Unsupervised Adversarially-Robust Representation Learning on Graphs

Jiarong Xu,[1] **Junru Chen,** [1] **Yang Yang,** [1] **Yizhou Sun,** [2] **Chunping Wang,** [3] **Jiangang Lu** [1]
[1] Zhejiang University
[2] University of California, Los Angeles
[3] FinVolution Group
xujr@zju.edu.cn, jrchen_cali@zju.edu.cn, yangya@zju.edu.cn, yzsun@cs.ucla.edu, wangchunping02@xinye.com, lujg@zju.edu.cn

Abstract

Recent works have demonstrated that deep learning on graphs is vulnerable to adversarial attacks, in that imperceptible perturbations on input data can lead to dramatic performance deterioration. In this paper, we focus on the underlying problem of learning robust representations on graphs via mutual information. In contrast to previous works measure the task-specific robustness based on the label space, we here take advantage of the representation space to study a task-free robustness measure given the joint input space w.r.t graph topology and node attributes. We formulate this problem as a constrained saddle point optimization problem and solve it efficiently in a reduced search space. Furthermore, we provably establish theoretical connections between our task-free robustness measure and the robustness of downstream classifiers. Extensive experiments demonstrate that our proposed method is able to enhance robustness against adversarial attacks on graphs, yet even increases natural accuracy.

1 Introduction

Despite their great success, recent studies have shown that deep learning on graphs such as graph neural networks (GNNs) and network embeddings, is vulnerable to adversarial examples (Dai et al. 2018a; Zügner and Günnemann 2019a; Bojchevski and Günnemann 2019a), *i.e.*, even imperceptible perturbations can easily mislead classifiers to make wrong predictions (Chen et al. 2020). This so-called adversarial vulnerability has given rise to tremendous concerns regarding their utilization in security-critical applications such as drug discovery (Gilmer et al. 2017) and financial surveillance (Paranjape, Benson, and Leskovec 2017), *etc.*

When tackling the adversarial vulnerability problem, we focus on the robustness of graph representation learning. Commonly, learning from graphs follows a basic setup: given a joint input space of graph topology (*i.e.*, $\mathcal{A}$) and node attributes (*i.e.*, $\mathcal{X}$), we learn an encoder $e : \mathcal{S} = (\mathcal{A}, \mathcal{X}) \to \mathcal{Z}$ that maps $\mathcal{S}$ to the representation space $\mathcal{Z}$, after which a classifier $f : \mathcal{Z} \to \mathcal{Y}$ further maps $\mathcal{Z}$ to the label space $\mathcal{Y}$. As their composition, $g = f \circ e$ is defined such that $(f \circ e)(\boldsymbol{s}) = f(e(\boldsymbol{s}))$, $\boldsymbol{s} \in \mathcal{S}$. Conventionally, the robustness of model g is often measured based on the label space $\mathcal{Y}$ (Hao-Chen et al. 2020; Zügner and Günnemann 2019b; Bojchevski and Günnemann 2019b); that is, we usually consider the model g to be robust if its predictions over the label space are insensitive to input perturbations. One of the most troubling is that there are many unsupervised learning models which can not access any information of the task-specific label space, thus they turn to take advantage of the representation space $\mathcal{Z}$ to measure the task-free robustness of encoder e. A recent work (Zhu, Zhang, and Evans 2020) proposes such a task-free mutual information-based robustness measure given a single input space $\mathcal{X}$ and explores its connection to the label space in image domain. However, adding perturbations both on the space $\mathcal{A}$ and $\mathcal{X}$ aligned with graph domain will make the robustness even harder to enhance – unlike image domain where the instances are assumed to be i.i.d., the adversarial perturbations of edges or node attributes are easy to propagate to other neighbors via the relational information on a graph. In this paper, we extend this robustness measure to the more challenging graph learning setting, where a joint input space $(\mathcal{A}, \mathcal{X})$ is considered.

In increasingly many interests in the field of unsupervised representation learning, there has been a revival of methods based on the mutual information (MI) maximization principle (Linsker 1988), *i.e.*, one aims to learn an encoder function e to maximize a tractable lower bound of MI between the input and its representation (Hjelm et al. 2019; Oord, Li, and Vinyals 2018; Kolesnikov, Zhai, and Beyer 2019; Bachman, Hjelm, and Buchwalter 2019). More recent efforts have also shed light on the graph domain, DGI (Veličković et al. 2019), achieves massive gains across unsupervised representation learning on graphs and even remains competitive with supervised architectures. However, to the best of our knowledge, the robustness of this unsupervised graph representation learning via MI is far from well-studied.

In this paper, we target on the critical yet far overlooked aspect of learning robust representation via MI on graphs. More specifically, we formulate our problem as a trade-off between the representation capability of nature and adversary in the objective of the MI maximization principle, providing an unsupervised way of learning robust representations on graphs. We further relax the problem to a constrained saddle point optimization problem. Considering the difficulty of solving this optimization problem on the joint input space w.r.t the graph topology and node attributes, we reduce the input search space to a more practical and explainable boundary and adopt a gradient-based method to efficiently find

Preprint.

solutions. We additionally establish theoretical connections between our extended robustness measure (relied on the representation space $\mathcal{Z}$) and the downstream classifiers (built on the label space $\mathcal{Y}$) on graphs. Extensive experiments demonstrate that the proposed model obtains an improvement of +2.9% in terms of accuracy on adversarial examples, and still achieves a +1.0% improvement on benign examples.

2 Preliminaries

Notation. 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. The corresponding lower-case letters (*e.g.*, $\boldsymbol{x}$ and $\boldsymbol{y}$) indicate the realizations of these variables. We denote the random variables of the probability distributions with 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 $\mathbf{A}_i$ denotes the vector at the i-th row. We use $(\mathcal{X}, d)$ to denote the metric space, where $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a distance function on $\mathcal{X}$. Let $\mathcal{B}(\boldsymbol{x}, \tau) = \{\boldsymbol{x}' \in \mathcal{X} : d(\boldsymbol{x}', \boldsymbol{x}) \leq \tau\}$. Furthermore, we let $\mathcal{M}(\mathcal{X})$ denote the set of all probability measures on $\mathcal{X}$.

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 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 certain attributes. Accordingly, we define our input as $\boldsymbol{s} = (\boldsymbol{a}, \boldsymbol{x}) \in \mathcal{S}$; thus we can think 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 of a node's subgraph under an inductive learning setting. We further define an encoder $e : \mathcal{S} \to \mathcal{Z}$ that 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 classifier $f : \mathcal{Z} \to \mathcal{Y}$ that maps a representation $\boldsymbol{z} \in \mathcal{Z}$ to a label $f(\boldsymbol{z}) \in \mathcal{Y}$. We go on to define $g = f \circ e$ as their composition, such that $(f \circ e)(\boldsymbol{a}, \boldsymbol{x}) = f(e(\boldsymbol{a}, \boldsymbol{x}))$. A more clear table of main notations can be found in the Appendix.

Admissible perturbations on graphs. We use the Wasserstein distance to ensure that the attackers can not modify the input graph completely. The Wasserstein distance (a.k.a. the Earth mover's distance) can be considered as an optimal transport problem: that is, we wish to move probability mass to change one distribution into another at minimal cost. Intuitively, we can imagine probability mass as piles of dirt or sand and Wasserstein distance can be interpreted as the minimal cost required to move the first pile to reconstruct the second pile. More specifically, consider the metric space $(\mathcal{S}, d)$; given two probability distributions μ_S and $\mu_{S'}$ on it, the p-th Wasserstein distance is defined as:

$$W_p(\mu_S, \mu_{S'}) = \left(\inf_{\pi \in \Pi(\mu_S, \mu_{S'})} \langle \pi, d^p \rangle \right)^{1/p}$$

where $\langle \pi, d^p \rangle = \int_{\mathcal{S}^2} d(\boldsymbol{s}, \boldsymbol{s}')^p \, \mathrm{d}\pi(\boldsymbol{s}, \boldsymbol{s}')$ and $\Pi(\mu_S, \mu_{S'})$ denotes the set of all transportation plans over $\mathcal{S}^2$ with marginals μ_S and $\mu_{S'}$ on the first and second factors, respectively. We choose ∞-Wasserstein distance (Champion, De Pascale, and Juutinen 2008), *i.e.*, the limit of Wasserstein distance, as the measure of two distributions throughout this work. We then define our ∞-Wasserstein perturbation bound:

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

where $\mu_{S'}$ denotes the adversarial distribution w.r.t μ_S here.

Conventional robustness measure. We here introduce some robustness measures built on the label space. Classification margin is commonly used to measure whether a node can be correctly classified, which has also been utilized to measure the robustness associated with GNNs (Zügner and Günnemann 2019b; Bojchevski and Günnemann 2019b).

Definition 1. *(Classification margin.) Let y^* denotes the ground truth class of the example $(\boldsymbol{a}, \boldsymbol{x})$, then the classification margin of $(\boldsymbol{a}, \boldsymbol{x})$ can be defined as:*

$$\textit{CM}(\boldsymbol{a}, \boldsymbol{x}, g, y^*) = \max_{y \in \mathcal{Y} \setminus \{y^*\}} \ln p(\hat{y} = y) - \ln p(\hat{y} = y^*),$$

where $\hat{y} = g(\boldsymbol{a}, \boldsymbol{x})$. The smaller the value of $\textit{CM}(\boldsymbol{a}, \boldsymbol{x}, g, y^*)$, the more robust g is w.r.t the example $(\boldsymbol{a}, \boldsymbol{x})$.

Drawn from the definition of classification margin, we can define adversarial risk and adversarial gap which measure the vulnerability of a given model under input perturbations on the joint input space (Zhu, Zhang, and Evans 2020).

Definition 2. *Let $(\mathcal{S}, d)$ denotes the input metric space. For any classification model $g : \mathcal{S} \to \mathcal{Y}$, we define the **adversarial risk** of g with the adversarial budget $\tau \geq 0$ as follows:*

$$\textit{AdvRisk}_\tau(g) = \mathbb{E}_{p(\boldsymbol{s}, y^*)}[\exists\, \boldsymbol{s}' = (\boldsymbol{a}', \boldsymbol{x}') \in \mathcal{B}(\boldsymbol{s}, \tau) \text{ s.t. } \textit{CM}(\boldsymbol{a}', \boldsymbol{x}', g, y^*) \geq 0].$$

*Based on $\textit{AdvRisk}_\tau(g)$, the **adversarial gap** is defined to measure the relative vulnerability of a given model g w.r.t τ as:*

$$\textit{AG}_\tau(g) = \textit{AdvRisk}_{\tau>0}(g) - \textit{AdvRisk}_{\tau=0}(g).$$

3 Adversarially Robust Representations

Recall that our focus is on learning robust graph representations via MI. In this section, we first formulate our problem as a constrained saddle point optimization problem (Section 3.1). Then, in Section 3.2, we provide an approximate solution for this constrained saddle point optimization problem, which enables us to solve this problem efficiently by adopting gradient-based approaches.

3.1 Optimization problem

It has been demonstrated in recent works that robustness is at odds with natural accuracy (Tsipras et al. 2019; Zhang et al. 2019). In line with these studies, we formulate our MI-based robust learning model as an optimization problem considering the trade-off between the representation capability of nature and adversary:

$$\text{maximize}_\Theta \quad \ell_1(\Theta) = \beta \mathrm{I}(S; e(S)) + (1-\beta) \mathrm{I}(S'; e(S')), \tag{1}$$

where Θ represents the learnable parameter matrix of encoder e_Θ (we simplify e_Θ as e), $S = (A, X)$ is the random variable following the benign data distribution, and $S' = (A', X')$ denotes the variable following the adversarial distribution.

A practically viable lower bound. To ensure the perturbations are imperceptible, the adversarial distribution $\mu_{S'}$ is bounded in $\mathcal{B}_\infty(\mu_S, \tau)$. For a generic model, it is tricky to randomly select an adversarial distribution to compute $\mathrm{I}(S'; e(S'))$. Thereby, the problem of maximizing $\ell_1(\Theta)$, i.e., Problem (1), can be properly relaxed to a problem of maximizing its lower bound $\ell_2(\Theta)$:

$$\begin{aligned}\ell_1(\Theta) &\geq \beta \mathrm{I}(S; e(S)) + (1-\beta) \inf_{\mu_{S'} \sim \mathcal{B}_\infty(\mu_S, \tau)} \mathrm{I}(S'; e(S')) \\ &= \mathrm{I}(S; e(S)) - (1-\beta)\mathrm{RV}_\tau(e) \triangleq \ell_2(\Theta), \qquad (2)\end{aligned}$$

where $\mathrm{RV}_\tau(e) = \mathrm{I}(S; e(S)) - \inf_{\mu_{S'} \sim \mathcal{B}_\infty(\mu_S, \tau)} \mathrm{I}(S'; e(S'))$.

The infimum in objective (2) is attained to learn robust representations from the worst-case mutual information. To take a close look at this formulation, we find that the representation vulnerability $\mathrm{RV}_\tau(e)$ is exactly an extended version of the notion proposed by (Zhu, Zhang, and Evans 2020) as a robustness measure of MI-based representation learning. A lower value of $\mathrm{RV}_\tau(e)$ implies that the encoder is more robust to perturbations of input data distribution. Formally, a representation is deemed (τ, m)-robust if $\mathrm{RV}_\tau(e) \leq m$.

Robust training. To ensure our model is (τ, m)-robust, we study an alternative robust hinge loss. To achieve this, we introduce a hinge loss penalty term to the standard formulation of $\mathrm{RV}_\tau(e)$ via the soft margin m. We directly apply β to replace $1-\beta$ in $\ell_2(\Theta)$ and thus obtain the objective:

$$\ell_3(\Theta) = \mathrm{I}(S; e(S)) - \beta \max(\mathrm{RV}_\tau(e), m).$$

The second term is positive if $\mathrm{RV}_\tau \geq m$ and zero otherwise – the model is robust with a margin of at most m. Effectively, if our model is sufficiently robust, the second term becomes zero, thus reducing $\ell_3(\Theta)$ to the standard mutual information maximization on natural (benign) examples with robustness guarantees. It can be further cast into a saddle point problem by setting $\beta = 1$ for simplicity in $\ell_3(\Theta)$, as follows:

$$\begin{cases} \max\limits_{\Theta} \inf\limits_{\mu_{S'} \sim \mathcal{B}_\infty(\mu_S, \tau)} \mathrm{I}(S'; e(S')), & \text{if } \mathrm{RV}_\tau \geq m \\ \max\limits_{\Theta} \mathrm{I}(S; e(S)), & \text{otherwise.} \end{cases} \qquad (3)$$

GNNs for parameterizing encoder. Commonly, graph neural networks (GNNs) take the graph topology and node attributes as inputs to learn node-level representations by applying convolutional filters (Wu et al. 2020). That being said, GNNs appear to be an expressive function for parameterizing the encoder e given the graph-structured data. Here, we adopt the encoder architecture proposed in (Veličković et al. 2019). More specifically, our encoder is a one-layer GNN that implements the following propagation rule (Kipf and Welling 2017): $e(\mathbf{A}, \mathbf{X}) = \sigma(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{X}\Theta)$, where $\hat{\mathbf{A}}$ is the adjacency matrix with self-loops, $\hat{\mathbf{D}}$ is the corresponding degree matrix, σ denotes the ReLU activation function and Θ is the learnable weight matrix.

3.2 Approximate solution

By far, we formulate the robust graph learning as a constrained saddle point optimization problem (as in Problem (3)). If we are to solve the problem using an efficient gradient-based method, we would like to first estimate MI via neural networks, since directly computing it would be intractable (see the **MI estimation** section for more details). The other challenge is that of how to find the adversarial distribution $\mu_{S'} \sim \mathcal{B}_\infty(\mu_S, \tau)$ in order to obtain the worst-case MI; thus, we aim to choose an appropriate search space and efficiently find the adversarial perturbations (see the **Adversarial distribution estimation** section for more details). Let us consider that we are under a graph transductive learning setting; that is, all nodes share the same $\boldsymbol{s} = (\boldsymbol{a}, \boldsymbol{x}) = (\mathbf{A}, \mathbf{X})$.

MI estimation. Directly optimizing $\mathrm{I}(S; e(S))$ in Problem (3) is intractable because estimating MI is a notoriously difficult task. In the field of MI-based unsupervised graph learning, DGI (Veličković et al. 2019) has achieved competitive performance. We thus follow DGI to provide a principled approach to adopt the Jensen Shannon divergence (JSD) MI estimator (Nowozin, Cseke, and Tomioka 2016):

$$\begin{aligned}\mathrm{I}^{(\mathrm{JSD})}(S; e(S)) =& \mathbb{E}_S[\log \mathcal{D}(\boldsymbol{z}, \boldsymbol{z}_\mathbf{G})] + \\ & \mathbb{E}_{\tilde{S}}[\log(1 - \mathcal{D}(\tilde{\boldsymbol{z}}, \boldsymbol{z}_\mathbf{G}))], \qquad (4)\end{aligned}$$

where $\boldsymbol{z}$ denotes the local representation, $\boldsymbol{z}_\mathbf{G} = \mathrm{sigmoid}(\mathbb{E}_S(\boldsymbol{z}))$ represents the global representation; $\tilde{S}$ is the random variable of negative examples, and $\tilde{\boldsymbol{z}}$ is the realization of $e(\tilde{S})$. Intuitively, the critic function $\mathcal{D}(\boldsymbol{z}, \boldsymbol{z}_\mathbf{G})$ represents the probability score assigned to a pair of local and global representation from the natural samples (*i.e.*, the original graph), while $\mathcal{D}(\tilde{\boldsymbol{z}}, \boldsymbol{z}_\mathbf{G})$ is that from negative samples. In practice, the expectation over an underlying distribution is usually approximated by the expectation of the empirical distribution over n independent samples $\{(\boldsymbol{a}^i, \boldsymbol{x}^i)\}_{i \in [n]}$.

Common choices for the critic function $\mathcal{D}$ include bilinear critics, separable critics, concatenated critics, or even inner product critics (Tschannen et al. 2020). Here, we choose the learnable bilinear critic as our critic function, *i.e.*, $\mathcal{D}_\Phi = \mathrm{sigmoid}(\boldsymbol{z}^T \Phi \boldsymbol{z}_\mathbf{G})$, where Φ is a learnable scoring matrix.

Adversarial distribution estimation. Given the MI estimator in Eq. (4), another challenge for solving Problem (3) is how to find the adversarial distribution $\mu_{S'} \sim \mathcal{B}_\infty(\mu_S, \tau)$ associated with the worst-case MI. Here, we address three difficulties of solving this challenge, as follows.

First, it is difficult to choose an appropriate metric d on the joint input space $\mathcal{S} = (\mathcal{A}, \mathcal{X})$ that faithfully measures the distance between each pair of point elements. Commonly, we can adopt the product metric to measure the Cartesian product $\mathcal{A} \times \mathcal{X}$ of two metric spaces $(\mathcal{A}, d_\mathcal{A})$ and $(\mathcal{X}, d_\mathcal{X})$, *i.e.*, $d = \|d_\mathcal{A}, d_\mathcal{X}\|_p$ (Deza and Deza 2009). More specifically, for any pair of points $(\boldsymbol{a}^1, \boldsymbol{x}^1), (\boldsymbol{a}^2, \boldsymbol{x}^2) \in \mathcal{S}$, the distance of them is calculated as $\|d_\mathcal{A}(\boldsymbol{a}^1, \boldsymbol{a}^2), d_\mathcal{X}(\boldsymbol{x}^1, \boldsymbol{x}^2)\|_p$. However, this approach fails when our goal is to ensure imperceptible perturbations on μ_A and μ_X separately. Taking our case as an example, we usually limit the number of edge flips (*i.e.*, ℓ_0) so that it aligns with the discrete properties of graph topology,

while we also limiting the ℓ_p norm of change in attributes as usually doing in image domain. Note that the budget for graph topology and the budget for node attributes are not in the same order of magnitude; thus, we reduce the search space $\mathcal{B}_\infty(\mu_S, \tau)$ (where $d = \|d_\mathcal{A}, d_\mathcal{X}\|_p$ and $\tau = \|\delta, \epsilon\|_p$) to its subset in order to bring it into alignment with the adversarial setting on graphs, *i.e.*,

$$\mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon) = \{(\mu_{A'}, \mu_{X'}) \in \mathcal{M}(\mathcal{A}) \times \mathcal{M}(\mathcal{X}) : W_\infty(\mu_A, \mu_{A'}) \leq \delta, W_\infty(\mu_X, \mu_{X'}) \leq \epsilon\}.$$

Indeed, we aim to find a suboptimal solution in $\mathcal{B}_\infty(\mu_S, \tau)$.

Second, despite the search space having been reduced, the ∞-Wasserstein constrained optimization problem remains intractable, as it is based on the assumption that the underlying probability distribution is known exactly. As the MI estimation does, we turn to replace the real data distributions by the empirical ones (*i.e.*, $\hat{\mu}_A^{(n)}$ and $\hat{\mu}_X^{(n)}$) from a finite sample set. However, directly solving ∞-Wasserstein constrained optimization problem from the empirical distribution is still nontrival due to the non-linearity and the absence of duality of ∞-Wasserstein distance (Champion, De Pascale, and Juutinen 2008). We thus reduce the constrains w.r.t ∞-Wasserstein via $\mathcal{B}(\boldsymbol{a}, \boldsymbol{x}, \delta, \epsilon)$, and define the reduced search space as follows:

$$\mathcal{A}(\mathcal{C}, \delta, \epsilon) = \{(\hat{\mu}^{(n)}_{\{\boldsymbol{a}^{i\prime}\}_{i\in[n]}}, \hat{\mu}^{(n)}_{\{\boldsymbol{x}^{i\prime}\}_{i\in[n]}}) : (\boldsymbol{a}^{i\prime}, \boldsymbol{x}^{i\prime}) \in \mathcal{B}(\boldsymbol{a}^i, \boldsymbol{x}^i, \delta, \epsilon) \ \forall i \in [n]\},$$

$$\text{where } \hat{\mu}^{(n)}_{\{*^{i\prime}\}_{i\in[n]}} = \frac{1}{n}\sum_{i\in[n]} \mathbf{1}_{\{*^{i\prime}\}},$$

$\mathcal{C} = \{(\boldsymbol{a}^i, \boldsymbol{x}^i)\}_{i\in[n]}$ denotes the set of n *i.i.d.* samples, and $\mathbf{1}$ represents the indicator function (*i.e.*, $\mathbf{1}_{\{*^{i\prime}\}}(x) = 1$ if $x \in \{*^{i\prime}\}$ and $\mathbf{1}_{\{*^{i\prime}\}}(x) = 0$ otherwise), and $\mathcal{B}(\boldsymbol{a}, \boldsymbol{x}, \delta, \epsilon) = \{(\boldsymbol{a}', \boldsymbol{x}') \in \{0,1\}^{|\mathbf{V}|\times|\mathbf{V}|} \times \mathbb{R}^{|\mathbf{V}|\times c} : \|\boldsymbol{a}' - \boldsymbol{a}\|_0 \leq \delta, \|\boldsymbol{x}' - \boldsymbol{x}\|_\infty \leq \epsilon\}$ is the empirical admissible perturbation bound around $\boldsymbol{a}$ and $\boldsymbol{x}$. Under the graph adversarial attack setting, we have $0 < \delta \ll |\mathbf{V}|^2$ as the budget constraint on the graph topology; furthermore, we use the ℓ_0 to measure the change to $\boldsymbol{a}$ due to its discrete nature. We also have $\epsilon \geq 0$ as the budget constraint on the node attributes and use the ℓ_∞ norm to measure the change to $\boldsymbol{x}$ due to its continuous nature in most cases [1]. It can thus easily be shown that $\mathcal{A}(\mathcal{C}, \delta, \epsilon) \subseteq \mathcal{B}_\infty(\hat{\mu}_A^{(n)}, \hat{\mu}_X^{(n)}, \delta, \epsilon)$.

Third, given the reduced search space $\mathcal{A}(\mathcal{C}, \delta, \epsilon)$, the remaining question is that of how to efficiently search the worst-case perturbations w.r.t $\boldsymbol{a}$ and $\boldsymbol{x}$. With the aid of first-order attack generation methods on images, it is easy to apply the projected gradient descent method (PGD) (Madry et al. 2018) to find the adversarial examples of $\boldsymbol{x}$ due to its continuous nature in most cases. However, continuous optimization methods can not be directly applied to the graph topology, which is a kind of Boolean variable. As suggested in (Xu et al. 2019a), we notice that projection onto the feasible set defined by the convex hull of the Boolean variables is doable. Thereby, we adopt this kind of projected gradient method on graphs, and then recover the binary worst-case solution of $\boldsymbol{a}$ via a randomization sampling approach. This kind of gradient-based method can help us efficiently find the adversarial examples w.r.t worst-case MI in Problem (3).

As a result, we can get our approximate solution of $\ell_3(\Theta)$:

$$\begin{cases} \max\limits_{\Theta,\Phi} \min\limits_{\hat{\mu}^{(n)}_{S'} \in \mathcal{A}(\mathcal{C},\delta,\epsilon)} \mathrm{I}^{(\mathrm{JSD})}\left(S'; e\left(S'\right)\right), & \text{if } \widehat{\mathrm{RV}}_\tau \geq m \\ \max\limits_{\Theta,\Phi} \mathrm{I}^{(\mathrm{JSD})}\left(S; e\left(S\right)\right), & \text{otherwise,} \end{cases} \tag{5}$$

where $\widehat{\mathrm{RV}}_\tau$ is the approximated value, the encoder's parameters Θ and critic's parameters Φ are jointly optimized in the outer maximization step. The overall algorithm can be found in the Appendix.

4 Theoretical Connection to Label Space

In this section, we provably establish a connection between our task-free robustness measure and the robustness of the downstream classifier on graphs. A brief summary of robustness measures in Table 1 recalls us that the traditional robustness of a model $\mathrm{AG}_*(g)$ is based on the label space $\mathcal{Y}$, while the MI-based robustness measure $\mathrm{RV}_*(e)$ used in this work is built upon the representation space $\mathcal{Z}$. Prior work (Zhu, Zhang, and Evans 2020), defining $\mathrm{RV}_\epsilon(e)$ on a single input space $\mathcal{X}$ in image domain has shown that $\mathrm{RV}_\epsilon(e)$ has a clear connection with the robustness of the classifier. Our extended notion $\mathrm{RV}_\tau(e)$, however, defined as it is on a joint input space $(\mathcal{A}, \mathcal{X})$ in graph domain, is different from images due to the existence of both discrete and continuous input data structures. In what follows, we demonstrate that an inherent relationship provably exists between the representation vulnerability $\mathrm{RV}_\tau(e)$ and the adversarial gap $\mathrm{AG}_\tau(g)$ under a joint input space, which is based on an assumption that is more aligned with the graph representation learning.

In exploring the connection to the label space $\mathcal{Y}$, one solution could be to simply assume that one of the input random variables (*i.e.*, A or X) and Y are independent. We first adopt the following two special cases, that is:

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

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 $\mathcal{X}$	$\mathcal{Z}$
$\mathrm{RV}_\tau(e)$	Graph	Joint $(\mathcal{A}, \mathcal{X})$	$\mathcal{Z}$

Table 1: Summary of robustness measures, which are defined based on the output space of g or e. Here, the adversarial gap (AG, Cf. Definition 2) is the task-specific measure, while representation vulnerability (RV) is a task-free MI-based measure proposed by (Zhu, Zhang, and Evans 2020). The subscript ϵ denotes the perturbation budget of $\boldsymbol{x}$ (*i.e.*, the image) on image domain, while the subscript τ denotes the perturbation budget of $(\boldsymbol{a}, \boldsymbol{x})$ on graph domain.

[1] We can also use the ℓ_0 to measure the change to $\boldsymbol{x}$ when the node attributes are discrete or binary in some cases.

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

Here, we first work on these two special cases under their assumptions. We then illustrate a more general case in which Y is dependent on both A and X. Detailed proofs of the following theorems can be found in the Appendix.

Special cases. Recall that in Section 3.1, we adopt GNNs to parameterize the encoder e. Here, we further simplify the encoder architecture to obtain a tractable surrogate model. That is, we remove the nonlinearities of GNNs following the method outlined by (Wu et al. 2019a), including the last layer, and adopt one-layer GNN, leading to $\boldsymbol{z} = \boldsymbol{a}^T\boldsymbol{x}\boldsymbol{\Theta}$. It is easy to show that the representation of each node depends only on its 1-hop neighbors. Therefore, rather than operating with the adjacency matrix $\mathbf{A}$, we can directly obtain the corresponding column of $\mathbf{A}$ to compute the representation for each node.

In Theorem 4.1 and 4.2, we use $\boldsymbol{a} \in \{0,1\}^{|\mathbf{V}|}$ to denote a column of $\mathbf{A}$ and $\boldsymbol{x} = \mathbf{X}$, the subscript δ, ϵ of RV, AdvRisk and AG represents that they are defined via $\mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon)$, $\mathcal{F} = \{f : z \mapsto y\}$ denotes the set of non-trivial downstream classifiers, $f^* = \arg\min_{f\in\mathcal{F}} \text{AdvRisk}_{\delta,\epsilon}(f \circ e)$ is the optimal classifier built upon e, and H_b is the binary entropy function. When indexing $\boldsymbol{a}$ and $\boldsymbol{x}$, $\boldsymbol{a}_i$ denotes 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:*

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

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

$$RV_{\delta,\epsilon}(e) = 1 - H_b(0.5 + AG_{\delta,\epsilon}(f^* \circ e)). \quad (7)$$

Next, consider a simpler case in which $\boldsymbol{y} \overset{u.a.r.}{\sim} U\{-1,+1\}$ and $\boldsymbol{a}_i \overset{i.i.d.}{\sim}$ Bernoulli$(0.5 + \boldsymbol{y}\cdot(p-0.5))$ hold but $\boldsymbol{x}_i = \mathbf{1}_c,\ i = 1,\ldots,|\boldsymbol{V}|$ and the set of encoders follows such that:

$$\mathcal{E} = \{e : (\boldsymbol{a}, \boldsymbol{x}) \mapsto \text{sgn}[(\boldsymbol{a}^T\boldsymbol{x} - 0.5|\boldsymbol{V}|\mathbf{1}_c^T)\boldsymbol{\Theta}]\ |\ \|\boldsymbol{\Theta}\|_2 = 1\},$$

which can be regarded as the non-attribute case. Then, given $\epsilon \geq 0, \delta \geq 0$, for any $e \in \mathcal{E}$, we have

$$\begin{aligned} 1 - H_b(0.5 - 0.5AG_{\delta,\epsilon}(f^* \circ e)) &\leq RV_{\delta,\epsilon}(e) \\ &\leq 1 - H_b(0.5 - AG_{\delta,\epsilon}(f^* \circ e)). \end{aligned} \quad (8)$$

Theorem 4.1 reveals that an explicit connection exists between representation vulnerability $\text{RV}_{\delta,\epsilon}(e)$ and adversarial gap $\text{AG}_{\delta,\epsilon}(f^* \circ e)$ achieved by the best classifier in the topology-aware case. We notice that $H_b(\theta)$ is concave on $(0,1)$ and that the maximum of H_b is attained uniquely at $\theta = 0.5$. Thus, lower representation vulnerability is the sufficient and necessary condition of a smaller adversarial gap.

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 the set of encoders is* (6)*. Assume the samples $(\boldsymbol{s}, \boldsymbol{y}) \sim \mu_{SY}$ are generated from $\boldsymbol{y} \overset{u.a.r.}{\sim} U\{-1,+1\}$, $\boldsymbol{a}_i \overset{i.i.d.}{\sim}$ Bernoulli(0.5) and $\boldsymbol{x}_i \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{y}\cdot\boldsymbol{\mu}, \sigma^2\boldsymbol{I}_c)$ where $i = 1,2,\ldots,|\boldsymbol{V}|$. Then, given $\epsilon \geq 0, \delta \geq 0$, for any $e \in \mathcal{E}$, we have:*

$$RV_{\delta,\epsilon}(e) = 1 - H_b(0.5 - AG_{\delta,\epsilon}(f^* \circ e)). \quad (9)$$

Next, consider a simpler case in which $\boldsymbol{y} \overset{u.a.r.}{\sim} U\{-1,+1\}$, $\boldsymbol{x}_i \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{y}\cdot\boldsymbol{\mu}, \sigma^2\boldsymbol{I}_c)$ 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 $\boldsymbol{y} = -1$. Suppose the set of encoders is (6)*. Then, given $\epsilon \geq 0, \delta \geq 0$,* (8) *also holds for any $e \in \mathcal{E}$.*

According to the attribute-aware case in Theorem 4.2, we can draw a similar conclusion as Theorem 4.1: *i.e.*, $\text{RV}_{\delta,\epsilon} \propto \text{AG}_{\delta,\epsilon}$. Note that Theorem 4.1 and Theorem 4.2 still hold when $\boldsymbol{a}$ is with self-loops. We leave the case in which Y is dependent on both A and X and the connection to other downstream tasks for future work.

General Case. In the general case, it is easy to extend (Zhu, Zhang, and Evans 2020, Theorem 3.4) to graph domain by directly replacing the perturbation bound of distribution μ_X with that of the joint distribution μ_{AX} (*i.e.*, $\mathcal{B}_\infty(\mu_{AX}, \tau)$). Given any encoders, the theorem below provides a general lower bound of adversarial risk over any downstream classifiers that involves both mutual information and representation vulnerability. We restate the theorem here.

Theorem 4.3. *(Zhu, Zhang, and Evans 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 $\tau \geq 0$,*

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

holds for any encoder e.

Theorem 4.3 suggests that smaller adversarial risk over all downstream classifiers cannot be achieved without either lower representation vulnerability or higher mutual information between S and $e(S)$. It turns out that jointly optimizing the objective of maximizing $I(S; e(S))$ and that of minimizing $\text{RV}_\tau(e)$ enables the learning of robust representations.

5 Experiments

5.1 Experimental setup

We conduct experiments on Cora, Citeseer (Sen et al. 2008) and Polblogs (Adamic and Glance 2005). As Polblogs is a dataset without node attributes, we construct an identity matrix as its node attribute matrix. In the training phase, we set m =5e-3 in Objective (5) and apply the graph PGD attack (Xu et al. 2019a) to find adversarial examples of $\boldsymbol{a}$ and the PGD attack (Madry et al. 2018) to find adversarial examples of $\boldsymbol{x}$ ($\delta = 0.4|\mathbf{E}|$ and $\epsilon = 0.1$) when estimating worst-case adversarial distribution. For Polblogs, we do not

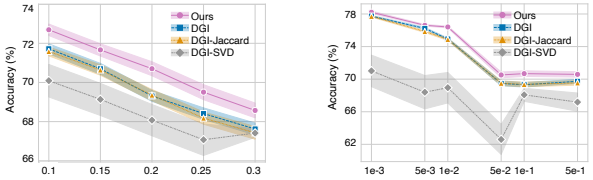
Method \ Dataset	Cora		Citeseer		Polblogs	
	Benign	Adv	Benign	Adv	Benign	Adv
Raw	60.1	57.4	52.0	49.7	84.2	73.9
DeepWalk	67.2	56.2	43.2	16.5	88.0	80.4
DeepWalk + $\mathbf{X}$	70.7	59.3	51.4	26.5	88.0	-
GAE	70.2	14.0	54.7	16.2	50.4	49.9
DGI	82.2	69.3	70.8	56.6	86.6	75.2
RSC	38.0	46.9	30.1	34.0	56.1	58.9
DGI-EdgeDrop	80.4	56.0	70.5	49.0	86.2	79.8
DGI-Jaccard	81.0	69.4	70.6	57.1	81.6	79.3
DGI-SVD	13.0	62.4	7.7	56.1	50.8	81.6
Ours-hinge	81.9	69.4	71.1	57.5	87.2	79.7
Ours	**82.9**	**70.7**	**71.8**	**60.2**	**88.8**	**82.7**

Table 2: Accuracy on benign and adversarial examples (*i.e.*, Adv) on the node classification task (the results with standard deviation can be found in the Appendix).

perform attacks on its constructed node attributes. In the testing phase, we use the same attack methods as in the training phase to evaluate model robustness, but set $\delta = 0.2|\mathbf{E}|$ due to the imperceptible constraint.

Baselines. We compare against the following unsupervised graph representation learning and defense baselines: 1). Raw: raw features concatenating the graph topology and the node attributes (we only use the graph topology on Polblogs); 2). DeepWalk (Perozzi, Al-Rfou, and Skiena 2014): a random walk-based unsupervised representation learning method that only considers the graph topology; 3). DeepWalk + $\mathbf{X}$: concatenating the Deepwalk embedding and the node attributes; 4). GAE (Kipf and Welling 2016): variational graph auto-encoder, an unsupervised representation learning method; 5). DGI (Veličković et al. 2019): another unsupervised representation learning method based on MI; 6). RSC (Bojchevski, Matkovic, and Günnemann 2017): a robust unsupervised representation learning method via spectral clustering; 7). DGI-EdgeDrop (Dai et al. 2018a): a cheap defense model that works by dropping 10% of edges during training DGI; 8). DGI-Jaccard (Wu et al. 2019b): preprocessing-based defense model by removing the edges whose two end nodes have low Jaccard similarity (we further utilize DGI to learn from the preprocessed matrix); 9). DGI-SVD (Entezari et al. 2020): another preprocessing-based defense model that operates on the truncated SVD to obtain the low-rank approximation (also following DGI) and 10). Ours-hinge: an ablation study of our model where the hinge-loss penalty is removed.

For evaluation, we train the logistic regression on the learned representations for node classification. For Cora and Citeseer, we choose the same dataset splits as in (Kipf and Welling 2017), but do not utilize the labels in the validation set. For Polblogs, we use 10% of the data for training and 80% for testing. In what follows, we report the accuracy score of all the experiments due to the balanced test set. The reported results are all averaged over 10 trials. In each trial, we record the average of 50 runs of logistic regression training. More detailed information of the datasets, implementations and analyses can be found in the Appendix.



(a) Increasing attack rate α (b) Increasing attack rate ϵ

Figure 1: Accuracy under increasing attack rate on graph topology (*i.e.*, α) and increasing attack rate on node attributes (*i.e.*, ϵ) on the Cora dataset. The shaded area indicates standard deviation ($\times 0.1$) over several random runs.

5.2 Results

Performance. Comparison results on benign and adversarial examples are summarized in Table 2. Note that as DeepWalk and RSC requires the whole graph to be available when learning the node embedding, we need to retrain it on adversarial examples. First, let us observe the results when meeting with adversarial examples, overall, we can observe a +2.9% improvement in the performance of our method. Compared with the unsupervised representation learning model DGI, our method exhibits an immediate improvement because it takes the worst-case mutual information into consideration. By contrast, the three defense models are only performed on graph topology, resulting in their sub-optimal defense accuracy. For instance, DGI-EdgeDrop performs poorly on some specific datasets because it is unable to distinguish the useful edges from the removed ones. Next, the results when dealing with benign examples surprisingly demonstrate that our method still achieves a +1.0% improvement. We explain this phenomenon by taking the worst-case adversarial examples generated in our model as augmented data (Madry et al. 2018). Yet, the defense model DGI-SVD fails on benign examples, even though it works well on adversarial cases. Another robust unsupervised method (RSC), however, performs somewhat poorly in both cases. As an ablation, Ours-hinge (that is, the hinge loss penalty is removed) drops to some extent in most cases. This demonstrates that it is better to enhance the accuracy if the encoder e is already (α, ϵ, m)-robust; moreover, considering the trade-off between accuracy and robustness would be helpful when the encoder e is more vulnerable to adversarial examples.

Robustness against increasing attack rate. We compare our method with some strong competitors in Figure 1. As shown, the performance of all the models decreases to some extent with increasing attack rate on the graph topology (*i.e.*, α) and the node attributes (*i.e.*, ϵ). Significantly, the performance of our model is always superior to that of other baselines. The most competitive DGI suggests us that taking advantage of negative examples can help distinguish the benign distribution from a noise one. The defense method, DGI-SVD, was found to have a high standard deviation. The reason is that its preprocessed matrix is a continuous low-rank approximation of an adjacency matrix, which disagrees with the discrete nature of graph topology.

	Degree	Betw	Eigen	DW
DGI	86.6	84.8	84.8	86.4
Ours	**89.3**	**86.3**	**86.7**	**89.0**

Table 3: Transferability against other attackers on Polblogs.

Transferability. The robustness against other attackers is important because, in practice, it is difficult to determine what attack method attackers will use. Since few general attacks follow the unsupervised learning setting, we compare against the following baselines: 1) Degree: flip edges based on the sum of the degree centrality of two end nodes; 2) Betw: flip edges based on the sum of the betweenness centrality of two end nodes; 3) Eigen: flip edges based on the sum of the eigenvector centrality of two end nodes; 4) DW (Bojchevski and Günnemann 2019a): a black-box attack method designed for DeepWalk. We set the size of the sampled candidate set to 20K, as suggested in DW (Bojchevski and Günnemann 2019a). We evaluate the robustness transferability of our model on the dataset whose graph topology matters most, *i.e.*, Polblogs, because the above-mentioned untargeted attacks only modify the graph topology. As can be seen from Table 3, the superiority of our model persists under different attackers and is superior to that of the strongest competitor DGI (+2.5% in accuracy). The results of our method against the less effective attackers Degree and DW are also higher than any of the models trained on benign examples (see Table 2). This is likely because the perturbed graph generated by these two attackers can be taken as augmented data similar with ours, which have been optimized via our worst-case adversarial examples. Another observation is that the graph PGD attack via MI (see Table 2) is more effective than the other attackers used here. This finding verifies that our attacker can find the worst-case adversarial examples and further learn from them in order to achieve robustness.

6 Related Works

Unsupervised representation learning on graphs. Learning graph representations in an unsupervised manner is a fundamental yet challenging problem due to the high-dimensional, sparse and discrete nature of graph data. More specifically, the goal is to learn an encoder function which maps the input graph data into low-dimensional representation space, yielding tractable downstream task solutions. Currently, the dominant algorithms for unsupervised graph representation learning methods rely on matrix factorization-based objectives (Von Luxburg 2007), random walk-based objectives (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016) and adjacency matrix reconstruction objectives (Kipf and Welling 2016; García-Durán and Niepert 2017). An alternative approach has been advocated recently, namely that of adopting the MI maximization principle (Linsker 1988) to learn graph representation (Sun et al. 2020; Veličković et al. 2019). Methods employing this approach achieve massive gains in standard metrics across unsupervised representation learning on graphs, and even remain competitive with supervised architectures. Whereas, when processing noisy or even adversarial graph data, unsupervised MI-based graph embeddings can easily lead to poor-quality learned representations. This prompts consideration of MI-based robust representations on graphs.

Robust graph learning. With a surge of adversarial attacks on graphs, several countermeasures have been proposed. Compared with pre/post-processing approaches, supported by empirical observations on specific attacks or models (mostly on GNNs), such as input denoising (Wu et al. 2019b; Entezari et al. 2020), adversarial detection (Bojchevski, Matkovic, and Günnemann 2017; Xu et al. 2019b; Ioannidis, Berberidis, and Giannakis 2019) and certifiable robustness guarantees on GNNs (Zügner and Günnemann 2019b; Bojchevski and Günnemann 2019b), several recent attempts have been made to formulate the graph defense problem as a minimax adversarial game (Deng, Dong, and Zhu 2019; Feng et al. 2019; Xu et al. 2019a; Chen et al. 2019; Wang, Liu, and Hsieh 2019). They often resort to adversarial training for optimization due to its excellent performance; however, these works usually require the additional label information and are tailored to attacks on GNNs. Sharing the similar adversarial training strategy, another line of works, on the other hand, do not depend on label information. For instance, a cheap method is to randomly drop edges during adversarial training (Dai et al. 2018a). Whereas, so long as no label information is exposed, there exist some works (Dai et al. 2019, 2018b; Yu et al. 2018) applying adversarial training to unsupervised representation learning, *i.e.*, DeepWalk and autoencoders. However, the robustness of unsupervised representation learning via MI on graphs is still an inherent blind spot.

The work most closely related to ours is that of (Zhu, Zhang, and Evans 2020), which develops a notion of representation vulnerability based on the worst-case MI in image domain. Inspired by this work, we extend this concept to graph domain and further demonstrate the theoretical connections between our extended robustness measure and the classifier robustness under a joint input space.

7 Conclusion

In this paper, we focus on the underlying problem of learning robust representations via mutual information guided by a task-free robustness measure on graphs. Accordingly, we formulate the robust learning problem as a constrained saddle point optimization problem and suggest solving this problem in a reduced search space. We further demonstrate that the robustness of our representation is theoretically related to the downstream classification task on the label space. However, we only provide a provable lower bound of adversarial risk in Theorem 4.3, which indicates that a robust classifier may not be guaranteed by optimizing our objective. Despite this limitation, our experimental results demonstrate the effectiveness of our proposed method on node classification task.

8 Ethics Statement

Recently, deep learning on graphs has shown excellent results in different real-world applications including molecule analysis (Hamilton, Ying, and Leskovec 2017), drug discovery (Gilmer et al. 2017), financial surveillance (Paranjape, Benson, and Leskovec 2017), recommendation sys-

tems (Wang et al. 2019), *etc*. Despite these impressive results, the research community has recently shown that graph learning models are vulnerable to adversarial attacks (Chen et al. 2020; Bojchevski and Günnemann 2019a; Zügner and Günnemann 2019a). A graph attacker can intentionally modify the original graph such that the modifications are almost human-imperceptible, which can cause wrong predictions in graph learning models. The real-life applications of such attacks can be very serious, for instance, one could create some false connections with rich people to increase his/her credit score in an online payment company, then he/she will be able to get loan and run away from the system. For that reason, there is a pressing need to learn robust representations on graphs against these adversarial attacks.

In this paper, we focus on the underlying problem of learning robust representations via mutual information on graphs and further establish a theoretical connection between our robustness measure and the robustness of downstream classifiers. Our approach and analysis provide us some novel perspectives to better understand the vulnerability of graph learning models when facing with graph adversarial attacks. Accordingly, we think our technology can bring beneficial impacts to the society. We would also encourage further works to develop better defense methods to maintain a more robust learning environment.

References

Adamic, L. A.; and Glance, N. 2005. The political blogosphere and the 2004 U.S. election: Divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*.

Bachman, P.; Hjelm, R. D.; and Buchwalter, W. 2019. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910* .

Bojchevski, A.; and Günnemann, S. 2019a. Adversarial attacks on node embeddings via graph poisoning. In *Proceedings of the 36th International Conference on Machine Learning*.

Bojchevski, A.; and Günnemann, S. 2019b. Certifiable robustness to graph perturbations. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*.

Bojchevski, A.; Matkovic, Y.; and Günnemann, S. 2017. Robust spectral clustering for noisy data: Modeling sparse corruptions improves latent embeddings. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Champion, T.; De Pascale, L.; and Juutinen, P. 2008. The ∞-wasserstein distance: Local solutions and existence of optimal transport maps. *SIAM Journal on Mathematical Analysis* 40(1): 1–20.

Chen, J.; Wu, Y.; Lin, X.; and Xuan, Q. 2019. Can adversarial network attack be defended? *arXiv preprint arXiv:1903.05994* .

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. 2018a. Adversarial attack on graph structured data. In *Proceedings of the 35th International Conference on Machine Learning*.

Dai, Q.; Li, Q.; Tang, J.; and Wang, D. 2018b. Adversarial network embedding. In *The 32nd AAAI Conference on Artificial Intelligence*.

Dai, Q.; Shen, X.; Zhang, L.; Li, Q.; and Wang, D. 2019. Adversarial training methods for network embedding. In *Proceedings of the 28th Web Conference*.

Deng, Z.; Dong, Y.; and Zhu, J. 2019. Batch virtual adversarial training for graph convolutional networks. *arXiv preprint arXiv:1902.09192* .

Deza, M. M.; and Deza, E. 2009. Encyclopedia of distances. In *Encyclopedia of distances*, 1–583. Springer.

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 *Proceedings of the 13th International Conference on Web Search and Data Mining*.

Feng, F.; He, X.; Tang, J.; and Chua, T.-S. 2019. Graph adversarial training: Dynamically regularizing based on graph structure. *IEEE Transactions on Knowledge and Data Engineering* .

García-Durán, A.; and Niepert, M. 2017. Learning graph representations with embedding propagation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*.

Hamilton, W. L.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*.

Hao-Chen, H. X. Y. M.; Deb, L. D.; Anil, H. L. J.-L. T.; and Jain, K. 2020. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing* 17(2): 151–178.

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 *Proceedings of the 7th International Conference on Learning Representations*.

Ioannidis, V. N.; Berberidis, D.; and Giannakis, G. B. 2019. GraphSAC: Detecting anomalies in large-scale graphs. *arXiv preprint arXiv:1910.09589* .

Kipf, T. N.; and Welling, M. 2016. Variational graph auto-Encoders. *NIPS Workshop on Bayesian Deep Learning* .

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*.

Kolesnikov, A.; Zhai, X.; and Beyer, L. 2019. Revisiting self-supervised visual representation learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Linsker, R. 1988. Self-organization in a perceptual network. *Computer* 21(3): 105–117.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards deep learning models resistant to adversarial attacks. In *Proceedings of the 6th International Conference on Learning Representations*.

Nowozin, S.; Cseke, B.; and Tomioka, R. 2016. f-GAN: Training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* .

Paranjape, A.; Benson, A. R.; and Leskovec, J. 2017. Motifs in temporal networks. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Sen, P.; Namata, G. M.; Bilgic, M.; Getoor, L.; Gallagher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI Magazine* 29(3): 93–106.

Sun, F.-Y.; Hoffman, J.; Verma, V.; and Tang, J. 2020. InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *Proceedings of the 8th International Conference on Learning Representations*.

Tschannen, M.; Djolonga, J.; Rubenstein, P. K.; Gelly, S.; and Lucic, M. 2020. On mutual information maximization for representation learning. In *Proceedings of the 8th International Conference on Learning Representations*.

Tsipras, D.; Santurkar, S.; Engstrom, L.; Turner, A.; and Madry, A. 2019. Robustness may be at odds with accuracy. In *Proceedings of the 7th International Conference on Learning Representations*.

Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep graph infomax. In *Proceedings of the 7th International Conference on Learning Representations*.

Von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and computing* 17(4): 395–416.

Wang, H.; Zhang, F.; Zhang, M.; Leskovec, J.; Zhao, M.; Li, W.; and Wang, Z. 2019. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Wang, X.; Liu, X.; and Hsieh, C.-J. 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 *Proceedings of the 36th International Conference on Machine Learning*.

Wu, H.; Wang, C.; Tyshetskiy, Y.; Docherty, A.; Lu, K.; and Zhu, L. 2019b. Adversarial examples on graph data: Deep insights into attack and defense. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 1–21.

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 *Proceedings of the 28th International Joint Conference on Artificial Intelligence*.

Xu, X.; Yu, Y.; Li, B.; Song, L.; Liu, C.; and Gunter, C. 2019b. Characterizing malicious edges targeting on graph neural networks .

Yu, W.; Zheng, C.; Cheng, W.; Aggarwal, C. C.; Song, D.; Zong, B.; Chen, H.; and Wang, W. 2018. Learning deep network representations with adversarially regularized autoencoders. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Zhang, H.; Yu, Y.; Jiao, J.; Xing, E. P.; Ghaoui, L. E.; and Jordan, M. I. 2019. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573* .

Zhu, S.; Zhang, X.; and Evans, D. 2020. Learning adversarially robust representations via Worst-Case mutual information maximization. In *Proceedings of the 37th International Conference on Machine Learning*.

Zügner, D.; and Günnemann, S. 2019a. Adversarial attacks on graph neural networks via meta learning. In *Proceedings of the 7th International Conference on Learning Representations*.

Zügner, D.; and Günnemann, S. 2019b. Certifiable robustness and robust training for graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

A Appendix

A.1 Notations

The main notations can be found in the Table 4.

Notation	Description
$\mathbf{G}$, $\mathbf{A}$, $\mathbf{X}$	The input graph, the adjacency matrix and the node attribute matrix of $\mathbf{G}$
$\mathbf{V}$, $\mathbf{E}$	The node set and edge set of $\mathbf{G}$
$\|\mathbf{V}\|$, $\|\mathbf{E}\|$	The number of nodes and the number of edges 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, he input space w.r.t node attributes, the joint input space of $(\mathcal{A}, \mathcal{X})$, the representation space, and the label space
$\mathcal{M}(\mathcal{A})$, $\mathcal{M}(\mathcal{X})$, $\mathcal{M}(\mathcal{S})$	The set of all probability distributions on $\mathcal{A}$, $\mathcal{X}$ and $\mathcal{S}$
μ_X	The probability distribution of random variable X
$\mu_{X'}$	The adversarial probability distribution of random variable X'
$\hat{\mu}_X^{(n)}$	The empirical distribution of random variable X
$\hat{\mu}_{X'}^{(n)}$	The adversarial empirical distribution of random variable X'
e, f, g	The encoder function, the classifier function and their composition
CM	Classification margin
AdvRisk	Adversarial risk
AG	Adversarial gap
RV	Representation Vulnerability

Table 4: Description of some major notations.

A.2 Dataset details

In the following experiments, 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 Table 5.

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

Table 5: Dataset statistics

A.3 Implementation Details

Here, we provide additional implementation details of our experiments.

Hardware Configuration. 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).

Software Configuration. All models are implemented in PyTorch [2] 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.

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

In the training phase, we set m =5e-3 in Objective (5) and apply the graph PGD attack (Xu et al. 2019a) to find adversarial examples of $\boldsymbol{a}$ and the PGD attack (Madry et al. 2018) to find adversarial examples of $\boldsymbol{x}$ ($\delta = 0.4|\mathbf{E}|$ and $\epsilon = 0.1$) when estimating worst-case adversarial distribution. For Polblogs, we do not perform attacks on its constructed node attributes. The step size of the graph PGD 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, we set the perturbation budget on graph topology δ to be 20% of the number of existing edges and the perturbation budget ϵ on node attributes to 0.1 due to the imperceptible constraints. The step size of PGD 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. When evaluating the learnt representations via the logistic regression classifier, we set its learning rate as 1e-2 and train 100 epochs.

Implementations of baselines. For all the baselines, we directly adopt their implementations and keep all the hyperparameters as the default values. The detailed implementations can be found in Table 6. Note that for GAE, we adopt their graph variational autoencoder version.

Method	Implementation code
Deepwalk	https://github.com/phanein/deepwalk
GAE	https://github.com/tkipf/gae
DGI	https://github.com/PetarV-/DGI
RSC	https://github.com/abojchevski/rsc
DGI-Jaccard	https://github.com/DSE-MSU/DeepRobust
DGI-SVD	https://github.com/DSE-MSU/DeepRobust
DW	https://github.com/abojchevski/node_embedding_attack

Table 6: Implementation details of baseline

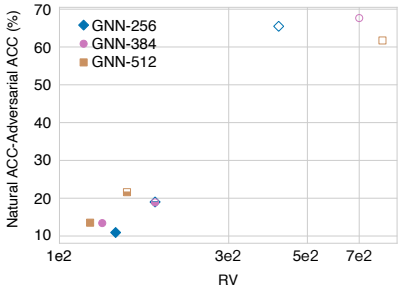
A.4 Additional results

Results with standard deviation. We here report the detailed accuracy with standard deviation on benign and adversarial examples on the node classification task in Table 7.

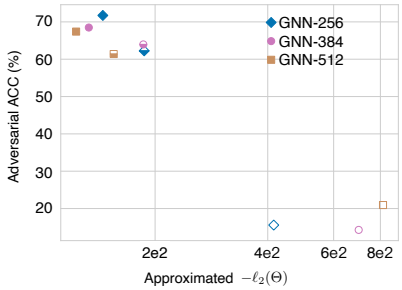
Methods \ Dataset	Cora		Citeseer		Polblogs	
	Benign	Adv	Benign	Adv	Benign	Adv
Raw	60.1±0.2	57.4±3.0	52.0±0.2	49.7±1.6	84.2±0.1	73.9±0.9
DeepWalk	67.2±0.8	56.2±1.1	43.2±0.7	16.5±0.9	88.0±0.4	80.4±0.5
DeepWalk + $\mathbf{X}$	70.7±0.9	59.3±0.4	51.4±0.8	26.5±0.5	88.0±0.5	-
GAE	70.2±1.9	14.0±1.2	54.7±2.6	16.2±1.1	50.4±1.4	49.9±1.2
DGI	82.2±0.5	69.3±2.8	70.8±0.6	56.6±3.4	86.6±0.9	75.2±2.4
RSC	38.0±0.7	46.9±3.5	30.1±1.2	34.0±2.2	56.1±1.6	58.9±1.7
DGI-EdgeDrop	80.4±0.2	56.0±4.3	70.5±0.2	49.0±4.5	86.2±0.2	79.8±1.7
DGI-Jaccard	81.0±0.6	69.4±2.8	70.6±0.7	57.1±1.3	81.6±0.6	79.3±0.8
DGI-SVD	13.0±0.0	62.4± 8.0	7.7±0.0	56.1±16.4	50.8±0.0	81.6±0.7
Ours-hinge	81.9±0.3	69.4±0.7	71.1±0.2	57.5±2.0	87.2±0.1	79.7±2.1
Ours	**82.9±0.5**	**70.7±0.9**	**71.8±0.7**	**60.2±2.3**	**88.8±0.5**	**82.7±2.2**

Table 7: The experimental results with standard deviation on node classification.

Empirical connection between RV and AG. Recall that in Section 4, we have established a theoretical connection between representation vulnerability RV and adversarial gap AG. However, these theoretical connections are conditioned on given distributions, given encoder architecture, *etc*. Here, we aim to figure out whether such kind of connection still exist in practice. Figure 2(a) presents the values of RV and AG of three kinds of encoder (*i.e.*, GNN-512/384/256: GNN with 512/384/256 dimension of the last layer) in the testing phase on Cora dataset. The lower the value of adversarial gap, the lower the value of natural-adversarial accuracy gap, the more robust the encoder. Models with low representation vulnerability tend to have low gap between natural and adversarial accuracy, which is aligned with the intuition of the definition of representation vulnerability. Based on the empirical connection between RV and AG, we further present the connection between our approximated $\ell_2(\Theta) = \mathrm{I}\left(S; e\left(S\right)\right) - \mathrm{RV}_\tau(e)$ ($\beta = 1$) and the adversarial accuracy. From Figure 2(b), we can clearly see that optimizing $\ell_2(\Theta)$ can help us enhance the adversarial accuracy.



(a) Connection between representation vulnerability and natural-adversarial accuracy gap.

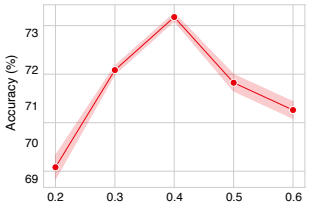


(b) Connection between our objective and adversarial accuracy.

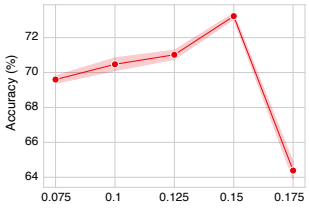
Figure 2: Filled points indicate our models trained with $\alpha = 0.4, \epsilon = 0.1$, half-filled represent our models trained with $\alpha = 0.1, \epsilon = 0.025$, and unfilled points are standard models DGI.

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 3. 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 Section 5.2, and present their results in Figure 4. 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.

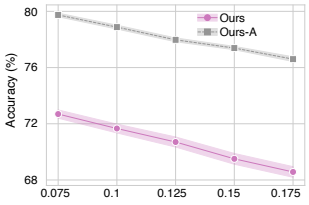


(a) Budget of graph topology α

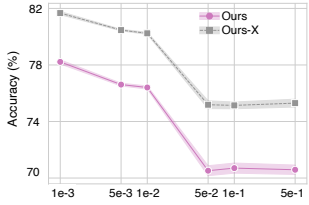


(b) Budget of node attributes ϵ

Figure 3: Hyperparameter analysis.



(a) Increasing attack rate on graph topology



(b) Increasing attack rate on node attributes

Figure 4: 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.

A.5 Proofs of Theorem 4.1

In Theorem 4.1 and 4.2, we use $\boldsymbol{a} \in \{0,1\}^{|\mathbf{V}|}$ to denote a column of the adjacency matrix and $\boldsymbol{x}$ to denote the attribute matrix, $\mathcal{F} = \{f : z \mapsto y\}$ denotes the set of non-trivial downstream classifiers, $f^* = \arg\min_{f \in \mathcal{F}} \text{AdvRisk}_{\delta,\epsilon}(f \circ e)$ is the optimal classifier built upon e, and H_b is the binary entropy function. When indexing $\boldsymbol{a}$ and $\boldsymbol{x}$, $\boldsymbol{a}_i$ denotes i-th entry of $\boldsymbol{a}$ and $\boldsymbol{x}_i$ denotes the i-th row of $\boldsymbol{x}$. When indexing A, A_i denotes i-th entry of A. The subscript δ, ϵ of RV, AdvRisk and AG represents that they are defined via $\mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon)$, *i.e.*, the separative bounds of A and X,

$$\text{RV}_{\delta,\epsilon}(e) = \text{I}\left(S; e\left(S\right)\right) - \inf_{(\mu_{A'}, \mu_{X'}) \sim \mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon)} \text{I}\left(S'; e\left(S'\right)\right),$$

$$\begin{aligned}\text{AdvRisk}_{\delta,\epsilon}(g) =& \mathbb{E}_{p(\boldsymbol{s},y)}[\exists\, \boldsymbol{s}' = (\boldsymbol{a}', \boldsymbol{x}') \in \mathcal{B}(\boldsymbol{a}, \boldsymbol{x}, \delta, \epsilon) \\ & \text{s.t. } \text{CM}(\boldsymbol{a}', \boldsymbol{x}', g, y^*) \geq 0],\end{aligned}$$

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:

$$\mathcal{E} = \{e : (\boldsymbol{a}, \boldsymbol{x}) \in \mathcal{S} \mapsto \operatorname{sgn}[\boldsymbol{a}^T \boldsymbol{x} \boldsymbol{\Theta}] |\ \|\boldsymbol{\Theta}\|_2 = 1\}. \tag{10}$$

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

$$\text{RV}_{\delta,\epsilon}(e) = 1 - H_b(0.5 + \text{AG}_{\delta,\epsilon}(f^* \circ e)). \tag{11}$$

Next, consider a simpler case in which $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1, +1\}$ and $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(0.5 + \boldsymbol{y} \cdot (p - 0.5))$ hold but $\boldsymbol{x}_i = \mathbf{1}_c,\ i = 1, \ldots, |\boldsymbol{V}|$ and the set of encoders follows such that:

$$\mathcal{E} = \{e : (\boldsymbol{a}, \boldsymbol{x}) \mapsto \operatorname{sgn}[(\boldsymbol{a}^T \boldsymbol{x} - 0.5|\boldsymbol{V}|\mathbf{1}_c^T)\boldsymbol{\Theta}] \mid \|\boldsymbol{\Theta}\|_2 = 1\},$$

which can be regarded as the non-attribute case. Then, given $\epsilon \geq 0, \delta \geq 0$, for any $e \in \mathcal{E}$, we have

$$1 - H_b(0.5 - 0.5\text{AG}_{\delta,\epsilon}(f^* \circ e)) \leq \text{RV}_{\delta,\epsilon}(e) \leq 1 - H_b(0.5 - \text{AG}_{\delta,\epsilon}(f^* \circ e)). \tag{12}$$

Proof. For simplicity, we denote $n := |\boldsymbol{V}|$. Let $R^{\boldsymbol{y}}$ represent 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 represent 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 $\text{RV}_{\delta,\epsilon}(e)$. Note that $\text{I}(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$. For any given $e \in \mathcal{E}$, we have

$$\begin{aligned}
\text{RV}_{\delta,\epsilon}(e) &= \text{I}(S; e(S)) - \inf_{(\mu_{A'}, \mu_{X'}) \sim \mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon)} \text{I}(S'; e(S')) \\
&= H_b(e(S)) - \inf_{(\mu_{A'}, \mu_{X'}) \sim \mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon)} H_b(e(S')) \\
&= H_b(1/2) - \inf_{(\mu_{A'}, \mu_{X'}) \sim \mathcal{B}_\infty(\mu_A, \mu_X, \delta, \epsilon)} H_b(e(S'))
\end{aligned}$$

where the second equality holds because $H(e(X)|X) = 0$, and the third equality holds because the distribution of $\boldsymbol{h} = \boldsymbol{a}^T \boldsymbol{x}$ (*i.e.*, 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. Thus, we 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 notice 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 $\Delta \boldsymbol{h} = \boldsymbol{h}' - \boldsymbol{h} = \boldsymbol{a}'^T \boldsymbol{x}' - \boldsymbol{a}^T \boldsymbol{x} = (\boldsymbol{a} + \Delta \boldsymbol{a})^T (\boldsymbol{x} + \Delta \boldsymbol{x}) - \boldsymbol{a}^T \boldsymbol{x} = \boldsymbol{a}^T \Delta \boldsymbol{x} + \Delta \boldsymbol{a}^T \boldsymbol{x} + \Delta \boldsymbol{a}^T \Delta \boldsymbol{x}$. According to the triangle inequality, we have,

$$\begin{aligned}
\|\Delta \boldsymbol{h}\|_p =& \|\boldsymbol{a}^T \Delta \boldsymbol{x} + \Delta \boldsymbol{a}^T \boldsymbol{x} + \Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p \\
\leq& \|\boldsymbol{a}^T \Delta \boldsymbol{x}\|_p + \|\Delta \boldsymbol{a}^T \boldsymbol{x}\|_p + \|\Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p.
\end{aligned}$$

According to $\boldsymbol{s}' \in \mathcal{B}(\boldsymbol{a}, \boldsymbol{x}, \delta, \epsilon)$, we can get $\|\Delta \boldsymbol{a}\|_0 \leq \delta$ and $\|\Delta \boldsymbol{x}\|_p \leq \epsilon$. The difficulty is that $\boldsymbol{x}$ is unbounded under Gaussian assumption $\boldsymbol{x}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}_c)$. But in practice, we can roughly consider $\boldsymbol{x}$ can be bounded. In addition, according to three-sigma rule, $\boldsymbol{x}$ is bounded as, $\|\boldsymbol{x}_i\|_p = (|\boldsymbol{x}_{i1}|^p + |\boldsymbol{x}_{i2}|^p + \cdots + |\boldsymbol{x}_{ic}|^p)^{\frac{1}{p}} \leq c^{\frac{1}{p}} 3\sigma =: \epsilon_\sigma < +\infty$.

Note that the bound $\|\Delta \boldsymbol{a}\|_0 \leq \delta$, is a constraint on the change in each column of adjacency matrix. In practice, there is another way to ensure the imperceptible perturbations on adjacency matrix, *i.e.*, $\|\Delta \boldsymbol{A}\|_0 \leq \delta^*$. In this case, we can still obtain the budget constraint $\delta = \min\{\delta^*, n\}$ for $\Delta \boldsymbol{a}$.

Then, we aim to find the upper bound of $\|\Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p$, *i.e.*, ϵ_1. We discuss two kinds of norm here. When we applying $\|\cdot\|_p$ as the operator norm, we have

$$\|\Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p = \|\Delta \boldsymbol{x}^T \Delta \boldsymbol{a}\|_p = \frac{\|\Delta \boldsymbol{x}^T \Delta \boldsymbol{a}\|_p}{\|\Delta \boldsymbol{a}\|_p} \|\Delta \boldsymbol{a}\|_p \leq \|\Delta \boldsymbol{a}\|_p \sup_{\Delta \boldsymbol{a} \neq \mathbf{0}} \frac{\|\Delta \boldsymbol{x}^T \Delta \boldsymbol{a}\|_p}{\|\Delta \boldsymbol{a}\|_p} = \|\Delta \boldsymbol{a}\|_p \|\Delta \boldsymbol{x}^T\|_{p,p}.$$

According to $\mathcal{B}(\boldsymbol{a}, \boldsymbol{x}, \delta, \epsilon)$, we have $\|\Delta \boldsymbol{x}\|_p \leq \epsilon$. In this case, we set our bound as $\|\Delta \boldsymbol{x}^T\|_{p,p} \leq \epsilon$. We further relax $\Delta \boldsymbol{a} \in (\{0, 1\}^n, \|\cdot\|_0)$ to $(\mathbb{R}^n, \|\cdot\|_p)$. Based on $\|\Delta \boldsymbol{a}\|_0 \leq \delta$, we can get $\|\Delta \boldsymbol{a}\|_p \leq \delta^{1/p}$, which indicates $\|\Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p \leq \delta^{1/p} \epsilon = \epsilon_1$.

When we applying $\|\cdot\|_p$ as the entrywise matrix norm, we have,

$$\begin{aligned}\|\Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p =& \|\sum_{i:\Delta \boldsymbol{a}_i=1} \Delta \boldsymbol{x}_i\|_p \\ =& (\sum_{j=1}^{c} |\sum_{i:\Delta \boldsymbol{a}_i=1} \Delta \boldsymbol{x}_{ij}|^p)^{1/p} \\ \leq& (\sum_{j=1}^{c} \sum_{i:\Delta \boldsymbol{a}_i=1} |\Delta \boldsymbol{x}_{ij}|^p)^{1/p} \\ \leq& (\sum_{i=1}^{n}\sum_{j=1}^{c} |\Delta \boldsymbol{x}_{ij}|^p)^{1/p} = \|\Delta \boldsymbol{x}\|_{p,p}.\end{aligned}$$

In this case, we have $\|\Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p \leq \|\Delta \boldsymbol{x}\|_{p,p} \leq \epsilon = \epsilon_1$.

Note that the bound $\|\Delta \boldsymbol{x}\|_p \leq \epsilon$, is a constraint on the change in $\boldsymbol{X}$. In practice, there is another way to ensure the imperceptible perturbations on $\boldsymbol{X}$, *i.e.*, $\|\Delta \boldsymbol{X}_i\|_p \leq \epsilon^*$. We can get the following bound under either operator norm or entrywise norm, that is,

$$\begin{aligned}\|\Delta \boldsymbol{a}^T \Delta \boldsymbol{x}\|_p =& \|\sum_{i:\Delta \boldsymbol{a}_i=1} \Delta \boldsymbol{x}_i\|_p \\ \leq& \sum_{i:\Delta \boldsymbol{a}_i=1} \|\Delta \boldsymbol{x}_i\|_p \\ \leq& \delta\epsilon^* = \epsilon_1.\end{aligned}$$

Similarly, we can achieve the upper bound of $\|\boldsymbol{a}^T \Delta \boldsymbol{x}\|_p$, *i.e.*, ϵ_2.

Finally, we can get the bound of $\|\Delta \boldsymbol{h}\|_p$ as,

$$\|\Delta \boldsymbol{h}\|_p \leq \epsilon_2 + \delta\epsilon_\sigma + \epsilon_1 =: B < +\infty.$$

Then, according to the Hölder's inequality, we have $|\Delta \boldsymbol{h}\Theta| \leq \|\Delta \boldsymbol{h}\|_p \|\Theta\|_q \leq B\|\Theta\|_q$, which indicates $P_{\boldsymbol{s}\sim\mu_S}(|\Delta \boldsymbol{h}\Theta| \leq B\|\Theta\|_q) \approx 1$. We have,

$$P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta - B\|\Theta\|_q \geq 0) \leq P_{\boldsymbol{s}'\sim\mu_{S'}}(\boldsymbol{h}'\Theta \geq 0) \leq P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta + B\|\Theta\|_q \geq 0).$$

Compute RV$_{\delta,\epsilon}$. Next, we will induce the more detailed formulations of the two bounds above. The lower bound is

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

This is because $\boldsymbol{h}\Theta \sim \mathcal{N}(0, \boldsymbol{r}\sigma^2\|\Theta\|_2)$ and we normalize $\boldsymbol{h}\Theta$ to $Z = (\boldsymbol{h}\Theta - 0)/\sqrt{\boldsymbol{r}\sigma^2\|\Theta\|_2} \sim \mathcal{N}(0,1)$.

$$\begin{aligned}① = P_{Z\sim\mathcal{N}(0,1)}[&\frac{1}{2} P_{\boldsymbol{r}^+\sim B(n,p)}(Z \geq \frac{B\|\Theta\|_q}{\sqrt{\boldsymbol{r}^+}\sigma\|\Theta\|_2}) \\ &+ \frac{1}{2} P_{\boldsymbol{r}^-\sim B(n,1-p)}(Z \geq \frac{B\|\Theta\|_q}{\sqrt{\boldsymbol{r}^-}\sigma\|\Theta\|_2})].\end{aligned}$$

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,

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

On the other hand, the upper bound is

$$
\begin{aligned}
② :=& P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta + B\|\Theta\|_q \geq 0) \\
=& P_{Z\sim\mathcal{N}(0,1)} P_{\boldsymbol{r}\sim\mu_R}(Z \geq \frac{-B\|\Theta\|_q}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}) \\
=& \frac{1}{2} P_{Z\sim\mathcal{N}(0,1)}[P_{\boldsymbol{r}^+\sim B(n,p)}(Z\sqrt{\boldsymbol{r}^+}\sigma\|\Theta\|_2 \geq -B\|\Theta\|_q) \\
&\qquad + P_{\boldsymbol{r}^-\sim B(n,1-p)}(Z\sqrt{\boldsymbol{r}^-}\sigma\|\Theta\|_2 \geq -B\|\Theta\|_q)] \\
=& \frac{1}{2} + \frac{1}{2} P_{Z\sim\mathcal{N}(0,1),Z<0}[P_{\boldsymbol{r}^+\sim B(n,p)}(Z\sqrt{\boldsymbol{r}^+}\sigma\|\Theta\|_2 \geq -B\|\Theta\|_q) \\
&\qquad + P_{\boldsymbol{r}^-\sim B(n,1-p)}(Z\sqrt{\boldsymbol{r}^-}\sigma\|\Theta\|_2 \geq -B\|\Theta\|_q)] \\
\approx& \frac{1}{2} + \frac{1}{2} P_{Z\sim\mathcal{N}(0,1),Z>0}[P_{Y\sim\mathcal{N}(0,1)}(Y \leq \frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2 - np}{\sqrt{npq}}) \\
&\qquad + P_{Y\sim\mathcal{N}(0,1)}(Y \leq \frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2 - nq}{\sqrt{npq}})] \\
=:& \frac{1}{2} + \frac{1}{2} P_2 \quad (0 \leq P_2 \leq 1).
\end{aligned}
$$

Therefore, $\mathrm{RV}_{\delta,\epsilon}(e) = H_b(1/2) - H_b(\max\{|P_1/2 - 1/2|, |P_2/2|\} + 1/2)$.

Compute $\mathrm{AG}_{\delta,\epsilon}$. Given the formulation of $\mathrm{RV}_{\delta,\epsilon}$, we further aim to establish its connection to $\mathrm{AG}_{\delta,\epsilon}$. Here we induce the detailed formulation of $\mathrm{AG}_{\delta,\epsilon}$. In our case, the only two classifiers to be discussed are $f_1(z) = z$ and $f_2(z) = -z$.

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

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

Because $|\Delta\boldsymbol{h}\Theta| \leq B\|\Theta\|_q$, $-\min_{\Delta\boldsymbol{s}\in\mathcal{B}(0,0,\delta,\epsilon)} \boldsymbol{y}\cdot\Delta\boldsymbol{h}\Theta = B\|\Theta\|_q$ holds for any $\boldsymbol{y}$.

$$
\begin{aligned}
③ =& \frac{1}{2} P_{\boldsymbol{h}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{r}^+\sigma^2\boldsymbol{I}),\boldsymbol{r}^+\sim B(n,p)}(\boldsymbol{h}\Theta \leq B\|\Theta\|_q) \\
&+ \frac{1}{2} P_{\boldsymbol{h}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{r}^-\sigma^2\boldsymbol{I}),\boldsymbol{r}^+\sim B(n,q)}(\boldsymbol{h}\Theta \geq -B\|\Theta\|_q) \\
=& \frac{1}{2} P_{Z\sim\mathcal{N}(0,1)}[P_{\boldsymbol{r}^+\sim B(n,p)}(Z \leq \frac{B\|\Theta\|_q}{\sqrt{\boldsymbol{r}^+}\sigma\|\Theta\|_2}) \\
&\qquad + P_{\boldsymbol{r}^-\sim B(n,q)}(Z \geq \frac{-B\|\Theta\|_q}{\sqrt{\boldsymbol{r}^-}\sigma\|\Theta\|_2})] \\
\approx& \frac{1}{2}\times\frac{1}{2} + \frac{1}{2} P_{Z\sim\mathcal{N}(0,1),Z>0}[P_{Y\sim\mathcal{N}(0,1)}(Y \leq \frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2 - np}{\sqrt{npq}})] \\
&+ \frac{1}{2}\times\frac{1}{2} + \frac{1}{2} P_{Z\sim\mathcal{N}(0,1),Z<0}[P_{Y\sim\mathcal{N}(0,1)}(Y \leq \frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2 - nq}{\sqrt{npq}})] \\
=& \frac{1}{2} + \frac{1}{2} P_{Z\sim\mathcal{N}(0,1),Z>0}[P_{Y\sim\mathcal{N}(0,1)}(Y \leq \frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2 - np}{\sqrt{npq}}) \\
&\qquad + P_{Y\sim\mathcal{N}(0,1)}(Y \leq \frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2 - nq}{\sqrt{npq}})] \\
=& \frac{1}{2} + \frac{1}{2} P_2.
\end{aligned}
$$

We also have $\text{AdvRisk}_{\delta=0,\epsilon=0}(f_1 \circ e)$ as

$$\begin{aligned}
④ :=&\text{AdvRisk}_{\delta=0,\epsilon=0}(f_1 \circ e)\\
=&P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\boldsymbol{y}\cdot\boldsymbol{h}\Theta \le 0]\\
=&\frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{r}^+\sigma^2\boldsymbol{I}),\boldsymbol{r}^+\sim B(n,p)}(\boldsymbol{h}\Theta \le 0)\\
+&\frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{r}^-\sigma^2\boldsymbol{I}),\boldsymbol{r}^+\sim B(n,q)}(\boldsymbol{h}\Theta \ge 0)\\
=&\frac{1}{2}P_{Z\sim\mathcal{N}(0,1)}[P_{\boldsymbol{r}^+\sim B(n,p)}(Z\le 0)+P_{\boldsymbol{r}^-\sim B(n,q)}(Z\ge 0)]\\
=&\frac{1}{2}.
\end{aligned}$$

Thus, $\text{AG}_{\delta,\epsilon}(f_1 \circ e) = ③ - ④ = P_2/2$.

Similarly, for given $e \in \mathcal{E}$, we have $\text{AdvRisk}_{\delta,\epsilon}(f_2 \circ e)$ as

$$\begin{aligned}
⑤ :=&\text{AdvRisk}_{\delta,\epsilon}(f_2 \circ e)\\
=&P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\max_{\boldsymbol{s}'\in\mathcal{B}(\boldsymbol{a},\boldsymbol{x},\delta,\epsilon)}\boldsymbol{y}\cdot\boldsymbol{h}'\Theta \ge 0]\\
=&P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\boldsymbol{y}\cdot\boldsymbol{h}\Theta \ge -\max_{\Delta\boldsymbol{s}\in\mathcal{B}(0,0,\delta,\epsilon)}\boldsymbol{y}\cdot\Delta\boldsymbol{h}\Theta]\\
=&\frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{r}^+\sigma^2\boldsymbol{I}),\boldsymbol{r}^+\sim B(n,p)}(\boldsymbol{h}\Theta \ge -B\|\Theta\|_q)\\
+&\frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{r}^-\sigma^2\boldsymbol{I}),\boldsymbol{r}^+\sim B(n,q)}(\boldsymbol{h}\Theta \le B\|\Theta\|_q)\\
\approx&\frac{1}{2}\times\frac{1}{2}+\frac{1}{2}P_{Z\sim\mathcal{N}(0,1),Z<0}[P_{Y\sim\mathcal{N}(0,1)}(Y\le\frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2-np}{\sqrt{npq}})]\\
+&\frac{1}{2}\times\frac{1}{2}+\frac{1}{2}P_{Z\sim\mathcal{N}(0,1),Z>0}[P_{Y\sim\mathcal{N}(0,1)}(Y\le\frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2-nq}{\sqrt{npq}})]\\
=&\frac{1}{2}+\frac{1}{2}P_2
\end{aligned}$$

We also have $\text{AdvRisk}_{\delta=0,\epsilon=0}(f_2 \circ e)$ as

$$⑥ := \text{AdvRisk}_{\delta=0,\epsilon=0}(f_2 \circ e) = P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\boldsymbol{y}\cdot\boldsymbol{h}\Theta \ge 0] = \frac{1}{2}.$$

We can get $\text{AG}_{\delta,\epsilon}(f_2 \circ e) = ⑤ - ⑥ = P_2/2$.

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

Connection between RV and AG. Now we aim to find the connection between $\text{AG}_{\delta,\epsilon}$ and $\text{RV}_{\delta,\epsilon}$. Given their formulations

$$\begin{cases}\text{RV}_{\delta,\epsilon}(e) = H_b(1/2) - H_b(\max\{|P_1/2-1/2|,|P_2/2|\}+1/2)\\ \text{AG}_{\delta,\epsilon}(f_1\circ e) = \text{AG}_{\delta,\epsilon}(f_2\circ e) = P_2/2\end{cases}$$

and the probability P_1 and P_2

$$\begin{cases}
P_1 = P_{Z\sim\mathcal{N}(0,1),Z>0}[P_{Y\sim\mathcal{N}(0,1)}(Y\ge\frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2-np}{\sqrt{npq}})\\
\qquad\qquad + P_{Y\sim\mathcal{N}(0,1)}(Y\ge\frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2-nq}{\sqrt{npq}})]\\
P_2 = P_{Z\sim\mathcal{N}(0,1),Z>0}[P_{Y\sim\mathcal{N}(0,1)}(Y\le\frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2-np}{\sqrt{npq}})\\
\qquad\qquad + P_{Y\sim\mathcal{N}(0,1)}(Y\le\frac{B^2\|\Theta\|_q^2/Z^2\sigma^2\|\Theta\|_2^2-nq}{\sqrt{npq}})],
\end{cases}$$

We can 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,

$$\begin{aligned}
\text{RV}_{\delta,\epsilon}(e) =&H_b(1/2) - H_b(P_2/2+1/2)\\
=&H_b(1/2) - H_b(1/2+\text{AG}_{\delta,\epsilon}(f^*\circ e)).
\end{aligned}$$

which completes the proof.

Simpler case. Consider a simpler case that $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1,+1\}$ and $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(0.5+\boldsymbol{y}\cdot(p-0.5))$ hold but $\boldsymbol{x}_i = \mathbf{1}_c,\ i = 1,2,\dots n$, and the set of encoders follows that

$$\mathcal{E} = \{e : (\boldsymbol{a},\boldsymbol{x}) \mapsto \text{sgn}[(\boldsymbol{a}^T\boldsymbol{x} - 0.5n\mathbf{1}_c^T)\boldsymbol{\Theta}] \mid \|\boldsymbol{\Theta}\|_2 = 1\}.$$

This simpler case removes the randomness in $\boldsymbol{x}$. We let $H^{\boldsymbol{y}} = A^TX - 0.5n\mathbf{1}_c^T \sim (R^{\boldsymbol{y}} - 0.5n)^c$, then its realization $\boldsymbol{h} = \boldsymbol{a}^T\boldsymbol{x} - 0.5n\mathbf{1}_c^T$. We again use the De Moivre-Laplace Central Limit Theorem to approximate $H^{\boldsymbol{y}}$,

$$\begin{aligned}\boldsymbol{h}^+ =& \boldsymbol{r}^+ - 0.5n \to \mathcal{N}((np-\frac{n}{2})^c, \text{diag}_c(npq)),\\ \boldsymbol{h}^- =& \boldsymbol{r}^- - 0.5n \to \mathcal{N}((nq-\frac{n}{2})^c, \text{diag}_c(npq)).\end{aligned}$$

We notice that $np - n/2 + nq - n/2 = 0$ and $H^{\boldsymbol{y}}$ satisfies the conditions of Gaussian Mixture model in (Zhu, Zhang, and Evans 2020). Thus, we can directly reuse (Zhu, Zhang, and Evans 2020, Theorem 3.4) by replacing $\boldsymbol{\theta}^*$ with $np - n/2$ and $\boldsymbol{\Sigma}^*$ with $\text{diag}_c(npq)$, which completes the proof.

Note that Theorem 4.1 also holds if we add self-loops into A, *i.e.*, $\boldsymbol{a}$ has only $n-1$ random entries.

■

D.6 Proof of Theorem 4.2

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 the set of encoders is (10). Assume the samples $(\boldsymbol{s},\boldsymbol{y}) \sim \mu_{SY}$ are generated from $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1,+1\}$, $\boldsymbol{a}_i \overset{i.i.d.}{\sim} \text{Bernoulli}(0.5)$ and $\boldsymbol{x}_i \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{y}\cdot\boldsymbol{\mu}, \sigma^2 I_c)$ where $i = 1,2,\dots,|\boldsymbol{V}|$. Then, given $\epsilon \geq 0, \delta \geq 0$, for any $e \in \mathcal{E}$, we have:

$$\text{RV}_{\delta,\epsilon}(e) = 1 - H_b(0.5 - \text{AG}_{\delta,\epsilon}(f^* \circ e)). \tag{13}$$

Next, consider a simpler case in which $\boldsymbol{y} \overset{\text{u.a.r.}}{\sim} U\{-1,+1\}$, $\boldsymbol{x}_i \overset{i.i.d.}{\sim} \mathcal{N}(\boldsymbol{y}\cdot\boldsymbol{\mu}, \sigma^2 I_c)$ 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 $\boldsymbol{y} = -1$. Suppose the set of encoders is (10). Then, given $\epsilon \geq 0, \delta \geq 0$, (12) also holds for any $e \in \mathcal{E}$.

Proof. For simplicity, we denote $n := |\boldsymbol{V}|$. 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 represent the number of samples with $\boldsymbol{y} = +1$ connected to a node while $\boldsymbol{r} - p$ represent 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.*, $H = A^TX \sim \mathcal{N}(C_p(R)\boldsymbol{\mu}, R\sigma^2\boldsymbol{I})$), then its realization $\boldsymbol{h} = \boldsymbol{a}^T\boldsymbol{x}$ is exactly the aggregation operator of GNNs.

We have,

$$\begin{aligned}P_{\boldsymbol{s}\sim\mu_S}(l(\boldsymbol{h}\Theta) \geq 0) =& P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}} P_{\boldsymbol{h}\sim\mathcal{N}(C_p(\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta) \geq 0)]\\ =& \frac{1}{2}P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}} P_{\boldsymbol{h}\sim\mathcal{N}(C_p(\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta) \geq 0)\\ &+ \sum_{r-p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}} P_{\boldsymbol{h}\sim\mathcal{N}(C_p(\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta) \geq 0)]\\ =& \frac{1}{2}P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p=\max\{0,\boldsymbol{r}-n/2\}}^{\min\{\boldsymbol{r},n/2\}} (P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta) \geq 0)\\ &+ P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p)\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta) \geq 0))].\end{aligned}$$

where $l(\boldsymbol{h}\Theta)$ represent the function of $\boldsymbol{h}\Theta$. 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 $(P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta) \geq 0), P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p)\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta) \geq 0))$ 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, Zhang, and Evans 2020). So we can directly reuse the formulations.

Note that self-loops in attribute-aware case are important, so we highlight the slight difference with stochastic self-loops case (*under the assumption of* $n-1$ *random entries of* $\boldsymbol{a}$). In this case, we let R represent the random variable following the Binomial distribution according to A (*i.e.*, $R=\sum_{i=1}^{n}A_i\sim B(n-1,0.5)$) and its realization $\boldsymbol{r}=\sum_{i=1}^{n}\boldsymbol{a}_i$. We define $C_p^{\boldsymbol{y}}(\boldsymbol{r})=p-(\boldsymbol{r}-p)+\boldsymbol{y}=2p-\boldsymbol{r}+\boldsymbol{y}$. We then have,

$$\begin{aligned}
P_{\boldsymbol{s}\sim\mu_S}(l(\boldsymbol{h}\Theta)\geq 0) =&\frac{1}{2}P_{\boldsymbol{r}\sim B(n-1,0.5)}[\sum_{p=\max\{0,\boldsymbol{r}-n/2+1\}}^{\min\{\boldsymbol{r},n/2-1\}}P_{\boldsymbol{h}\sim\mathcal{N}(C_p^{+}(\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta)\geq 0)\\
&+\sum_{r-p=\max\{0,\boldsymbol{r}-n/2+1\}}^{\min\{\boldsymbol{r},n/2-1\}}P_{\boldsymbol{h}\sim\mathcal{N}(C_p^{-}(\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta)\geq 0)]\\
=&\frac{1}{2}P_{\boldsymbol{r}\sim B(n-1,0.5)}[\sum_{p=\max\{0,\boldsymbol{r}-n/2+1\}}^{\min\{\boldsymbol{r},n/2-1\}}(P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r}+1)\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta)\geq 0)\\
&+P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p-1)\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(l(\boldsymbol{h}\Theta)\geq 0))].
\end{aligned}$$

We can see that it is easy to extend the formulations to case that self-loops is considered.

In the following derivations, we just list some key components and the other components are similar to the proof of Theorem 4.1.

First, we compute $P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta\geq 0)$**.**

$$\begin{aligned}
P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta\geq 0) =&\frac{1}{2}P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p}(P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(\boldsymbol{h}\Theta\geq 0)\\
&+P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p)\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(\boldsymbol{h}\Theta\geq 0))]\\
=&\frac{1}{2}P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p}(P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p)\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(\boldsymbol{h}\Theta< 0)\\
&+P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(\boldsymbol{h}\Theta< 0))]\\
=&P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta< 0)
\end{aligned}$$

Thus, $H_b(e(S))=H_b(1/2)$.

Second, we compute the bound of $\|\Delta\boldsymbol{h}\|_p$**.** According to three-sigma rule, $\boldsymbol{x}_i\overset{i.i.d.}{\sim}\mathcal{N}(\boldsymbol{y}\cdot\boldsymbol{\mu},\sigma^2I_c)$ is bounded as, $\|\boldsymbol{x}_i\|_p=(|\boldsymbol{x}_{i1}|^p+|\boldsymbol{x}_{i2}|^p+\cdots+|\boldsymbol{x}_{ic}|^p)^{\frac{1}{p}}\leq c^{\frac{1}{p}}(\max_{j=1}^{c}|\boldsymbol{\mu}_j|+3\sigma)=:\epsilon_\sigma<+\infty$. And then we can still get a bound of $\|\Delta\boldsymbol{h}\|_p$ similar with Theorem 4.1.

Third, we compute $\mathbf{RV}_{\delta,\epsilon}$**.** We first compute the lower bound and upper bound of $\|\Delta\boldsymbol{h}\|_p$ as,

$$\begin{aligned}
P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta-B\|\Theta\|_q\geq 0) =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p}(\frac{1}{2}-\frac{1}{2}P_{Z\sim\mathcal{N}(0,1)}(\frac{-B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}\leq Z\leq\frac{B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}))]\\
=&\frac{1}{2}-\frac{1}{2}P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p}P_{Z\sim\mathcal{N}(0,1)}(\frac{-B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}\leq Z\leq\frac{B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})],
\end{aligned}$$

$$P_{\boldsymbol{s}\sim\mu_S}(\boldsymbol{h}\Theta+B\|\Theta\|_q\geq 0) =\frac{1}{2}+\frac{1}{2}P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p}P_{Z\sim\mathcal{N}(0,1)}(\frac{-B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}\leq Z\leq\frac{B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})].$$

Because $H_b(\theta)$ is symmetric w.r.t 0.5, we can achieve the formulation of $\mathrm{RV}_{\delta,\epsilon}(e)$ as,

$$\begin{aligned}
\mathrm{RV}_{\delta,\epsilon}(e) =&H_b(\frac{1}{2})-H_b(\frac{1}{2}-\frac{1}{2}P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{p}P_{Z\sim\mathcal{N}(0,1)}(\frac{-B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}\\
&\leq Z\leq\frac{B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})]).
\end{aligned}$$

Fourth, we compute $\text{AG}_{\delta,\epsilon}$. In order to compute $\text{AG}_{\delta,\epsilon}$, we first compute its two components as below:

$$\begin{aligned}\text{AdvRisk}_{\delta,\epsilon}(f_1\circ e) =&P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\boldsymbol{y}\cdot\boldsymbol{h}\Theta\le-\min_{\Delta\boldsymbol{s}\in\mathcal{B}(0,0,\delta,\epsilon)}\boldsymbol{y}\cdot\Delta\boldsymbol{h}\Theta]\\ =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p(\frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}((2p-\boldsymbol{r})\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(\boldsymbol{h}\Theta\le B\|\Theta\|_q)\\ &+\frac{1}{2}P_{\boldsymbol{h}\sim\mathcal{N}((\boldsymbol{r}-2p)\boldsymbol{\mu},\boldsymbol{r}\sigma^2\boldsymbol{I})}(\boldsymbol{h}\Theta\ge-B\|\Theta\|_q))]\\ =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p P_{Z\sim\mathcal{N}(0,1)}(Z\le\frac{B\|\Theta\|_q-(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})],\end{aligned}$$

$$\begin{aligned}\text{AdvRisk}_{\delta=0,\epsilon=0}(f_1\circ e) =&P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}(\boldsymbol{y}\cdot\boldsymbol{h}\Theta\le0)\\ =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p P_{Z\sim\mathcal{N}(0,1)}(Z\le\frac{-(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})].\end{aligned}$$

Thus we have the formulation of $\text{AG}_{\delta,\epsilon}(f_1\circ e)$ as,

$$\begin{aligned}\text{AG}_{\delta,\epsilon}(f_1\circ e) =&\text{AdvRisk}_{\delta,\epsilon}(f_1\circ e)-\text{AdvRisk}_{\delta=0,\epsilon=0}(f_1\circ e)\\ =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p P_{Z\sim\mathcal{N}(0,1)}(\frac{-B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}\le Z\le\frac{(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})].\end{aligned}$$

Similarly, we can derive the formulation of $\text{AG}_{\delta,\epsilon}(f_2\circ e)$ as,

$$\begin{aligned}\text{AdvRisk}_{\delta,\epsilon}(f_2\circ e) =&P_{(\boldsymbol{s},\boldsymbol{y})\sim\mu_{SY}}[\boldsymbol{y}\cdot\boldsymbol{h}\Theta\ge-\max_{\Delta\boldsymbol{s}\in\mathcal{B}(0,0,\delta,\epsilon)}\boldsymbol{y}\cdot\Delta\boldsymbol{h}\Theta]\\ =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p P_{Z\sim\mathcal{N}(0,1)}(Z\le\frac{B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})],\end{aligned}$$

$$\text{AdvRisk}_{\delta=0,\epsilon=0}(f_2\circ e) =P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p P_{Z\sim\mathcal{N}(0,1)}(Z\le\frac{(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})],$$

$$\begin{aligned}\text{AG}_{\delta,\epsilon}(f_2\circ e) =&\text{AdvRisk}_{\delta,\epsilon}(f_2\circ e)-\text{AdvRisk}_{\delta=0,\epsilon=0}(f_2\circ e)\\ =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p P_{Z\sim\mathcal{N}(0,1)}(\frac{(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2}\le Z\le\frac{B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})].\end{aligned}$$

Note that $\text{AG}_{\delta,\epsilon}(f_2\circ e)=\text{AG}_{\delta,\epsilon}(f_1\circ e)$, because

$$\begin{aligned}\text{AdvRisk}_{\delta,\epsilon}(f_1\circ e) =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_p P_{Z\sim\mathcal{N}(0,1)}(Z\le\frac{B\|\Theta\|_q-(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})]\\ =&P_{\boldsymbol{r}\sim B(n,0.5)}[\sum_{r-p} P_{Z\sim\mathcal{N}(0,1)}(Z\le\frac{B\|\Theta\|_q+(\boldsymbol{r}-2p)\Theta^T\boldsymbol{\mu}}{\sqrt{\boldsymbol{r}}\sigma\|\Theta\|_2})]\\ =&\text{AdvRisk}_{\delta,\epsilon}(f_2\circ e).\end{aligned}$$

The third equality holds because the range of p and $\boldsymbol{r}-p$ is the same and given $\boldsymbol{r}$, each value p in the range has the same probability. It's also easy to prove that $\text{AdvRisk}_{\delta=0,\epsilon=0}(f_2\circ e)=\text{AdvRisk}_{\delta=0,\epsilon=0}(f_1\circ e)$ and $\text{AG}_{\delta,\epsilon}(f_2\circ e)=\text{AG}_{\delta,\epsilon}(f_1\circ e)$. we thus have,

$$\text{RV}_{\delta,\epsilon}(e)=H_b(1/2)-H_b(1/2-\text{AG}_{\delta,\epsilon}(f^*\circ e)).$$

which completes the proof.

Simpler case. Consider a simpler case that $\boldsymbol{y}\overset{\text{u.a.r.}}{\sim}U\{-1,+1\}$, $\boldsymbol{x}_i\overset{i.i.d.}{\sim}\mathcal{N}(\boldsymbol{y}\cdot\boldsymbol{\mu},\sigma^2I_c)$ but $\boldsymbol{a}\in\{0,1\}^n$, $\sum_{i=1}^n\boldsymbol{a}_i=n_0+n_1$, where $n_0=n/4+\boldsymbol{y}\cdot(p-n/4)$, $n_1=n/4+\boldsymbol{y}\cdot(q-n/4)$ and $p+q=n/2,\ 0\le p,q\le n/2,\ p,q\in\mathbb{Z}$, that is, $\boldsymbol{a}^T\boldsymbol{x}$ will aggregate n_0 samples with $\boldsymbol{y}=+1$ and n_1 samples with $\boldsymbol{y}=-1$. Suppose the set of encoders is (10). The simpler case removes the randomness of $\boldsymbol{a}$. We let $H=A^TX\sim\mathcal{N}(C_p(R)\boldsymbol{\mu},R\sigma^2\boldsymbol{I})$, then its realization $\boldsymbol{h}=\boldsymbol{a}^T\boldsymbol{x}$. In this simpler case, $\boldsymbol{r}$ depends

on the label $\boldsymbol{y}$, *i.e.*, $C_p(\boldsymbol{r}^+) = p - q$ and $C_p(\boldsymbol{r}^-) = q - p$. Thus , we get the similar Gaussian Mixture distribution with Theorem 4.1 as

$$\boldsymbol{h}^+ \sim \mathcal{N}((p-q)\boldsymbol{\mu}, \frac{n}{2}\sigma^2\boldsymbol{I}),$$
$$\boldsymbol{h}^- \sim \mathcal{N}((q-p)\boldsymbol{\mu}, \frac{n}{2}\sigma^2\boldsymbol{I}).$$

We can directly reuse (Zhu, Zhang, and Evans 2020, Theorem 3.4) by replacing $\boldsymbol{\theta}^*$ with $p - q$ and $\boldsymbol{\Sigma}^*$ with $(n/2)\sigma^2\boldsymbol{I}$. Intuitively, the conditions $n_0 = n/4 + \boldsymbol{y} \cdot (p - n/4)$, $n_1 = n/4 + \boldsymbol{y} \cdot (q - n/4)$ and $p + q = n/2$ implies that if a positive node aggregates p positive samples and q negative ones, then a negative node have to aggregate $n/2 - p$ positive samples and $n/2 - q$ negative ones, and each node has to be connected to $n/2$ samples.

■