Local-Global Defense against Unsupervised Adversarial Attacks on Graphs

Di Jin[1], Bingdao Feng[1], Siqi Guo[1], Xiaobao Wang[1]*, Jianguo Wei[1], Zhen Wang[2]*

[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]School of Cybersecurity, Northwestern Polytechnical University, Xi'an, Shaanxi, China
{jindi, fengbingdao, guosiqi, wangxiaobao, jianguo}@tju.edu.cn, w-zhen@nwpu.edu.cn

Abstract

Unsupervised pre-training algorithms for graph representation learning are vulnerable to adversarial attacks, such as first-order perturbations on graphs, which will have an impact on particular downstream applications. Designing an effective representation learning strategy against white-box attacks remains a crucial open topic. Prior research attempts to improve representation robustness by maximizing mutual information between the representation and the perturbed graph, which is sub-optimal because it does not adapt its defense techniques to the severity of the attack. To address this issue, we propose an unsupervised defense method that combines local and global defense to improve the robustness of representation. Note that we put forward the **P**erturbed **E**dges **H**armfulness (**PEH**) metric to determine the riskiness of the attack. Thus, when the edges are attacked, the model can automatically identify the risk of attack. We present a method of attention-based protection against high-risk attacks that penalizes attention coefficients of perturbed edges to encoders. Extensive experiments demonstrate that our strategies can enhance the robustness of representation against various adversarial attacks on three benchmark graphs.

Introduction

Graphs are commonly used to simulate real-world relationships (Wu et al. 2022), such as social networks (Zhang et al. 2020), biological interaction graphs (Vlaic et al. 2018) and e-commerce networks (Eswaran et al. 2017). In recent years, graph neural networks (GNNs) (Welling and Kipf 2016) based on graph-structured data have gained a lot of attention due to their outstanding performance in many applications, such as node classification (Jin et al. 2021a; Yu et al. 2021), link prediction (Kipf and Welling 2016), and graph clustering (Bo et al. 2020; Jin et al. 2021b).

Due to the high cost of labels and complexity of graph neural network training, many studies (Velickovic et al. 2019; You et al. 2020; Peng et al. 2020; Qiu et al. 2020) have moved towards establishing pretraining graph models on unlabeled data and feeding the learned representations to off-the-shelf machine learning models for applicable downstream tasks. Although pre-trained models on graphs have shown encouraging outcomes, (Xu et al. 2022) indicates that these models based on GNN are also more vulnerable to adversarial attacks on graphs (Zügner, Akbarnejad, and Günnemann 2018; Zügner and Günnemann 2019; Xu et al. 2019), which affect the representation ability of the entire graph and then transmit the incorrect representation to all downstream tasks. Even subtle perturbations have a considerable impact on the learned graph representation, thereby degrading the performance of downstream tasks such as severe rumor detection (Sun et al. 2022) and financial supervision (Paranjape, Benson, and Leskovec 2017). For example, hackers invading a bank system and making subtle modifications to the clean data cause the system to provide the same credit limit to two unconnected clients in accordance with the attacker's instructions.

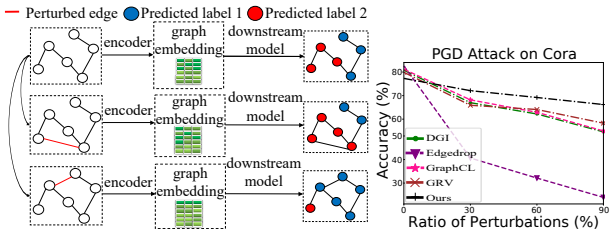


Figure 1: The overview of (a) a graph attacked under different peturbed edges and (b) accuracy of different unsurpervised models under various perturbation rates.

Figure 1(a) provides an overview of several attacks on the graph pre-training process. The main purpose of these attacks is to perturb a clean graph to alter its representation and jeopardize applications that are used afterward. However, we find that numerous studies (Xu et al. 2019; Zügner, Akbarnejad, and Günnemann 2018; Zügner and Günnemann 2019) have demonstrated that perturbing various edges can result in different degrees of damage to representations. The degree of damage produced by perturbing an edge is primarily determined by the edge's sensitivity, and a small number of perturbations on edges with high sensitivity can significantly reduce the representation capability of the model. The existing robust GNN pre-training models (Xu et al. 2022; Zhu et al. 2022; Li et al. 2022a; Yang, Zhang, and Yang 2021)

*Corresponding authors

disregard the requirement for sensitive edge protection. Figure 1(b) shows the trend of decreasing classification accuracy of traditional global defense-based robust representation learning methods as the proportion of perturbed edges increases. It is evident that the decreasing trend in classification accuracy of conventional approaches is turbulent and unable to protect critical edges properly. During the pre-training process of the graph neural network, we anticipate locating sensitive edges and injecting vaccinations, as well as performing global robust representation learning, thus ensuring that the classification accuracy declines steadily and gradually.

Consequently, there are two issues that need to be resolved: 1) How should sensitive edges be identified? 2) How can sensitive edges be protected from harmful attacks?

In this paper, we first present the **P**erturbed **E**dges **H**armfulness (**PEH**) to distinguish whether the attacked edges are sensitive; then, an information theory-based measure is then used to quantify whether the attacked edges are harmful. Next, we design an algorithm to improve the robustness of the representation from local and global perspectives, using a combination of local sensitive edge defense and global defense methods. In local defense, we propose a penalty attention mechanism to mitigate the detrimental effects of perturbations on sensitive edges without sacrificing the representations of other nodes. In addition, we concentrated on adding edges rather than removing them, as adding edges is more effective than removing them and requires more protection (Wu et al. 2019; Li et al. 2022a). Adding an edge between two distant nodes will have a significant effect on the graph's structure, whereas the nodes of the deleted edge may be connected via higher-order neighbors. Furthermore, locating sensitive edges is an enormous challenge that is directly tied to their position. To acquire perturbed data under the most perilous attack, we employ greedy and gradient descent-based topological attacks and project the most influential perturbation to the constraint set using convex relaxation on the boolean variables (Xu et al. 2019). Ultimately, we propose a whole optimization problem to investigate the trade-off between global and local defense. Overall, our main contributions are:

- We present a new robust unsupervised pre-training approach that combines global and local defense for improving robustness.
- We propose a novel vaccination method for protecting sensitive edges. Through the suggested penalized aggregation mechanism, harmful effects of perturbed sensitive edges can be mitigated.
- We conduct several experiments on real-world datasets to demonstrate the robustness of our approach against a variety of adversarial attacks.

Preliminaries

In this section, we first introduce the notations used in this paper, then briefly describe the preliminaries of our method.

Graph Representation Learning

For unsupervised graph representation learning, usually $G{=}(V,E)$ can represent a graph, where $V = \{v_1, v_2 \dots v_n\}$ denotes the set of nodes, $E \in V \times V$ denotes the set of edges, and we also use an adjacency matrix $\mathrm{A} \in \{0,1\}^{|V|\times|V|}$ to represent the set of edges E, which is a symmetric matrix with elements $\mathrm{A}_{ij} = 1$ if $(v_i, v_j) \in E$, $\mathrm{A}_{ij} = 0$ otherwise. $\mathrm{X} \in \mathbb{R}^{|V|\times d}$ denotes the feature matrix. In the following discussion, we use $G = (\mathrm{A}, \mathrm{X})$ to denote the graph.

The objective is to learn an encoder e: $\mathbb{R}^{|V|\times|V|} \times \mathbb{R}^{|V|\times d} \rightarrow \mathbb{R}^{|V|\times d'}$, which maps input nodes to a low-dimensional representation $\mathbf{z}$ without label information, where $\mathbf{z}$ can be used in downstream tasks such as node classification and graph clustering.

Mutual Information

Mutual information $I(X;Y)$ is an entropy-based measure of the mutual dependence between variables *X* and *Y*, and can be interpreted as the degree of uncertainty reduction of another random variable *Y* once the value of variable *X* is known. It is related to conditional entropy, defined as:

$$I(X;Y) = H(X) - H(X|Y), \tag{1}$$

where $H(X)$ denotes the entropy of variable X, and $H(X|Y)$ denotes the entropy of the conditional probability of *X* given *Y*. Currently, many researchers use mutual information in the application of graph data, such as recommendation (Yuhao et al. 2022), sociology (Coutrot et al. 2022), and bioinformatics (Li et al. 2022b). DGI (Velickovic et al. 2019) is built upon the InfoMax principle (Hjelm et al. 2018), which prescribes to learn an encoder e that maximizes the mutual information between the graph and its representation., i.e., $I(G; e(G))$.

Projected Gradient Descent Attack

The Projected Gradient Descent (PGD) (Madry et al. 2018) attack, one of the most effective first-order adversarial methods, is a greedy attack method onto the l_∞-bound at the end of each iteration. The PGD adversarial example can be written as:

$$x^{t+1} = \prod_{\|x+s\|_\infty < \epsilon} (x^t + \alpha \mathrm{sgn}(\nabla_{\mathrm{x}} L(\theta, x))) \tag{2}$$

where $\mathrm{sgn}(\cdot)$ is a signum function, α is the attack step, which is similar to learning rate, *t* denotes the iteration index of PGD, and $\prod_{\|x+s\|}$ is the projection operator over the constraint set $x + s$ on the ϵ-ball in the l_∞-norm.

Methods

This section begins with a discussion of min-max adversarial attacks. Then, we present a mechanism for determining the harmfulness of adversarial attacks, followed by the introduction of two distinct defense strategies, namely local and global defense. Finally, we present the technical details and optimization issues associated with our methods.

Min-Max Adversarial Attack

Here, we introduce how to attack the graph during the graph pre-training. By perturbing the structures of the original graph G, it becomes a new graph $G^* = (\mathrm{A}^*, \mathrm{X})$, where the adversarial attack can be defined as a bilevel optimization problem:

$$\begin{aligned} &\max_{G^* \in P_\Delta^G} L(G^*, f_{\theta^*}(G^*)) \\ &\text{s.t.} \theta^* = \arg\min_\theta L(G, f_\theta(G)) \end{aligned}, \tag{3}$$

where P_Δ^G is the space of the perturbation matrix Δ on the input graph G. $L(\cdot)$ is a contrastive loss that is the negative mutual information between the local representations and the global graph summary, and $f_\theta(\cdot)$ is a surrogate model. The attacker obtains the parameters θ^* by minimizing the training loss $L(\cdot)$ of the target model. How to quickly get the polluted graph remains an urgent problem, so we choose the projected gradient descent (PGD) method (Madry et al. 2018), which is proved to be the best first-order attack. However, PGD is not appropriate for boolean-type graph structures. Inspired by (Xu et al. 2019), we perform convex relaxation on the Boolean variables and then use the PGD method on the basis of the continuation assumption. If the attack causes high-risk harm to the model's representation, conventional learning approaches cannot obtain robust representations effectively. In the subsequent section, we introduce the overall framework of our robust representation learning for the adversarial attack.

Quantifying Harmfulness of Adversarial Attack

We propose the **PEH** to quantify the harmfulness of adversarial attacks. Intuitively, attacked edges could be considered sensitive if learned representations degrade significantly following adversarial attacks on contrastive learning. We measure the difference in representation quality before and after adversarial attacks using the variation in mutual information, which also reflects the side effects of the attack.

We provide a comprehensive overview of identifying the harmfulness of an attack by utilizing mutual information. If the mutual information between the attacked graph and the encoded representation reduces dramatically after the adversarial attack, this implies that the attack has severely degraded the representation capacity. In other words, other nodes would be required to learn the perturbed graph, hence limiting the expressive power of the encoder. Therefore, the attacked edges deemed sensitive should be vaccinated against adversarial attacks to improve the representation capability. Furthermore, we ought not specially to protect the attacked edge if the value of PEH(θ) is not readily high. PEH(θ) can be defined as:

$$\mathrm{PEH}(\theta) = I\left(G; f_\theta\left(G\right)\right) - I\left(G^*; f_\theta\left(G^*\right)\right), \tag{4}$$

where $I\left(G^*; f_\theta\left(G^*\right)\right)$ refers to the mutual information between the perturbed graph and its representation. Therefore, by definition, PEH(θ) describes the divergence between the encoder's capacity to represent clean and perturbed graphs. The higher the value of PEH(θ), the current encoder is less expressible in the perturbed graph, which also means that this attack is dangerous to the current encoder. Formally, perturbed edges are sufficiently hazardous when $\mathrm{PEH}(\theta) > h$.

Though some works (Xu et al. 2022) propose graph representation vulnerability (GRV) to describe the robustness of a representation and quantify the vulnerability of graph encoders, they have failed to notice the degree of vulnerability. Imagine that if the sensitive edges are attacked without specific protection, the encoder will forcibly change the representation structure based on the perturbed graph, which would affect the overall graph representation and then affect downstream tasks, resulting in poorer node classification performance. Following, we introduce our proposed strategies for protecting sensitive edges.

Defense Methods

After judging whether the attacked edge is sensitive, we propose two different defense strategies, i.e., local and global defense. If the $\mathrm{PEH}(\theta) > h$, we adopt the local defense strategy, otherwise global defense, where θ denotes the current parameter.

Local defense. To formulate the protection strategy for sensitive edges, we first attempt to explain why the graph encoder is susceptible to adversarial attacks. Graph encoders employ aggregation to acquire adjacent node representations during unsupervised learning. If the graph's edges are disrupted, "false" neighbors will be treated as "true" neighbors throughout the aggregating process and their information will be propagated to other nodes. So, we use an attention-based penalty mechanism on the encoder of unsupervised pre-training. Firstly, we use Graph Attention Networks (GAT) (Veličković et al. 2018) as the graph encoder and define the attention coefficient from node v_i to node v_j:

$$r_{ij} = \mathrm{LeakyReLU}(\mathbf{a}[\mathrm{W}\mathbf{z}_i || \mathrm{W}\mathbf{z}_j]), \tag{5}$$

where W is the parameter responsible for increasing the dimension of the characteristics of nodes, $||$ indicates the concatenation of vectors, **a** represents the mapping of the spliced high-dimensional features to a real number, and we used a single-layer feedforward neural network in there. For node *i*, calculate its first-order neighbor node $j \in N_i$ and their attention coefficient r_{ij} one by one. After that, the attention coefficient associated with node v_i is further normalized:

$$a_{ij} = \mathrm{softmax}(r_{ij}) = \frac{\exp(r_{ij})}{\sum_{k \in N_i} \exp(r_{ik})}. \tag{6}$$

Once obtained, the features of adjacent nodes are weighted and summed according to the calculated attention coefficient to update the features of each node i.

$$\mathbf{z}_i = ||_{k=1}^{K} \sigma\Big(\frac{1}{K} \sum\nolimits_{j \in N_i} a_{ij}^k \mathrm{W}^k \mathbf{z}_j\Big), \tag{7}$$

where $\sigma(\cdot)$ represents a nonlinear activation function and a_{ij}^k denotes the weight coefficient calculated by the k-th attention mechanism. However, this method of message passing can not distinguish the perturbed edges, and naturally obtains the "fake message" passed from them. We hope to reduce negative effects by reducing the attention coefficients of all perturbed edges in the polluted graph. Inspired by the

PA-GNN method (Tang et al. 2020), we evaluate the influence of perturbed edges on the whole system by accumulating the attention coefficient of the perturbed edges:

$$S_p = \sum\nolimits_{k=1}^{K} \sum\nolimits_{i,j \in E^*} a_{i,j}^k, \tag{8}$$

where E^* denotes the perturbed edges, i.e., added edges. K represents the number of independent attention mechanisms. The smaller S_p, the less influence the perturbed edges have. In order to further minimize S_p, a loss function is designed to simultaneously decrease the attention values of the perturbed edges and increase the attention values of the clean edges:

$$L_{\text{att}} = \min_{\Theta} \frac{\sum_{k=1}^{K} \left(\sum_{i,j \in E^*} a_{i,j}^k - \sum_{i,j \in E} a_{i,j}^k \right)}{K(E + E^*)}, \tag{9}$$

where E and E^* represent the number of original and perturbed edges respectively. Note that we try to directly reduce the attention coefficients a_{ij} received by all added edges, resulting in the overall loss being unable to converge. In addition, we also need to consider the expressive power of the graph encoder. In our scenario, we combine L_{att} with the mutual information entropy loss L_{m} to produce the local protection loss, that is:

$$L_l = L_m + \beta L_{att}, \tag{10}$$

where β is a hyper-parameter to control the trade-off between the local protection coefficient. L_m is the negative mutual information between the clean graph and the encoded representation.

Let's review the procedure once more. We first extract the sensitive edges from the clean graph using the PGD attack method, and then we propose the local defense (injection vaccination) strategy to reduce the sensitive edges' message-passing ability. Naturally, we also rely on clean graphs to enhance the expressive power of graph encoders. In the following section, we will discuss how to implement the global defense when $\text{PEH}(\theta) < h$.

Global defense. A lower value of $PEH(\theta)$ implies the attack causes less danger to the current encoder, we can implement global defense that is not so targeted to the perturbed edges. We choose a simpler approach to improve the global robustness of the model for unsupervised graph representation learning. That is, by maximizing the mutual information of the parameters and perturbed graphs, we update the global representation. We maximize the mutual information between the perturbed graph and its representation, hence improving the representation's overall robustness. The following are the comparative learning objectives for global defense:

$$L_g = -I\left(G^*; f\left(G^*\right)\right). \tag{11}$$

However, calculating mutual information is challenging. In the following section, we discuss how to solve optimization issues and construct a robust framework for learning graph representations.

Algorithm 1: Optimization algorithm

Input: Graph $G = (A, X)$, learning rate δ_l, δ_g, hyper-parameters α, β.
Output: Model parameters Θ

```
Randomly initialize Θ.
While not early-stop do
    generate adversarial graph G* based on Eq.3
    PEH(θ) ← I(G; f_θ(G)) − I(G*; f_θ(G*))
    if PEH(θ) > h then
        update parameters Θ based on Eq.13
    else
        update parameters Θ based on Eq.14
    end if
end while
Return Θ
```

Optimization

In this section, we introduce the optimization of our proposed model. The main challenge is how to maximize the mutual information between encoded representations and global summaries of graphs. Inspired by DGI, we use the binary cross-entropy loss between the positive examples and the negative examples from the original graph as follows:

$$L_m(G, e) = E_G[\log D(\mathbf{z}, \mathbf{s})] + E_{\tilde{G}}[\log(1 - D(\widetilde{\mathbf{z}}, \mathbf{s}))], \tag{12}$$

where $\mathbf{z}$ denotes the local representation encoded by GAT and $\mathbf{s}$ is a readout function to summarize the global graph-level representation, i.e., accumulating all local representations. $D(\mathbf{z},\mathbf{s})$ represents the learnable bilinear discriminator by default, the probability scores, assigned to a patch-summary pair of local and global representations. $\widetilde{\mathbf{z}}$ denotes the representation of negative samples $(\widetilde{\mathrm{X}}, \widetilde{\mathrm{A}})$ with the corruption function C, i,e., $C\left(\mathrm{X}, \mathrm{A}\right) = (\widetilde{\mathrm{X}}, \widetilde{\mathrm{A}})$. To effectively maximize mutual information of positive examples, we use the Jensen-Shannon divergence based on the product of joint distribution and marginal distribution. We further fine-tune the polluted graph G^* by generating perturbations on the clean graph to enhance the robustness of the graph, i.e., the worst-case adversarial attack. We then calculate the PEH to determine whether the perturbed edges are sensitive. If $\text{PEH}(\theta) > h$, we try to inject the vaccine against sensitive edges, i.e., local protection. The model parameters Θ are updated as follows:

$$\Theta \leftarrow \Theta - \delta_l \nabla_{\Theta} L_l. \tag{13}$$

When $\text{PEH}(\theta) < h$, the risk of this attack is low, and the model only has to be taught global representation robustness using stochastic gradient descent (SGD) in Eq.11.

	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert Feature\rvert$	$\lvert Class\rvert$
Cora	2708	5429	1433	7
Citeseer	3327	4732	3703	6
Polblogs	1490	16714	-	2

Table 1: Statistics of the experimental data

Dataset	Model	Node classification (Acc%))				Node clustering (NMI%)			
		0	0.3	0.6	0.9	0	0.3	0.6	0.9
Cora	GCN	80.5±0.5	72.7±0.3	66.5±0.7	63.7±0.8	55.3±0.6	45.8±0.5	42.3±0.4	35.5±1.1
	GAT	80.9±0.6	**74.2±0.9**	69.1±2.2	65.3±1.3	51.8±0.4	45.6±1.3	41.5±0.8	38.6±2.0
	DGI	80.8±2.0	67.0±1.8	62.2±2.0	54.6±1.0	60.7±2.6	40.6±3.8	33.4±2.3	26.5±3.3
	RGCN	79.4±1.1	68.6±2.4	67.9±1.3	60.5±1.7	**61.2±0.9**	**49.3±3.9**	37.3±2.0	31.7±2.4
	EdgeDrop	**82.6±3.5**	43.5±7.0	35.0±4.9	26.8±6.7	36.7±5.1	16.5±6.5	9.7±3.2	7.1±3.5
	GraphCL	81.2±0.2	68.2±0.2	62.9±0.2	55.0±0.2	60.8±0.3	40.9±0.2	33.6±0.2	26.6±0.3
	GRV	79.9±1.5	65.9±1.2	64.1±1.3	58.3±0.7	52.4±1.5	38.4±3.1	35.1±1.9	28.5±1.3
	Ours-Wl	75.6±1.5	64.2±0.8	63.5±1.1	58.8±1.2	47.5±2.0	38.4±0.8	35.2±1.7	24.7±1.1
	Ours	77.3±2.8	72.1±2.2	**69.2±1.3**	**66.2±1.4**	51.1±0.8	46.4±2.6	**44.2±3.2**	**42.1±3.9**
Citseer	GCN	69.8±1.2	58.5±0.7	51.2±0.8	49.7±0.2	42.7±1.0	**33.8±1.2**	23.3±0.6	20.5±0.3
	GAT	70.5±0.8	59.3±0.5	52.3±0.3	50.9±1.0	**44.9±0.7**	32.0±0.3	25.9±0.2	20.4±0.7
	DGI	71.2±0.8	56.5±1.1	49.8±1.2	45.5±1.6	47.5±0.5	32.3±0.7	19.1±0.5	14.8±0.3
	RGCN	66.9±1.7	59.5±2.2	51.5±1.3	49.8±0.8	39.8±2.4	29.8±3.0	26.7±2.1	17.5±1.8
	EdgeDrop	**76.8±3.7**	54.2±3.5	39.6±3.2	27.8±2.4	15.3±2.7	13.5±1.8	9.1±1.4	6.2±0.6
	GraphCL	72.9±0.2	56.4±0.1	52.6±0.2	43.2±0.2	46.6±0.2	25.1±0.2	20.1±0.2	12.9±0.1
	GRV	69.2±0.4	59.2±0.4	51.2±0.3	48.0±0.3	45.0±1.7	28.7±0.3	20.2±1.4	16.3±1.2
	Ours-Wl	65.8±0.8	52.3±1.5	47.5±1.1	42.8±1.0	37.2±1.6	24.4±1.2	19.0±1.4	15.5±1.3
	Ours	67.8±3.1	**59.9±2.7**	**55.6±0.8**	**51.1±1.1**	42.4±2.9	30.1±3.0	**28.8±3.7**	**20.8±4.3**
Polblogs	GCN	86.3±0.3	81.1±1.5	80.3±0.7	76.1±2.5	42.3±0.7	35.2±2.5	26.6±1.1	23.8±1.0
	GAT	86.7±0.5	82.8±2.8	80.5±0.9	78.7±2.2	44.2±1.5	37.5±3.9	33.8±1.2	28.5±3.0
	DGI	84.9±0.6	81.8±0.6	78.3±0.8	75.3±0.7	34.3±2.3	35.5±1.7	30.8±0.4	25.7 ±3.1
	RGCN	85.3±0.8	81.7±0.8	79.1±0.4	78.5±0.7	40.7±0.4	32.8±1.3	30.4±1.0	26.7±0.6
	EdgeDrop	86.7±3.3	76.7±2.4	72.5±2.0	68.1±1.3	33.2±2.5	21.8±6.6	15.2±4.3	9.7±0.5
	GraphCL	86.2±0.2	78.3±0.4	69.0±0.3	67.6±0.4	30.6±0.4	19.3±0.7	10.6±0.4	9.3±0.4
	GRV	87.0±0.8	84.2±1.1	82.6±1.5	80.6±2.7	43.5±2.3	37.5±1.8	32.8±0.9	28.1±1.1
	Ours-Wl	85.8±0.5	82.9±1.2	80.1±1.4	78.6±0.7	40.3±1.7	40.2±2.1	35.5±1.3	29.8±1.7
	Ours	**87.4±0.8**	**85.8±1.3**	**84.5±1.7**	**83.3±0.6**	**45.0±2.7**	**42.5±1.5**	**39.1±2.4**	**35.6±1.2**

Table 2: Node classification and clustering performance under PGD attack. 0, 0.3, 0.6, and 0.9 represent the perturbation rates.

$$\Theta \leftarrow \Theta - \delta_g \nabla_\Theta L_{\mathrm{g}}, \tag{14}$$

where δ_l and δ_g control the learning rate. Overall, Algorithm 1 summarizes the framework for learning local-global robustness that we presented.

Experiments

We evaluated the robustness of unsupervised graph representation learning using three real-world datasets. Specifically, we design experiments to investigate the following questions. Q1: How well can our approach perform in downstream tasks, e.g., node classification and clustering? Q2: How does our method fare against various adversarial attacks? Q3: What is the distribution of the attention score over the original and perturbed edges? Q4: How sensitive is the model to its parameters?

Experimental Setup

Datasets. We use three real-world datasets in our experiments, i.e., Cora, Citeseer (Sen et al. 2008) and Polblogs (Adamic and Glance 2005). Their detailed statistics are given in Table 1. Cora and Citeser are citation networks with nodes representing documents and edges representing citation relationships. The attributes of nodes are represented as bag-of-words. Polblogs is a network of political blogs from a crawl of the front page of the blog. Since the dataset lacks attributes, we set the attribute matrix to be an identity matrix.

Baselines. We compare our methods with seven baselines in two categories as listed below:

1) Non-robust graph representation learning

GCN (Welling and Kipf 2016) is a graph convolutional network model which learns node representations via message passing.

GAT (Veličković et al. 2018) leverages multi-head self-attention to aggregate node features.

DGI (Velickovic et al. 2019) is an unsupervised representation learning method that relies on maximizing mutual information between representations and global summaries of graphs.

2) Robust graph representation learning

RGCN (Zhu et al. 2019) adapts Gaussian distributions as hidden representations to "fortify" GCNs against adversarial attacks.

EdgeDrop (Rong et al. 2020) is a novel and flexible technique to increase robustness via randomly removing edges and the message passing reducer technology is introduced, and we delete 10% edges during training on the DGI surrogate model.

GraphCL (You et al. 2020), is a graph contrastive learning framework that learns unsupervised graph representations by augmentation for the sake of better robustness.

GRV (Xu et al. 2022) is also an unsupervised learning method, which designs a robust representation learning al-

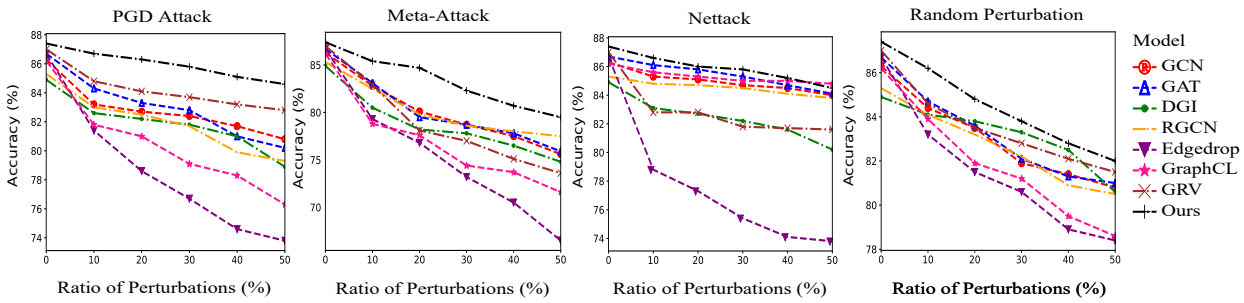


Figure 2: Node classification performance (Acc%) on Polblogs under different attackers

gorithm by using a mutual information-based measure.

Ours-Wl is a variant of our model without local defense.

Attack Methods. We also evaluate how robust our model is under different adversarial attacks and select four adversarial attack methods.

Nettack (Zügner, Akbarnejad, and Günnemann 2018) is a targeted attack method. We randomly perturb various nodes with Nettack, next, we count the total number of perturbed edges to make it approximate the default perturbation rate.

Meta-attack (Zügner and Günnemann 2019) is a non-targeted adversarial attack and perturbs the discrete graph structure via meta-gradients. We set a different perturbation rate from 0% to 50%, with a step of 10%.

Random perturbation is an attack method that randomly connects or removes edges from a clean graph, and we design different perturbation ratios with a step of 10%.

PGD (Xu et al. 2019) is a non-targeted attack, the most effective first-order adversarial method.

Settings and Parameters. For Cora and Citeseer, randomly assign them to the training, verification, and test sets in the ratio of 1:1:8. For Polblogs, we randomly select 10% nodes for training and 80% nodes for testing. We adopt PGD to generate attacked datasets on clean graphs by default in the training phase. For our model, we set the parameters $h = 0.2$, $\alpha = 1$ and $\beta = 0.4$. At the stage of evaluating, we consider both the performance and the robustness of the model, so we employ the four attack methods indicated above, and set different perturbation ratios with a step of 10%. All comparative learning baselines use a two-layer GCN as the encoder and use the default setting. For node clustering tasks, we use normalized mutual information (NMI) as the measure. In addition, we run 10 trials and report the mean and the standard deviation.

Performances in Downstream Tasks against PGD Attack(Q1)

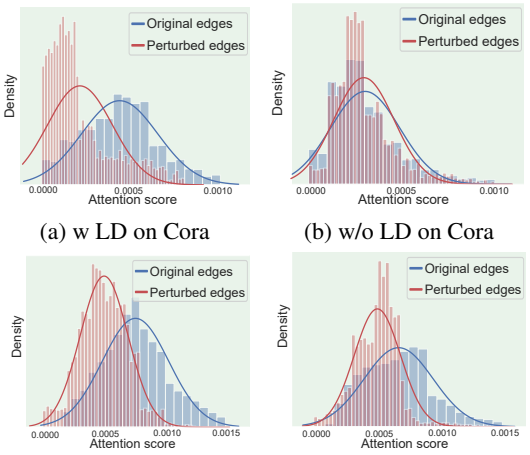
In this section, we compare the node classification and clustering performance of our proposed model against PGD adversarial attacks at different perturbation rates with seven baselines. We employ the PGD's default parameter settings, obtained from its authors. From Table 2, we find that (i) as the perturbation ratio rises, so does the performance of our technique compared with baselines, highlighting the need for early vaccination; (ii) we can see that our model performs better than Ours-Wl. This also illustrates that the attention penalty mechanism is advantageous for representation ability in the face of adversarial attacks; (iii) the accuracy of a semi-supervised model deviates more noticeably as the perturbation rate rises. It may be that the increase in the perturbation rate leads to the inconsistency between the label information and the perturbed graph.

Performances against Different Adversarial Attacks(Q2)

In this section, we further evaluate the performance of representations against different adversarial attacks, where perturbations are defined as adding edges. We adapt some common attack strategies to the baselines, i.e., PGD, Nettack, meta-attack, and random perturbation. This time, we examine the node classification problem on the Polblogs dataset as an example. This choice is convincing because the aforementioned attack strategies only modify the graph topology, which is the only piece of information we know regarding the Polblogs dataset. The experimental results are shown in Figure 2. From the figures, we observe the following results: (i) The performance of our model decays slowly when the perturbation rate goes higher, while other robust models drop rapidly in most cases. This proves the importance of vaccinating the sensitive edges; (ii) compared with other methods, our pre-trained model can defend against different adversarial attacks in the downstream tasks effectively. This is because our method successfully combines and balances local and global defense to improve the robustness of the model; (iii) in general, most models perform better when under Nettack. Because Netstack is primarily used to target unnoticeable edges, many sensitive edges are not attacked for this reason; (iv) our model can also defend against random attacks. The reason is that these attack methods more or less change the sensitive edges that need to be specially protected.

Effects of the Attention Score(Q3)

An interesting question is why our defense strategy using a global-sensitive protection approach is so robust. If we can determine what makes representation so robust, we can avoid costly meta-gradient computations and potentially use this information to defend against adversarial attacks. We



(a) w LD on Cora (b) w/o LD on Cora

(c) w LD on Polblogs (d) w/o LD on Polblogs

Figure 3: Distributions of attention coefficients. We implement GCN w and w/o LD (local defense) on Cora and polblogs datasets

design an attention visualization task for which we use Eq.6 to calculate attention coefficients. For a fair comparison, we compute attention scores for each node pair ($|V| \times |V|$) in the encoder, then naturally normalize them and evaluate all attention in terms of all node pairs. Generally, the more relevant two representations are, the more attention coefficients are scored between them. So we investigate the four attention coefficient factors with and without the vaccination mechanism as well as from two datasets respectively in Figure 3. As shown in Figure 3(a) and Figure 3(c), normal edges receive relatively higher attention scores when compared to the method of global defense. Simultaneously, perturbed edges are compelled to reduce the attention coefficient between them. We confirm that the ability to punish perturbation can also be transferred to the encoder. These figures prove the effectiveness of the penalized aggregation mechanism of the encoder and the global-local defense algorithm.

Sensitivity Analysis(Q4)

To prove the effectiveness of the design of our model, we evaluate the sensitivity of our model on the 20% ratio of PGD perturbation to its two main parameters, i.e., h, and β. h controls the local defense or global defense and β balances the robustness and ability of representations at local defense. We explore the sensitivity of the Polblogs dataset. We vary h from 0 to 0.5 and β from 0 to 1. The results in Figure 4 demonstrate that our model is not markedly sensitive to changes within certain ranges. However, compared with values of h, we find extremely minimum values β result in low performances under perturbation, demonstrating the proposed local and global defense are both essential. Furthermore. It is worth noting that we fix β at 0.4 and h at 0.2 to achieve the best performance on the Cora and Polblogs datasets, implying just one kind of defense is not enough and we need to combine local and global defense to resist adversarial attacks.

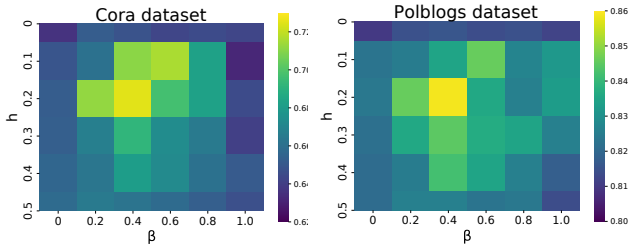


Figure 4: Parameter sensitivity analysis

Related Works

In this section, we review related work about robust representation training on graphs. Contrastive learning is known for its cheapness and strong performance, from traditional methods such as DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) and node2vec (Grover and Leskovec 2016), to graph contrastive learning, e.g., DGI (Velickovic et al. 2019). Then hybrid method GMI (Peng et al. 2020) and adaptive augmentation GCA (Zhu et al. 2021) are proposed. However, representations on graphs are known to be vulnerable to adversarial attacks. At present, aiming to increase the robustness of representations has drawn increasing research interest in the past few years. Most of these works use attacked graphs as a part of data augmentation to learn representations. GraphACL (Guo et al. 2022) maximizes the mutual information using global representations of a perturbed graph. RoSA (Zhu et al. 2022) utilizes non-aligned augmented views and introduces adversarial training to increase its robustness. ArieL (Feng et al. 2022) uses an adversal attack and information regulation to obtain comparison samples of the reasonable constraint range that satisfy the conditions. However, they mostly use perturbed graphs to learn better representations and do not directly defend against adversarial attacks. (Xu et al. 2022) introduces graph representation vulnerability (GRV) to successfully identify and apply the information of perturbed graphs. Nevertheless, they fail to recognize that different perturbations result in various destructions of the representation, making it difficult to identify and safeguard problematic edges.

Conlusion

In this paper, we propose a novel robust model that successfully defends adversarial attacks by combining global and local defense strategies. By penalizing attention coefficients of perturbed edges to encoders, our method can effectively protect dangerous edges in advance. Experimental results illustrate our methods can learn robust representation to defend against various adversarial attack strategies, particularly for minor but extremely dangerous perturbations. In the future, we will explore representation learning in dynamic adversarial attack scenarios. At the same time, we also apply the representation to the link prediction task.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grants No. U22B2036, 62272340, 62276187, 11931015), National Science Fund for Distinguished Young Scholars (Grants No. 62025602), Tencent Foundation, and XPLORER PRIZE.

References

Adamic, L. A.; and Glance, N. 2005. The political blogosphere and the 2004 US election: Divided they blog. In *LINKDD*, 36–43.

Bo, D.; Wang, X.; Shi, C.; Zhu, M.; Lu, E.; and Cui, P. 2020. Structural deep clustering network. In *WWW*, 1400–1410.

Coutrot, A.; Manley, E.; Goodroe, S.; Gahnstrom, C.; Filomena, G.; Yesiltepe, D.; Dalton, R.; Wiener, J. M.; Hölscher, C.; Hornberger, M.; et al. 2022. Entropy of city street networks linked to future spatial navigation ability. *Nature*, 604(7904): 104–110.

Eswaran, D.; Günnemann, S.; Faloutsos, C.; Makhija, D.; and Kumar, M. 2017. Zoobp: Belief propagation for heterogeneous networks. *Proceedings of the VLDB Endowment*, 10(5): 625–636.

Feng, S.; Jing, B.; Zhu, Y.; and Tong, H. 2022. Adversarial graph contrastive learning with information regularization. In *WWW*, 1362–1371.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*, 855–864.

Guo, J.; Li, S.; Zhao, Y.; and Zhang, Y. 2022. Learning robust representation through graph adversarial contrastive learning. In *DASFAA*, 682–697. Springer.

Hjelm, R. D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Bachman, P.; Trischler, A.; and Bengio, Y. 2018. Learning deep representations by mutual information estimation and maximization. In *ICLR*.

Jin, D.; Huo, C.; Liang, C.; and Yang, L. 2021a. Heterogeneous graph neural network via attribute completion. In *WWW*, 391–400.

Jin, D.; Yu, Z.; Jiao, P.; Pan, S.; He, D.; Wu, J.; Yu, P.; and Zhang, W. 2021b. A survey of community detection approaches: From statistical modeling to deep learning. *IEEE Transactions on Knowledge and Data Engineering*.

Kipf, T. N.; and Welling, M. 2016. Variational graph auto-encoders. *STAT*, 1050: 21.

Li, K.; Liu, Y.; Ao, X.; Chi, J.; Feng, J.; Yang, H.; and He, Q. 2022a. Reliable representations make a stronger defender: Unsupervised structure refinement for robust GNN. In *SIGKDD*, 925–935. ACM.

Li, Y.; Qiao, G.; Gao, X.; and Wang, G. 2022b. Supervised graph co-contrastive learning for drug–target interaction prediction. *Bioinformatics*, 38(10): 2847–2854.

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

Paranjape, A.; Benson, A. R.; and Leskovec, J. 2017. Motifs in temporal networks. In *WSDM*, 601–610.

Peng, Z.; Huang, W.; Luo, M.; Zheng, Q.; Rong, Y.; Xu, T.; and Huang, J. 2020. Graph representation learning via graphical mutual information maximization. In *WWW*, 259–270.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*, 701–710.

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

Rong, Y.; Huang, W.; Xu, T.; and Huang, J. 2020. DropEdge: Towards deep graph convolutional networks on node classification. In *ICLR*.

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

Sun, T.; Qian, Z.; Dong, S.; Li, P.; and Zhu, Q. 2022. Rumor detection on social media with graph adversarial contrastive learning. In *WWW*, 2789–2797.

Tang, X.; Li, Y.; Sun, Y.; Yao, H.; Mitra, P.; and Wang, S. 2020. Transferring robustness for graph neural network against poisoning attacks. In *WSDM*, 600–608.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph attention networks. In *ICLR*.

Velickovic, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep graph infomax. *ICLR (Poster)*, 2(3): 4.

Vlaic, S.; Conrad, T.; Tokarski-Schnelle, C.; Gustafsson, M.; Dahmen, U.; Guthke, R.; and Schuster, S. 2018. ModuleDiscoverer: identification of regulatory modules in protein-protein interaction networks. *Scientific reports*, 8(1): 1–11.

Welling, M.; and Kipf, T. N. 2016. Semi-supervised classification with graph convolutional networks. In *ICLR*.

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

Wu, L.; Cui, P.; Pei, J.; and Zhao, L. 2022. *Graph Neural Networks: Foundations, Frontiers, and Applications*. Singapore: Springer Singapore.

Xu, J.; Yang, Y.; Chen, J.; Jiang, X.; Wang, C.; Lu, J.; and Sun, Y. 2022. Unsupervised adversarially robust representation learning on graphs. In *AAAI*, volume 36, 4290–4298.

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

Yang, L.; Zhang, L.; and Yang, W. 2021. Graph adversarial self-supervised learning. *NeurIPS*, 34: 14887–14899.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph contrastive learning with augmentations. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *NeurIPS*, volume 33, 5812–5823. Curran Associates, Inc.

Yu, Z.; Jin, D.; Liu, Z.; He, D.; Wang, X.; Tong, H.; and Han, J. 2021. AS-GCN: Adaptive semantic architecture of graph convolutional networks for text-rich networks. In *ICDM*, 837–846. IEEE.

Yuhao, Y.; Huang, C.; Xia, L.; and Li, C. 2022. Knowledge graph contrastive learning for recommendation. In *SIGIR*, 1434–1443.

Zhang, Y.; Xiong, Y.; Ye, Y.; Liu, T.; Wang, W.; Zhu, Y.; and Yu, P. S. 2020. SEAL: Learning heuristics for community detection with generative adversarial networks. In *SIGKDD*, 1103–1113.

Zhu, D.; Zhang, Z.; Cui, P.; and Zhu, W. 2019. Robust graph convolutional networks against adversarial attacks. In *SIGKDD*, 1399–1407.

Zhu, Y.; Guo, J.; Wu, F.; and Tang, S. 2022. RoSA: A robust self-aligned framework for node-node graph contrastive learning. In *IJCAI*, 3795–3801.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2021. Graph contrastive learning with adaptive augmentation. In *WWW*.

Zügner, D.; Akbarnejad, A.; and Günnemann, S. 2018. Adversarial attacks on neural networks for graph data. In *SIGKDD*, 2847–2856.

Zügner, D.; and Günnemann, S. 2019. Adversarial attacks on graph neural networks via meta learning. In *ICLR*.