e) \approx 1/2 + P_2/2$. We also have $\mathrm{AdvRisk}_{\rho=0}(f_2 \circ e) = 1/2$. We can get $\mathrm{AG}_\rho(f_2 \circ e) = ⑤ - ⑥ = P_2/2$.

As a result, we have $\mathrm{AG}_\rho(f_1 \circ e) = \mathrm{AG}_\rho(f_2 \circ e) = P_2/2$.

Connection between GRV and AG. Now we aim to find the connection between AG_ρ and GRV_ρ. Given their formulations derived above, it is easy to show that $P_1 + P_2 = 1$ is equivalent to $1/2 - P_1/2 = P_2/2$ and $|P_1/2 - 1/2| = |P_2/2|$. Then we have, $\mathrm{GRV}_\rho(e) = H_b(1/2) - H_b(P_2/2 + 1/2) = H_b(1/2) - H_b(1/2 + \mathrm{AG}_\rho(f^* \circ e))$.

■

Proof of Theorem 4.2. Similarly, let R represent the random variable following the Binomial distribution according to A (*i.e.*, $R = \sum_{i=1}^{n} A_i \sim B(n, 0.5)$) and its realization $\boldsymbol{r} = \sum_{i=1}^{n} \boldsymbol{a}_i$. We define $C_p(\boldsymbol{r}) = p - (\boldsymbol{r} - p) = 2p - \boldsymbol{r}$, where p represents the number of samples with $\boldsymbol{y} = +1$ connected to a node and $\boldsymbol{r} - p$ represents the number of samples with $\boldsymbol{y} = -1$ connected to a node (note that a node can be connected to itself). Let H represent the random variable following the Gaussian distribution according to A and X (*i.e.*, $H = A^T X \sim \mathcal{N}(C_p(R)\boldsymbol{\mu}, R\sigma^2\boldsymbol{I})$), then its realization $\boldsymbol{h} = \boldsymbol{a}^T\boldsymbol{x}$ is exactly the aggregation operator of GNNs.

We have the probability $P_{\boldsymbol{s}\sim\mu_S}(M)$ (where $M = l(\boldsymbol{h}\Theta \geq$

0) and $l(\boldsymbol{h}\Theta)$ represents some functions of $\boldsymbol{h}\Theta$) as,

$$\begin{aligned}
P_{\boldsymbol{s}\sim\mu_S}(M) =& P_{\boldsymbol{r}}[\sum_{p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}} P_{\boldsymbol{h}\sim\mathcal{N}(C_p(\boldsymbol{r})\mu,\boldsymbol{r}\sigma^2\boldsymbol{I})}(M)] \\
=& \frac{1}{2}P_{\boldsymbol{r}}[(\sum_{p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}} + \sum_{r-p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}}) \\
& P_{\boldsymbol{h}\sim\mathcal{N}(C_p(\boldsymbol{r})\mu,\boldsymbol{r}\sigma^2\boldsymbol{I})}(M)] \\
=& \frac{1}{2}P_{\boldsymbol{r}}[\sum_{p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}} (P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r})\mu,\boldsymbol{r}\sigma^2\boldsymbol{I})}(M) \\
& + P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p)\mu,\boldsymbol{r}\sigma^2\boldsymbol{I})}(M))].
\end{aligned}$$

where $\boldsymbol{r} \sim B(n, 0.5)$. The first equality holds because $C_p(\boldsymbol{r})$ is independent of $\boldsymbol{y}$. Note that if a node is connected with p samples with $\boldsymbol{y} = +1$, then it can also be connected with p samples with $\boldsymbol{y} = -1$ with the same probability given $\boldsymbol{r}$. We thus get a pair of probability $(P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r})\mu,\boldsymbol{r}\sigma^2\boldsymbol{I})}(M), P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p)\mu,\boldsymbol{r}\sigma^2\boldsymbol{I})}(M))$ for given $\boldsymbol{r}$ and p. The pair of Gaussian distribution has opposite mean and the same variance which perfectly match the conditions in (Zhu et al. 2020). So we can directly reuse the formulations.

The rest of the proof follows in a similar way as Theorem 4.1, so we omit it here.

■