Smoothing Adversarial Training for GNN

Jinyin Chen, Xiang Lin, Hui Xiong, Yangyang Wu, Haibin Zheng, and Qi Xuan, *Member, IEEE*

***Abstract*— Recently, a graph neural network (GNN) was proposed to analyze various graphs/networks, which has been proven to outperform many other network analysis methods. However, it is also shown that such state-of-the-art methods suffer from adversarial attacks, i.e., carefully crafted adversarial networks with slight perturbation on clean one may invalid these methods on lots of applications, such as network embedding, node classification, link prediction, and community detection. Adversarial training has been testified as an efficient defense strategy against adversarial attacks in computer vision and graph mining. However, almost all the algorithms based on adversarial training focus on global defense through overall adversarial training. In a more practical scene, certain users would be targeted to attack, i.e., specific labeled users. It is still a challenge to defend against target node attack by existing adversarial training methods. Therefore, we propose smoothing adversarial training (SAT) to improve the robustness of GNNs. In particular, we analytically investigate the robustness of graph convolutional network (GCN), one of the classic GNNs, and propose two smooth defensive strategies: smoothing distillation and smoothing cross-entropy loss function. Both of them smooth the gradients of GCN and, consequently, reduce the amplitude of adversarial gradients, benefiting gradient masking from attackers in both global attack and target label node attack. The comprehensive experiments on five real-world networks testify that the proposed SAT method shows state-of-the-art defensibility against different adversarial attacks on node classification and community detection. Especially, the average attack success rate of different attack methods can be decreased by about 40% by SAT at the cost of tolerable embedding performance decline of the original network.**

***Index Terms*— Adversarial attack, adversarial training, complex network, cross-entropy loss, smoothing distillation (SD).**

I. Introduction

THE on-going process of datafication continues to turn many aspects of our lives into computerized data [1]. Modern computational society pays more and more attention to the relationships among individuals, which can be well described by various graphs or networks, such as social networks, communication networks, biological networks, and traffic networks. Network structure plays an important role in many real-world applications. Many tools and data analysis methods have been advocated for network analysis and network embedding. Among them, deep learning methods [2]–[7] are becoming attractive due to their promising modeling abilities. It learns the representation of the network structure by graph neural networks (GNNs), which has shown promising results in various applications, such as link prediction [8], [9], node classification [10], [11], graph classification [12], community detection [13], and social network analysis [14]. Since the representation of the network structure learned by the deep network embedding methods usually directly determines the performances of downstream tasks, it has attracted more and more attention in the past decades.

However, despite the great success of GNNs, recent studies indicate that they are quite vulnerable to various adversarial attacks [15]–[17]. Such attacks add carefully designed perturbations on network structures or node features to degrade the performance of GNNs. For instance, it could disguise Mohamed Atta's leading position within the WTC terrorist network [18] by rewiring a strikingly small number of connections. Therefore, for one thing, we enjoy the benefit from the great efficiency of GNNs; for another thing, we also suffer from the potential security threats when someone misuses them. Adversarial attacks pose a great puzzle for applying GNNs to real-world networks. How to develop an efficient GNN model robust against attacks is becoming an urgent priority.

Adversarial training, one of the most efficient defense methods through training the deep model with adversarial examples labeled as ground truth, is also successfully applied to GNNs [19]. Currently existing defenses for GNNs are mostly based on adversarial training, such as virtual adversarial training (VAT) [19], ReFeX [20], and robust training [21]. They can indeed protect the GNNs from adversarial attacks toward the whole network. More specifically, they perform global adversarial training (Global-AT) for the whole network consisted of all nodes, while, in more practical applications, they cannot protect some important nodes escape target adversarial attacks because their defense benefits all nodes. They adopt adversarial training to smooth the classification boundary, without modifying the model structures or objective functions of GNNs.

To defend against both global attack and target label node attack, we propose a novel smoothing adversarial training (SAT) method that utilizes adversarial training and smoothing strategies to improve the robustness of GNN models. In particular, we first select graph convolution network (GCN) [22] as a surrogate model and then use two effective attack methods, i.e., FGA [23] and NETTACK [24],

Manuscript received January 22, 2020; revised July 19, 2020 and October 13, 2020; accepted December 1, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 62072406 and Grant 61973273, in part by the Natural Science Foundation of Zhejiang Provincial under Grant LY19F020025 and Grant LR19F030001, in part by the Major Special Funding for "Science and Technology Innovation 2025" in Ningbo under Grant 2018B10063, and in part by the Key Laboratory of the Public Security Ministry Open Project in 2020 under Grant 2020DSJSYS001. *(Corresponding authors: Jinyin Chen, Qi Xuan.)*

The authors are with the College of Information Engineering, Institute of Cyberspace Security, Zhejiang University of Technology, Hangzhou 310023, China (e-mail: chenjinyin@zjut.edu.cn; lynnzlnx@163.com; bearlight080329@gmail.com; zjuwuyy@zju.edu.cn; haibinzheng320@gmail.com; xuanqi@zjut.edu.cn).

Digital Object Identifier 10.1109/TCSS.2020.3042628

to generate adversarial examples for adversarial training. Furthermore, in the process of adversarial training, distillation and smoothing loss function are proposed to smooth the gradients of GCN and increase the cost of adversarial attacks. Especially, we make the following main contributions.

1) We propose an SAT method to enhance the robustness of the GCN model against adversarial attacks. In this method, we adopt a smoothing distillation (SD) and smoothing cross-entropy loss (SCEL) function to realize the gradient mask, making it difficult to design an attack via a gradient.
2) We design different adversarial training methods for different types of attacks. Aiming at the global network attack, we propose Global-AT to improve the robustness of GCN, and we also propose target label adversarial training against target adversarial attack. Both the proposed defense strategies can achieve state-of-the-art defense performance.
3) Comprehensive experimental results validate that SAT performs significantly better than the adversarial training strategies, in terms of higher average defense rate (ADR) and lower average confidence different (ACD) on a number of real-world networks, achieving state-of-the-art performance.

The rest of this article is organized as follows. In Section II, we introduce the related works of GNN models, adversarial attack methods, and defense methods on graph data. In Section III, we introduce SAT step by step. In Section IV, we empirically evaluate SAT on node classification and community detection compared with other defense methods on several real-world networks. Section V concludes this article and highlights future research directions.

II. Related Work

In this section, we briefly review the literature of GNN models, adversarial attack methods, and defense methods on graph data.

A. Graph Neural Network Models

Recently, graph mining with machine learning methods has received much attention. Especially, GNNs are widely used for their outstanding performance. Scarselli *et al.* [25] first proposed the formal definition of GNN. Afterward, they validated its computation and approximation ability through comprehensive experiments [26]. Inspired by the architecture of the convolutional neural network, Kipf and Welling [22] proposed the graph convolutional network (GCN) for semisupervised graph learning. It learns hidden layer representations by encoding both local graph structure and node features. Similarly, Pham *et al.* [27] designed a column network, named CLN, which is a deep model for collective classification. Compared with GCN, CLN can process multirelational data. Based on GCN, Kipf and Welling [28] proposed the GAE model. Adding a decoder behind GCN to reconstruct the relationship between the nodes, GAE achieves outstanding link prediction results. Monti *et al.* [29] proposed a unified framework MoNet that generalizes CNN architectures to non-Euclidean domains. Moreover, this method also generalizes several spectral methods on graphs. To deal with large scale graph data, Hamilton *et al.* [30] proposed GraphSAGE to generate node embeddings for previously unseen data, instead of training individual embeddings for each node.

B. Adversarial Attack on GNN

Recently, various adversarial attack techniques are proposed for GNN models. Zügner *et al.* [24] studied adversarial attacks on neural networks for graph data, named NETTACK. It generates the adversarial graph iteratively according to the confidence value change by adding perturbations. Based on NETTACK, Zügner and Günnemann [31] developed a general attack on the training procedure of GCN using metagradients, which treats the graph as a hyperparameter to optimize. Bojcheski and Günnemann [32] proposed a data poisoning attack on unsupervised node representation learning. It uses a principled strategy to attack downstream tasks. Inspired by reinforcement learning, Dai *et al.* [15] proposed RL-S2V that learns the generalizable attack strategy based on the prediction labels alone. Based on gradient information, Chen *et al.* [23] proposed a fast gradient attack (FGA) method that can change the embedding of the target node by modifying a few links. Similarly, Xu *et al.* [33] proposed a generic framework of adversarial attack. They used projected gradient descent (PGD) topology attack and min–max topology attack to mislead the GNN classifier. To solve the problem of data discreteness in graph data, Wu *et al.* [34] introduced the concept of the integrated gradient to generate the adversarial network by disturbing both node features and links. Chang *et al.* [35] proposed GF-Attack that is a black box attack. It performs the attack only on the graph filter instead of providing any information on the target classifiers.

C. Defense Strategies for GNN

In order to enhance the robustness of GNN models to resist the adversarial attack, Dai *et al.* [15] sought to use a cheap method for adversarial training, which simply drops the links globally at random during each training step for defense gain. However, in most cases, such a random strategy is not good enough for the DNNs when they face more powerful adversarial attacks. Based on PGD topology attack and min–max topology attack, Xu *et al.* [33] proposed an adversarial training method for GNNs to improve their robustness. Zügner and Günnemann [21] proposed robustness certificate to evaluate whether a node is robust or not. Furthermore, they developed the semisupervised nature of the GNN by taking also the unlabeled nodes into account for robust training. Sun *et al.* [19] proposed VAT that adds an adversarial regularization item on the supervised loss of GCN to enhance its generalization performance. It can improve the symmetrical Laplacian smoothness of semisupervised learning models. Miller *et al.* [20] proposed ReFeX that uses a support vector machine as an alternative classifier and selects nodes of the highest degree in each class to train the alternative classifier. It can effectively increase the attack cost of attackers. Wang *et al.* [36] proposed GraphDefense. This algorithm uses

the GCN model to obtain the confidence of the unlabeled nodes set and constructs the adversarial training loss function that is combined with a labeled node and an unlabeled node. Zhu *et al.* [37] proposed robust GCN (RGCN) that uses Gaussian distributions to represent nodes in each convolutional hidden layer. In this method, a variance-based attention mechanism is also considered. The experiments validate the effectiveness of RGCN against adversarial attacks. They all protect the whole network from attacks.

III. Method

We propose the SAT for GNN that consists of three strategies, including adversarial training, distillation, and smoothing loss function. In this article, we only focus on evasion attacks on graph structures, and SAT will defend against this type of attack by different defense strategies.

A. Problem Definition

We first give the definitions of adversarial attack and adversarial defense as follows.

1) *Adversarial Attack:* Let $G(V, E, A)$ denote a network, where V denotes the node set of size $n = |V|$, E denotes the edge set, and $A \in \{0, 1\}^{n\times n}$ denotes the adjacent matrix. Given the attacked nodes $V_a \subseteq V$ and the training node subset $V_t \subseteq V$, for a prediction model $f_\theta(A)$ parameterized by θ, the adversarial attack is used to disturb the adjacent matrix A and mislead the prediction model. It can be formulated as

$$\begin{aligned} &\arg\max_{A'} \sum_{u\in V_a} \mathcal{L}(f_{\theta^*}(A')) \\ &\text{s.t. } \theta^* = \arg\min_{\theta} \sum_{i\in V_t} L(f_\theta(A), Y_i) \end{aligned} \tag{1}$$

where u denotes the node in V_a, i denotes the node in V_t, and Y_i denotes the one-hot label of node i. $L(\cdot, \cdot)$ denotes the cross-entropy loss function minimized by θ. A' is the generated adversarial network. $\mathcal{L}(f_{\theta^*}(A'))$ is defined as the loss measuring the attack damage. Lower loss corresponds to higher quality. During the process of attack, the attackers achieve higher $\mathcal{L}$ by changing A to A'. θ^* are learned on the clean network that remains unchanged.

2) *Adversarial Defense:* The adversarial defense is used to defend model F_θ from attacks triggered by the adversarial network A'. It can be formulated as

$$\arg\min_{\theta} \mathscr{L}(\theta; A') \tag{2}$$

similar to $\mathcal{L}(A', Z)$ in adversarial attack, $\mathscr{L}(\theta; A')$ is defined as the loss measuring the robustness of F_θ, and lower loss corresponds to better robustness. $\mathscr{L}$ can also be set to be the same as L. At this time, the entire defense process only uses A' to train the parameters θ of F_θ. In this case, the defense method is named adversarial training.

B. Surrogate Model

In this work, we focus on node classification employing the GCN. GCN learns the hidden layer representations that encode both local structure and features of nodes. For most data sets, two-layer GCN can achieve good performance for network embedding. Therefore, we consider a two-layer GCN model as the surrogate model. A is the adjacency matrix, which changes to $\tilde{A} = A + I_N$, with I_N being the identity matrix, if we add self-connections into the network. The forward propagation process of the surrogate model takes the following simple form:

$$Y' = \text{softmax}(\bar{A}\sigma(\bar{A}XW_0)W_1) \tag{3}$$

where X is a matrix of node feature vectors, and $\bar{A} = \tilde{D}^{-(1/2)}\tilde{A}\tilde{D}^{-(1/2)} = \tilde{D}^{-(1/2)}(A + I_N)\tilde{D}^{-(1/2)}$. $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ is the degree matrix of $\tilde{A}$. $W_0 \in R^{C\times H}$ and $W_1 \in R^{H\times|F|}$ are the input-to-hidden and hidden-to-output weight matrices, respectively, with the hidden layer of H feature maps; σ is the Relu activation function. Here, the softmax activation function is applied rowwise.

For node classification, the common training objective of GCN is to minimize the cross-entropy loss, defined as

$$L = -\sum_{l=1}^{|S_{\text{train}}|}\sum_{k=1}^{|F|} Y_{lk} \ln\left(Y'_{lk}\right) \tag{4}$$

where S_{train} is the set of nodes with labels, $F = [\tau_1, \ldots, \tau_{|F|}]$ is the category set for the nodes in the network, $|F|$ denotes the number of categories, Y is the ground-truth label matrix with $Y_{lk} = 1$ if node v_l belongs to category τ_k and $Y_{lk} = 0$ otherwise, and Y' is the output of the model calculated by (3).

C. Adversarial Training

Szegedy *et al.* [38] showed that, by training on a mixture of adversarial and clean examples, a neural network could be regularized to a certain extent. That is, adversarial training is efficient to enhance the defensive capability of deep neural models against adversarial attacks [38], [39].

Given an original clean network G_{cln} with adjacency matrix A, here, we adopt adversarial training to enhance the defensive capability of the GCN model. In particular, we propose two adversarial training strategies: Global-AT to protect all the nodes and target-label adversarial training (Target-AT) to protect the nodes of the target label, which are introduced in the following.

1) Global Adversarial Training: In Global-AT, we generate the adversarial network iteratively. The adversarial links are selected by adversarial attack methods on the training node set $S_{\text{train}} = [v_1, \ldots, v_m]$. In particular, for all nodes in S_{train}, the generation process of tth adjacency matrix $\hat{A}^t$ can be described by the following steps.

1) *Selecting the Adversarial Links:* First, we use an adversarial attack method to select the adversarial links that can maximize the negative cross-entropy loss of the training nodes in S_{train}. These adversarial links can be saved as a matrix Λ. The size of Λ is the same as that of the adjacency matrix A. Its element $\Lambda_{ij} \in \{-1, 0, 1\}$

Algorithm 1 Training GCN With Global-AT

Input: Original clean network G_{cln} with adjacency matrix A, the training set $S_{train} = [v_1, \ldots, v_m]$.
Output: The classification result.

```
Train the GCN model with G_cln;
Initialize the adjacency matrix of the adversarial network by Â^0 = A;
for t = 1 to m do
    Λ = AdversarialAttack(Â^0, S_train);
    Update the adjacency matrix Â^t by Â^t_ij = Â^{t-1}_ij + Λ_ij;
end
Train the GCN model with the G_adv that constructed by Â^m, and output the classification result;
return The classification result.
```

indicates the modification of links. If $\Lambda_{ij} = 1$, we will add an adversarial link between node v_i and v_j. If $\Lambda_{ij} = -1$, we will delete the relationship between v_i and v_j. If $\Lambda_{ij} = 0$, the relationship between v_i and v_j will not be modified.

2) *Updating Adversarial Network:* We use Λ to update the $(t-1)$th adversarial network. The specific update process is defined as

$$\hat{A}^t_{ij} = \hat{A}^{t-1}_{ij} + \Lambda_{ij} \tag{5}$$

where $\hat{A}^t_{ij}$, $\hat{A}^{t-1}_{ij}$ and Λ_{ij} denote the elements of $\hat{A}^t$, $\hat{A}^{t-1}$, and Λ, respectively.

The pseudocode of Global-AT is given in Algorithm 1.

2) Target-Label Adversarial Training: Different from Global-AT, Target-AT only focuses on defensing adversarial attacks on the nodes with the target label. For example, for target label τ_p, the adversarial attack methods only generate the adversarial links on the node set S_{τ_p}. The rest steps of Target-AT are the same as Global-AT.

D. Smoothing Strategy

Adversarial training attempts to classify the target node in the adversarial network correctly, leading to the change of decision boundary. However, Szegedy *et al.* [38] suggested that a smart enough adversary can always find new adversarial examples to successfully attack the target classifier, no matter how the decision boundary changes.

Traditionally, adversarial attacks exploit gradients to estimate the sensitivity of a DNN model to its input dimensions. For network data, if adversarial gradients are relatively high, crafting adversarial networks for target nodes becomes quite easy since small adversarial perturbation will induce a high variation of classification output for target nodes. To defend against such adversarial attacks, one needs to reduce the introduced variations around the output confidence and, furthermore, the amplitude of adversarial gradients. In other words, for GCN, we must smooth the trained model, leading to higher generalization ability. We, thus, propose two smooth strategies for the GCN model after adversarial training, which are described in the following.

1) Smoothing Distillation: GCN always produces class probabilities by using a softmax layer, within which a given neuron, corresponding to a class indexed by $i \in [0, \ldots, |F| - 1]$ ($|F|$ denotes the number of categories), computes the ith element in the output confidence vector Y'_x for the sample $x \in S_{\text{train}}$

$$Y'_{xi} = \frac{\exp(z_{xi}/T)}{\sum_j \exp(z_{xj}/T)} \tag{6}$$

where $[z_{x1}, \ldots, z_{x|F|-1}]$ are the $|F|$ logits that correspond to the hidden layer outputs for each of the $|F|$ classes, and T is a temperature that is normally set to 1. Using a higher value for T produces a softer probability distribution over classes.

Suggested by Hinton *et al.* [40], the distilled model is trained on a transferred data set. Instead of its ground-truth label, the set is labeled with soft target distribution produced by the cumbersome model with a high temperature. More specifically, the proposed distillation model consists of two modules: the first one is the soft classification module for labeling the unlabeled nodes set; the second one is the distillation module that uses the soft labels of the data, instead of the ground truth, to train the classifier. Benefitting from the two modules, the proposed distillation model can improve the robustness of the classifier.

Inspired by distillation, we propose an SD to improve the robustness of GCN against adversarial perturbations. Compared with the original GCN, the proposed distilled GCN model can not only promise high classification accuracy but also be beneficial to transferability. Meanwhile, the information extracted by the distillation will help to filter the perturbations purposely added to the network and, thus, increase the robustness of the model.

Different from the original distillation, in SD, we keep the same architecture to train both the original GCN model and the distilled one. The framework of SD is illustrated in Fig. 1, and its training procedure is described as follows.

1) *Training the Initial GCN Model:* Given the training set $S_{\text{train}} = [v_1, \ldots, v_m]$ and its hard label matrix Y, first, we train the initial GCN model with a softmax output layer at temperature T. Based on (3), we set Y' as the output confidence of the trained model.
2) *Encoding Nodes by Soft Label:* Instead of using hard label matrix Y, we use the soft label Y' to encode the confidence probabilities over the labels for all the trained nodes.
3) *Training the Distilled GCN Model:* Based on the soft label matrix Y' and hard label matrix Y, we train the distilled GCN model, where the objective function L_{all} is composed of the soft loss function L_s and original loss function L, defined as

$$L_{\text{all}} = \frac{T^2 L_s}{T^2 + 1} + \frac{L}{T^2 + 1} \tag{7}$$

with

$$L_s = -\sum_{l=1}^{|S_{\text{train}}|} \sum_{k=1}^{|F|} Y'_{lk} \ln\left(Y''_{lk}\right) \tag{8}$$

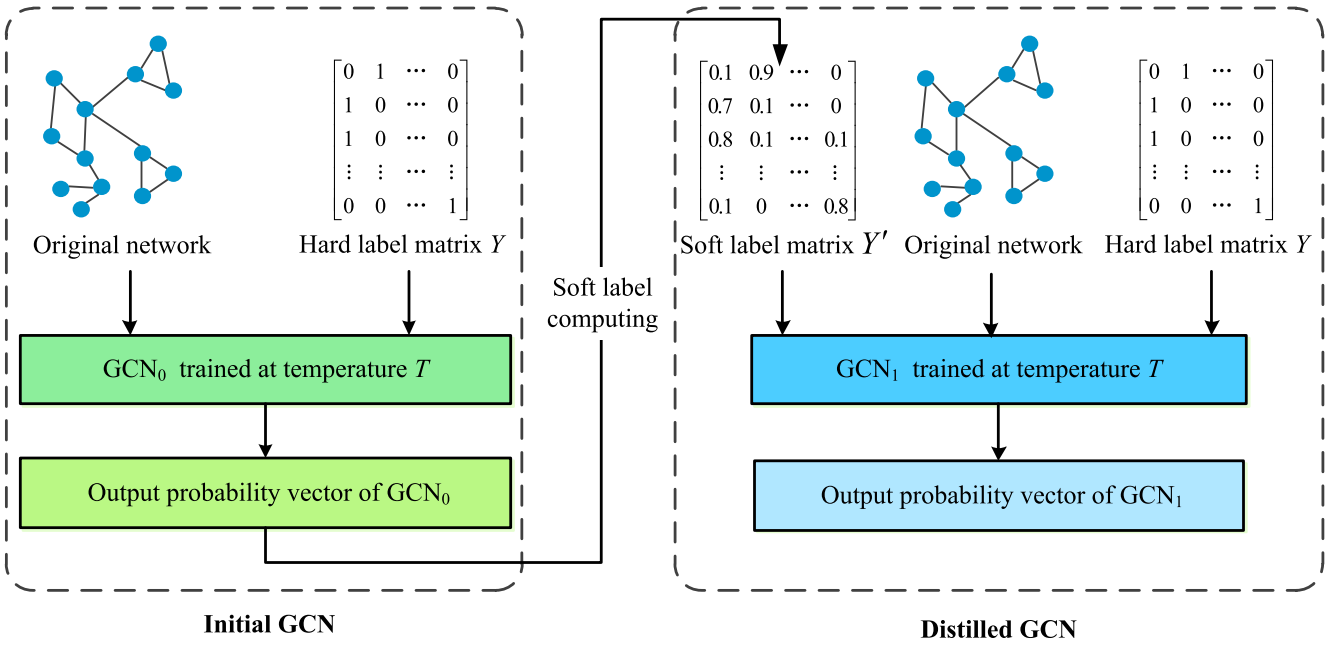

Fig. 1. Illustration of SD. In this model, we train the initial GCN with a softmax output layer at temperature T as first. Then, we encode training nodes by the soft label that is the output confidence of the trained initial GCN model. Finally, we train the distilled GCN with its training objective function that is composed of the soft loss function and original loss function.

where Y'' denotes the output of the distilled GCN model. Note that, in the soft loss L_s, we also use a softmax function with temperature T.

2) Smoothing Cross-Entropy: Inspired by the model regularization method [41], we also propose an SCEL function that encourages the GCN to return high confidence on the true label, while a smoothing confidence distribution on the false label for each node. The SCEL function is defined as

$$L_{\text{smooth}} = -\sum_{l=1}^{|S_{\text{train}}|}\sum_{k=1}^{|F|}\hat{Y}_{lk}\ln\left(Y'_{lk}\right) \tag{9}$$

where $\hat{Y}$ is a smoothing matrix with $\hat{Y}_{lk} = 1$ if node v_l belongs to category τ_k and $\hat{Y}_{lk} = (1/|F|)$ otherwise, $F = [\tau_1, \ldots, \tau_{|F|}]$ is the category set for the nodes in the network, $|F|$ denotes the number of categories, and Y' is the output of the model calculated by (3).

In Sections III-C and III-D, we have proposed two types of defense methods: adversarial training and smoothing strategies. Here, we analyze their difference and relationship. Adversarial training and smoothing strategies protect the model from different aspects. Adversarial training improves the robustness by fine-tuning the classification boundary with adversarial networks generated in prior so that it helps the classifier, i.e., GCN, to defend against the attacks. Smoothing strategies hide the gradient information of the GCN model by setting a gradient mask, which makes it difficult for hackers to grasp the precise gradient information. Consequently, both defenses protect the model and complement each other. Furthermore, we can conclude that adversarial training usually focuses on adjusting the classification boundary with unstable classification performance, that is, it will not adjust each classification boundary at the same time. In contrast, smoothing strategies are designed to reduce the probability that the target node is wrongly categorized, which is performed simultaneously. Consequently, we believe that adversarial training and smoothing strategies protect the GCN model from different aspects. They both improve the robustness of the GCN model by fine-tuning the model's weight parameters. Therefore, we could design a more effective defense method by combining them. The experiments also show their contributions to improving model robustness.

E. Time Complexity and Memory Consumption

In this section, we analyze the time complexity and memory consumption of Global-AT, Target-AT, SD, and SCEL and compare them with the baseline method, GraphDefense [36]. For time complexity, Global-AT and Target-AT strategies consist of two stages: adversarial network generation and adversarial training. Since the complexity of adversarial network generation is dependent on the attack methods, in the experiment, we use NETTACK [24] and FGA [23] to generate adversarial networks, whose time complexity can be represented as $O(m \times |V|^2)$, where m is the number of attack iterations, and $|V|$ indicates the number of nodes in the network. More specifically, Global-AT and Target-AT adopt different training data and training mechanisms; for instance, their attack iterations are different, denoted by m_G and m_T, respectively. For adversarial training, its time complexity is mainly determined by the number of epochs. We use ϕ_{train} to represent the time complexity of 200 epochs training. Therefore, the total time complexity of Global-AT and Target-AT are $O(m_G \times |V|^2 + \phi_{\text{train}})$ and $O(m_T \times |V|^2 + \phi_{\text{train}})$, respectively. For SD and SCEL, they only need to smooth the gradient information by training their GCN models. However, in order

TABLE I
TIME COMPLEXITY AND MEMORY CONSUMPTION OF DIFFERENT NETWORK DEFENSE METHODS

Methods	Time complexity	Memory consumption
Global-AT	$O\left(m_G \times \lvert V\rvert^2 + \phi_{train}\right)$	$O\left(\lvert V\rvert^2 + \varphi_{model}\right)$
Target-AT	$O\left(m_T \times \lvert V\rvert^2 + \phi_{train}\right)$	$O\left(\lvert V\rvert^2 + \varphi_{model}\right)$
SD	$O\left(2 \times \phi_{train}\right)$	$O\left(2 \times \varphi_{model}\right)$
SCEL	$O\left(2 \times \phi_{train}\right)$	$O\left(\varphi_{model}\right)$
GraphDefense	$O\left(m_D \times \left(\lvert V\rvert^2 + \phi_{train}\right)\right)$	$O\left(\lvert V\rvert^2 + \varphi_{model}\right)$

TABLE II
BASIC STATISTICS OF THE THREE NETWORK DATA SETS

Dataset	#Nodes	#Links	#Classes	#Train/Validation/Test
Pol.Blogs	1,490	19,090	2	121/121/1,248
Cora	2,708	5,429	7	267/267/2,174
Citeseer	3,312	4,732	6	330/330/2,652

to obtain a defensive model, they require more training epochs. In the experiment, their numbers of training epochs are set as twice that of Global-AT and Target-AT, which means that their time complexity is $O(2 \times \phi_{\text{train}})$. GraphDefense is an adversarial training-based method, and it achieves adversarial attack and adversarial training iteratively, whose time complexity is $O(m_D \times (|V|^2 + \phi_{\text{train}}))$.

For memory consumption, we use φ_{model} to represent the memory cost occupied by the GCN model. Compared with SCEL, Global-AT, Target-AT, and GraphDefense need memory for adversarial network storage, while SD needs twice the memory cost because of its complex model structure.

The specific time complexity and memory consumption of these defense methods are presented in Table I.

IV. EXPERIMENTS

In order to testify the effectiveness of our SAT method, we use it to defend against two types of attack methods by performing a number of experiments for node classification and community detection. Our experimental environment consists of i7-7700K 3.5-GHz × 8 (CPU), TITAN Xp 12 GiB (GPU), 16 GB × 4 memory (DDR4), and Ubuntu 16.04 (OS).

A. Data Sets

The following three networks are used in our experiments, with their basic statistics summarized in Table II.

1) *Pol.Blogs:* The Pol.Blogs network is compiled by Adamic and Glance [42]. This network is about political leaning, which is collected from blog directories. The blogs are divided into two classes. The links between blogs were automatically extracted from the front pages of the blogs. It contains 1490 blogs and 19 090 links in total.
2) *Cora:* This network contains a number of machine-learning papers of seven classes [43]. The links between papers represent the citation relationships. It contains 2708 papers and 5429 links in total.
3) *Citeseer:* Citeseer is also a paper citation network with the papers divided into six classes [43]. It contains 3312 papers and 4732 citation links in total.

B. Attack Methods and Metrics of Defense

To testify the defense capacity of SAT, we adopt FGA [23] and NETTACK [24] as the adversarial attacks. FGA is a gradient-based attack method, and NETTACK is a score-based attack method. They are two typical adversarial attack methods; both of them achieve a good attack success rate (ASR) on GNN models. Therefore, we choose them to generate adversarial networks for SAT. We first evaluate the performance of the two adversarial network attacks using average ASR on three data sets. ASR is defined as

$$\text{ASR} = \frac{n_{\text{suc}}}{n_{\text{atk}}} \times 100\% \tag{10}$$

where n_{suc} is the number of misclassified nodes after attack and n_{atk} is the total number of attacked nodes. The higher ASR indicates the better attack results.

FGA and NETTACK are briefly described as follows.

1) *FGA [23]:* It utilizes the iterative gradient information of pairwise nodes to generate an adversarial network, which is based on a trained GCN model. Here, we only consider changing the links around the target nodes.
2) *NETTACK [24]:* It generates adversarial network iteratively. In each iteration, first, it selects candidate links based on the degree distribution; then, it defines a scoring function for each link, meaning the confidence loss of the target node in the GCN model when the link is changed; and finally, it utilizes the scores of the candidate links to update the adversarial network.

We use the following metrics to measure the defense effectiveness on the attacked node set, where the nodes are classified correctly before attacked in the test set.

1) *ADR:* It indicates the relative difference between the ASR of attack on GCN with and without defense. It can be calculated as

$$\text{ADR} = \text{ASR}_{\text{atk}} - \text{ASR}_{\text{def}} \tag{11}$$

where ASR_{atk} is the ASR without defense methods, and ASR_{def} is the ASR after defense methods. The higher ADR corresponds to the better defense effect.
2) *ACD:* The average confidence difference, i.e., the average difference between the confidences of the nodes in n_{suc} before and after attack, which is defined as follows:

$$ACD = \frac{1}{n_{\text{suc}}} \sum_{v_i \in N_s} CD_i(\hat{A}_{v_i}) - CD_i(A) \tag{12}$$

$$CD_i(A) = \max_{c \neq R_i} Y'_{i,c}(A) - Y'_{i,R_i}(A) \tag{13}$$

where $\hat{A}_{v_i}$ is the adversarial network of the target node v_i, R_i is the real label of the target node v_i, and Y' is the output of the model calculated by (3). The lower ACD corresponds to the better defense effect.

C. Experimental Settings

In this article, we conduct attack and defense experiments on a trained two-layer GCN model. For the GCN model, we set the number of hidden units $H = 16$, learning_rate $= 0.01$, drop_out $= 0.5$, weight_deacy $= 5 \times 10^{-4}$, and the training

TABLE III
DEFENSE RESULTS OF VARIOUS DEFENSE STRATEGIES AGAINST TWO ADVERSARIAL NETWORK ATTACKS ON MULTIPLE NETWORKS

Datasets	Attack	ASR(%)	ADR(%)				ACD			
			SCEL	SD	Global-AT	Target-AT	SCEL	SD	Global-AT	Target-AT
Polblogs	FGA	84.47	25.38	20.71	24.89	**62.41**	-0.309	-0.259	-0.299	**-0.631**
	NETTACK	77.54	**28.93**	17.36	24.15	22.39	-0.714	-0.744	-0.743	**-0.748**
	Average	81.01	27.16	19.04	24.52	**42.40**	-0.511	-0.502	-0.521	**-0.690**
Cora	FGA	98.21	31.07	12.37	28.59	**76.26**	0.019	-0.072	0.04	**-0.596**
	NETTACK	95.01	20.51	18.62	**21.84**	12.87	-0.013	0.017	**-0.030**	0.075
	Average	96.61	25.79	15.50	25.21	**44.57**	0.003	-0.027	0.005	**-0.261**
Citeseer	FGA	97.89	31.68	6.84	31.22	**72.31**	-0.001	-0.033	0.021	**-0.493**
	NETTACK	93.43	**5.72**	4.29	5.27	4.45	0.338	0.314	**0.319**	0.340
	Average	95.66	18.70	5.56	18.24	**38.38**	0.169	0.141	0.17	**-0.076**

epoch is 200. For FGA and NETTACK, we achieve global attacks by maximizing the negative cross-entropy loss of all training nodes and select 40 nodes for each data set to realize target attacks. Especially, these 40 nodes can be grouped into three categories: 1) ten nodes with the highest classification confidence; 2) ten nodes with the lowest classification confidence; and 3) 20 more nodes randomly. The results of the target attack and Target-AT are the average values obtained by these nodes. In NETTACK, we set its critical value for the likelihood ratio test of the power-law distributions $\Delta(A, A') = 0.004$. In FGA, for each attacked node v_i, we limit its number of modified links $\omega_i \leq 0.5d_i$, where d_i is the degree of v_i. The maximum number of attack iterations of FGA and NETTACK is $(1/2)|S_{\text{atk}}|d_{\text{ave}}$, where $|S_{\text{atk}}|$ indicates the number of attacked nodes and d_{ave} is the average degree of attacked nodes. Moreover, in each iteration of an adversarial attack, FGA and NETTACK only change one link.

For the proposed defense methods in this article, we implement adversarial training and smoothing strategies based on the trained GCN model. Therefore, during adversarial training and smoothing process, we maintain the original learning_rate, drop_out, and weight_deacy of GCN. In Global-AT and Target-AT, we set the number of adversarial training epochs to 200. For SD, considering the complexity of its model, we set its number of training epochs to 400. For SCEL, in order to make its loss converge stably, we also set its number of training epochs to 400.

D. Defense on Node Classification

1) Parameter Setting of T in Smoothing Distillation: First, we need to test the effect of T on the SD model. In this section, we use FGA and NETTACK as the attack baselines and calculate the ADR of three-node classification data sets under different T values. The results are shown in Fig. 2, where the value of T varies from 1/5 to 5. For the three data sets, their curves show a trend of increasing first and then decreasing as T increases. Because the too small value of T cannot play a good gradient smoothing, while the too large value of T will make smoothing excessive. For Pol.Blogs, the distillation model achieves the best defense effect when $T = 3$. For Cora and Citeseer, when $T = 2$, the defense effect is the best. Therefore, in the following experiments, we set $T = 3$ for Pol.Blogs and $T = 2$ for Cora and Citeseer.

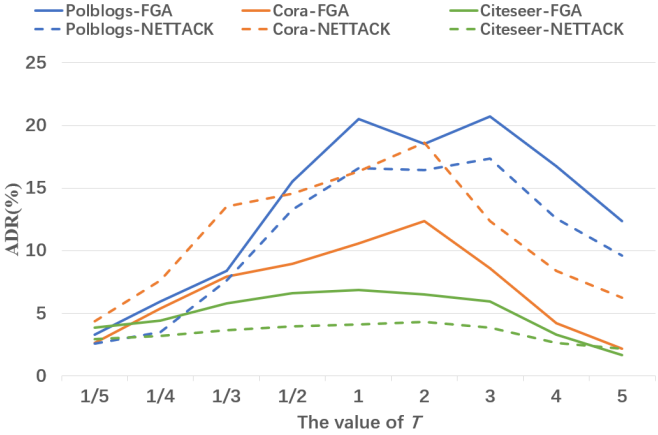

Fig. 2. ADR curves of three data sets under different T values in the SD model.

2) Defense Effect of Proposed Methods: In this section, we test the defense capabilities of Global-AT, Target-AT, SD, and SCEL against FGA and NETTACK. These algorithms are briefly described as follows.

1) *Global-AT:* It first uses attack methods to generate the adversarial networks based on all the training nodes and then utilizes the generated adversarial networks to achieve adversarial training.
2) *Target-AT:* Similar to Global-AT, Target-AT also needs to generate the adversarial networks and achieve adversarial training, but, unlike Global-AT, Target-AT generates the adversarial networks only based on a certain type of nodes.
3) *SD:* It calculates the soft label for each node and uses the soft label instead of the ground truth to train the classifier.
4) *SCEL:* It encourages the GCN to return high confidence on the true label. It trains the GCN model by a smoothing confidence distribution on the false label for each node.

The attack results of FGA and NETTACK and defense results of SCEL, SD, Global-AT, and Target-AT are shown in Table III. It can be seen that FGA and NETTACK achieve good attack results with high ASR. In defense, Target-AT outperforms other defense strategies in most cases, which achieves the highest ADR and the lowest ACD. Global-AT

TABLE IV

DEFENSE RESULTS OF VARIOUS SAT STRATEGIES AND GRAPHDEFENSE AGAINST TWO ADVERSARIAL NETWORK ATTACKS ON MULTIPLE NETWORKS

Datasets	Attack	ASR(%)	ADR(%)					ACD				
			G-SCEL	G-SD	T-SCEL	T-SD	GraphDefense	G-SCEL	G-SD	T-SCEL	T-SD	GraphDefense
Polblogs	FGA	84.47	35.28	29.37	**72.54**	67.41	23.54	-0.624	-0.547	**-0.758**	-0.579	-0.235
	NETTACK	77.54	37.64	31.54	**42.37**	38.65	25.97	-0.796	-0.684	**-0.846**	-0.745	-0.628
	Average	81.01	36.46	30.46	**57.46**	53.03	24.76	-0.710	-0.616	**-0.802**	-0.662	-0.432
Cora	FGA	98.21	42.32	33.03	**84.37**	79.58	28.40	-0.147	-0.129	**-0.703**	-0.687	-0.073
	NETTACK	95.01	**30.47**	27.53	25.74	25.13	26.57	-0.036	-0.003	**-0.128**	-0.087	-0.054
	Average	96.61	36.40	30.28	**55.06**	52.36	27.49	-0.092	-0.066	**-0.416**	-0.387	-0.064
Citeseer	FGA	97.89	39.57	32.17	**82.33**	78.57	23.82	-0.028	-0.004	**-0.472**	-0.423	0.012
	NETTACK	93.43	11.73	10.54	8.73	9.24	**12.21**	0.137	0.122	0.056	0.037	**-0.032**
	Average	95.66	25.65	21.36	**45.53**	43.91	18.02	0.055	0.059	**-0.208**	-0.193	-0.010

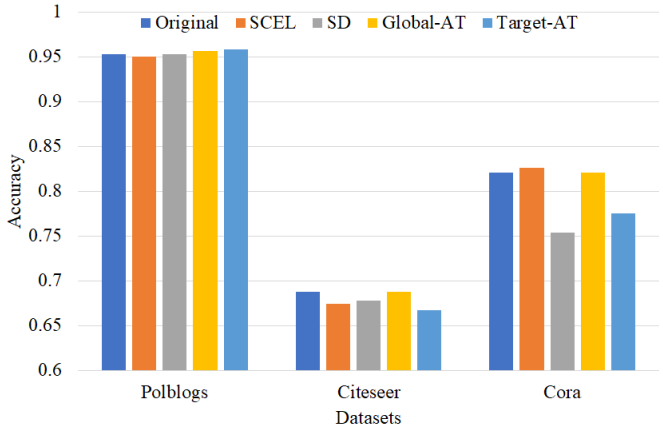

Fig. 3. Classification accuracy of the GCN model with different defense strategies on three network data sets.

has the same defense effect as SCEL. SD has a relatively low defense effect. Compared with Global-AT, Target-AT only needs to perform adversarial training on a certain type of nodes, which is not affected by other nodes in adversarial training. For two smoothing strategies, SCEL and SD, since SD needs to generate a soft label for each node, when some soft labels are wrong, the defense effect of the distillation model will be affected. In addition, compared with NETTACK, these defense methods achieve better defense effects on FGA. Because FGA directly attacks the GCN model according to its gradient information, NETTACK considers the concealment of the disturbance while attacking the GCN model. Therefore, although the ASR of FGA is higher, the adversarial attack of FGA is also easier to be defended. In addition, in order to test the impact of these defense methods for model classification accuracy, we also give comparison between the classification results of the GCN model trained with and without different defense strategies, as shown in Fig. 3. We find that, in most cases, the performance of GCN trained with most of the defense strategies is comparable with the performance of GCN trained without defense. However, the performance of GCN with SD and Target-AT seems poorer on the Cora network data set because Cora has the most classes of the three data sets, which causes SD to achieve oversmoothing to the GCN model. Target-AT enhances the robustness of the target class nodes while reducing the classification accuracy of other nodes.

TABLE V

BASIC STATISTICS OF THE TWO COMMUNITY NETWORK DATA SETS

Dataset	#Nodes	#Links	#Classes	#Train/Validation/Test
PolBook	105	441	3	13/12/80
Dolphin	62	159	2	10/10/42

Moreover, we pairwisely combine two adversarial training strategies and two smoothing strategies and get four types of SAT methods: G-SCEL (Global-AT + SCEL), G-SD (Global-AT + SD), T-SCEL (Target-AT + SCEL), and T-SD (Target-AT + SD). The training epochs of these methods are all set to 400. To verify the performance of these SAT methods, we compare them with the state-of-the-art adversarial training method: GraphDefense [36]. The description of GraphDefense is given as follows.

1) *GraphDefense [36]:* GraphDefense uses the GCN model to obtain the confidence of the unlabeled nodes set and constructs the adversarial training loss function by using both labeled and unlabeled nodes.

The ADR and ACD of these methods are shown in Table IV, where we can see that combining adversarial training methods and smoothing strategies can, indeed, enhance the defense effect. This might be because adversarial training and smoothing strategies could complement each other, i.e., the former plays a defensive role by adjusting the classification boundary, while the latter constructs gradient masks to improve the robustness of the model. For the four SAT strategies in Table IV, T-SCEL achieves the highest ADR and the lowest ACD. Compared with SD, SCEL can better enhance two types of adversarial training methods, which means that the loss function of SCEL can prevent adversarial attacks more effectively. Compared with GraphDefense, these SAT strategies provide a full range of defense against the GCN model, achieving higher ADR and lower ACD.

3) Running Time Comparison of Different Defense Methods: In addition, we also display the specific running time of defense methods on three node classification data sets, as shown in Fig. 4. It can be seen that it takes a longer running time to defend against the attacks when the network consists of more nodes because large network data sets require more time to train the GCN model and generate adversarial networks. Especially, the running time of Global-AT and GraphDefense is significantly higher than other defense methods because

TABLE VI
DEFENSE RESULTS OF VARIOUS DEFENSE STRATEGIES AGAINST TWO ADVERSARIAL NETWORK ATTACKS ON COMMUNITY NETWORKS

Datasets	Attack	Model	ASR(%)	ADR(%)								
				SCEL	SD	Global-AT	Target-AT	G-SCEL	G-SD	T-SCEL	T-SD	GraphDefense
PolBook	FGA	DeepWalk	93.55	23.20	42.40	54.00	69.04	62.37	72.58	78.64	**82.07**	61.03
		node2vec	92.06	47.77	27.60	34.71	61.74	67.36	54.91	**83.21**	72.68	54.44
		Louvain	90.48	70.50	29.67	14.26	57.84	74.62	41.08	**86.97**	68.52	64.97
		Average	92.03	47.16	33.22	34.32	62.87	68.12	56.19	**82.94**	74.42	60.15
	NETTACK	DeepWalk	87.06	47.69	13.85	32.00	68.92	65.43	41.07	**79.36**	72.05	65.08
		node2vec	86.30	54.67	32.01	13.60	54.14	59.52	40.83	**80.33**	76.37	49.87
		Louvain	84.12	48.84	43.89	24.92	53.50	66.37	65.48	74.66	**77.84**	52.19
		Average	85.83	50.40	29.92	23.51	58.85	63.77	49.13	**78.12**	75.42	55.72
Dolphins	FGA	DeepWalk	100.00	21.92	41.44	50.00	70.00	63.74	76.58	78.65	**82.41**	65.86
		node2vec	100.00	40.00	20.00	38.52	71.10	64.76	52.63	**84.57**	79.32	63.47
		Louvain	98.69	54.44	55.54	11.12	33.80	57.69	55.43	**68.57**	65.33	42.58
		Average	99.56	38.79	38.99	33.21	58.30	62.06	61.55	**77.26**	75.69	5730
	NETTACK	DeepWalk	91.05	48.16	9.12	32.64	63.68	67.32	36.46	**72.38**	66.07	46.53
		node2vec	90.00	47.19	4.29	32.34	60.20	63.28	33.52	**81.06**	58.96	57.73
		Louvain	93.26	48.84	43.89	24.92	63.50	60.54	52.61	**81.36**	78.83	53.68
		Average	91.44	48.06	19.10	29.97	62.46	63.71	4086	**78.27**	67.95	52.65

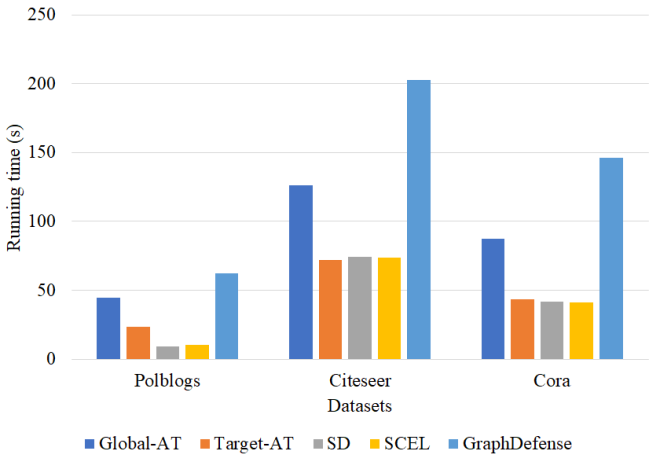

Fig. 4. Running time comparison of different defenses on three network data sets.

Global-AT needs to launch the attack iteratively when generating adversarial networks, and GraphDefense requires multiple numbers of adversarial trainings of GCN model. For SD and SCEL, they share the same number of training epochs, so their running time is basically the same. For Target-AT, due to the dense connection of Polblogs, it needs to modify a large number of links in adversarial network generation process, which makes its running time higher than SD and SCEL. To successfully attack Citeseer and Cora, the number of links that need to be modified is greatly reduced. Therefore, the running time of Target-AT is equivalent to that of SD and SCEL.

E. Defense on Community Detection

We further use community detection methods [44]–[46] to detect target nodes in the adversarial networks that are generated by the GCN model with or without defense. In particular, we use the following network embedding methods and Leuven [47] to detect target nodes in the adversarial network.

1) *DeepWalk [48]:* It first transforms the network into node sequences by random walk and then uses it as input to the Skip-Gram model to learn representations.
2) *node2vec [49]:* It develops a second-order biased random walk procedure to explore the neighborhood of a node, which can strike a balance between local and global properties of a network.
3) *Louvain [47]:* It is a modularity-based method that can generate a hierarchical community structure by compressing the communities.

We use the following two data sets in this experiment. Their basic statistics are summarized in Table V.

1) *PloBook:* This network represents copurchasing of books about U.S. politics sold by the online bookseller [50]. In this network, nodes represent books about U.S. politics sold by the online bookseller Amazon.com, and two nodes are connected if the corresponding two books were copurchased by the same buyers. There are 105 books and 441 links in total, and the books belong to three communities: liberal, neutral, and conservative.
2) *Dolphins:* This social network represents the common interactions observed between a group of dolphins in a community living off Doubtful Sound, New Zealand [51]. There are 62 dolphins and 159 links in total, and the dolphins are partitioned into two groups by the temporary disappearance of dolphin numbers.

The attack and defense results on community detection are presented in Table VI, where we can see that the results are quite consistent with those in node classification, i.e., for each community detection method on each data set, T-SCEL outperforms the other defense strategies in most cases. The combination of adversarial training and smoothing strategies can obtain better defense results.

To better show the defense effect of various strategies, in this part, we visualize the networks generated by the attack with and without defense and their representations, as shown in Figs. 5 and 6, respectively. Particularly, in most cases, since the T-SCEL model performs better than other

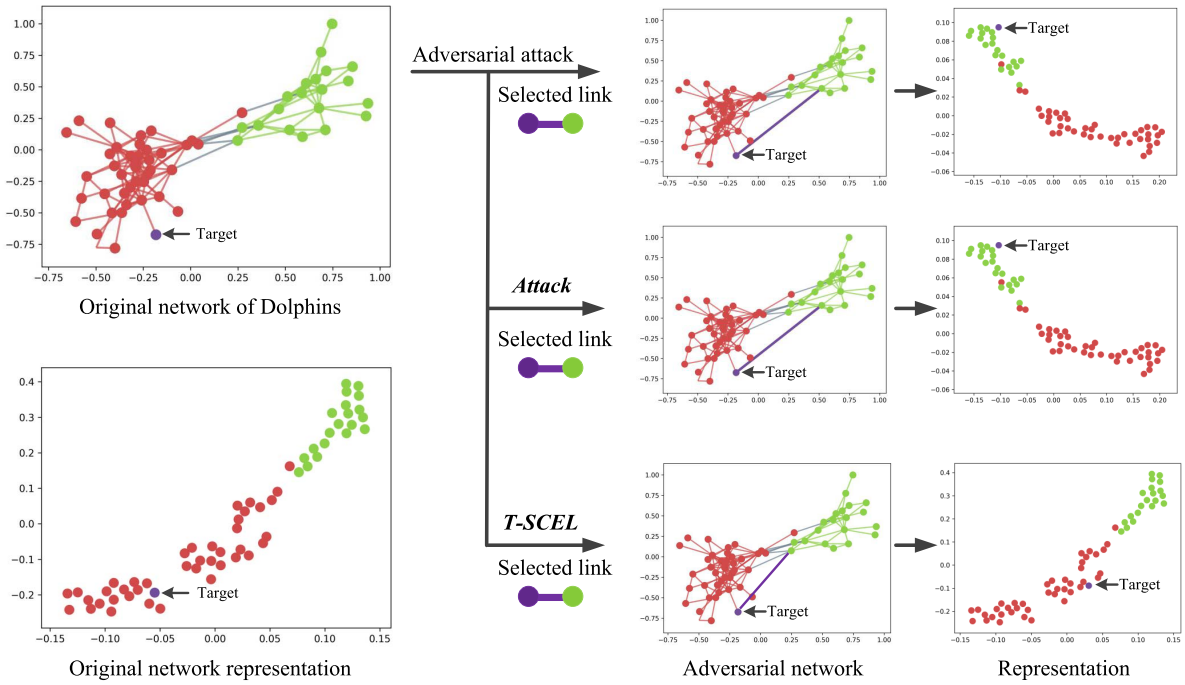

Fig. 5. Visualization of FGA under different defense strategies on network embedding of a random target node in Dolphins. The purple node represents the target node, and the purple link is selected by our FGA due to its largest gradient. Except for the target node, the nodes of the same color belong to the same community before the attack.

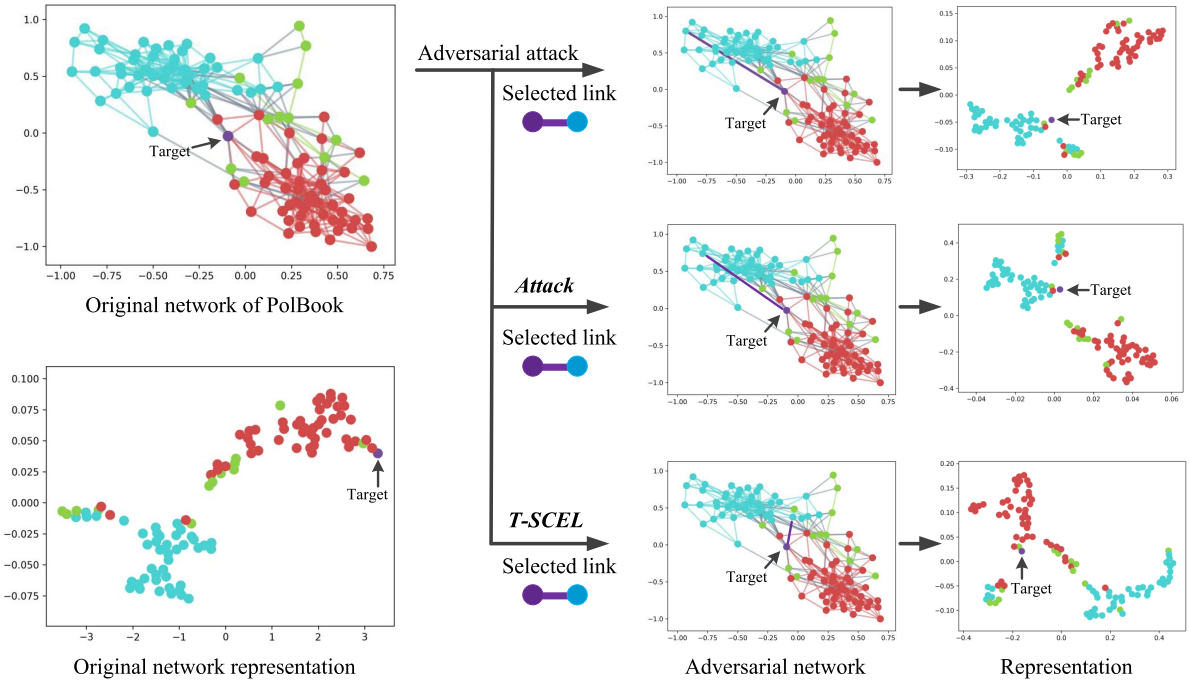

Fig. 6. Visualization of FGA under different defense strategies on network embedding of a random target node in PolBook. The purple node represents the target node, and the purple link is selected by our FGA due to its largest gradient. Except for the target node, the nodes of the same color belong to the same community before the attack.

strategies, we only show the result of T-SCEL in this part. Here, we use the low-dimensional network representations learned by DeepWalk [48] as the input to the visualization tool t-SNE [52]. We find that the embedding vector of the target node changes significantly even when only one link is changed in each network. Moreover, T-SCEL achieves effective defense ability against FGA.

V. Conclusion

In this article, we focus on defense against adversarial attacks on graphs/networks and propose an SAT method for the GCN model, which consists of two parts: adversarial training and gradient smoothing. In adversarial training, we design two different adversarial training strategies for different defense objects, which can achieve better defense

results for important nodes. In gradient smoothing, we propose a distillation architecture for the GCN model and design an SCEL function to train this model. Both of them can reduce the amplitude of adversarial gradients, which makes gradient-based attacks ineffective. In experiments, we verified the performance of the proposed defense strategies, and the results show that adversarial training and gradient smoothing perform good defensibility against different attack methods in each real-world data set.

Studying the robustness of GNNs for networks is an important problem. Our future work includes developing more effective defense algorithms against adversarial network attacks and trying to propose robust network embedding methods that are more robust to adversarial network attacks.

REFERENCES

[1] E. Dumbill, "A revolution that will transform how we live, work, and think: An interview with the authors of big data," *Big Data*, vol. 1, no. 2, pp. 73–77, 2013.

[2] H. Wang *et al.*, "Graphgan: Graph representation learning with generative adversarial nets," 2017, *arXiv:1711.08267*. [Online]. Available: https://arxiv.org/abs/1711.08267

[3] Y. Liu, C. Yang, Z. Gao, and Y. Yao, "Ensemble deep kernel learning with application to quality prediction in industrial polymerization processes," *Chemometric Intell. Lab. Syst.*, vol. 174, pp. 15–21, Mar. 2018.

[4] Q. Xuan *et al.*, "Automatic pearl classification machine based on a multistream convolutional neural network," *IEEE Trans. Ind. Electron.*, vol. 65, no. 8, pp. 6538–6547, Aug. 2018.

[5] J. Chen, K. Hu, Y. Yang, Y. Liu, and Q. Xuan, "Collective transfer learning for defect prediction," *Neurocomputing*, vol. 416, pp. 103–116, Nov. 2020.

[6] Y. Liu, K. Liu, J. Yang, and Y. Yao, "Spatial-neighborhood manifold learning for nondestructive testing of defects in polymer composites," *IEEE Trans. Ind. Informat.*, vol. 16, no. 7, pp. 4639–4649, Jul. 2020.

[7] Q. Xuan, Z. Chen, Y. Liu, H. Huang, G. Bao, and D. Zhang, "Multiview generative adversarial network and its application in pearl classification," *IEEE Trans. Ind. Electron.*, vol. 66, no. 10, pp. 8244–8252, Oct. 2019.

[8] J. Chen *et al.*, "E-LSTM-D: A deep learning framework for dynamic network link prediction," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Aug. 22, 2019, doi: 10.1109/TSMC.2019.2932913.

[9] C. Fu *et al.*, "Link weight prediction using supervised learning methods and its application to yelp layered network," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 8, pp. 1507–1518, Aug. 2018.

[10] J. Tang, M. Qu, and Q. Mei, "PTE: Predictive text embedding through large-scale heterogeneous text networks," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2015, pp. 1165–1174.

[11] S. Wang, J. Tang, C. Aggarwal, and H. Liu, "Linked document embedding for classification," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2016, pp. 115–124.

[12] Q. Xuan *et al.*, "Subgraph networks with application to structural feature space expansion," *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 5, 2020, doi: 10.1109/TKDE.2019.2957755.

[13] F. Tian, B. Gao, Q. Cui, E. Chen, and T. Y. Liu, "Learning deep representations for graph clustering," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1293–1299.

[14] L. Liu, W. K. Cheung, X. Li, and L. Liao, "Aligning users across social networks using network embedding," in *Proc. IJCAI*, 2016, pp. 1774–1780.

[15] H. Dai *et al.*, "Adversarial attack on graph structured data," 2018, *arXiv:1806.02371*. [Online]. Available: http://arxiv.org/abs/1806.02371

[16] S. Yu *et al.*, "Target defense against link-prediction-based attacks via evolutionary perturbations," *IEEE Trans. Knowl. Data Eng.*, early access, Aug. 8, 2019, doi: 10.1109/TKDE.2019.2933833.

[17] J. Chen *et al.*, "GA-based Q-attack on community detection," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 3, pp. 491–503, Jun. 2019.

[18] V. Krebs, "Mapping networks of terrorist cells," *Connections*, vol. 24, no. 3, pp. 43–52, 2002.

[19] K. Sun, Z. Lin, H. Guo, and Z. Zhu, "Virtual adversarial training on graph convolutional networks in node classification," in *Proc. Chin. Conf. Pattern Recognit. Comput. Vis. (PRCV)*. Cham, Switzerland: Springer, 2019, pp. 431–443.

[20] B. A. Miller, M. Çamurcu, A. J. Gomez, K. Chan, and T. Eliassi-Rad, "Improving robustness to attacks against vertex classification," in *Proc. MLG Workshop*, 2019, pp. 1–8.

[21] D. Zügner and S. Günnemann, "Certifiable robustness and robust training for graph convolutional networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 246–256.

[22] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907

[23] J. Chen, Y. Wu, X. Xu, Y. Chen, H. Zheng, and Q. Xuan, "Fast gradient attack on network embedding," 2018, *arXiv:1809.02797*. [Online]. Available: http://arxiv.org/abs/1809.02797

[24] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, London, U.K., Aug. 2019, pp. 2847–2856.

[25] F. Scarselli, M. Gori, A. Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[26] F. Scarselli, M. Gori, A. Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "Computational capabilities of graph neural networks," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 81–102, Jan. 2009.

[27] T. Pham, T. Tran, D. Q. Phung, and S. Venkatesh, "Column networks for collective classification," in *Proc. AAAI*, 2017, pp. 2485–2491.

[28] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*. [Online]. Available: http://arxiv.org/abs/1611.07308

[29] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5115–5124.

[30] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.

[31] D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," 2019, *arXiv:1902.08412*. [Online]. Available: http://arxiv.org/abs/1902.08412

[32] A. Bojchevski and S. Günnemann, "Adversarial attacks on node embeddings via graph poisoning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 695–704.

[33] K. Xu *et al.*, "Topology attack and defense for graph neural networks: An optimization perspective," 2019, *arXiv:1906.04214*. [Online]. Available: http://arxiv.org/abs/1906.04214

[34] H. Wu, C. Wang, Y. Tyshetskiy, A. Docherty, K. Lu, and L. Zhu, "Adversarial examples on graph data: Deep insights into attack and defense," 2019, *arXiv:1903.01610*. [Online]. Available: http://arxiv.org/abs/1903.01610

[35] H. Chang *et al.*, "A restricted black-box adversarial framework towards attacking graph embedding models," 2019, *arXiv:1908.01297*. [Online]. Available: http://arxiv.org/abs/1908.01297

[36] X. Wang, X. Liu, and C.-J. Hsieh, "GraphDefense: Towards robust graph convolutional networks," 2019, *arXiv:1911.04429*. [Online]. Available: http://arxiv.org/abs/1911.04429

[37] D. Zhu, Z. Zhang, P. Cui, and W. Zhu, "Robust graph convolutional networks against adversarial attacks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 1399–1407.

[38] C. Szegedy *et al.*, "Intriguing properties of neural networks," in *Proc. ICLR*, Canada, Apr. 2014, pp. 803–813.

[39] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*. [Online]. Available: http://arxiv.org/abs/1412.6572

[40] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*. [Online]. Available: http://arxiv.org/abs/1503.02531

[41] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[42] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 U.S. election: Divided they blog," in *Proc. 3rd Int. Workshop Link Discovery (LinkKDD)*, 2005, pp. 36–43.

[43] A. K. Mccallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of Internet portals with machine learning," *Inf. Retr.*, vol. 3, no. 2, pp. 127–163, 2000.

[44] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, "Learning community embedding with community detection and node embedding on graphs," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 377–386.
[45] V. W. Zheng, S. Cavallari, H. Cai, K. C.-C. Chang, and E. Cambria, "From node embedding to community embedding," 2016, *arXiv:1610.09950*. [Online]. Available: http://arxiv.org/abs/1610.09950
[46] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Proc. AAAI*, 2017, pp. 203–209.
[47] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008.
[48] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 701–710.
[49] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. SIGKDD*, 2016, pp. 855–864.
[50] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci. USA*, vol. 103, no. 23, pp. 8577–8582, 2006.
[51] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," *Behav. Ecol. Sociobiol.*, vol. 54, no. 4, pp. 396–405, Sep. 2003.
[52] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

Jinyin Chen received the B.S. and Ph.D. degrees from the Zhejiang University of Technology, Hangzhou, China, in 2004 and 2009, respectively.

She is currently an Associate Professor with the College of Information Engineering, Zhejiang University of Technology. Her research interests cover intelligent computing, social network data mining, optimization, and network security.

Xiang Lin is currently pursuing the master's degree with the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China.

His research interests include data mining and applications, and network analysis.

Hui Xiong is currently pursuing the master's degree with the College of Information Engineering, Zhejiang University of Technology, Hangzhou, China.

His research interests are image processing and artificial intelligence.

Yangyang Wu received the bachelor's degree from the Ningbo Institute of Technology, Zhejiang University, Hangzhou, China, in 2016. He is currently pursuing the master's degree with the Institute of Information Engineering, Zhejiang University of Technology, Hangzhou.

His research interest covers data mining and applications, social network data mining, and clustering analysis.

Haibin Zheng received the bachelor's degree from the Zhejiang University of Technology, Hangzhou, China, in 2017, where he is currently pursuing the master's degree with the College of Information Engineering.

His research interests include data mining and applications, and bioinformatics.

Qi Xuan (Member, IEEE) received the B.S. and Ph.D. degrees in control theory and engineering from Zhejiang University, Hangzhou, China, in 2003 and 2008, respectively.

He was a Post-Doctoral Researcher with the Department of Information Science and Electronic Engineering, Zhejiang University, from 2008 to 2010, and a Research Assistant with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, in 2010 and 2017. From 2012 to 2014, he was a Post-Doctoral Fellow with the Department of Computer Science, University of California at Davis, Davis, CA, USA. He is currently a Professor with the College of Information Engineering, Institute of Cyberspace Security, Zhejiang University of Technology, Hangzhou. His current research interests include network science, graph data mining, cyberspace security, machine learning, and computer vision.