Practical Adversarial Attacks on Graph Neural Networks

Jiaqi Ma [*1] **Shuangrui Ding** [*2] **Qiaozhu Mei** [1 2]

Abstract

We study the black-box attacks on graph neural networks (GNNs) under a novel and realistic constraint: attackers have access to only a subset of nodes in the network, and they can only attack a small number of them. A node selection step is essential under this setup. We demonstrate that the structural inductive biases of GNN models can be an effective source for this type of attacks. Specifically, by exploiting the connection between the backward propagation of GNNs and random walks, we show that the common gradient-based white-box attacks can be generalized to the black-box setting via the connection between the gradient and an importance score similar to PageRank. In practice, we find attacks based on this importance score indeed increase the classification loss by a large margin, but they fail to significantly increase the mis-classification rate. Our further analyses suggest that there is a discrepancy between the loss and mis-classification rate, as the latter presents a diminishing-return pattern when the number of attacked nodes increases. Therefore, we propose a greedy procedure to correct the importance score that takes into account of the diminishing-return pattern. Experimental results show that the proposed procedure can significantly increase the mis-classification rate of common GNNs on real-world data without access to model parameters nor predictions.

1. Introduction

Graph neural networks (GNNs) (Wu et al., 2020), the family of deep learning models on graphs, have shown promising empirical performance on various applications of machine learning to graph data, such as recommender systems (Ying et al., 2018), social network analysis (Li et al., 2017), and drug discovery (Shi et al., 2020). Like other deep learning models, GNNs have also been shown to be vulnerable under adversarial attacks (Zügner et al., 2018), which has recently attracted increasing research interest (Jin et al., 2020). Indeed, adversarial attacks have been an efficient tool to analyze both the theoretical properties as well as the practical accountability of graph neural networks. As graph data have more complex structures than image or text data, researchers have come up with diverse adversarial attack setups.

Despite these research efforts, there is still a considerable gap between the existing attack setups and the reality. It is unreasonable to assume that an attacker can alter the input of a large proportion of nodes, and even if there is a budget limit, it is unreasonable to assume that they can attack any node as they wish. For example, in a real-world social network, the attackers usually only have access to a few bot accounts, and they are unlikely to be among the top nodes in the network; it is difficult for the attackers to hack and alter the properties of celebrity accounts. Moreover, an attacker usually has limited knowledge about the underling machine learning model used by the platform (e.g., they may roughly know what types of models are used but have no access to the model parameters or model predictions). Motivated by the real-world scenario of attacks, in this paper we study a new type of black-box adversarial attack for node classification tasks, which is more restricted and more realistic, assuming that the attacker has no access to the model parameters or predictions. Our setup differs from existing work with a novel constraint on node access, where attackers only have access to a subset of nodes in the graph, and they can only manipulate a small number of them.

The proposed black-box adversarial attack requires a two-step procedure: 1) selecting a small subset of nodes to attack under the limits of node access; 2) altering the node attributes or edges under a per-node budget. In this paper, we focus on the first step and study the node selection strategy. The key insight of the proposed strategy lies in the observation that, with no access to the GNN parameters or predictions, the strong *structural inductive biases* of the GNN models can be exploited as an effective information source of attacks. GNNs have explicit structural inductive biases due to the graph structure and their heavy weight

[*]Equal contribution [1]School of Information, University of Michigan, Ann Arbor, Michigan, USA [2]Department of EECS, University of Michigan, Ann Arbor, Michigan, USA. Correspondence to: Jiaqi Ma, Shuangrui Ding, Qiaozhu Mei <jiaqima@umich.edu, markding@umich.edu, qmei@umich.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 108, 2020.

sharing design. Our work demonstrates that such structural inductive biases turn into security concerns, as the graph structure is usually exposed to the attackers.

Following this insight, we derive a node selection strategy with a formal analysis of the proposed black-box attack setup. By exploiting the connection between the backward propagation of GNNs and random walks, we first generalize the gradient-norm in a white-box attack into a model-independent importance score similar to the PageRank. In practice, attacking the nodes with high importance scores increases the classification loss significantly but does not generate the same effect on the mis-classification rate. Our analyses suggest that such discrepancy is due to the diminishing-return effect of the mis-classification rate. We further propose a greedy correction procedure for calculating the importance scores. Experiments on three real-world benchmark datasets and popular GNN models show that the proposed attack strategy significantly outperforms baseline methods. We summarize our main contributions as follows. 1) We propose a novel setup of black-box attacks for GNNs with a constraint of limited node access, which is by far the most restricted and realistic compared to existing work. 2) We demonstrate that effective practical adversarial attacks are still possible under the restricted setup, due to the structural inductive biases of GNNs. 3) We empirically verify the effectiveness of the proposed method on three benchmark datasets with popular GNN models.

2. Related Work

The study of adversarial attacks on graph neural networks has surged recently. Following a taxonomy of existing work given by Jin et al. (2020), our work belongs to untargeted evasion attacks. For the adversarial perturbation, most existing works of untargeted attacks apply global constraints on the proportion of node features or the number of edges to be altered. Our work sets a novel local constraint on node access, which is more realistic in practice: perturbation on top (e.g., celebrity) nodes is prohibited and only a small number of nodes can be perturbed. Finally, depending on the attacker's knowledge about the GNN model, existing work can be split into three categories: white-box attacks (Xu et al., 2019; Chen et al., 2018; Wu et al., 2019) have access to full information about the model, including model parameters, input data, and labels; grey-box attacks (Zügner and Günnemann, 2019; Zügner et al., 2018; Sun et al., 2019) have partial information about the model and the exact setups vary in a range; in the most challenging setting, black-box attacks (Dai et al., 2018; Bojchevski and Günnemann, 2018; Chang et al., 2020) can only access the input data and sometimes the black-box predictions of the model. In this work, we consider an even more strict black-box attack, where model predictions are invisible to the attackers. As far as we know, the only existing works that conduct untargeted black-box attacks without access to model predictions are Bojchevski and Günnemann (2018) and Chang et al. (2020). However both of them require the access to embeddings of nodes, which are prohibited as well in our setup.

3. Principled Black-Box Attack Strategies with Limited Node Access

In this section, we derive principled attack strategies on GNNs under the novel black-box setup.

3.1. Preliminary Notations

We first introduce necessary notations. We denote a graph as $G = (V, E)$, where $V = \{1, 2, \dots, N\}$ is the set of N nodes, and $E \subseteq V \times V$ is the set of edges. For a node classification problem, the nodes of the graph are collectively associated with node features $X \in \mathbb{R}^{N \times D}$ and labels $y \in \{1, 2, \dots, K\}^N$, where D is the dimensionality of the feature vectors and K is the number of classes. Each node i's local neighborhood including itself is denoted as $\mathcal{N}_i = \{j \in V \mid (i, j) \in E\} \cup \{i\}$, and its degree as $d_i = |\mathcal{N}_i|$. To ease the notation, for any matrix $A \in \mathbb{R}^{D_1 \times D_2}$ in this paper, we refer A_j to the transpose of the j-th row of the matrix, i.e., $A_j \in \mathbb{R}^{D_2}$.

GNN models. Given the graph G, a GNN model is a function $f_G : \mathbb{R}^{N \times D} \to \mathbb{R}^{N \times K}$ that maps the node features X to output logits of each node. We denote the output logits of all nodes as a matrix $H \in \mathbb{R}^{N \times K}$ and $H = f_G(X)$.

Random walks. A random walk (Lovász et al., 1993) on G is specified by the matrix of transition probabilities, $M \in \mathbb{R}^{N \times N}$, where $M_{ij} = 1/d_i$ if $(i, j) \in E$ or $j = i$, and $M_{ij} = 0$ otherwise. M_{ij} represents the probability of transiting from i to j at each step of random walk. Powering M by t gives us the t-step transition matrix M^t.

3.2. White-Box Attacks with Limited Node Access

Problem formulation. Given a classification loss $\mathcal{L} : \mathbb{R}^{N \times K} \times \{1, \dots, K\}^N \to \mathbb{R}$, the problem of white-box attack with limited node access can be formulated as an optimization problem as follows:

$$\max_{S \subseteq V} \quad \mathcal{L}(H, y) \triangleq \sum_{j=1}^{N} \mathcal{L}_j(H_j, y_j) \tag{1}$$

$$\text{subject to} \quad |S| \leq r, d_i \leq m, \forall i \in S, H = f(\tau(X, S)),$$

where $r, m \in \mathbb{Z}^+$ respectively specify the maximum number of nodes and the maximum degree of nodes that can be attacked. Intuitively, we treat high-degree nodes as a proxy of celebrity accounts in a social network. For simplicity, we have omitted the subscript G of the learned GNN

classifier f_G. The function $\tau : \mathbb{R}^{N\times D} \times 2^V \to \mathbb{R}^{N\times D}$ perturbs the feature matrix X based on the selected node set S (i.e., *attack set*). Under the white-box setup, theoretically τ can also be optimized to maximize the loss. However, as our goal is to study the node selection strategy under the black-box setup, we set τ as a pre-determined function. In particular, we define the j-th row of the output of τ as $\tau(X, S)_j = X_j + \mathbb{1}[j \in S]\epsilon$, where $\epsilon \in \mathbb{R}^D$ is a small constant noise vector constructed by attackers' domain knowledge about the features.

Our analysis uses the Carlili-Wagner loss, a close approximation of cross-entropy loss and has been used in adversarial attacks on image classifiers (Carlini and Wagner, 2017):$\mathcal{L}_j(H_j, y_j) \triangleq \max_{k\in\{1,\dots,K\}} H_{jk} - H_{jy_j}$.

The change of loss under perturbation. Next we investigate how the overall loss changes when we perturb different nodes. We define the change of loss when perturbing the node i as a function of the perturbed feature vector x: $\Delta_i(x) = \mathcal{L}(f(X'), y) - \mathcal{L}(f(X), y)$ where $X'_i = x$ and $X'_j = X_j, \forall j \neq i$. To concretize the analysis, we consider the GCN (Kipf and Welling, 2016) model in our following derivations. Suppose f is an L-layer GCN. With the connection between GCN and random walk (Xu et al., 2018) and Assumption 1 on the label distribution, we can show that, in expectation, the first-order Taylor approximation $\widetilde{\Delta}_i(x) \triangleq \Delta_i(X_i) + (\nabla_x \Delta_i(X_i))^T(x - X_i)$ is related to the sum of the i-th column of the L-step random walk transition matrix M^L. We summarize this finding in Proposition 1.

Assumption 1 (Label Distribution)**.** *Assume the label distribution of all nodes follows the same constant categorical distribution, i.e.,* $\Pr[y_j = k] = q_k, \forall j = 1, 2, \dots, N$, *where* $0 < q_k < 1$ *for* $k = 1, 2, \dots, K$ *and* $\sum_{k=1}^K q_k = 1$. *Moreover, since the classifier f has been well-trained, the prediction of f should capture certain relationships among the K classes. Specifically, we assume the chance for f predicting any node j as any class $k \in \{1, \dots, K\}$, conditioned on the node label $y_j = l \in \{1, \dots, K\}$, confines to a certain distribution $p(k \mid l)$, i.e.,* $\Pr\left[\left(\text{argmax}_{c\in\{1,\dots,K\}} H_{jc}\right) = k \mid y_j = l\right] = p(k \mid l)$.

Proposition 1. *For an L-layer GCN model, if Assumption 1 and a technical assumption about the GCN*[1] *hold, then* $\delta_i \triangleq \mathbb{E}\left[\widetilde{\Delta}_i(x)\,|_{x=\tau(X,\{i\})_i}\right] = C\sum_{j=1}^N [M^L]_{ji}$, *where C is a constant independent of i.*

3.3. Adaptation from White-Box to Black-Box

Now we turn to the *black-box* setup where we cannot access to the model parameters or predictions. This means we are not able to evaluate the objective function $\mathcal{L}(H, y)$ of the optimization problem (1). Proposition 1 shows that the relative ratio of δ_i/δ_j between nodes $i \neq j$ only depends on the transition matrix M, which implies that we can still optimize the problem (1) in the black-box setup.

Node selection with importance scores. Consider the change of loss under the perturbation of a set of nodes S. If we write the change of loss δ as a function of the perturbed features and take the first order Taylor expansion, we have $\delta = \sum_{i\in S} \delta_i$. Therefore δ is maximized by the set of r nodes with degrees less than m and the largest possible δ_i, where m, r are the limits of node access defined in the problem (1). Therefore, we can define an *importance score* for each node i as the sum of the i-th column of M^L, i.e., $I_i = \sum_{j=1}^N [M^L]_{ji}$, and simply select the nodes with the highest importance scores to attack. We denote this strategy as **RWCS** (Random Walk Column Sum). We note that RWCS is similar to PageRank. The difference between RWCS and PageRank is that the latter uses the stationary transition matrix M^∞ for a random walk with restart.

3.4. Diminishing-Return Effect and its Correction

Empirically, We find RWCS significantly increases the classification loss (See Figure 1 in Appendix A.3). The nonlinear loss actually increases linearly w.r.t. the perturbation strength (the norm of the perturbation noise ϵ) for a wide range, which indicates that $\widetilde{\Delta}_i$ is a good approximation of Δ_i. Surprisingly, RWCS fails to continue to increase the mis-classification rate (which matters more in real applications) when the perturbation strength becomes larger.

Intuitively, the discrepancy between the classification loss and the mis-classification rate is due to the diminishing-return effect of the latter: after the prediction of a certain target node is flipped to a wrong class, further enhancing that wrong prediction will not contribute to mis-classification rate. Therefore, we further apply two heuristic correction steps on top of the RWCS scores, and develop a greedy iterative node selection procedure.

The first heuristic is that the nodes within a local neighborhood may influence similar target nodes. Following this heuristic, after each node is selected into the attack set, we exclude a k-hop neighborhood of the selected node for next iteration, for a given constant integer k. The second heuristic is that the prediction of a target node will be flipped after a couple of related nodes are perturbed. According to this heuristic, we adopt an adaptive version of RWCS scores. We binarize the L-step random walk transition matrix M^L as $\widetilde{M}$, where $[\widetilde{M}]_{ij} = 1$ if $[M^L]_{ij}$ is among Top-l of $[M^L]_i$ and $[M^L]_{ij} \neq 0$. Here, l is a given constant integer. Next, we define a new adaptive influence score as a function of a matrix Q: $\widetilde{I}_i(Q) = \sum_{j=1}^N [Q]_{ji}$. In the iterative node selection procedure, we initialize Q as $\widetilde{M}$. We select the node

[1] This is an assumption made by Xu et al. (2018), which we list as Assumption 2 in Appendix A.1.

Table 1. Summary of the attack performance. The lower the accuracy (in %) the better the attacks. The **bold** marker denotes the best performance. The asterisk (*) means the difference between the best strategy and the second-best strategy is statistically significant by a t-test at significance level 0.05. The error bar (±) denotes the standard error of the mean by 40 independent trials.

	Cora			Citeseer			Pubmed		
Method	GCN	JKNetConcat	JKNetMaxpool	GCN	JKNetConcat	JKNetMaxpool	GCN	JKNetConcat	JKNetMaxpool
No-Attack	85.6 ± 0.3	86.2 ± 0.2	85.8 ± 0.3	75.1 ± 0.2	72.9 ± 0.3	73.2 ± 0.3	85.7 ± 0.1	85.8 ± 0.1	85.7 ± 0.1
Random	82.6 ± 0.4	70.7 ± 1.1	71.8 ± 1.1	72.6 ± 0.3	62.7 ± 0.8	63.9 ± 0.8	82.6 ± 0.2	77.3 ± 0.4	77.4 ± 0.5
Degree	**80.7** ± 0.4	64.9 ± 1.4	67.0 ± 1.5	70.4 ± 0.4	56.9 ± 0.8	58.7 ± 0.9	81.5 ± 0.4	72.4 ± 0.7	72.3 ± 0.7
PageRank	82.6 ± 0.3	79.6 ± 0.4	79.7 ± 0.4	72.9 ± 0.2	70.2 ± 0.3	70.3 ± 0.3	83.0 ± 0.2	79.3 ± 0.3	79.6 ± 0.3
Betweenness	81.8 ± 0.4	64.1 ± 1.3	65.9 ± 1.4	70.7 ± 0.3	56.3 ± 0.8	58.3 ± 0.9	81.3 ± 0.3	74.1 ± 0.5	74.6 ± 0.5
RWCS	82.8 ± 0.3	79.3 ± 0.5	79.5 ± 0.4	72.9 ± 0.2	69.8 ± 0.3	70.1 ± 0.3	82.1 ± 0.2	77.8 ± 0.3	78.4 ± 0.3
GC-RWCS	**80.7** ± 0.5	**59.1** ± 1.6*	**61.1** ± 1.6*	**67.8** ± 0.5*	**49.0** ± 0.9*	**50.7** ± 1.1*	**80.3** ± 0.5*	**69.2** ± 0.7*	**70.0** ± 0.7*

with highest score $\widetilde{I}_i(Q)$ subsequently. After each iteration, suppose we have selected the node i in this iteration, we will update Q by setting to zero for all the rows where the elements of the i-th column are 1. The underlying assumption is that, adding i to the selected set is likely to mis-classify all the target nodes corresponding to the aforementioned rows. We name this iterative procedure as the **GC-RWCS** (Greedily Corrected RWCS) strategy, and summarize it in Algorithm 1 in Appendix A.2.

4. Experiments

4.1. Experiment Setup

We test the proposed attack strategies by attacking three GNN models on three benchmark datasets, Cora, Citeseer, and Pubmed (Yang et al., 2016). The three GNN models include GCN (Kipf and Welling, 2016) and two variants of JKNet (Xu et al., 2018), JKNetConcat and JKNetMaxpool.

Baseline methods for comparison. As we summarized in Section 2, our proposed black-box adversarial attack setup is by far the most restricted, and none of existing attack strategies for GNN can be applied. We compare the proposed attack strategies with baseline strategies by selecting nodes with top centrality metrics. We compare with three well-known network metrics capturing different aspects of node centrality: **Degree**, **Betweenness**, and **PageRank** and name the attack strategies correspondingly. In classical network analysis literature (Newman, 2018), real-world networks are shown to be fragile under attacks to high-centrality nodes. Therefore we believe these centrality metrics serve as reasonable baselines under our restricted black-box setup. For the purpose of sanity check, we also include a trivial baseline **Random**, which randomly selects the nodes to be attacked.

Nuisance parameters of the attack procedure. For each dataset, we fix the maximum number of nodes to attack, r, as 1% of the graph size. We also fix the node degree limit, m, as the lowest degree of the top 30% nodes. After the node selection step, we also need to specify the perturbation vector $\epsilon \in \mathbb{R}^D$, ideally with domain knowledge about the feature semantics and the task. In our experiments, we have to simulate this idea as we do not know the semantic meaning of the features in the benchmark datasets. Formally, we specify ϵ as follows. For $j = 1, 2, \cdots, D$, $\epsilon_j = \lambda \text{sign}(\sum_{i=1}^N \frac{\partial \mathcal{L}(H,y)}{\partial X_{ij}})$ if $j \in \text{arg top-}J \left(\left[\left| \sum_{i=1}^N \frac{\partial \mathcal{L}(H,y)}{\partial X_{il}} \right| \right]_{l=1,2,\ldots,D} \right)$, and $\epsilon_j = 0$ otherwise, where λ is the magnitude of modification. We fix $J = \lfloor 0.02D \rfloor$ and $\lambda = 1$ for all datasets. While gradients of the model are involved, we emphasize that only use extremely limited information of the gradients, i.e., selecting a few number of important features and the rough individual directions to perturb at the *global level* by averaging gradients on all nodes. We believe such coarse information is usually available from domain knowledge about the classification task.

4.2. Experiment Results

The results clearly demonstrate the effectiveness of the proposed GC-RWCS strategy. GC-RWCS achieves the best attack performance on almost all experiment settings, and the difference to the second-best strategy is significant in almost all cases. It is also worth noting that the proposed GC-RWCS achieves a 70% larger decrease of the accuracy than the Random baseline in most cases (see Table 4 in Appendix A.4). And this is achieved by merely adding the same constant perturbation vector to the features of 1% of the nodes in the graph. This verifies that the explicit structural inductive biases of GNN models make them vulnerable even in the extremely restricted black-box attack setup.

5. Conclusion

In this paper, we propose a novel black-box adversarial attack setup for GNN models with constraint of limited node access, which we believe is by far the most restricted and realistic. Nonetheless, through both theoretical analyses and empirical experiments, we demonstrate that effective practical adversarial attacks are still possible due to the strong and explicit structural inductive biases of GNN models.

Acknowledgement

This work was in part supported by the National Science Foundation under grant number 1633370.

References

Aleksandar Bojchevski and Stephan Günnemann. Adversarial attacks on node embeddings via graph poisoning. *arXiv preprint arXiv:1809.01093*, 2018.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

Heng Chang, Yu Rong, Tingyang Xu, Wenbing Huang, Honglei Zhang, Peng Cui, Wenwu Zhu, and Junzhou Huang. A restricted black-box adversarial framework towards attacking graph embedding models. In *AAAI Conference on Artificial Intelligence*, 2020.

Jinyin Chen, Yangyang Wu, Xuanheng Xu, Yixian Chen, Haibin Zheng, and Qi Xuan. Fast gradient attack on network embedding. *arXiv preprint arXiv:1809.02797*, 2018.

Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. Adversarial attack on graph structured data. *arXiv preprint arXiv:1806.02371*, 2018.

Wei Jin, Yaxin Li, Han Xu, Yiqi Wang, and Jiliang Tang. Adversarial attacks and defenses on graphs: A review and empirical study. *arXiv preprint arXiv:2003.00653*, 2020.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Cheng Li, Jiaqi Ma, Xiaoxiao Guo, and Qiaozhu Mei. Deepcas: An end-to-end predictor of information cascades. In *Proceedings of the 26th international conference on World Wide Web*, pages 577–586, 2017.

László Lovász et al. Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46, 1993.

Mark Newman. *Networks*. Oxford university press, 2018.

Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382*, 2020.

Yiwei Sun, Suhang Wang, Xianfeng Tang, Tsung-Yu Hsieh, and Vasant Honavar. Node injection attacks on graphs via reinforcement learning. *arXiv preprint arXiv:1909.06543*, 2019.

Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, et al. Deep graph library: Towards efficient and scalable deep learning on graphs. *arXiv preprint arXiv:1909.01315*, 2019.

Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. Adversarial examples for graph data: Deep insights into attack and defense. In *IJCAI*, 2019.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. Topology attack and defense for graph neural networks: An optimization perspective. *arXiv preprint arXiv:1906.04214*, 2019.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. *arXiv preprint arXiv:1806.03536*, 2018.

Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 974–983, 2018.

Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *International Conference on Learning Representations (ICLR)*, 2019.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2847–2856, 2018.

A. Appendix

A.1. Proof of Proposition 1

A GNN f_G is usually built by stacking a certain number (L) of layers, with the l-th layer, $1 \leq l \leq L$, taking the following form:

$$H_i^{(l)} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \boldsymbol{W}_l H_j^{(l-1)}\right), \tag{2}$$

where $H^{(l)} \in \mathbb{R}^{N \times D_l}$ is the hidden representation of nodes with D_l dimensions, output by the l-th layer; $\boldsymbol{W}_l$ is a learnable linear transformation matrix; σ is an element-wise nonlinear activation function; and different GNNs have different normalization terms α_{ij}. For instance, $\alpha_{ij} = 1/\sqrt{d_i d_j}$ or $\alpha_{ij} = 1/d_i$ in Graph Convolutional Networks (GCN) (Kipf and Welling, 2016). We first remind the reader for some notations, a GCN model is denoted as a function f, the feature matrix is $X \in \mathbb{R}^{N \times D}$, and the output logits $H = f(X) \in \mathbb{R}^{N \times K}$. The L-step random walk transition matrix is M^L. More details can be found in in Section 3.1

We first give in Lemma 1 the connection between GCN models and random walks. Lemma 1 relies on a technical assumption about the GCN model (Assumption 2) and the proof can be found in Xu et al. (2018).

Assumption 2 (Xu et al. (2018))**.** *All paths in the computation graph of the given GCN model are independently activated with the same probability of success ρ.*

Lemma 1. *(Xu et al. (2018).) Given an L-layer GCN with averaging as $\alpha_{i,j} = 1/d_i$ in Eq. 2, assume that all path in the computation graph of the model are activated with the same probability of success ρ (Assumption 2). Then, for any node $i, j \in V$,*

$$\mathbb{E}\left[\frac{\partial H_j}{\partial X_i}\right] = \rho \cdot \prod_{l=L}^{1} \boldsymbol{W}_l [M^L]_{ji}, \tag{3}$$

where $\boldsymbol{W}_l$ is the learnable parameter at l-th layer.

Then we are able to prove Proposition 1 below.

Proof. First, we derive the gradient of the loss $\mathcal{L}(H, y)$ w.r.t. the feature X_i of node i,

$$\begin{aligned}
\nabla_{X_i} \mathcal{L}(H, y) &= \nabla_{X_i} \left(\sum_{j=1}^{N} \mathcal{L}_j(H_j, y_j)\right) \\
&= \sum_{j=1}^{N} \nabla_{X_i} \mathcal{L}_j(H_j, y_j) \\
&= \sum_{j=1}^{N} \left(\frac{\partial H_j}{\partial X_i}\right)^T \frac{\partial \mathcal{L}_j(H_j, y_j)}{\partial H_j},
\end{aligned} \tag{4}$$

where H_j is the jth row of H but being transposed as column vectors and y_j is the true label of node j. Note that $\frac{\partial \mathcal{L}_j(H_j, y_j)}{\partial H_j} \in \mathbb{R}^K$, and $\frac{\partial H_j}{\partial X_i} \in \mathbb{R}^{K \times D}$.

Next, we plug Eq. 4 into $\widetilde{\Delta}_i(\tau(X, \{i\})_i)$. For simplicity, We write $\widetilde{\Delta}_i(\tau(X, \{i\})_i)$ as $\widetilde{\Delta}_i$ in the rest of the proof.

$$\begin{aligned}
\widetilde{\Delta}_i &= \left(\nabla_{X_i} \mathcal{L}(H, y)\right)^T \epsilon \\
&= \sum_{j=1}^{N} \left(\frac{\partial \mathcal{L}_j(H_j, y_j)}{\partial H_j}\right)^T \frac{\partial H_j}{\partial X_i} \epsilon.
\end{aligned} \tag{5}$$

Denote $a^j \triangleq \frac{\partial \mathcal{L}_j(H_j, y_j)}{\partial H_j} \in \mathbb{R}^K$. From the definition of loss

$$\mathcal{L}_j(H_j, y_j) = \max_{k \in \{1, \ldots, K\}} H_{jk} - H_{jy_j},$$

we have

$$a_k^j = \begin{cases} -1, & \text{if } k = y_j \text{ and } y_j \neq \operatorname{argmax}_{c \in \{1, \ldots, K\}} H_{jc}, \\ 1, & \text{if } k \neq y_j \text{ and } k = \operatorname{argmax}_{c \in \{1, \ldots, K\}} H_{jc}, \\ 0, & \text{otherwise,} \end{cases}$$

for $k = 1, 2, \ldots, K$. Under Assumption 1, the expectation of each element of a^j is

$$\mathbb{E}[a_k^j] = -q_k(1 - p(k \mid k)) + \sum_{w=1, w \neq k}^{K} p(k \mid w) q_w, k = 1, \ldots, K$$

which is a constant independent of H_j and y_j. Therefore, we can write

$$\mathbb{E}[a^j] = c, \forall j = 1, 2, \ldots, N,$$

where $c \in \mathbb{R}^K$ is a constant.

Taking expectation of Eq. (5) and plug in the result of Lemma 1,

$$\begin{aligned}
\mathbb{E}\left[\widetilde{\Delta}_i\right] &\approx \mathbb{E}\left[\sum_{j=1}^{N} \left(\frac{\partial \mathcal{L}_j(H_j, y_j)}{\partial H_j}\right)^T \frac{\partial H_j}{\partial X_i} \epsilon\right] \\
&= \sum_{j=1}^{N} \mathbb{E}[a^j]^T \left(\rho \prod_{l=L}^{1} W_l [M^L]_{ji}\right) \epsilon \\
&= \left(\rho c^T \prod_{l=L}^{1} W_l \epsilon\right) \sum_{j=1}^{N} [M^L]_{ji} \\
&= C \sum_{j=1}^{N} [M^L]_{ji},
\end{aligned}$$

where $C = \rho c^T \prod_{l=L}^{1} W_l \epsilon$ is a constant scalar independent of i. □

A.2. Algorithm Details of GC-RWCS

Algorithm 1 The GC-RWCS Strategy for Node Selection.

Input: number of nodes limit r; maximum degree limit m; neighbor hops k; binarized transition matrix $\widetilde{M}$; the adaptive influence score function $\widetilde{I}_i, \forall i \in V$.
Output: the attack set S.

Initialize the candidate set $P = \{i \in V \mid d_i \leq m\}$, and the score matrix $Q = \widetilde{M}$
Initialize $S = \emptyset$
for $t = 1, 2, \ldots, r$ **do**
  $z \leftarrow \text{argmax}_{i \in P} \widetilde{I}_i(Q)$;
  $S \leftarrow S \cup \{z\}$;
  $P \leftarrow P \setminus \{i \in P \mid \text{shortest-path}(i, z) \leq k\}$;
  $q \leftarrow Q_{\cdot,z}$;
  for $i \in V$ **do**
    if q_i is 1 **then**
      $Q_i \leftarrow \mathbf{0}$;
    end if
  end for
end for
return S;

A.3. Experiment Details

GNN models. We evaluate the proposed attack strategies on two common GNN models, GCN (Kipf and Welling, 2016) and JK-Net (Xu et al., 2018). For JK-Net, we test on its two variants, JKNetConcat and JKNetMaxpool, which apply concatenation and element-wise max at last layer respectively. We set the number of layers for GCN as 2 and the number of layers for both JK-Concat and JK-Maxpool as 7. The hidden size of each layer is 32. For the training, we closely follow the hyper-parameter setup in Xu et al. (2018).

Datasets. We adopt three citation networks, Citeseer, Cora, and Pubmed, which are standard node classification benchmark datasets (Yang et al., 2016). We load those dataset from DGL library (Wang et al., 2019) in our experiments[2]. Statistics of three dataset are summarized in Table 2.

Table 2. Dataset statistics

Dataset	Nodes	Edges	Classes	Features
Citeseer	3,327	4,552	6	3,703
Cora	2,708	5,278	7	1,433
Pubmed	19,717	44,324	3	500

Following the setup of JK-Net (Xu et al., 2018), we randomly split each dataset by 60%, 20%, and 20% for training, validation, and testing. And we draw 40 random splits.

[2]The statistic of edges in table 2 is listed after eliminating self-loops.

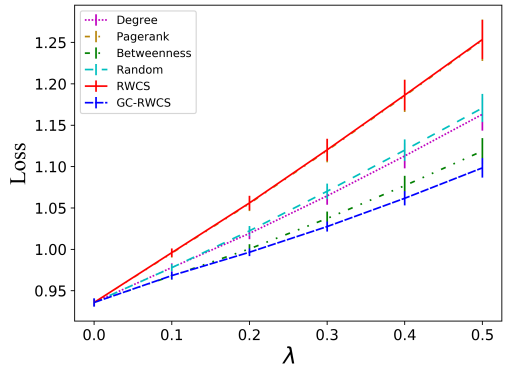
Hyper-parameters for GC-RWCS. For the proposed GC-RWCS strategy, we fix the number of step $L = 4$, the neighbor-hop parameter $k = 1$ and the parameter $l = 30$ for the binarized $\widetilde{M}$ for all models on all datasets. Note that $L = 4$ is different from the number of layers of both GCN and JK-Nets in our experiments. But we achieve effective attack performance. We also conduct a sensitivity analysis in Appendix A.4 and demonstrate the proposed method is not sensitive w.r.t. L.

A.4. Additional Results

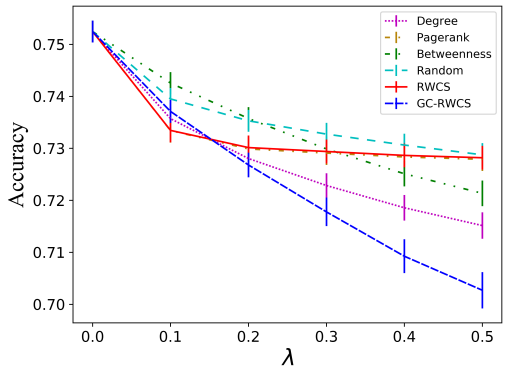
Verifying the discrepancy between the loss and the mis-classification rate. We first provide empirical evidence for the discrepancy between classification loss (cross-entropy) and mis-classification rate. We compare the RWCS strategy with baseline strategies with varying perturbation strength as measured by λ. The results shown in Figure 1 are obtained by attacking GCN on Citeseer. First, we observe that RWCS increases the classification loss almost linearly as λ increases, indicating our approximation of the loss by first-order Taylor expansion actually works pretty well in practice. Not surprisingly, RWCS performs very similarly as PageRank. And RWCS performs much better than other centrality metrics in increasing the classification loss, showing the effectiveness of Proposition 1. However, we see the decrease of classification accuracy when attacked by RWCS (and PageRank) quickly saturates as λ increases. The GC-RWCS strategy that is proposed to correct the importance scores is able to decreases the classification accuracy the most as λ becomes larger, although it increases the classification loss the least.

Table 3. Summary of the accuracy (in %) when $L = \{3, 4, 5, 6, 7\}$. The **bold number** and the asterisk (*) denotes the same meaning as Table 1. The underline marker denotes the values of GC-RWCS outperforms all the baseline.

	Cora			Citeseer			Pubmed		
Method	GCN	JKNetConcat	JKNetMaxpool	GCN	JKNetConcat	JKNetMaxpool	GCN	JKNetConcat	JKNetMaxpool
None	85.6 ± 0.3	86.2 ± 0.2	85.8 ± 0.3	75.1 ± 0.2	72.9 ± 0.3	73.2 ± 0.3	85.7 ± 0.1	85.8 ± 0.1	85.7 ± 0.1
Threshold 10%									
Random	81.3 ± 0.3	68.8 ± 0.8	68.8 ± 1.3	71.3 ± 0.3	60.8 ± 0.8	61.7 ± 0.9	82.0 ± 0.3	75.9 ± 0.7	75.2 ± 0.7
Degree	78.2 ± 0.4	60.7 ± 1.0	59.9 ± 1.5	67.5 ± 0.4	52.5 ± 0.8	53.7 ± 1.0	78.9 ± 0.5	63.4 ± 1.0	63.2 ± 1.2
PageRank	79.4 ± 0.4	71.6 ± 0.6	70.0 ± 1.0	70.1 ± 0.3	61.5 ± 0.5	62.6 ± 0.6	80.3 ± 0.3	71.3 ± 0.8	71.2 ± 0.8
Betweenness	79.7 ± 0.4	60.5 ± 0.9	60.3 ± 1.6	68.9 ± 0.3	53.5 ± 0.8	55.1 ± 1.0	78.5 ± 0.6	67.1 ± 1.1	66.1 ± 1.1
RWCS	79.4 ± 0.4	71.7 ± 0.5	70.3 ± 0.9	69.9 ± 0.3	62.4 ± 0.4	63.1 ± 0.6	79.8 ± 0.3	70.7 ± 0.8	70.7 ± 0.8
GC-RWCS-3	78.6 ± 0.5	**52.1** ± 1.1*	**53.0** ± 1.9*	**64.8** ± 0.5*	**46.4** ± 0.8*	**48.2** ± 1.0*	78.1 ± 0.6	62.3 ± 1.2	61.6 ± 1.5
GC-RWCS-4	78.5 ± 0.5	52.7 ± 1.0*	53.3 ± 1.9*	65.1 ± 0.5*	46.6 ± 0.8*	48.2 ± 1.1*	**77.3** ± 0.7	**62.1** ± 1.2	**60.6** ± 1.4*
GC-RWCS-5	78.9 ± 0.5	53.5 ± 1.1*	54.2 ± 1.9*	65.3 ± 0.5*	46.6 ± 0.8*	48.4 ± 1.0*	78.4 ± 0.5	64.2 ± 1.2	62.5 ± 1.4
GC-RWCS-6	78.5 ± 0.5	54.3 ± 1.1*	54.9 ± 1.9*	65.5 ± 0.5*	47.1 ± 0.8	48.9 ± 1.1*	78.0 ± 0.6	63.7 ± 1.1	62.6 ± 1.4
GC-RWCS-7	**78.1** ± 0.5	54.2 ± 1.1*	54.8 ± 1.9*	66.1 ± 0.4*	47.5 ± 0.8	49.3 ± 1.1*	78.7 ± 0.5	64.9 ± 1.2	63.3 ± 1.3
Threshold 20%									
Random	82.3 ± 0.3	71.7 ± 1.1	69.8 ± 1.1	72.1 ± 0.3	62.1 ± 0.7	62.6 ± 0.9	82.6 ± 0.2	77.9 ± 0.5	77.5 ± 0.5
Degree	**79.3** ± 0.4	64.2 ± 1.2	61.6 ± 1.3	69.2 ± 0.4	56.0 ± 0.8	56.4 ± 1.0	80.6 ± 0.4	69.5 ± 0.8	69.4 ± 1.0
PageRank	80.8 ± 0.3	74.5 ± 0.8	73.0 ± 0.8	72.1 ± 0.3	68.3 ± 0.3	68.2 ± 0.4	82.2 ± 0.2	77.7 ± 0.4	77.8 ± 0.4
Betweenness	80.7 ± 0.4	62.2 ± 1.4	60.1 ± 1.4	70.1 ± 0.4	54.8 ± 0.8	55.8 ± 1.1	80.2 ± 0.4	72.4 ± 0.8	72.0 ± 0.7
RWCS	81.4 ± 0.3	76.8 ± 0.6	76.0 ± 0.6	72.4 ± 0.3	68.9 ± 0.3	69.0 ± 0.4	81.3 ± 0.2	76.0 ± 0.4	76.5 ± 0.4
GC-RWCS-3	79.4 ± 0.5	**57.5** ± 1.6*	**53.1** ± 1.5*	**67.1** ± 0.4*	48.4 ± 0.9*	49.3 ± 1.2*	**79.0** ± 0.5*	**67.4** ± 0.9*	**66.3** ± 1.0*
GC-RWCS-4	79.4 ± 0.5	57.5 ± 1.7*	53.2 ± 1.4*	67.3 ± 0.5*	**47.9** ± 0.9*	**48.8** ± 1.3*	**79.0** ± 0.5*	67.4 ± 1.0*	**66.3** ± 1.0*
GC-RWCS-5	79.4 ± 0.5	59.0 ± 1.7*	54.5 ± 1.4*	67.3 ± 0.4*	48.4 ± 0.9*	49.4 ± 1.3*	79.2 ± 0.5*	68.5 ± 0.9	68.1 ± 0.9
GC-RWCS-6	79.5 ± 0.5	59.3 ± 1.7	54.9 ± 1.5*	68.1 ± 0.4*	49.2 ± 0.9*	50.2 ± 1.3*	79.1 ± 0.5*	68.4 ± 0.9	68.5 ± 1.0
GC-RWCS-7	79.4 ± 0.5	59.3 ± 1.6	55.3 ± 1.5*	68.1 ± 0.4*	50.0 ± 0.9*	50.8 ± 1.3*	79.2 ± 0.5*	68.7 ± 0.9	68.2 ± 0.8
Threshold 30%									
Random	82.6 ± 0.4	70.7 ± 1.1	71.8 ± 1.1	72.6 ± 0.3	62.7 ± 0.8	63.9 ± 0.8	82.6 ± 0.2	77.3 ± 0.4	77.3 ± 0.5
Degree	80.7 ± 0.4	64.9 ± 1.4	67.0 ± 1.5	70.4 ± 0.4	56.9 ± 0.8	58.7 ± 0.9	81.5 ± 0.4	72.4 ± 0.7	72.1 ± 0.8
PageRank	82.6 ± 0.3	79.6 ± 0.4	79.7 ± 0.4	72.9 ± 0.2	70.2 ± 0.3	70.3 ± 0.3	83.0 ± 0.2	79.3 ± 0.3	79.5 ± 0.3
Betweenness	81.8 ± 0.4	64.1 ± 1.3	65.9 ± 1.4	70.7 ± 0.3	56.3 ± 0.8	58.3 ± 0.9	81.3 ± 0.3	74.1 ± 0.5	74.5 ± 0.5
RWCS	82.9 ± 0.3	79.7 ± 0.4	80.0 ± 0.4	72.9 ± 0.2	70.2 ± 0.3	70.4 ± 0.3	82.1 ± 0.2	77.8 ± 0.3	78.4 ± 0.3
GC-RWCS-3	**80.2** ± 0.6	**57.3** ± 1.7*	**59.0** ± 1.6*	67.9 ± 0.5*	49.1 ± 0.9*	50.8 ± 1.1*	80.3 ± 0.5*	**69.0** ± 0.7*	**69.8** ± 0.7*
GC-RWCS-4	80.7 ± 0.5	59.1 ± 1.6*	61.1 ± 1.6*	**67.8** ± 0.5*	**49.0** ± 0.9*	**50.7** ± 1.1*	80.3 ± 0.5*	69.2 ± 0.7*	70.0 ± 0.7*
GC-RWCS-5	80.8 ± 0.5	59.8 ± 1.6*	61.5 ± 1.6*	68.4 ± 0.5*	49.2 ± 0.9*	51.2 ± 1.1*	**80.2** ± 0.5*	70.4 ± 0.6*	71.5 ± 0.6
GC-RWCS-6	80.7 ± 0.5	59.8 ± 1.5*	61.4 ± 1.5*	68.5 ± 0.5*	50.5 ± 0.9*	52.2 ± 1.1*	**80.2** ± 0.5*	70.5 ± 0.5*	71.6 ± 0.6
GC-RWCS-7	80.7 ± 0.5	60.2 ± 1.5*	61.9 ± 1.5*	68.7 ± 0.5*	50.7 ± 0.9*	52.6 ± 1.1*	80.3 ± 0.4*	70.9 ± 0.5*	71.9 ± 0.6



(a) Loss on Test Set



(b) Accuracy on Test Set

Figure 1. Experiments of attacking GCN on Citeseer with increasing perturbation strength λ. Results are averaged over 40 random trials and error bars indicate standard error of mean.

More experiment setups and sensitivity analysis the hyper-parameter L. We provide more experiment setups with varying max degree limits m and test the sensitivity of the parameter L. We show the results in table 3. Thresholds 10%, 20%, and 30% respectively indicate m equals to the lowest degree of the top 10%, 20%, and 30% nodes. We also show the results of GC-RWCS with $L = 3, 4, 5, 6, 7$. In conclusion, it can be seen that our strategy is not sensitive w.r.t. the choice of L in a wide range.

Further, we also compare the relative decrease of accuracy between the proposed GC-RWCS strategy ($L = 4$) and the Random strategy in Table 4. GC-RWCS is able to decrease the node classification accuracy by up to 33.5%, and achieves a 70% larger decrease of the accuracy than the Random baseline in most cases. As the GC-RWCS and Random use exactly the same feature perturbation and the node selection step of Random does not include any information of the graph structure, this relative comparison can be roughly viewed as an indicator of the attack effectiveness attributed to the structural inductive biases of the GNN models.

Table 4. Accuracy decrease (in %) comparison with clean dataset

	Cora			Citeseer			Pubmed		
Method	GCN	JKNetConcat	JKNetMaxpool	GCN	JKNetConcat	JKNetMaxpool	GCN	JKNetConcat	JKNetMaxpool
Random	3.0	15.5	14	2.5	10.2	9.3	3.1	8.5	8.3
GC-RWCS	4.9	27.1	24.7	7.3	23.9	22.5	5.4	16.6	15.7
GC-RWCS/Random	163.33%	174.84%	176.43%	292.00%	234.31%	241.94%	174.19%	195.29%	189.16%