

Quantifying the Noise of Structural Perturbations on Graph Adversarial Attacks

JUNYUAN FANG, Department of Electrical Engineering, City University of Hong Kong, China

HAN YANG, School of Computer Science and Engineering, Sun Yat-sen University, China

HAIXIAN WEN, School of Computer Science and Engineering, Sun Yat-sen University, China

JIAJING WU*, School of Software Engineering, Sun Yat-sen University, China

ZIBIN ZHENG, School of Software Engineering, Sun Yat-sen University, China

CHI K. TSE, Department of Electrical Engineering, City University of Hong Kong, China

Graph neural networks have been widely utilized to solve graph-related tasks because of their strong learning power in utilizing the local information of neighbors. However, recent studies on graph adversarial attacks have proven that current graph neural networks are not robust against malicious attacks. Yet much of the existing work has focused on the optimization objective based on attack performance to obtain (near) optimal perturbations, but paid less attention to the strength quantification of each perturbation such as the injection of a particular node/link, which makes the choice of perturbations a black-box model that lacks interpretability. In this work, we propose the concept of noise to quantify the attack strength of each adversarial link. Furthermore, we propose three attack strategies based on the defined noise and classification margins in terms of single and multiple steps optimization. Extensive experiments conducted on benchmark datasets against three representative graph neural networks demonstrate the effectiveness of the proposed attack strategies. Particularly, we also investigate the preferred patterns of effective adversarial perturbations by analyzing the corresponding properties of the selected perturbation nodes.

CCS Concepts: • **Computing methodologies** → **Neural networks**; • **Security and privacy** → **Network security**.

Additional Key Words and Phrases: Graph Neural Networks, Graph Adversarial Attacks, Robustness, Noise Propagation

ACM Reference Format:

Junyuan Fang, Han Yang, Haixian Wen, Jiajing Wu, Zibin Zheng, and Chi K. Tse. 2018. Quantifying the Noise of Structural Perturbations on Graph Adversarial Attacks. *J. ACM* 37, 4, Article 111 (August 2018), 22 pages. <https://doi.org/XXXXXXXX.XXXXXXX>

*Corresponding author.

Authors' Contact Information: Junyuan Fang, Department of Electrical Engineering, City University of Hong Kong, Hong Kong SAR, China, junyufang2-c@my.cityu.edu.hk; Han Yang, School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China, yangh396@mail2.sysu.edu.cn; Haixian Wen, School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China, wenhx6@mail2.sysu.edu.cn; Jiajing Wu, School of Software Engineering, Sun Yat-sen University, Zhuhai, China, wujiajing@mail.sysu.edu.cn; Zibin Zheng, School of Software Engineering, Sun Yat-sen University, Zhuhai, China, zhizbin@mail.sysu.edu.cn; Chi K. Tse, Department of Electrical Engineering, City University of Hong Kong, Hong Kong SAR, China, chitse@cityu.edu.hk.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM 1557-735X/2018/8-ART111

<https://doi.org/XXXXXXXX.XXXXXXX>

1 Introduction

Graphs, where nodes indicate the entities and links represent the corresponding relationships between entities, have been widely employed to characterize the interaction of various social systems. In recent years, due to the strong learning power of the neighborhood aggregation mechanism, graph neural networks (GNNs) have obtained great success on various graph-related downstream tasks, such as node classification, link prediction, and community detection [7, 11, 25, 26, 28].

Although GNNs perform well in these graph-based tasks, recent studies [5, 18] pointed out that current GNNs are not robust to carefully designed attacks (i.e., adversarial attacks). In other words, GNNs may be easily deceived after being injected with some small perturbations into the data, such as modifying the node features, adding or removing the links, injecting fake nodes, etc. Therefore, a series of works focusing on graph adversarial attacks has been proposed to further evaluate the robustness of current GNNs from different perspectives [4, 6, 19, 20, 24, 32].

Most existing adversarial attack methods focus on the optimization objective based on final attack performance by utilizing gradient information [4, 15], classification margins [6], reinforcement learning (RL) algorithm [6, 19], etc., to obtain (near) optimal perturbations. Yet these methods pay less attention to the strength quantification of each perturbation, such as the injection of a particular node/link into the clean graph, which makes the choice of perturbations a black-box model that lacks interpretability. Therefore, we want to identify the contributions of adversarial links and further uncover the selection preference of the optimal perturbation of graph adversarial attacks.

In this work, we first propose the concept of *noise* to quantify the attack strength of each adversarial link, and then propose three simple yet effective attack methods based on this noise concept to further evaluate the robustness of current GNNs. Specifically, following the classic work [6] in the graph adversarial attacks, we focus on the scenario of targeted attacks on the node classification task, which means the attack goal is to make GNNs predict the targeted node to the wrong classes. Based on neighborhood aggregations of GNNs, we theoretically propose and discuss the noise of different adversarial links. In addition, we formulate a reasonable noise function to quantify the noise of each adversarial link on the structural attacks. We then refine the adversarial links based on their rankings of noise value. According to the defined noise function and the traditional classification margins, we then propose three different attack strategies from the perspectives of both single step and multiple steps optimization, namely noise-based greedy attacks (**NGA**), noise and margin-based attacks (**NMA**), and noise and margin-based attacks-boost (**NMAB**). Finally, we verify the attack performance of the proposed methods on the benchmark datasets against three representative GNNs. Moreover, we also measure the properties of the adversarial links from several perspectives, including their class, degree, homophily ratio [8, 16, 30], etc. The major contributions of this work are summarized as follows.

- (1) **New Concept.** We theoretically analyze the perturbation of adversarial links from the perspective of noise propagation based on the neighborhood aggregation mechanism of GNNs, and then present the concept of *noise* to quantify the perturbation strength of each adversarial link on neighborhood aggregations.
- (2) **New Methods.** Based on the defined noise function and classification margins of adversarial links, we then propose three simple yet effective attack strategies by considering both the single step and multiple steps optimizations.
- (3) **Comprehensive Experiments.** We conducted extensive experiments on three benchmark datasets against representative GNNs to verify the superiority of the proposed methods. Moreover, we further analyze the properties and patterns of the chosen adversarial links from several perspectives.

2 Related Work

2.1 Graph Neural Network Models

Graph neural networks (GNNs) have obtained great success on different graph-related tasks by utilizing the message passing mechanism. Through the neighborhood aggregations of GNNs, each node in the graph can utilize the information of its neighbors to further characterize itself. A series of well-known GNNs have been proposed in the past few years. For example, the graph convolution network (GCN) [13] achieved the feature propagation via a first-order approximation of localized spectral filters on graphs. GraphSAGE [10] utilized neighborhood sampling mechanism and proposed more advanced aggregators to improve the scalability of GNNs. Graph attention network (GAT) [21] assigned different weights to different neighbors adaptively via a self-attention mechanism. Simplified graph convolution network (SGC) [23] further reduced the complexity of the original GCN by removing the non-linear activation functions in the middle aggregation layers.

2.2 Categories of Graph Adversarial Attacks

Despite the strong potential shown on various downstream tasks, recent GNNs tend to be vulnerable to the particularly designed perturbations, which are called graph adversarial attacks. Based on the claim from previous studies [5, 18], graph adversarial attacks can be divided into graph modification attacks and node injection attacks from the perspective of attack operations. The former assumes that the attackers can directly modify the original graphs by changing features or flipping links, while the latter considers that the attackers prefer to inject some fake nodes into the original graph. Moreover, from the perspective of attack goals, graph adversarial attacks can be divided into targeted attacks and global attacks, indicating that the attackers will focus on attacking a single node and the overall performance of all nodes each time, respectively. In addition, based on different attack stages, graph adversarial attacks can also be divided into evasion attacks (i.e., test time) which evaluate the attack performance on the pre-trained GNNs, and poisoning attacks (i.e., training time) which evaluate the attack performance on the retrained GNNs.

As the main objective of this work is to investigate the specific difference of the adversarial links in influencing the neighborhood aggregation mechanism of GNNs, we will focus on the *graph modification attacks* and *targeted evasion attacks* on the node classification task. In other words, our goal is to make GNNs to predict the targeted node to the wrong class via structural perturbations in the testing phase.

2.3 Existing Methods of Graph Adversarial Attacks

Zügner *et al.* [6] first pointed out that traditional GNNs can be easily deceived through small unnoticeable perturbations by proposing NETTACK method by selecting the adversarial links causing the largest classification margins. After that, a bunch of works have been proposed to investigate the robustness of GNNs. For instance, Dai *et al.* [6] proposed GradArgmax via flipping the corresponding link with the largest magnitude of gradients. Chen *et al.* [4] presented another gradient-based attack method, FGA, by adding/removing the corresponding valid link with the largest absolute gradient value. Zhang *et al.* [27] utilized the cross entropy to denote the similarity of different nodes, then heuristically connect/disconnect the corresponding node pairs. Li *et al.* [15] proposed SGA to achieve the gradient-based attacks by leveraging subgraph to reduce the time and space of gradient calculation. Geisler *et al.* [9] developed an attack strategy by adopting the sparsity-aware first-order optimization attacks and a novel surrogate loss to improve the scalability of attacks. Zhu *et al.* [29] introduced the partial graph attack strategy by adopting a hierarchical selection policy to select the vulnerable nodes as the attack targets and a cost-effective anchor-picking policy to pick the most promising anchor nodes for modifying edges. Recently, Alom *et al.* [1] proposed a

Table 1. Main notations in this paper.

Notation	Definition
G	Graph
V	Node set
E	Link set
X, x_u	Node feature matrix, feature vector of node u
Y	Label set
$\mathcal{N}(u)$	Neighbor set of node u including itself
f_θ	GNN model
$h_u^{(k)}$	Representation of node u in the k -th layer
Δ	Attack budget, i.e., number of links to be added
$\mathcal{L}_{\text{train}}$	Training loss of GNN models
\mathcal{L}_{atk}	Attack loss of the adversaries
$\widetilde{LN}(u, v)$	Link noise for link (u, v)
$\text{DIS}(u, v)$	Dissimilarity between nodes u and v
$\widetilde{LN}(u, v)$	Appropriate link noise for link (u, v)

subtle and effective attack method, GOttack, by manipulating the graph orbit vector of each node. Besides, there are also a series of strong attack methods from other optimization perspectives, such as the reinforcement learning algorithms [19], meta-gradients [33], evolutionary algorithms [3], etc. However, few of them have analyzed the specific difference between different adversarial links, especially for targeted attacks. Therefore, we want to bridge this gap and quantitatively evaluate the difference of structural perturbations, then propose powerful attack strategies from the perspective of noise propagation on neighborhood aggregations.

3 Preliminaries

3.1 Graph Neural Networks

A graph can be denoted as $G = (V, E, X)$ where $V = \{v_1, v_2, \dots, v_n\}$ represents the node set with n nodes and $E = \{e_1, e_2, \dots, e_m\}$ indicates the link set with m links. $X \in \mathbb{R}^{n \times d}$ represents the feature matrix where each node has a d -dimension feature vector. We summarize the main notations used in this paper in Table 1.

The details of a general semi-supervised node classification task [13] are as follows. Assuming the n nodes belong to the label set $Y = \{0, 1, \dots, C - 1\}$ with C different classes, we will first train a GNN model f_θ based on the labeled/training nodes $V_{\text{train}} \subset V$ by minimizing a training loss $\mathcal{L}_{\text{train}}$, and then utilize f_θ to predict the possible class y_i of remaining/test nodes $\bar{V} = \{v_i | v_i \in V \setminus V_{\text{train}}\}$. Specifically, a k -layer GCN can be formally defined as follows.

$$h_u^{(k)} = \sigma(W^{(k)} \cdot \text{Agg}(h_v^{(k-1)} | v \in \mathcal{N}(u)), \quad (1)$$

where $h_u^{(k)}$ indicates the representation of node u in the k -th layer, $\mathcal{N}(u)$ is the neighbors of node u including self-loop (i.e., $u \in \mathcal{N}(u)$). σ is the activation function such as ReLU in the middle layers or softmax in the last layer, and $W^{(k)}$ is the learning parameter in the k -th layer. Moreover, $\text{Agg}(\cdot)$ is the aggregation function that combines the representation of neighboring nodes in the prior layer, which is as follows.

$$\text{Agg}(h_v^{(k-1)} | v \in \mathcal{N}(u)) = \sum_{v \in \mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}_u| \cdot |\mathcal{N}_v|}} \cdot h_v^{(k-1)}, \quad (2)$$

where $|\mathcal{N}_u|$ and $|\mathcal{N}_v|$ represent the number of neighbors of nodes u and v , respectively. Therefore, term $\frac{1}{\sqrt{|\mathcal{N}_u| \cdot |\mathcal{N}_v|}}$ can be considered as the weight or contribution of each neighbor v to the future representations of the central node u .

The learning objective of the node classification task is to minimize the training loss (e.g., entropy loss), which is as follows.

$$\min_{\theta} \mathcal{L}_{\text{train}} = - \sum_{v \in V_{\text{train}}} \ln Z_{v,c}, \quad Z = f_{\theta}(G), \quad (3)$$

where $Z \in \mathbb{R}^{n \times C}$ is the output of the last layer of GNN model f_{θ} , $Z_{v,c}$ indicates the probability that node v belongs to the true class c .

3.2 Graph Adversarial Attacks

As we mentioned before, the goal of node classification is to obtain the optimal model f_{θ^*} by minimizing $\mathcal{L}_{\text{train}}$. Graph adversarial attacks, on the contrary, aim to fool a GNN model f_{θ} by generating a new adversarial graph G' via injecting some small perturbations into the original graph G under a given budget Δ , where the budget Δ represents the number of links can be flipped in structural attacks. More specifically, the goal of graph adversarial attacks can be formulated as follows.

$$\begin{aligned} \min \mathcal{L}_{\text{atk}}(f_{\theta'}(G')) \\ \text{s.t. } |G' - G| \leq \Delta, \end{aligned} \quad (4)$$

where $f_{\theta'}$ represents the GNN model being attacked, which will be either trained on the clean graph G in the evasion attacks (i.e., this work) or trained on the perturbed graph G' in the poisoning attacks. \mathcal{L}_{atk} represents the attack loss of the adversaries. As we focus on the targeted evasion attacks, we define the loss of the target node u as the classification margin (CM) between the predicted probabilities after and before the attacks, which can be given as follows.

$$\mathcal{L}_{\text{atk}}(u) = f_{\theta'}(G')_{u,c} - f_{\theta'}(G)_{u,c}. \quad (5)$$

We try to minimize the CM between the probabilities that node u be predicted to the ground truth label c after and before the adversarial attacks under the same GNN model $f_{\theta'}$. Obviously that $\text{CM} \in [-1, 1]$, and a smaller CM indicates a better attack performance.

4 Proposed Methods

In this section, we first introduce the definition of the noise of adversarial links, and then give the details of the proposed three attack methods based on noise propagation.

4.1 Noise Propagation

In the graph adversarial attacks, the key point to mislead the target node to a wrong class is to influence the neighborhood aggregations of GNNs. For example, the attackers can add some fake links between the target node and the nodes that may be harmful to the future representation of the target node. In addition to adding links, the attackers also can remove some original links between the target node and their neighbors. As previous studies [2, 12] have shown that adding link attacks would be more powerful than removing link attacks, we only consider adding link attacks in this work for simplicity.

Typically, we assume that our neighbors will help obtain a better representation via neighborhood aggregations. However, *what will happen if the information from our neighbors is harmful to the target node?* Recall to (2), we can simply consider that the term $\frac{1}{\sqrt{|\mathcal{N}_u| \cdot |\mathcal{N}_v|}}$ (*Weight* for short) as the aggregated weight of neighbor node v to the central node u . Particularly, in the following, we define the potential node v that chose to connect with the target node u as the **adversarial node**, and the corresponding link (u, v) as the **adversarial link**. As we only consider adding link attacks, the corresponding adversarial link (u, v) should not exist in the clean graph. Moreover, we focus on the noise propagation on homophilic networks where most of the original neighbors of each node belong to the same class because traditional GNNs, such as GCN, usually cannot perform well on heterophilic networks, and corresponding heterophilic GNNs usually employ some new aggregation designs. Thus, the findings in this work maybe cannot be directly extended to the heterophilic networks, and we leave it for future work.

In the targeted attacks, for a specific target node u , if the aggregation between the target node u and each possible adversarial node v only contains noise and the specific noise value of each adversarial link is the same, we can have the following proposition based on (2).

PROPOSITION 4.1. *Let $G = (A, X, E)$ be a simple graph, and $Y = \{0, 1, \dots, C - 1\}$ be the possible label. We simplify the feature of each node to be a one-hot vector corresponding to its label, denoted as $\mu(Y)$. Namely, the feature vector of node u is $x_u = \mu(Y_u)$. Assuming that most of the original neighbors of each node belong to the same class, and the specific noise value of each adversarial link is the same. Consider a one-layer GCN where the output of node u is $h_u = \sigma(W \cdot \sum_{v \in \mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}_u| \cdot |\mathcal{N}_v|}} \cdot x_v)$, σ is the softmax activation function, we have the following.*

- (1) *From the perspective of target nodes, nodes with a lower degree will be easier to be attacked than those with a higher degree.*
- (2) *From the perspective of adversarial nodes, nodes with a lower degree will influence the representation of the target node more than those with a higher degree.*

PROOF. See Appendix A.1 for the detailed proof. □

we can further explain Proposition 4.1 as follows. Intuitively, a target node with a higher degree will have a larger value of $|\mathcal{N}_u|$, thus leading to the fact that each neighbor will only contribute a small part in the neighborhood aggregation procedure as each of them has a small value of *Weight*. Therefore, if we only inject a small budget of adversarial links, it is unlikely to attack the node (i.e., influence the node representations) having a high degree successfully.

From another perspective, for attacking a specific target node u , $|\mathcal{N}_u|$ is the same for all adversarial links, and thus a lower $|\mathcal{N}_v|$ indicates a much larger *Weight*. Therefore, these kinds of neighbors will contribute more to the target node in the aggregation procedure, indicating that nodes with a lower degree will be more aggressive than the nodes with a higher degree under this assumption (i.e., each adversarial link has the same noise value).

However, the above observations are obtained based on the assumption that each adversarial link has the same noise value, but obviously the information from neighbors is more complicated, each link will have a different noise value on the propagation. Therefore, we have the further proposition as follows.

PROPOSITION 4.2. *Except for the specific noise value of adversarial links varying from each other, we let all of the other assumptions be the same as Proposition 4.1. Then, we have the following. For a specific target node u , if the adversarial nodes have the same degree, the adversarial nodes which are dissimilar to node u influence the aggregation of node u more than those similar to node u .*

PROOF. See Appendix A.2 for the detailed proof. □

Based on proposition 4.2, to better quantify the specific noise value of each adversarial link on the neighborhood aggregation process of target node u , we then propose a metric *link noise* (LN) considering both degree and similarity, which is defined as follows.

$$\text{LN}(u, v) = \frac{1}{\sqrt{(|\mathcal{N}_u| + 1) \cdot (|\mathcal{N}_v| + 1)}} \cdot \text{DIS}(u, v), \quad (u, v) \notin E, \quad (6)$$

where the first term represents the latest *Weight* of the adversarial node v or the adversarial link (u, v) . As we only focus on adding link attacks, the degree of both u and v will increase by one. The second term $\text{DIS}(u, v)$ indicates the dissimilarity between the target node u and corresponding adversarial node v , or in other words, the total noise between nodes u and v on aggregations. Based on (2), we know only $\frac{1}{\sqrt{(|\mathcal{N}_u|+1) \cdot (|\mathcal{N}_v|+1)}}$ of the total noise will be aggregated and further influence the future representation of node u . Referring to previous work [14, 27], we utilize the entropy of corresponding representations of nodes obtained from GNNs to characterize the noise or dissimilarity between the target node and the adversarial nodes, which is as follows.

$$\text{DIS}(u, v) = - \sum_{i=0}^{C-1} (h_u^{(k)})_i \log(h_v^{(k)})_i, \quad (u, v) \notin E, \quad (7)$$

where $(h_u^{(k)})_i$ is the i -th hidden representation of GNN model of node u in the k -th layer. Actually, $(h_u^{(k)})_i$ is the probability that node u belongs to class $i \in Y$ as if the total layer of the corresponding GNN is k . Particularly, since the attackers cannot exactly know the specific GNN model that is being attacked, we utilize the representation of classic GCN [13] as the surrogate model in this work. From (7), we can find that a higher similarity between the target node u and adversarial node v indicates a lower dissimilarity $\text{DIS}(u, v)$.

Combining (6) and (7), for a specific target node u , we can simplify the calculation by removing the same term as the target node u is the same for all adversarial links. Therefore, we further define *appropriate link noise* ($\widetilde{\text{LN}}$) as follows.

$$\widetilde{\text{LN}}(u, v) = \frac{- \sum_{i=0}^{C-1} (h_u^{(k)})_i \log(h_v^{(k)})_i}{\sqrt{|\mathcal{N}_v| + 1}}. \quad (8)$$

Based on the above analysis, we can intuitively consider that a specific adversarial link (u, v) with a higher $\widetilde{\text{LN}}$ will influence the neighborhood aggregations of target node u more than those with smaller $\widetilde{\text{LN}}$. Therefore, it can be regarded as an importance measure of adversarial links. In the following, we further propose three attack strategies considering the proposed *appropriate link noise* metric.

4.2 Noise-Based Greedy Attacks

As analyzed above, $\widetilde{\text{LN}}(u, v)$ indicates the noise value of potential adversarial links connecting u and v . Therefore, we first propose a simple noise-based greedy attack method (NGA) by directly utilizing the noise value of each valid adversarial link. Specifically, we greedily add Δ valid adversarial links with the highest noise to the target node u . As the intuitive idea is that adversarial links with higher noise will negatively influence the neighborhood aggregation of the target node on a larger scale, we believe this greedy attack method will also achieve considerable performance.

4.3 Noise and Margin-Based Attacks

Although the noise value of adversarial links can reflect the potential harmfulness of future aggregations to some extent, greedily adding the corresponding adversarial links based on the noise value may not yield the best attack performance. As NGA only utilizes the original noise value from the

Algorithm 1 Noise and Margin-based Attack (NMA)**Input:** Original graph $G = (V, E, X)$, target node u , attack budget Δ , size of candidates δ_1 .**Output:** Generated adversarial graph $G' = (V, E', X)$.

- 1: Train a surrogate GCN model f_θ^s based on original G .
- 2: $Z \leftarrow$ Record the output of the last layer of model f_θ^s for all nodes.
- 3: $\widetilde{\text{LN}} \leftarrow$ Calculate $\widetilde{\text{LN}}(u, v)$ via Z for all valid adversarial links (u, v) based on (8).
- 4: $\widetilde{\text{LN}} \leftarrow$ Sort the obtained $\widetilde{\text{LN}}$ in descending order.
- 5: $\hat{E} \leftarrow$ Transform the sorted $\widetilde{\text{LN}}$ to the corresponding link list.
- 6: $E_{\text{cand}} \leftarrow$ Construct the candidates by only retaining the top valid δ_1 links with the highest noise, i.e., $\hat{E}[0 : \delta_1 - 1]$.
- 7: $E' \leftarrow E$.
- 8: **for each** $i \in \{0, \dots, \Delta - 1\}$ **do**
- 9: Calculate \mathcal{L}_{atk} for all valid links in E_{cand} based on (5), respectively.
- 10: $e \leftarrow$ Select the valid link with the minimal \mathcal{L}_{atk} .
- 11: $E' = E' \cup e$.
- 12: **end for**
- 13: **return** $G' = (V, E', X)$.

Table 2. The top 10 attack sequences ordered by CM of a target node (i.e. id = 422) on Cora dataset where the attack budgets are from 1 to 3. We simplify the adversarial links as the combination of the adversarial nodes as they all have the same target node. The boldfaced results indicate that the corresponding sequence exists in the top 10 sequences on the prior budget.

Index	$\Delta = 1$		$\Delta = 2$		$\Delta = 3$	
	Sequence	CM	Sequence	CM	Sequence	CM
1	1669	-0.3245	1669 , 2167	-0.4459	1669 , 2167 , 2259	-0.4742
2	2167	-0.3202	1669 , 2259	-0.4452	1669 , 2168, 2259	-0.4739
3	2259	-0.3181	1669 , 2168	-0.4443	1281, 1669 , 2167	-0.4733
4	2168	-0.3135	2167 , 2259	-0.4439	1281, 1669 , 2259	-0.4733
5	1281	-0.3057	2168, 2259	-0.4424	1281, 1669 , 2168	-0.4731
6	234	-0.2908	1281, 1669	-0.4417	1669 , 2167 , 2168	-0.4730
7	697	-0.2899	1281, 2167	-0.4401	1281, 2167 , 2259	-0.4728
8	535	-0.2563	1281, 2259	-0.4397	1281, 2168 , 2259	-0.4725
9	342	-0.2548	1281, 2168	-0.4388	1669 , 2167 , 234	-0.4725
10	2404	-0.2527	2167 , 2168	-0.4383	2167 , 2168, 2259	-0.4724

clean graph, it ignores the change of the latest noise value after the injection of some links. Therefore, combining the targeted attack loss in (5), we propose a noise and classification margin-based attack (NMA) strategy, in which the major steps are similar to NETTACK [32]. However, NETTACK needs to take all possible adversarial links into consideration, while the proposed NMA only retains a small ratio of links with the strong attack effect (i.e., higher noise value) to be the final candidates.

Specifically, we first utilize the ranking of $\widetilde{\text{LN}}$ to control the size of final candidates, as the links with higher noise tend to mislead the target node more in the neighborhood aggregations. The detailed procedure of NMA is given in Algorithm 1. We only reserve the top δ_1 valid links with higher noise to be the final candidates before we move into the margin calculation step (i.e., line 6). By employing

Algorithm 2 Noise and Margin-based Attack-Boost (NMAB)

Input: Original graph $G = (V, E, X)$, target node u , attack budget Δ , size of candidates δ_2 , size of single optimal list len_{sin} , size of retain list len_{re} .

Output: Generated adversarial graph $G' = (V, E', X)$.

- 1: Train a surrogate GCN model f_θ^s based on original G .
- 2: $Z \leftarrow$ Record the output of the last layer of model f_θ^s for all nodes.
- 3: $\widetilde{\text{LN}} \leftarrow$ Calculate $\widetilde{\text{LN}}(u, v)$ via Z for all valid adversarial links (u, v) based on (8).
- 4: $\widetilde{\text{LN}} \leftarrow$ Sort the obtained $\widetilde{\text{LN}}$ in descending order.
- 5: $\hat{E} \leftarrow$ Transform the sorted $\widetilde{\text{LN}}$ to the corresponding link list.
- 6: $E_{\text{cand}} \leftarrow$ Construct the candidates by only retaining the top δ_2 valid links with the highest noise, i.e., $\hat{E}[0 : \delta_2 - 1]$.
- 7: Calculate \mathcal{L}_{atk} for all links in E_{cand} based on (5), respectively.
- 8: $E_{\text{sin}} \leftarrow$ Record the top len_{sin} links which have the highest \mathcal{L}_{atk} among all δ_2 candidate links.
- 9: $E_{\text{optimal}} \leftarrow$ Record the top len_{re} links which have the highest \mathcal{L}_{atk} among all δ_2 candidate links.
- 10: **for each** $i \in \{1, \dots, \Delta - 1\}$ **do**
- 11: $E_{\text{current}} \leftarrow$ Construct the valid $(i + 1)$ -length attack list by combining the optimal (1) -length attack list E_{sin} and the optimal (i) -length attack list E_{optimal} .
- 12: Calculate \mathcal{L}_{atk} for all link sequences in E_{current} based on (5), respectively.
- 13: $E_{\text{optimal}} \leftarrow$ Select the top len_{re} attack sequence with the minimal \mathcal{L}_{atk} .
- 14: **end for**
- 15: $E^* \leftarrow$ Select the sequence with the minimal \mathcal{L}_{atk} in E_{optimal} as the optimal attack sequence.
- 16: $E' = E \cup E^*$.
- 17: **return** $G' = (V, E', X)$.

the above candidate refining mechanism, the searching space is greatly reduced as we only need to evaluate a small part of link perturbations.

4.4 Noise and Margin-Based Attacks-Boost

In the above, we propose NMA by taking the noise value of links as a candidate selection metric, and further utilising the classification margin as the final indicator. But similar to NETTACK, our NMA also focuses on single step optimization as we will select the current optimal adversarial link decreasing the classification margin the most during each attack budget. As a result, we may ignore other adversarial link combinations which have more powerful attack performance in a multiple steps optimization perspective. Based on the traditional robustness evaluation study of complex networked systems [31], an interesting finding is that the powerful $(i + 1)$ -length attack sequences usually contain the strong (i) -length attack sequences. It is also straightforward that the combination of strong sequences will lead to a better attack performance.

To investigate whether this phenomenon exists in our problem, we conduct a similar empirical study on a specific target node (i.e., id = 422) of the Cora dataset, whose statistics will be given in Section 5.1. Particularly, to reduce the full searching space, we only consider the adversarial links in NMA that are controlled by the size of candidates δ_1 as the valid links for simplicity. By utilizing the classification margin as the evaluation metric, we observe that this phenomenon also exists in our task, as shown in Table 2. For example, the optimal (2) -link attack sequence (i.e., {1669, 2167}) contains the optimal (1) -link attack sequence (i.e., {1669}), the optimal (3) -link attack sequence (i.e., {1669, 2167, 2259}) contains the optimal (2) -link attack attack sequence (i.e., {1669, 2167}), etc.

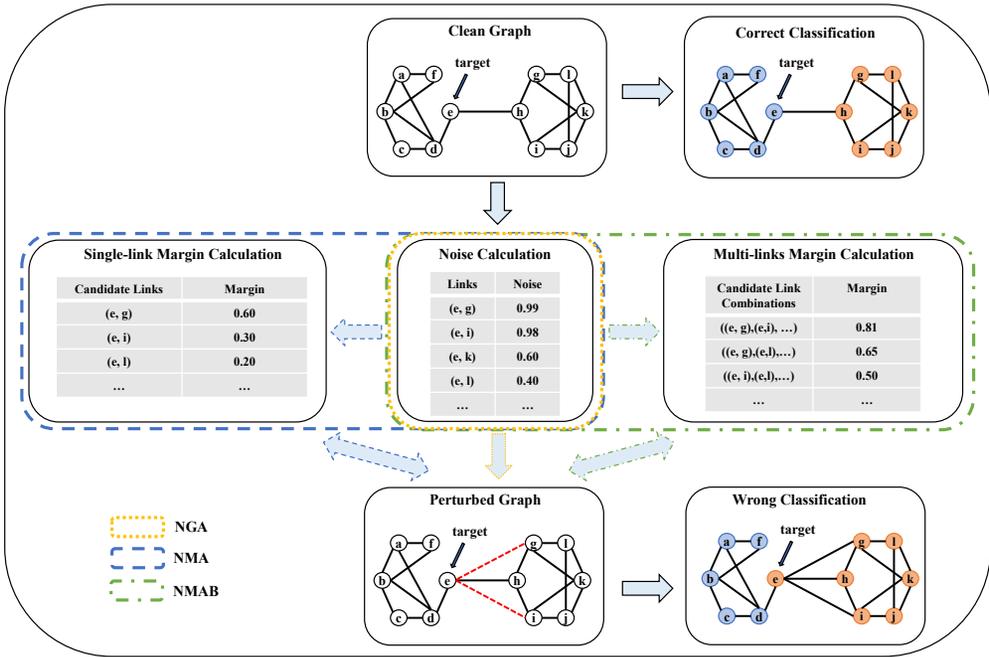


Fig. 1. A systematic framework of targeted attacks based on the proposed three strategies.

Therefore, we know that the strong attack sequences in prior steps can help search for more powerful attack sequences in the next step, which can be considered a multiple steps optimization process.

Therefore, following the above findings, we propose an improved strategy, NMAB, to boost the attack performance of NMA. The major steps of NMAB are similar to NMA, as shown in Algorithm 2. The only difference is, NMAB will not greedily select the adversarial link with the lowest CM during each step, but maintain two optimal lists with pre-defined sizes including the optimal (1)-link attack sequence and optimal (i)-link attack sequence (i.e., lines 8-14). Specifically, E_{sin} represents the optimal (1)-link attack sequence with the size of len_{sin} , and E_{optimal} indicates the optimal (i)-link attack sequence with the size of len_{re} . For each attack budget, we will construct the current attack candidates (i.e., sequence length = $i + 1$) by combining all the valid combinations of list E_{optimal} (i.e., sequence length = i) and list E_{sin} (i.e., sequence length = 1). Finally, we will return the optimal attack sequence in the (Δ)-length attack list (i.e., E^*) to generate the optimal adversarial links. It is worth noting that, compared to NMA which focuses on selecting the current optimal link during each attack budget from a local perspective, NMAB can select the optimal link(s) in multiple attack budgets from a more global perspective.

4.5 Overall Framework

To sum up, based on the proposed noise of nodes, we propose three attack strategies including NGA, NMA, and NMAB from different perspectives. The overall framework of the attack procedure based on the proposed three strategies is given in Fig. 1. In the targeted attack scenario, we will first select a node among all possible nodes as the target node, and then employ the specific strategies (i.e., NGA, NMA, or NMAB) to generate the optimal adversarial links. Finally, we will evaluate the attack

Table 3. Attack success rate of the baselines against GNNs on three datasets. The last column indicates the average rank of each baseline among different models and datasets. The top 3 results in each column and the best rank are boldfaced.

Methods	Cora			Citeseer			Pubmed			Ranks
	GCN	SGC	GAT	GCN	SGC	GAT	GCN	SGC	GAT	
NETTACK	0.9044	0.9278	0.7388	0.8344	0.8614	0.6496	0.9684	0.9874	0.9114	2.22
FGA	0.9030	0.8184	0.5472	0.8842	0.8526	0.5156	0.9772	0.9630	0.9014	3.56
SGA	0.8540	0.8242	0.5866	0.6686	0.6722	0.4708	0.9604	0.9546	0.8510	4.89
NGA (Ours)	0.7468	0.6746	0.4626	0.6222	0.6854	0.6022	0.9072	0.8746	0.7426	5.67
NMA (Ours)	0.9360	0.8738	0.6082	0.9342	0.8782	0.6368	0.9802	0.9536	0.8872	3.00
NMAB (Ours)	0.9676	0.9166	0.6580	0.9532	0.9074	0.6702	0.9842	0.9598	0.8954	1.67

performance via the GNN model trained on the clean graph by checking whether the target node will be predicted to a wrong class.

5 Experiments

In this section, we conduct various experiments to verify the effect of the proposed three attack strategies. To be specific, we first introduce the statistical details of the datasets. Then, we demonstrate the baselines for comparisons and the GNNs being attacked. Following that, we introduce the specific parameter settings of models and methods. Finally, we present the experimental results together with the corresponding analysis. The code for reproduction will be publicly available at <https://github.com/alexfanjn/Noise-propagation-attack>, depending on the acceptance.

5.1 Datasets

We evaluated the proposed strategies on three representative benchmark datasets including Cora, Citeseer, and Pubmed [17]. Following the settings in the previous study [6], we extracted the largest connected component of them, and the statistical information is given in Table 4.

5.2 Baselines

We chose three recent methods as baselines to compare with the proposed NGA, NMA, and NMAB. The details are as follows.

- (1) **NETTACK** [32]: NETTACK is a strong baseline that attacks the structures and features based on classification margin. Since we focus on structure attacks, NETTACK is only allowed to attack the topological connections for fair comparison.
- (2) **FGA** [4]: FGA is a gradient-based targeted attack method. Based on the assumption that the adversarial links with the maximum absolute gradient will influence the loss function the most, FGA greedily chooses the corresponding links.
- (3) **SGA** [15]: SGA is another gradient-based targeted attack method by only considering the k -hop subgraph rather than the entire graph. Specifically, SGA largely reduces the size of the candidate set because the adding link operations will only occur when the nodes belong to the second possible class of the target node.

Among them, NETTACK and SGA used SGC as the surrogate model, while FGA and our methods adopted GCN as the surrogate model. Particularly, we do not compare our methods with other attack strategies such as GradArgmax [6], DICE [22], etc., as FGA and SGA already achieved a better attack performance than them.

Table 4. The statistics of datasets.

Datasets	#Nodes	#Links	#Features	#Classes	Avg. Degree
Cora	2,485	5,069	1,433	7	4.08
Citeseer	2,100	3,668	3,703	6	3.48
Pubmed	19,717	44,324	500	3	4.50

Table 5. Attack success rate of the proposed methods against GCN under different dissimilarity metrics on three datasets, where EUC, COS, and ENT refer to Euclidean distance, Cosine distance, and entropy distance, respectively. The best results in each method under the same dataset are boldfaced.

Methods	Cora			Citeseer			Pubmed		
	EUC	COS	ENT	EUC	COS	ENT	EUC	COS	ENT
NGA	0.679	0.778	0.781	0.451	0.666	0.647	0.721	0.749	0.879
NMA	0.865	0.869	0.948	0.807	0.821	0.914	0.894	0.901	0.976
NMAB	0.946	0.906	0.975	0.917	0.868	0.936	0.931	0.913	0.978

5.3 Targeted Models

For the GNN models to be attacked, we selected three traditional GNNs including GCN, SGC, and GAT as representatives. The details of these methods are as follows.

- (1) **GCN** [13]: GCN is a traditional GNN that obtains the low dimensional representation of nodes via aggregating the local structural and feature information of neighbors.
- (2) **SGC** [23]: SGC is the simplified version of GCN by removing the activation functions in the middle layers of GCN, which achieves comparable performance as GCN but requires smaller training complexity.
- (3) **GAT** [21]: GAT further improves the performance of the original GCN by proposing a masked self-attention mechanism. In this way, GAT gives different weights to different neighbors, rather than directly assigning the weight of each neighbor based on its degree information.

Particularly, as the proposed attack strategies were trained based on the surrogate GCN model in this work, the attack performance on other GNNs can show the generalization ability of our methods.

5.4 Parameter Settings

Each datasets is randomly split into the training set (10%), validation set (10%), and test set (80%). All of the experimental results are the average performance among 5 different splits. Moreover, all of the baselines and attacked GNNs are implemented by utilizing the opensource platform *DeepRobust* [12].

For the specific attacks, we randomly selected 1000 nodes in the test set as the target nodes for each dataset. To ensure the unnoticeable perturbations, we followed the common configuration of several studies [15, 32] by setting the attack budget Δ as the degree of target node for all attack methods. For NMA, we set the size of candidate δ_1 as 5Δ . For NMAB, we set the size of candidate δ_2 , the size of single optimal list len_{sin} , and size of retain list len_{re} as 10Δ , 3Δ , and 10, respectively. Except for the above settings, all other parameter settings of the baselines and attacked models are adopted to the default settings.

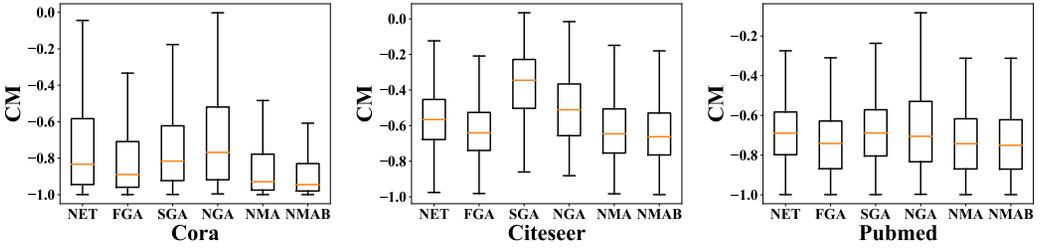


Fig. 2. Classification margin comparisons between different methods on three datasets.

5.5 Experimental Results and Analysis

5.5.1 Attack Success Rate. We first investigate the attack success rate of different methods under different datasets and models. Attack success rate refers to the ratio between the number of target nodes that are attacked successfully and the total number of target nodes. From Table 3, we can find that NGA obtains a good attack performance in some cases even though its idea is simple. For the proposed three strategies, NMAB performs the best, NMA is better than NETTACK in most cases, and both NMA and NMAB are better than NGA to a large extent. We think the slight gaps between NMA and NETTACK in some cases are acceptable as NMA has a much smaller size of candidates than NETTACK, so NETTACK will reasonably have more chances to obtain better solutions. For the comparisons with other baselines based on the overall ranks (i.e., the last column of Table 3), NMAB obtains the overall best attack success rate over all other methods, indicating the superiority of its multiple steps optimization. Following NMAB, NMA obtains a comparable performance as NETTACK, since they rank third and second place, respectively. The above results suggest that, even though NETTACK has a larger candidate set than NMAB, NETTACK may drop into local optimums due to its single step optimization. Moreover, although we utilize GCN as the surrogate model, the generated adversarial links also achieve remarkable attack performance on both SGC and GAT models, showing the generalizability of the proposed methods.

5.5.2 Classification Margin. We also analyze the classification margin (CM) obtained by each baseline. Fig. 2 is the box plot of the corresponding classification margins of the GCN model on three datasets. We can find that NMA and NMAB are the best two strategies that obtain the optimal CM than all of the other baselines, and NMAB is slightly better than NMA. More importantly, the worst cases (i.e., highest CM) of NMA and NMAB are also better than other strategies mostly. The above findings demonstrate that the proposed strategies can obtain a comparable or even better attack performance than current methods even though the size of candidate adversarial links has been largely reduced via the rankings of noise values, especially for NMA and NMAB.

5.5.3 Impact of Different Dissimilarity Metrics. In the default settings, we adopt entropy to measure the dissimilarity of different node pairs. To further investigate the impact of different dissimilarity metrics, we employ two other classic distance metrics including Euclidean distance and Cosine distance, to characterize the difference of the representation vectors of nodes. Specifically, we replace the calculation of entropy in (7) to the specific calculation of Euclidean distance and Cosine distance during the candidate selection process in the proposed methods. Then we analyze the corresponding attack success rate under different dissimilarity measurements. As shown in Table 5, entropy distance obtains the overall best attack performance in the proposed three methods among all datasets. The above results indicate that, compared with traditional Euclidean distance and Cosine

Table 6. Class distributions of selected adversarial nodes on three datasets. ‘Sec. Poss.’ refers to the second possible class.

Datasets	Methods	Same	Sec. Poss.	Others
Cora	NMA	0	0.3054	0.6946
	NMAB	0	0.3130	0.6870
Citeseer	NMA	0.0100	0.3370	0.6619
	NMAB	0.0016	0.3548	0.6436
Pubmed	NMA	0	0.8936	0.1064
	NMAB	0	0.8891	0.1109

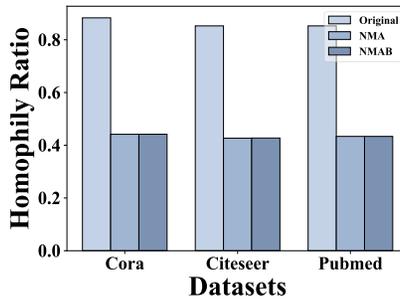


Fig. 3. Average homophily ratios of target nodes before and after NMA/NMAB attacks on three datasets.

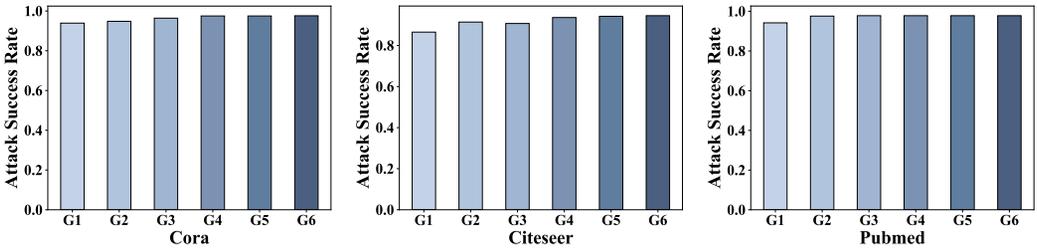


Fig. 4. Attack success rate of the proposed NMA and NMAB methods on three datasets under different parameter combinations.

distance, entropy distance may be a better metric to characterize the dissimilarity of different nodes during the neighborhood aggregations of GNNs.

5.5.4 Preference of Adversarial Node Selections. In this subsection, we analyze the preference of adversarial node selections of the proposed methods. As NMA and NMAB perform better than NGA, we utilize the prior two as representative strategies. Specifically, the investigated properties include class distribution, degree, and predicted confidence of the selected nodes.

Class. To explore the class of the selected adversarial nodes, we first analyze the change of homophily of target nodes before and after the attacks. Homophily [16, 30] is widely adopted to

characterize the similarity of the target node and its neighbors. Specifically, the homophily of a specific node i is given by the ratio of the number of neighbors with the same label to the number of all neighbors, which is as follows.

$$\text{homophily}_i = \frac{\# i's \text{ neighbors with same label}}{\# i's \text{ all neighbors}}. \quad (9)$$

Fig. 3 illustrates the average homophily of target nodes before and after the attacks on three datasets, we can easily find that both NMA and NMAB tend to decrease the homophily of target nodes. In other words, both of them will likely connect the target node with those nodes whose classes are different from the target node.

To investigate the class distributions of the new neighbors, we divide different classes into three groups, namely the same class, the second possible class, and others. The first group refers to the case where the new neighbor belongs to the same class as the target node. The second group refers to the case where the new neighbors belong to the second possible class of the target node, which is consistent with the node selection idea of SGA, while the last group indicates other classes that do not meet the prior two groups.

As shown in Table 6, almost none of the new neighbors are selected from the same class as the targeted node, and most of them belong to different classes (i.e., either the second possible class or others). The above finding is consistent with the intuitive fact that neighbors with the same class usually will help central nodes obtain a better representation, while the nodes with different classes usually will be harmful to the feature aggregation of central nodes. As for the other two groups, our methods do not always select nodes from the second possible class but give other classes large weights, except for the Pubmed dataset since it is only a three-class dataset. This phenomenon indicates the idea that SGA follows where the second possible class is the easiest way to achieve the wrong prediction may not always hold. Connecting the target node with the new neighbors belonging to other classes may lead to a better attack performance than connecting them with the nodes belonging to the second possible class.

Degree. The first part of Table 7 shows the degree properties of the selected adversarial nodes. As we can observe, the adversarial nodes usually have a low degree, which is even lower than the average degree of nodes in the clean datasets. The above finding supports our analysis in Section 4.1 that low degree nodes usually will have a larger influence on the neighborhood aggregation mechanism than high degree nodes.

Predicted Confidence. Besides the above properties, we further analyze the average predicted confidence of the adversarial nodes. As shown in the second part of Table 7, Both NMA and NMAB tend to choose the nodes with relatively high confidence in the predicted class, which also refers to the fact that these kinds of low degree nodes usually have a higher value of noise than other nodes as they belong to classes different from the target node.

To sum up, we can conclude that successful attacks usually select those adversarial nodes with different classes from the target nodes, low degree, and high predicted confidence. Connecting the specific target node with these nodes usually will lead to a more powerful attack performance than connecting with other nodes.

5.5.5 Comparison of Searching Space and Time Cost. Next, we compare the searching space of the proposed attack strategies with baselines. As we try to select the optimal perturbations to mislead the prediction of target nodes, our issue can be considered as a node/link selection problem. Assuming there are n nodes in the clean graph and the attack budget for the target node is Δ , the theoretical maximal searching space will be close to the combination number $C(n, \Delta)$, which will be a huge number with the increase of n . For NETTACK and FGA, as they will greedily select the

Table 7. Average degree and confidence (%) of selected adversarial nodes on three datasets.

Datasets	NMA		NMAB	
	Degree	Confidence	Degree	Confidence
Cora	1.66	99.88	1.94	99.91
Citeseer	1.60	74.64	1.75	76.75
Pubmed	1.15	99.64	1.22	99.47

Table 8. Average time (s) to generate adversarial samples of different methods.

Datasets	NETTACK	FGA	SGA	NGA	NMA	NMAB
Cora	0.880	0.103	0.022	0.004	0.237	1.226
Citeseer	0.676	0.093	0.042	0.003	0.188	0.904
Pubmed	30.852	18.016	0.287	0.010	1.431	7.294

optimal perturbation among all possible perturbations during each budget, their searching space is close to $\Delta \cdot n$. Particularly, the searching space of NETTACK will be slightly smaller than $\Delta \cdot n$ as it further requires the unnoticeable perturbations. Compared to them, SGA further reduces the searching space by only considering the nodes belonging to the second possible class. If we assume the number of nodes in the second possible class is $|C_s|$, then the searching space of SGA would be close to $\Delta \cdot |C_s|$.

In terms of the proposed three methods, NGA directly selects the top Δ links with the highest noise, and thus the searching space of NGA is Δ . For NMA, as we only consider the top δ_1 links with the highest noise value during each budget, the searching space of NMA will be $\Delta \cdot \delta_1$. Finally, for NMAB, we need to evaluate the multiple optimal links under two recorded lists E_{sin} and E_{optimal} whose length are len_{sin} and len_{re} , respectively. Therefore, the total searching space of NMAB should be $\delta_2 + len_{\text{sin}} \cdot len_{\text{re}} \cdot (\Delta - 1)$, where the first term means the searching of top len_{sin} links among all δ_2 candidates (i.e., *line 8 in Algorithm 2*), and the second term indicates the possible searching space by combining the (1)-link optimal attack list E_{sin} and (i)-link optimal attack list E_{optimal} (i.e., *line 12 in Algorithm 2*). Particularly, the searching space of NMAB will equal to NMA once the condition $\delta_1 = \delta_2 = len_{\text{sin}} \cdot len_{\text{re}}$ is satisfied. Based on the above discussions, as $n \gg |C_s| > \delta$ or len_{sin} or len_{re} , the proposed methods will largely reduce searching space while promising a remarkable attack performance.

To better demonstrate the efficiency of the proposed method, we further compare the average time cost for generating the adversarial samples of different attack strategies. As shown in Table 8, NGA obtains the global optimal results because of its heuristic selections. As expected, NMA performs more efficient than NETTACK as NMA avoids unnecessary searching on the nodes with lower noise. Although NMAB requires relatively larger time cost to generate adversarial samples on Cora and Citeseer, it shows good scalability on Pubmed than NETTACK and FGA. Furthermore, SGA obtains the smallest time cost among all methods, but there are trade-offs between its attack performance and time cost. Combining with previous experiments on attack performance, we can observe that the proposed methods not only yield a strong attack effect but also show remarkable efficiency.

5.5.6 Parameters Analysis. Finally, we study the attack performance of NMA and NMAB under different parameter combinations. For NMA, we analyze the influence of the size of candidates δ_1 . For NMAB, we analyze the size of candidates δ_2 , the size of single optimal list len_{sin} , and the size of

retaining list len_{re} . Generally speaking, a larger value of these parameters indicates a larger searching space, which refers to a larger possibility of obtaining the global optimal performance.

Specifically, we set δ_1 of NMA as 3Δ , 5Δ , and 10Δ . We then set the combination of $\{\delta_2, len_{sin}, len_{re}\}$ of NMAB as $\{10\Delta, 3\Delta, 10\}$, $\{10\Delta, 3\Delta, 2\Delta\}$ and $\{10\Delta, 10\Delta, 3\Delta\}$. We labeled the above 6 groups as G1 to G6. Fig. 4 shows the corresponding results of different settings of the above parameters of NMA and NMAB. For NMA (i.e., G1-G3), a larger value of δ_1 will truly lead to better performance in most cases, and NMA can achieve an excellent performance even when δ_1 is a small value. In other words, we can achieve a good attack performance by only considering the candidate nodes with higher noise. The possible reason for the unstable performance of G3 on Citeseer is that NMA may drop into the local optimal in the early steps. As for NMAB (i.e., G4-G6), a similar trend can be obtained. NMAB also achieves a strong attack performance by limiting the searching space to a relatively small value via the noise value. The above observations support that the proposed noise can help fast localize the powerful adversarial nodes that negatively influence the aggregations of target nodes the most. Therefore, our candidate selection mechanism based on noise can be considered as a plugin to integrate into further attack methods in the pre-processing step, which can omit some unnecessary searching on those poorly-performed candidates.

6 Conclusion

In this work, we theoretically discuss the attack strength of different adversarial structural perturbations of graph neural networks, and then put forward the concept of noise to characterize them. By quantifying the noise value of adversarial links via entropy, we further propose three simple yet effective targeted attack strategies, namely the noise-based method (NGA), the noise and margin-based method (NMA), and the boost version of the noise and margin-based method considering multiple budgets at the same time (NMAB). The latter two methods can greatly reduce the searching space of traditional margin-based methods while yielding a strong attack effect. Comprehensive experimental results against various graph neural network models on the benchmark datasets demonstrate the superiority of the proposed methods. Particularly, the analysis of the properties of selected adversarial nodes also supports the effectiveness of the proposed noise concept. Theoretical proof on the toy model, together with extensive empirical experiments, also shows the rationality of the proposed noise concept. Future attack methods can integrate the proposed candidate refining framework to avoid unnecessary searching for perturbations with small noise. In addition, there are some interesting directions that need further investigation in the future. Firstly, we will investigate whether the proposed attack strategies can be applied to large-scale graphs to improve the generality. Secondly, we will explore how to ensure the effectiveness of our attack strategies against defense methods.

Acknowledgments

References

- [1] Zufikar Alom, Tran Gia Bao Ngo, Murat Kantarcioglu, and Cuneyt Gurcan Akcora. 2025. G0ttack: Universal Adversarial Attacks on Graph Neural Networks via Graph Orbits Learning. In *Proc. ICLR*.
- [2] Aleksandar Bojchevski and Stephan Günnemann. 2019. Certifiable robustness to graph perturbations. *Proc. NeurIPS* 32 (2019).
- [3] Jinyin Chen, Lihong Chen, Yixian Chen, Minghao Zhao, Shanqing Yu, Qi Xuan, and Xiaoni Yang. 2019. GA-based q-attack on community detection. *IEEE Trans. Comput. Soc. Syst.* 6, 3 (2019), 491–503.
- [4] Jinyin Chen, Yangyang Wu, Xuanheng Xu, Yixian Chen, Haibin Zheng, and Qi Xuan. 2018. Fast gradient attack on network embedding. *arXiv preprint arXiv:1809.02797* (2018).
- [5] Liang Chen, Jintang Li, Jiaying Peng, Tao Xie, Zengxu Cao, Kun Xu, Xiangnan He, and Zibin Zheng. 2020. A survey of adversarial learning on graphs. *arXiv preprint arXiv:2003.05730* (2020).
- [6] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial attack on graph structured data. In *Proc. ICML*. PMLR, 1115–1124.

- [7] Junyuan Fang, Haixian Wen, Jiajing Wu, Qi Xuan, Zibin Zheng, and Chi K Tse. 2024. Gani: Global attacks on graph neural networks via imperceptible node injections. *IEEE Trans. Comput. Soc. Syst.* 11, 4 (2024), 5374–5387.
- [8] Junyuan Fang, Han Yang, Jiajing Wu, Zibin Zheng, and Chi K Tse. 2025. What Contributes More to the Robustness of Heterophilic Graph Neural Networks? *IEEE Trans. Syst. Man Cybern.: Syst.* (2025).
- [9] Simon Geisler, Tobias Schmidt, Hakan Şirin, Daniel Zügner, Aleksandar Bojchevski, and Stephan Günnemann. 2021. Robustness of graph neural networks at scale. *Proc. NeurIPS* 34 (2021), 7637–7649.
- [10] Will Hamilton, Zhitaoying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Proc. NeurIPS* 30 (2017).
- [11] Weiwei Jiang and Jiayun Luo. 2021. Graph neural network for traffic forecasting: A survey. *arXiv preprint arXiv:2101.11174* (2021).
- [12] Wei Jin, Yaxin Li, Han Xu, Yiqi Wang, Shuiwang Ji, Charu Aggarwal, and Jiliang Tang. 2020. Adversarial attacks and defenses on graphs: A review, a tool and empirical studies. *arXiv preprint arXiv:2003.00653* (2020).
- [13] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proc. ICLR*. OpenReview.net.
- [14] Jintang Li, Zishan Gu, Qibiao Peng, Kun Xu, Liang Chen, and Zibin Zheng. 2021. Deep insights into graph adversarial learning: An empirical study perspective. In *Proc. IJCAI-HBAI*. Springer, 87–101.
- [15] Jintang Li, Tao Xie, Chen Liang, Fenfang Xie, Xiangnan He, and Zibin Zheng. 2021. Adversarial attack on large scale graph. *IEEE Trans. Knowl. Data Eng.* (2021).
- [16] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-GCN: Geometric graph convolutional networks. In *Proc. ICLR*.
- [17] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Mag.* 29, 3 (2008), 93–93.
- [18] Lichao Sun, Yingtong Dou, Carl Yang, Ji Wang, Philip S Yu, Lifang He, and Bo Li. 2018. Adversarial attack and defense on graph data: A survey. *arXiv preprint arXiv:1812.10528* (2018).
- [19] Yiwei Sun, Suhang Wang, Xianfeng Tang, Tsung-Yu Hsieh, and Vasant Honavar. 2020. Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach. In *Proc. WWW*. 673–683.
- [20] Shuchang Tao, Qi Cao, Huawei Shen, Junjie Huang, Yunfan Wu, and Xueqi Cheng. 2021. Single node injection attack against graph neural networks. In *Proc. CIKM*. 1794–1803.
- [21] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *Proc. ICLR*.
- [22] Marcin Waniek, Tomasz P Michalak, Michael J Wooldridge, and Talal Rahwan. 2018. Hiding individuals and communities in a social network. *Nat. Hum. Behav.* 2, 2 (2018), 139–147.
- [23] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *Proc. ICML*. PMLR, 6861–6871.
- [24] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. 2019. Adversarial examples for graph data: Deep insights into attack and defense. In *Proc. IJCAI*. 4816–4823.
- [25] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2020. Graph neural networks in recommender systems: a survey. *ACM Comput. Surv.* (2020).
- [26] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* 32, 1 (2020), 4–24.
- [27] Qifan Zhang, Junyuan Fang, Jie Zhang, Jiajing Wu, Yongxiang Xia, and Zibin Zheng. 2020. Cross entropy attack on deep graph infomax. In *Proc. ISCAS*. IEEE, 1–5.
- [28] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81.
- [29] Guanghui Zhu, Mengyu Chen, Chunfeng Yuan, and Yihua Huang. 2024. Simple and efficient partial graph adversarial attack: A new perspective. *IEEE Trans. Knowl. Data Eng.* (2024).
- [30] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Proc. NeurIPS*.
- [31] Yihai Zhu, Jun Yan, Yan Sun, and Haibo He. 2014. Revealing cascading failure vulnerability in power grids using risk-graph. *IEEE Trans. Parallel Distrib. Syst.* 25, 12 (2014), 3274–3284.
- [32] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial attacks on neural networks for graph data. In *Proc. KDD*. 2847–2856.
- [33] Daniel Zügner and Stephan Günnemann. 2019. Adversarial attacks on graph neural networks via meta learning. In *Proc. ICLR*.

A Proof of Propositions

A.1 Proof of Proposition 4.1

PROPOSITION A.1. Let $G = (A, X, E)$ be a simple graph, and $Y = \{0, 1, \dots, C - 1\}$ be the possible label. We simplify the feature of each node to be a one-hot vector corresponding to the label of itself, denoted as $\mu(Y)$. Namely, the feature vector of node u is $x_u = \mu(Y_u)$. Assuming that most of the original neighbors of each node belong to the same class, and the specific noise value of each adversarial link is the same. Consider a one-layer GCN where the output of node u is $h_u = \sigma(W \cdot \sum_{v \in \mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}_u| \cdot |\mathcal{N}_v|}} \cdot x_v)$, σ is the softmax activation function, we have the following.

- (1) From the perspective of target nodes, nodes with a lower degree will be easier to be attacked than those with a higher degree.
- (2) From the perspective of adversarial nodes, nodes with a lower degree will influence the representation of the target node more than those with a higher degree.

PROOF. As we can know, $W \in \mathbb{R}^{C \times C}$ is the learning parameter that needs to be optimized in our assumption. Since we only utilize a single layer GCN, the aggregated features from the aggregator are the weighted sum of original features (i.e., one-hot vectors corresponding to their labels). Moreover, we can easily obtain that, a well-performed GCN can be trained with the optimal W^* which is similar to follows. W^* would be a diagonal-like matrix where the diagonal elements are non-zero while all other elements are mostly close to zero. Only under this condition, the final output vector h_u will have a large value on u 's corresponding label index, indicating that the corresponding GCN can classify the nodes to the ground truth label with a large probability.

For a specific node u with k neighbors, the original output h_u is given as follows.

$$\begin{aligned} h_u &= \sigma(W^* \cdot \sum_{v \in \mathcal{N}(u)} \frac{1}{\sqrt{|\mathcal{N}_u| \cdot |\mathcal{N}_v|}} \cdot x_v) \\ &= \sigma\left[\frac{W^*}{\sqrt{|\mathcal{N}_u|}} \cdot \left(\frac{x_i}{\sqrt{|\mathcal{N}_i|}} + \frac{x_j}{\sqrt{|\mathcal{N}_j|}} + \dots \right. \right. \\ &\quad \left. \left. + \frac{x_k}{\sqrt{|\mathcal{N}_k|}}\right)\right] \end{aligned} \quad (10)$$

Therefore, for the first claim of Proposition 4.1, assuming we have another targeted node a having neighbors l, m, \dots, n . The label of node a is the same as u (i.e., $Y_a = Y_u$) and the degree of node a is higher than the degree of node u (i.e., $|\mathcal{N}_a| > |\mathcal{N}_u|$), which can be given as follows.

$$h_a = \sigma\left[\frac{W^*}{\sqrt{|\mathcal{N}_a|}} \cdot \left(\frac{x_l}{\sqrt{|\mathcal{N}_l|}} + \frac{x_m}{\sqrt{|\mathcal{N}_m|}} + \dots + \frac{x_n}{\sqrt{|\mathcal{N}_n|}}\right)\right] \quad (11)$$

Then, after we connect the same adversarial node e to the target nodes, respectively, the latest output of GCN under W^* will be as follows.

$$\begin{aligned}
h_u &= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_u|+1}} \cdot \left(\frac{x_i}{\sqrt{|\mathcal{N}_i|}} + \frac{x_j}{\sqrt{|\mathcal{N}_j|}} + \dots \right. \right. \\
&\quad \left. \left. + \frac{x_k}{\sqrt{|\mathcal{N}_k|}} + \frac{x_e}{\sqrt{|\mathcal{N}_e|+1}} \right) \right] \\
h_a &= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_a|+1}} \cdot \left(\frac{x_l}{\sqrt{|\mathcal{N}_l|}} + \frac{x_m}{\sqrt{|\mathcal{N}_m|}} + \dots \right. \right. \\
&\quad \left. \left. + \frac{x_n}{\sqrt{|\mathcal{N}_n|}} + \frac{x_e}{\sqrt{|\mathcal{N}_e|+1}} \right) \right],
\end{aligned} \tag{12}$$

where $|\mathcal{N}_e|$ is the original degree of node e .

Since we assume the label of original neighbors (i.e., x_i, x_j, \dots, x_k) of node u are the same as center node, we can utilize one-hot feature $\mu(Y_u)$ to simplify (12). Moreover, if we further assume the degree of all neighbors equal to the average degree of the original graph, denoted as $\langle d \rangle$, we can further have follows.

$$\begin{aligned}
h_u &= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_u|+1}} \cdot \underbrace{\left(\frac{\mu(Y_u)}{\sqrt{\langle d \rangle}} + \frac{\mu(Y_u)}{\sqrt{\langle d \rangle}} + \dots + \frac{\mu(Y_u)}{\sqrt{\langle d \rangle}} \right)}_{|\mathcal{N}_u|} \right. \\
&\quad \left. + \frac{\mu(Y_e)}{\sqrt{|\mathcal{N}_e|+1}} \right] \\
&= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_u|+1}} \cdot \left(\frac{|\mathcal{N}_u| \cdot \mu(Y_u)}{\sqrt{\langle d \rangle}} + \frac{\mu(Y_e)}{\sqrt{|\mathcal{N}_e|+1}} \right) \right] \\
&= \sigma \left[\frac{|\mathcal{N}_u| \cdot W^* \cdot \mu(Y_u)}{\sqrt{(|\mathcal{N}_u|+1) \cdot \langle d \rangle}} + \frac{W^* \cdot \mu(Y_e)}{\sqrt{(|\mathcal{N}_u|+1) \cdot (|\mathcal{N}_e|+1)}} \right] \\
h_a &= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_a|+1}} \cdot \underbrace{\left(\frac{\mu(Y_a)}{\sqrt{\langle d \rangle}} + \frac{\mu(Y_a)}{\sqrt{\langle d \rangle}} + \dots + \frac{\mu(Y_a)}{\sqrt{\langle d \rangle}} \right)}_{|\mathcal{N}_a|} \right. \\
&\quad \left. + \frac{\mu(Y_e)}{\sqrt{|\mathcal{N}_e|+1}} \right] \\
&= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_a|+1}} \cdot \left(\frac{|\mathcal{N}_a| \cdot \mu(Y_a)}{\sqrt{\langle d \rangle}} + \frac{\mu(Y_e)}{\sqrt{|\mathcal{N}_e|+1}} \right) \right] \\
&= \sigma \left[\frac{|\mathcal{N}_a| \cdot W^* \cdot \mu(Y_a)}{\sqrt{(|\mathcal{N}_a|+1) \cdot \langle d \rangle}} + \frac{W^* \cdot \mu(Y_e)}{\sqrt{(|\mathcal{N}_a|+1) \cdot (|\mathcal{N}_e|+1)}} \right]
\end{aligned} \tag{13}$$

As $|\mathcal{N}_a| > |\mathcal{N}_u|$, we can easily obtain the following from (13).

$$\frac{|\mathcal{N}_u|}{\sqrt{(|\mathcal{N}_u| + 1) \cdot \langle d \rangle}} - \frac{|\mathcal{N}_a|}{\sqrt{(|\mathcal{N}_a| + 1) \cdot \langle d \rangle}} < 0$$

$$\frac{1}{\sqrt{(|\mathcal{N}_u| + 1) \cdot (|\mathcal{N}_e| + 1)}} - \frac{1}{\sqrt{(|\mathcal{N}_a| + 1) \cdot (|\mathcal{N}_e| + 1)}} > 0$$
(14)

Therefore, after multiplying with the diagonal-like optimal weight matrix W^* , and the final adjustment of the monotonic softmax activation function, the coefficient in each term represents the specific weight of the corresponding label. For the first term of nodes u and a in (13), we can easily obtain that for the original features denoted by $\mu(Y_u)$ or $\mu(Y_a)$, node a will assign a larger weight than node u . On the contrary, for the second term of nodes u and a in (13), we can obtain that for the adversarial features $\mu(Y_e)$ induced by the adversarial link, node a will assign a smaller weight than node u . Therefore, we have $h_a[Y_u] > h_u[Y_u]$. That is to say, in the Y_u -th/ Y_a -th index of h_a/h_u , the value of node a will larger than node u . Therefore, for the target nodes, nodes with a lower degree are easier to be influenced than those with a higher degree.

For the second claim of Proposition 4.1, we can proof following the similar procedures as before. To attack the target node u , assuming we have two different adversarial nodes p and q where $|\mathcal{N}_p| > |\mathcal{N}_q|$. Then, for the latest output of node u after attacking by nodes p and q , respectively, we can have follows.

$$h_{u_p} = \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_u| + 1}} \cdot \left(\frac{x_i}{\sqrt{|\mathcal{N}_i|}} + \frac{x_j}{\sqrt{|\mathcal{N}_j|}} + \dots + \frac{x_k}{\sqrt{|\mathcal{N}_k|}} + \frac{x_p}{\sqrt{|\mathcal{N}_p| + 1}} \right) \right]$$

$$h_{u_q} = \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_u| + 1}} \cdot \left(\frac{x_i}{\sqrt{|\mathcal{N}_i|}} + \frac{x_j}{\sqrt{|\mathcal{N}_j|}} + \dots + \frac{x_k}{\sqrt{|\mathcal{N}_k|}} + \frac{x_q}{\sqrt{|\mathcal{N}_q| + 1}} \right) \right]$$
(15)

Similar to (13), we can simplify (15) under the same assumptions, which is as follows.

$$h_{u_p} = \sigma \left[\frac{|\mathcal{N}_u| \cdot W^* \cdot \mu(Y_u)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot \langle d \rangle}} + \frac{W^* \cdot \mu(Y_p)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot (|\mathcal{N}_p| + 1)}} \right]$$

$$h_{u_q} = \sigma \left[\frac{|\mathcal{N}_u| \cdot W^* \cdot \mu(Y_u)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot \langle d \rangle}} + \frac{W^* \cdot \mu(Y_q)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot (|\mathcal{N}_q| + 1)}} \right]$$
(16)

Similarly, after multiplying with the diagonal-like optimal weight matrix W^* and the adjustment of the monotonic softmax function, the coefficient of each term represents the specific weight of the corresponding possible label. From (16), we know that the first term of h_{u_p} and h_{u_q} is the same. While for the second term, the latter situation (i.e., connects with node q) will assign a larger weight to the adversarial nodes/features than the first situation (i.e., connects with node p) since $|\mathcal{N}_p| > |\mathcal{N}_q|$, so we have $h_{u_p}[Y_u] > h_{u_q}[Y_u]$. That is to say, in the Y_u -th index of final outputs, the value of u_p will larger than u_q . Therefore, for the adversarial nodes, nodes with a lower degree can influence the aggregation of the target node more than those with a higher degree. \square

A.2 Proof of Proposition 4.2

PROPOSITION A.2. *Except for the specific noise value of adversarial links varying from each other, we let all of the other assumptions be the same as Proposition 4.1. Then, we have the following. For a specific target node u , if the adversarial nodes have the same degree, the adversarial nodes which are dissimilar to node u influence the aggregation of node u more than those similar to node u .*

PROOF. Assuming we have two different adversarial nodes p, q , and $|\mathcal{N}_p| = |\mathcal{N}_q|$, but their similarities with the target node u are different. Specifically, $\text{SIM}(u, p) > \text{SIM}(u, q)$ where $\text{SIM}(\cdot, \cdot)$ means the similarity function.

$$\begin{aligned}
 h_{u_p} &= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_u| + 1}} \cdot \left(\frac{x_i}{\sqrt{|\mathcal{N}_i|}} + \frac{x_j}{\sqrt{|\mathcal{N}_j|}} + \cdots + \frac{x_k}{\sqrt{|\mathcal{N}_k|}} \right. \right. \\
 &\quad \left. \left. + \frac{x_p}{\sqrt{|\mathcal{N}_p| + 1}} \right) \right] \\
 &= \sigma \left[\frac{|\mathcal{N}_u| \cdot W^* \cdot \mu(Y_u)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot \langle d \rangle}} + \frac{W^* \cdot \mu(Y_p)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot (|\mathcal{N}_p| + 1)}} \right]
 \end{aligned} \tag{17}$$

$$\begin{aligned}
 h_{u_q} &= \sigma \left[\frac{W^*}{\sqrt{|\mathcal{N}_u| + 1}} \cdot \left(\frac{x_i}{\sqrt{|\mathcal{N}_i|}} + \frac{x_j}{\sqrt{|\mathcal{N}_j|}} + \cdots + \frac{x_k}{\sqrt{|\mathcal{N}_k|}} \right. \right. \\
 &\quad \left. \left. + \frac{x_q}{\sqrt{|\mathcal{N}_q| + 1}} \right) \right] \\
 &= \sigma \left[\frac{|\mathcal{N}_u| \cdot W^* \cdot \mu(Y_u)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot \langle d \rangle}} + \frac{W^* \cdot \mu(Y_q)}{\sqrt{(|\mathcal{N}_u| + 1) \cdot (|\mathcal{N}_q| + 1)}} \right]
 \end{aligned}$$

Since $\text{SIM}(u, p) > \text{SIM}(u, q)$, if we further utilize their features to characterize the similarities, we can transform it to $\text{SIM}(\mu(Y_u), \mu(Y_p)) > \text{SIM}(\mu(Y_u), \mu(Y_q))$. As a result, in (17), we can consider that, a large part of the second term of h_{u_p} can be combined with the first term, while only a smaller part of the second term h_{u_q} can be combined with the first term. Similarly, after multiplying with the diagonal-like optimal weight matrix W^* and the adjustment of monotonic softmax function, in the Y_u -th index of the latest outputs, the value of u_p will larger than u_q , that is $h_{u_p}[Y_u] > h_{u_q}[Y_u]$. From the above, we can obtain that the adversarial nodes which are dissimilar with node u will influence the aggregation of node u more than those that are similar with node u , as the latter has a smaller probability in the Y_u -th class than the former. \square

Received May 1, 2025