# Topology Attack and Defense for Graph Neural Networks: An Optimization Perspective

**Kaidi Xu**[1*], **Hongge Chen**[2*], **Sijia Liu**[3], **Pin-Yu Chen**[3], **Tsui-Wei Weng**[2], **Mingyi Hong**[4] and **Xue Lin**[1]
[1]Electrical & Computer Engineering, Northeastern University, Boston, USA
[2]Electrical Engineering & Computer Science, Massachusetts Institute of Technology, Cambridge, USA
[3]MIT-IBM Watson AI Lab, IBM Research
[4]Electrical & Computer Engineering, University of Minnesota, Minneapolis, USA
xu.kaid@husky.neu.edu, chenhg@mit.edu, sijia.Liu@ibm.com, pin-yu.chen@ibm.com, twweng@mit.edu, mhong@umn.edu, xue.lin@northeastern.edu

## Abstract

Graph neural networks (GNNs) which apply the deep neural networks to graph data have achieved significant performance for the task of semi-supervised node classification. However, only few work has addressed the adversarial robustness of GNNs. In this paper, we first present a novel gradient-based attack method that facilitates the difficulty of tackling discrete graph data. When comparing to current adversarial attacks on GNNs, the results show that by only perturbing a small number of edge perturbations, including addition and deletion, our optimization-based attack can lead to a noticeable decrease in classification performance. Moreover, leveraging our gradient-based attack, we propose the first optimization-based adversarial training for GNNs. Our method yields higher robustness against both different gradient based and greedy attack methods without sacrificing classification accuracy on original graph.

## 1 Introduction

Graph structured data plays a crucial role in many AI applications. It is an important and versatile representation to model a wide variety of datasets from many domains, such as molecules, social networks, or interlinked documents with citations. Graph neural networks (GNNs) on graph structured data have shown outstanding results in various applications [Kipf and Welling, 2016; Veličković *et al.*, 2017; Xu *et al.*, 2019a]. However, despite the great success on inferring from graph data, the inherent challenge of lacking adversarial robustness in deep learning models still carries over to security-related domains such as blockchain or communication networks.

In this paper, we aim to evaluate the robustness of GNNs from a perspective of first-order optimization adversarial attacks. It is worth mentioning that first-order methods have achieved great success for generating adversarial attacks on audios or images [Carlini and Wagner, 2018; Xu *et al.*, 2019b; Chen *et al.*, 2018b; Xu *et al.*, 2019c; Chen *et al.*, 2018a]. However, some recent works [Dai *et al.*, 2018; Bojcheski and Günnemann, 2018] suggested that conventional (first-order) continuous optimization methods do not directly apply to attacks using edge manipulations (we call *topology attack*) due to the discrete nature of graphs. We close this gap by studying the problem of generating topology attacks via convex relaxation so that gradient-based adversarial attacks become plausible for GNNs. Benchmarking on node classification tasks using GNNs, our gradient-based topology attacks outperform current state-of-the-art attacks subject to the same topology perturbation budget. This demonstrates the effectiveness of our attack generation method through the lens of convex relaxation and first-order optimization. Moreover, by leveraging our proposed gradient-based attack, we propose the first optimization-based adversarial training technique for GNNs, yielding significantly improved robustness against gradient-based and greedy topology attacks.

Our new attack generation and adversarial training methods for GNNs are built upon the theoretical foundation of spectral graph theory, first-order optimization, and robust (mini-max) optimization. We summarize our main contributions as follows:

- We propose a general first-order attack generation framework under two attacking scenarios: a) attacking a pre-defined GNN and b) attacking a re-trainable GNN. This yields two new topology attacks: projected gradient descent (PGD) topology attack and min-max topology attack. Experimental results show that the proposed attacks outperform current state-of-the-art attacks.
- With the aid of our first-order attack generation methods, we propose an adversarial training method for GNNs to improve their robustness. The effectiveness of our method is shown by the considerable improvement of robustness on GNNs against both optimization-based and greedy-search-based topology attacks.

## 2 Related Works

Some recent attentions have been paid to the robustness of graph neural network. Both [Zügner *et al.*, 2018] and [Dai *et*

[*]Equal contribution

*al.*, 2018] studied adversarial attacks on neural networks for graph data. [Dai *et al.*, 2018] studied test-time non-targeted adversarial attacks on both graph classification and node classification. Their work restricted the attacks to perform modifications on discrete structures, that is, an attacker is only allowed to add or delete edges from a graph to construct a new graph. White-box, practical black-box and restricted black-box graph adversarial attack scenarios were studied. Authors in [Zügner *et al.*, 2018] considered both test-time (evasion) and training-time (data poisoning) attacks on node classification task. In contrast to [Dai *et al.*, 2018], besides adding or removing edges in the graph, attackers in [Zügner *et al.*, 2018] may modify node attributes. They designed adversarial attacks based on a static surrogate model and evaluated their impact by training a classifier on the data modified by the attack. The resulting attack algorithm is for targeted attacks on single nodes. It was shown that small perturbations on the graph structure and node features are able to achieve misclassification of a target node. A data poisoning attack on unsupervised node representation learning, or node embeddings, has been proposed in [Bojcheski and Günnemann, 2018]. This attack is based on perturbation theory to maximize the loss obtained from DeepWalk [Perozzi *et al.*, 2014]. In [Zügner and Günnemann, 2019], training-time attacks on GNNs were also investigated for node classification by perturbing the graph structure. The authors solved a min-max problem in training-time attacks using meta-gradients and treated the graph topology as a hyper-parameter to optimize.

## 3 Problem Statement

We begin by providing preliminaries on GNNs. We then formalize the attack threat model of GNNs in terms of edge perturbations, which we refer as 'topology attack'.

### 3.1 Preliminaries on GNNs

It has been recently shown in [Kipf and Welling, 2016; Veličković *et al.*, 2017; Xu *et al.*, 2019a] that GNN is powerful in transductive learning, e.g., node classification under graph data. That is, given a single network topology with node features and a known subset of node labels, GNNs are efficient to infer the classes of unlabeled nodes. Prior to defining GNN, we first introduce the following graph notations. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote an undirected and unweighted graph, where $\mathcal{V}$ is the vertex (or node) set with cardinality $|\mathcal{V}| = N$, and $\mathcal{E} \in (\mathcal{V} \times \mathcal{V})$ denotes the edge set with cardinality $|\mathcal{E}| = M$. Let $\mathbf{A}$ represent a binary adjancency matrix. By definition, we have $A_{ij} = 0$ if $(i, j) \notin \mathcal{E}$. In a GNN, we assume that each node $i$ is associated with a feature vector $\mathbf{x}_i \in \mathbb{R}^{M_0}$ and a scalar label $y_i$. The goal of GNN is to predict the class of an unlabeled node under the graph topology $\mathbf{A}$ and the training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N_{\text{train}}}$. Here GNN uses input features of all nodes but only $N_{\text{train}} < N$ nodes with labeled classes in the training phase.

Formally, the $k$th layer of a GNN model obeys the propagation rule of the generic form

$$\mathbf{h}_i^{(k)} = g^{(k)}\left(\{\mathbf{W}^{(k-1)}\mathbf{h}_j^{(k-1)}\tilde{A}_{ij},\ \forall j \in \mathcal{N}(i)\}\right),\ \forall i \in [N] \quad (1)$$

where $\mathbf{h}_i^{(k)} \in \mathbb{R}^{M_k}$ denotes the feature vector of node $i$ at layer $k$, $\mathbf{h}_i^{(0)} = \mathbf{x}_i \in \mathbb{R}^{M_0}$ is the input feature vector of node $i$, $g^{(k)}$ is a possible composite mapping (activation) function, $\mathbf{W}^{(k-1)} \in \mathbb{R}^{M_k \times M_{k-1}}$ is the trainable weight matrix at layer $(k-1)$, $\tilde{A}_{ij}$ is the $(i,j)$th entry of $\tilde{\mathbf{A}}$ that denotes a linear mapping of $\mathbf{A}$ but with the same sparsity pattern, and $\mathcal{N}(i)$ denotes node $i$'s neighbors together with itself, i.e., $\mathcal{N}(i) = \{j | (i,j) \in \mathcal{E}, \text{ or } j = i\}$.

A special form of GNN is graph convolutional networks (GCN) [Kipf and Welling, 2016]. This is a recent approach of learning on graph structures using convolution operations which is promising as an embedding methodology. In GCNs, the propagation rule (1) becomes [Kipf and Welling, 2016]

$$\mathbf{h}_i^{(k)} = \sigma\left(\sum_{j \in \mathcal{N}_i}\left(\mathbf{W}^{(k-1)}\mathbf{h}_j^{(k-1)}\tilde{A}_{ij}\right)\right), \quad (2)$$

where $\sigma(\cdot)$ is the ReLU function. Let $\tilde{A}_{i,:}$ denote the $i$th row of $\tilde{\mathbf{A}}$ and $\mathbf{H}^{(k)} = \left[(\mathbf{h}_1^{(k)})^\top; \ldots; (\mathbf{h}_N^{(k)})^\top\right]$, we then have the standard form of GCN,

$$\mathbf{H}^{(k)} = \sigma\left(\tilde{\mathbf{A}}\mathbf{H}^{(k-1)}(\mathbf{W}^{(k-1)})^\top\right). \quad (3)$$

Here $\tilde{\mathbf{A}}$ is given by a normalized adjacency matrix $\tilde{\mathbf{A}} = \hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}$, where $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, and $\hat{\mathbf{D}}_{ij} = 0$ if $i \neq j$ and $\hat{\mathbf{D}}_{ii} = \mathbf{1}^\top \hat{\mathbf{A}}_{:,i}$.

### 3.2 Topology Attack in Terms of Edge Perturbation

We introduce a Boolean symmetric matrix $\mathbf{S} \in \{0,1\}^{N \times N}$ to encode whether or not an edge in $\mathcal{G}$ is modified. That is, the edge connecting nodes $i$ and $j$ is modified (added or removed) if and only if $S_{ij} = S_{ji} = 1$. Otherwise, $S_{ij} = 0$ if $i = j$ or the edge $(i,j)$ is not perturbed. Given the adjacency matrix $\mathbf{A}$, its supplement is given by $\bar{\mathbf{A}} = \mathbf{1}\mathbf{1}^T - \mathbf{I} - \mathbf{A}$, where $\mathbf{I}$ is an identity matrix, and $(\mathbf{1}\mathbf{1}^T - \mathbf{I})$ corresponds to the fully-connected graph. With the aid of edge perturbation matrix $\mathbf{S}$ and $\bar{\mathbf{A}}$, a perturbed graph topology $\mathbf{A}'$ against $\mathbf{A}$ is given by

$$\mathbf{A}' = \mathbf{A} + \mathbf{C} \circ \mathbf{S},\ \mathbf{C} = \bar{\mathbf{A}} - \mathbf{A}, \quad (4)$$

where $\circ$ denotes the element-wise product. In (4), the positive entry of $\mathbf{C}$ denotes the edge that can be added to the graph $\mathbf{A}$, and the negative entry of $\mathbf{C}$ denotes the edge that can be removed from $\mathbf{A}$. We then formalize the concept of *topology attack* to GNNs: Finding minimum edge perturbations encoded by $\mathbf{S}$ in (4) to mislead GNNs. A more detailed attack formulation will be studied in the next section.

## 4 Topology Attack Generation: A First-Order Optimization Perspective

In this section, we first define attack loss (beyond the conventional cross-entropy loss) under different attacking scenarios. We then develop two efficient attack generation methods by leveraging first-order optimization. We call the resulting attacks projected gradient descent (PGD) topology attack and min-max topology attack, respectively.

### 4.1 Attack Loss & Attack Generation

Let $\mathbf{Z}(\mathbf{S}, \mathbf{W}; \mathbf{A}, \{\mathbf{x}_i\})$ denote the prediction probability of a GNN specified by $\mathbf{A}'$ in (4) and $\mathbf{W}$ under input features $\{\mathbf{x}_i\}$. Then $Z_{i,c}$ denotes the probability of assigning node $i$ to class $c$. It has been shown in existing works [Goodfellow *et al.*, 2015; Kurakin *et al.*, 2017] that the *negative cross-entropy (CE) loss* between the true labels ($y_i$) and the predicted labels ($\{Z_{i,c}\}$) can be used as an attack loss at node $i$, denoted by $f_i(\mathbf{S}, \mathbf{W}; \mathbf{A}, \{\mathbf{x}_i\}, y_i)$. We can also propose a *CW-type loss* similar to Carlili-Wagner (CW) attacks for attacking image classifiers [Carlini and Wagner, 2017],

$$f_i(\mathbf{S}, \mathbf{W}; \mathbf{A}, \{\mathbf{x}_i\}, y_i) = \max\left\{ Z_{i,y_i} - \max_{c \neq y_i} Z_{i,c}, -\kappa \right\}, \quad (5)$$

where $\kappa \geq 0$ is a confidence level of making wrong decisions.

To design topology attack, we seek $\mathbf{S}$ in (4) to minimize the per-node attack loss (CE-type or CW-type) given a finite budge of edge perturbations. We consider two threat models: a) attacking a pre-defined GNN with known $\mathbf{W}$; b) attacking an interactive GNN with re-trainable $\mathbf{W}$. In the case a) of fixed $\mathbf{W}$, the attack generation problem can be cast as

$$\begin{array}{ll} \underset{\mathbf{s}}{\text{minimize}} & \sum_{i \in \mathcal{V}} f_i(\mathbf{s}; \mathbf{W}, \mathbf{A}, \{\mathbf{x}_i\}, y_i) \\ \text{subject to} & \mathbf{1}^\top \mathbf{s} \leq \epsilon,\ \mathbf{s} \in \{0,1\}^n, \end{array} \quad (6)$$

where we replace the symmetric matrix variable $\mathbf{S}$ with its vector form that consists of $n := N(N-1)/2$ unique perturbation variables in $\mathbf{S}$. We recall that $f_i$ could be either a CE-type or a CW-type per-node attack loss. In the case b) of re-trainable $\mathbf{W}$, the attack generation problem has the following min-max form

$$\underset{\mathbf{1}^\top \mathbf{s} \leq \epsilon, \mathbf{s} \in \{0,1\}^n}{\text{minimize}} \ \underset{\mathbf{W}}{\text{maximize}} \sum_{i \in \mathcal{V}} f_i(\mathbf{s}, \mathbf{W}; \mathbf{A}, \{\mathbf{x}_i\}, y_i), \quad (7)$$

where the inner maximization aims to constrain the attack loss by retraining $\mathbf{W}$ so that attacking GNN is more difficult.

Motivated by targeted adversarial attacks against image classifiers [Carlini and Wagner, 2017], we can define targeted topology attacks that are restricted to perturb edges of targeted nodes. In this case, we require to linearly constrain $\mathbf{S}$ in (4) as $S_{i,\cdot} = 0$ if $i$ is not a target node. As a result, both attack formulations (6) and (7) have extra linear constraints with respect to $s$, which can be readily handled by the optimization solver introduced later. Without loss of generality, we focus on untargeted topology attacks in this paper.

### 4.2 PGD Topology Attack

Problem (6) is a combinatorial optimization problem due to the presence of Boolean variables. For ease of optimization, we relax $\mathbf{s} \in \{0,1\}^n$ to its convex hull $s \in [0,1]^n$ and solve the resulting continuous optimization problem,

$$\begin{array}{ll} \underset{\mathbf{s}}{\text{minimize}} & f(\mathbf{s}) := \sum_{i \in \mathcal{V}} f_i(\mathbf{s}; \mathbf{W}, \mathbf{A}, \{\mathbf{x}_i\}, y_i) \\ \text{subject to} & \mathbf{s} \in \mathcal{S}, \end{array} \quad (8)$$

where $\mathcal{S} = \{\mathbf{s} \,|\, \mathbf{1}^T s \leq \epsilon, \mathbf{s} \in [0,1]^n\}$. Suppose that the solution of problem (8) is achievable, the remaining question is how to recover a binary solution from it. Since the variable $\mathbf{s}$ in (8) can be interpreted as a probabilistic vector, a randomization sampling [Liu *et al.*, 2016] is suited for generating a near-optimal binary topology perturbation; see details in Algorithm 1.

**Algorithm 1** Random sampling from probabilistic to binary topology perturbation

1: Input: probabilistic vector $\mathbf{s}$, $K$ is # of random trials
2: **for** $k = 1, 2, \ldots, K$ **do**
3: draw binary vector $\mathbf{u}^{(k)}$ following

$$u_i^{(k)} = \begin{cases} 1 & \text{with probability } s_i \\ 0 & \text{with probability } 1 - s_i \end{cases}, \forall i \quad (9)$$

4: **end for**
5: choose a vector $\mathbf{s}^*$ from $\{\mathbf{u}^{(k)}\}$ which yields the smallest attack loss $f(\mathbf{u}^{(k)})$ under $\mathbf{1}^T \mathbf{s} \leq \epsilon$.

We solve the continuous optimization problem (8) by projected gradient descent (PGD),

$$\mathbf{s}^{(t)} = \Pi_{\mathcal{S}}\left[\mathbf{s}^{(t-1)} - \eta_t \hat{\mathbf{g}}_t\right], \quad (10)$$

where $t$ denotes the iteration index of PGD, $\eta_t > 0$ is the learning rate at iteration $t$, $\hat{\mathbf{g}}_t = \nabla f(\mathbf{s}^{(t-1)})$ denotes the gradient of the attack loss $f$ evaluated at $\mathbf{s}^{(t-1)}$, and $\Pi_{\mathcal{S}}(\mathbf{a}) := \arg\min_{\mathbf{s} \in \mathcal{S}} \|\mathbf{s} - \mathbf{a}\|_2^2$ is the projection operator at $\mathbf{a}$ over the constraint set $\mathcal{S}$. In Proposition 1, we show that the projection operation yields the closed-form solution.

**Proposition 1** *Given $\mathcal{S} = \{\mathbf{s} \,|\, \mathbf{1}^T s \leq \epsilon, \mathbf{s} \in [0,1]^n\}$, the projection operation at the point $\mathbf{a}$ with respect to $\mathcal{S}$ is*

$$\Pi_{\mathcal{S}}(\mathbf{a}) = \begin{cases} P_{[0,1]}[\mathbf{a} - \mu \mathbf{1}] & \text{If } \mu > 0 \text{ and} \\ & \mathbf{1}^T P_{[0,1]}[\mathbf{a} - \mu \mathbf{1}] = \epsilon, \\ P_{[0,1]}[\mathbf{a}] & \text{If } \mathbf{1}^T P_{[0,1]}[\mathbf{a}] \leq \epsilon, \end{cases} \quad (11)$$

*where $P_{[0,1]}(x) = x$ if $x \in [0,1]$, $0$ if $x < 0$, and $1$ if $x > 1$.*

**Proof**: We express the projection problem as

$$\begin{array}{ll} \underset{\mathbf{s}}{\text{minimize}} & \frac{1}{2}\|\mathbf{s} - \mathbf{a}\|_2^2 + \mathcal{I}_{[\mathbf{0},\mathbf{1}]}(\mathbf{s}) \\ \text{subject to} & \mathbf{1}^\top \mathbf{s} \leq \epsilon, \end{array} \quad (12)$$

where $\mathcal{I}_{[\mathbf{0},\mathbf{1}]}(\mathbf{s}) = 0$ if $\mathbf{s} \in [0,1]^n$, and $\infty$ otherwise.

The Lagrangian function of problem (12) is given by

$$\begin{aligned} &\frac{1}{2}\|\mathbf{s} - \mathbf{a}\|_2^2 + \mathcal{I}_{[\mathbf{0},\mathbf{1}]}(\mathbf{s}) + \mu(\mathbf{1}^\top \mathbf{s} - \epsilon) \\ &= \sum_i \left( \frac{1}{2}(s_i - a_i)^2 + \mathcal{I}_{[0,1]}(s_i) + \mu s_i \right) - \mu\epsilon, \end{aligned} \quad (13)$$

where $\mu \geq 0$ is the dual variable. The minimizer to the above Lagrangian function (with respect to the variable $\mathbf{s}$) is

$$\mathbf{s} = P_{[\mathbf{0},\mathbf{1}]}(\mathbf{a} - \mu \mathbf{1}), \quad (14)$$

where $P_{[\mathbf{0},\mathbf{1}]}$ is taken elementwise. Besides the stationary condition (14), other KKT conditions for solving problem (12) are

$$\mu(\mathbf{1}^\top \mathbf{s} - \epsilon) = 0, \quad (15)$$

$$\mu \geq 0, \quad (16)$$

$$\mathbf{1}^\top \mathbf{s} \leq \epsilon. \quad (17)$$

If $\mu > 0$, then the solution to problem (12) is given by (14), where the dual variable $\mu$ is determined by (14) and (15)

$$\mathbf{1}^T P_{[\mathbf{0},\mathbf{1}]}[\mathbf{a} - \mu \mathbf{1}] = \epsilon, \text{ and } \mu > 0. \quad (18)$$

If $\mu = 0$, then the solution to problem (12) is given by (14) and (17),

$$\mathbf{s} = P_{[\mathbf{0},\mathbf{1}]}(\mathbf{a}), \text{ and } \mathbf{1}^\top \mathbf{s} \leq \epsilon, \tag{19}$$

The proof is complete. □

In the projection operation (11), one might need to solve the scalar equation $\mathbf{1}^T P_{[\mathbf{0},\mathbf{1}]}[\mathbf{a} - \mu\mathbf{1}] = \epsilon$ with respect to the dual variable $\mu$. This can be accomplished by applying the bisection method [Boyd and Vandenberghe, 2004; Liu *et al.*, 2015] over $\mu \in [\min(\mathbf{a} - \mathbf{1}), \max(\mathbf{a})]$. That is because $\mathbf{1}^T P_{[\mathbf{0},\mathbf{1}]}[\mathbf{a} - \max(\mathbf{a})\mathbf{1}] \leq \epsilon$ and $\mathbf{1}^T P_{[\mathbf{0},\mathbf{1}]}[\mathbf{a} - \min(\mathbf{a} - \mathbf{1})\mathbf{1}] \geq \epsilon$, where $\max$ and $\min$ return the largest and smallest entry of a vector. We remark that the bisection method converges in the logarithmic rate given by $\log_2[(\max(\mathbf{a}) - \min(\mathbf{a} - \mathbf{1}))/\xi]$ for the solution of $\xi$-error tolerance. We summarize the PGD topology attack in Algorithm 2.

**Algorithm 2** PGD topology attack on GNN

1: Input: $\mathbf{s}^{(0)}$, $\epsilon > 0$, learning rate $\eta_t$, and iterations $T$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: gradient descent: $\mathbf{a}^{(t)} = \mathbf{s}^{(t-1)} - \eta_t \nabla f(\mathbf{s}^{(t-1)})$
4: call projection operation in (11)
5: **end for**
6: call Algorithm 1 to return $\mathbf{s}^*$, and the resulting $\mathbf{A}'$ in (4).

### 4.3 Min-max Topology Attack

We next solve the problem of min-max attack generation in (7). By convex relaxation on the Boolean variables, we obtain the following continuous optimization problem

$$\underset{\mathbf{s}\in\mathcal{S}}{\text{minimize}} \underset{\mathbf{W}}{\text{maximize}} \; f(\mathbf{s}, \mathbf{W}) = \sum_{i\in\mathcal{V}} f_i(\mathbf{s}, \mathbf{W}; \mathbf{A}, \{\mathbf{x}_i\}, y_i), \tag{20}$$

where $\mathcal{S}$ has been defined in (8). We solve problem (20) by first-order alternating optimization [Lu *et al.*, 2019a; 2019b], where the inner maximization is solved by gradient ascent, and the outer minimization is handled by PGD same as (10). We summarize the min-max topology attack in Algorithm 3. We remark that one can perform multiple maximization steps within each iteration of alternating optimization. This strikes a balance between the computation efficiency and the convergence accuracy [Chen *et al.*, 2017; Qian *et al.*, 2018].

## 5 Robust Training for GNNs

With the aid of first-order attack generation methods, we now introduce our adversarial training for GNNs via robust optimization. Similar formulation is also used in [Madry *et al.*, 2017]. In adversarial training, we solve a min-max problem for robust optimization:

$$\underset{\mathbf{W}}{\text{minimize}} \underset{\mathbf{s}\in\mathcal{S}}{\text{maximize}} \; -f(\mathbf{s}, \mathbf{W}), \tag{21}$$

where $f(\mathbf{x}, \mathbf{W})$ denotes the attack loss specified in (20). Following the idea of adversarial training for image classifiers in [Madry *et al.*, 2017], we restrict the loss function $f$ as the CE-type loss. This formulation tries to minimize the training loss at the presence of topology perturbations.

**Algorithm 3** Min-max topology attack to solve (20)

1: Input: given $\mathbf{W}^{(0)}$, $\mathbf{s}^{(0)}$, learning rates $\beta_t$ and $\eta_t$, and iteration numbers $T$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: inner maximization over $\mathbf{W}$: given $\mathbf{s}^{(t-1)}$, obtain

$$\mathbf{W}^t = \mathbf{W}^{t-1} + \beta_t \nabla_{\mathbf{W}} f(\mathbf{s}^{t-1}, \mathbf{W}^{t-1})$$

4: outer minimization over $\mathbf{s}$: given $\mathbf{W}^{(t)}$, running PGD (10), where $\hat{\mathbf{g}}_t = \nabla_{\mathbf{s}} f(\mathbf{s}^{t-1}, \mathbf{W}^t)$
5: **end for**
6: call Algorithm 1 to return $\mathbf{s}^*$, and the resulting $\mathbf{A}'$ in (4).

We note that problems (21) and (7) share a very similar min-max form, however, they are not equivalent since the loss $f$ is neither convex with respect to $\mathbf{s}$ nor concave with respect to $\mathbf{W}$, namely, lacking saddle point property [Boyd and Vandenberghe, 2004]. However, there exists connection between (7) and (21); see Proposition 2.

**Proposition 2** *Given a general attack loss function $f$, problem* (21) *is equivalent to*

$$\underset{\mathbf{W}}{\textit{maximize}} \underset{\mathbf{s}\in\mathcal{S}}{\textit{minimize}} \; f(\mathbf{s}, \mathbf{W}), \tag{22}$$

*which further yields* (22) $\leq$ (7).

**Proof**: By introducing epigraph variable $p$ [Boyd and Vandenberghe, 2004], problem (21) can be rewritten as

$$\begin{array}{ll} \underset{\mathbf{W},p}{\text{minimize}} & p \\ \text{subject to} & -f(\mathbf{s}, \mathbf{W}) \leq p, \forall \mathbf{s} \in \mathcal{S}. \end{array} \tag{23}$$

By changing variable $q := -p$, problem (23) is equivalent to

$$\begin{array}{ll} \underset{\mathbf{W},q}{\text{maximize}} & q \\ \text{subject to} & f(\mathbf{s}, \mathbf{W}) \geq q, \forall \mathbf{s} \in \mathcal{S}. \end{array} \tag{24}$$

By eliminating the epigraph variable $q$, problem (24) becomes (22). By *max-min inequality* [Boyd and Vandenberghe, 2004, Sec. 5.4], we finally obtain that

$$\underset{\mathbf{W}}{\text{maximize}} \underset{\mathbf{s}\in\mathcal{S}}{\text{minimize}} \; f(\mathbf{s}, \mathbf{W}) \leq \underset{\mathbf{s}\in\mathcal{S}}{\text{minimize}} \underset{\mathbf{W}}{\text{maximize}} \; f(\mathbf{s}, \mathbf{W}).$$

The proof is now complete. □

We summarize the robust training algorithm in Algorithm 4 for solving problem (22). Similar to Algorithm 3, one usually performs multiple inner minimization steps (with respect to $\mathbf{s}$) within each iteration $t$ to have a solution towards minimizer during alternating optimization. This improves the stability of convergence in practice [Qian *et al.*, 2018; Madry *et al.*, 2017].

## 6 Experiments

In this section, we present our experimental results for both topology attack and defense methods on a graph convolutional networks (GCN) [Kipf and Welling, 2016]. We demonstrate the misclassification rate and the convergence of the proposed 4 attack methods: negative cross-entropy loss via PGD attack (CE-PGD), CW loss via PGD attack (CW-PGD), negative cross-entropy loss via min-max attack (CE-min-max), CW loss via min-max attack (CW-min-max). We then show the improved robustness of GCN by leveraging our proposed robust training against topology attacks.

**Algorithm 4** Robust training for solving problem (22)

1: Input: given $\mathbf{W}^{(0)}$, $\mathbf{s}^{(0)}$, learning rates $\beta_t$ and $\eta_t$, and iteration numbers $T$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: inner minimization over $\mathbf{s}$: given $\mathbf{W}^{(t-1)}$, running PGD (10), where $\hat{\mathbf{g}}_t = \nabla_{\mathbf{s}} f(\mathbf{s}^{t-1}, \mathbf{W}^{t-1})$
4: outer maximization over $\mathbf{W}$: given $\mathbf{s}^{(t)}$, obtain

$$\mathbf{W}^t = \mathbf{W}^{t-1} + \beta_t \nabla_{\mathbf{W}} f(\mathbf{s}^t, \mathbf{W}^{t-1})$$

5: **end for**
6: return $\mathbf{W}^T$.

### 6.1 Experimental Setup

We evaluate our methods on two well-known datasets: Cora and Citeseer [Sen *et al.*, 2008]. Both datasets contain unweighted edges which can be generated as symmetric adjacency matrix $\mathbf{A}$ and sparse bag-of-words feature vectors which can be treated the input of GCN. To train the model, all node feature vectors are fed into GCN but with only 140 and 120 labeled nodes for Cora and Citeseer, respectively. The number of test labeled nodes is 1000 for both datasets. At each experiment, we repeat 5 times based on different splits of training/testing nodes and report *mean* $\pm$ *standard deviation* of misclassification rate (namely, $1 -$ prediction accuracy) on testing nodes. Source code avaliable at `https://github.com/KaidiXu/GCN_ADV_Train`

### 6.2 Attack Performance

We compare our four attack methods (CE-PGD, CW-PGD, CE-min-max, CW-min-max) with DICE ('delete edges internally, connect externally') [Waniek *et al.*, 2018], Meta-Self attack [Zügner and Günnemann, 2019] and greedy attack, a variant of Meta-Self attack without weight re-training for GCN. The greedy attack is considered as a fair comparison with our CE-PGD and CW-PGD attacks, which are generated on a fixed GCN without weight re-training. In min-max attacks (CE-min-max and CW-min-max), we show misclassification rates against both natural and retrained models from Algorithm 3, and compare them with the state-of-the-art Meta-Self attack. For a fair comparison, we use the same performance evaluation criterion in Meta-Self, testing nodes' predicted labels (not their ground-truth label) by an independent pre-trained model that can be used during the attack. In the attack problems (6) and (7), unless specified otherwise the maximum number of perturbed edges is set to be $5\%$ of the total number of existing edges in the original graph. In Algorithm 1, we set the iteration number of random sampling as $K = 20$ and choose the perturbed topology with the highest misclassification rate which also satisfies the edge perturbation constraint.

In Table 1, we present the misclassification rate of different attack methods against both natural and retrained model from (20). Here we recall that the retrained model arises due to the scenario of attacking an interactive GCN with re-trainable weights (Algorithm 3). For comparison, we also show the misclassification rate of a natural model with the true topology (denoted by 'clean'). As we can see, to attack the natural model, our proposed attacks achieve better misclassification rate than the existing methods. We also observe that compared to min-max attacks (CE-min-max and CW-min-max), CE-PGD and CW-PGD yield better attacking performance since it is easier to attack a pre-defined GCN. To attack the model that allows retraining, we set 20 steps of inner maximization per iteration of Algorithm 3. The results show that our proposed min-max attack achieves very competitive performance compared to Meta-Self attack. Note that evaluating the attack performance on the retrained model obtained from (20) is not quite fair since the retrained weights could be suboptimal and induce degradation in classification.

| | | Cora | Citeseer |
|---|---|---|---|
| fixed natural model | clean | $18.2 \pm 0.1$ | $28.9 \pm 0.3$ |
| | DICE | $18.9 \pm 0.2$ | $29.8 \pm 0.4$ |
| | Greedy | $25.2 \pm 0.2$ | $34.6 \pm 0.3$ |
| | Meta-Self | $22.7 \pm 0.3$ | $31.2 \pm 0.5$ |
| | **CE-PGD** | $\mathbf{28.0 \pm 0.1}$ | $36.0 \pm 0.2$ |
| | **CW-PGD** | $27.8 \pm 0.4$ | $\mathbf{37.1 \pm 0.5}$ |
| | **CE-min-max** | $26.4 \pm 0.1$ | $34.1 \pm 0.3$ |
| | **CW-min-max** | $26.0 \pm 0.3$ | $34.7 \pm 0.6$ |
| retrained model from (20) | Meta-Self | $29.6 \pm 0.4$ | $\mathbf{39.7 \pm 0.3}$ |
| | **CE-min-max** | $\mathbf{30.8 \pm 0.2}$ | $37.5 \pm 0.3$ |
| | **CW-min-max** | $30.5 \pm 0.5$ | $39.6 \pm 0.4$ |

Table 1: Misclassification rates (%) under 5% perturbed edges

In Fig. 1, we present the CE-loss and the CW-loss of the proposed topology attacks against the number of iterations in Algorithm 2. Here we choose $T = 200$ and $\eta_t = 200/\sqrt{t}$. As we can see, the method of PGD converges gracefully against iterations. This verifies the effectiveness of the first-order optimization based attack generation method.

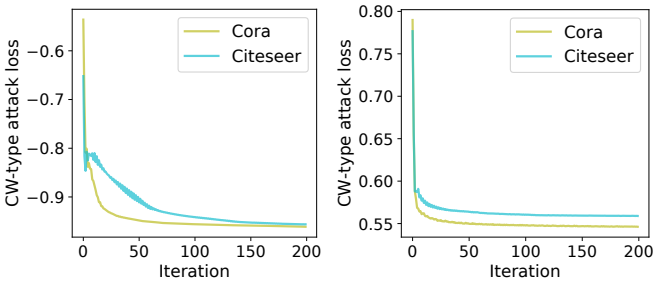


Figure 1: CE-PGD and CW-PGD attack losses on Cora and Citeseer datasets.

### 6.3 Defense Performance

In what follows, we invoke Algorithm 4 to generate robust GCN via adversarial training. We set $T = 1000$, $\beta_t = 0.01$ and $\eta_t = 200/\sqrt{t}$. We run 20 steps for inner minimization. Inspired by [Madry *et al.*, 2017], we increase the hidden units from 16 to 32 in order to create more capacity for this more complicated classifier. Initially, we set the maximum number of edges we can modify as $5\%$ of total existing edges.

In Figure 2, we present convergence of our robust training. As we can see, the loss drops reasonably and the $1,000$ iterations are necessary for robust training rather than normal training process which only need $200$ iterations. We also observe that our robust training algorithm does not harm the test accuracy when $\epsilon = 5\%$, but successfully improves the robustness as the attack success rate drops from $28.0\%$ to $22.0\%$ in Cora dataset as shown in Table 2,

After showing the effectiveness of our algorithm, we explore deeper in adversarial training on GCN. We aim to show how large $\epsilon$ we can use in robust training. So we set $\epsilon$ from $5\%$ to $20\%$ and apply CE-PGD attack following the same $\epsilon$ setting. The results are presented in Table 3. Note that when $\epsilon = 0$, the first row shows misclassification rates of test nodes on natural graph as the baseline for *lowest* misclassification rate we can obtain; the first column shows the CE-PGD attack misclassification rates of natural model as the baseline for *highest* misclassification rate we can obtain. We can conclude that when a robust model trained under an $\epsilon$ constraint, the model will gain robustness under this $\epsilon$ distinctly. Considering its importance to keep the original graph test performance, we suggest generating robust model under $\epsilon = 0.1$. Moreover, please refer to Figure 3 that a) our robust trained model can provide universal defense to CE-PGD, CW-PGD and Greedy attacks; b) when increasing $\epsilon$, the difference between both test accuracy and CE-PGD attack accuracy increases substantially, which also implies the robust model under larger $\epsilon$ is harder to obtain.

| | Cora | Citeseer |
|---|---|---|
| $\mathbf{A}$/natural model | $18.2 \pm 0.1$ | $28.9 \pm 0.1$ |
| $\mathbf{A}$/robust model | $18.1 \pm 0.3$ | $28.7 \pm 0.4$ |
| $\mathbf{A}'$/natural model | $28.0 \pm 0.1$ | $36.0 \pm 0.2$ |
| $\mathbf{A}'$/robust model | $22.0 \pm 0.2$ | $32.2 \pm 0.4$ |

Table 2: Misclassification rates (%) of robust training (smaller is better for defense task) with at most $5\%$ of edge perturbations. $\mathbf{A}$ means the natural graph, $\mathbf{A}'$ means the generated adversarial graph under $\epsilon = 5\%$. $\mathbf{X}/M$ means the misclassification rate of using model $M$ on graph $\mathbf{X}$.

| | $\epsilon$ in robust training (in %) | | | | |
|---|---|---|---|---|---|
| $\epsilon$ in attack (in %) | 0 | 5 | 10 | 15 | 20 |
| 0 | 18.1 | 18.2 | 19.0 | 20.2 | 21.3 |
| 5 | 27.9 | 22.0 | 23.9 | 24.8 | 26.5 |
| 10 | 32.7 | 32.1 | 26.4 | 27.7 | 31.0 |
| 15 | 36.7 | 36.2 | 33.4 | 29.7 | 32.9 |
| 20 | 40.2 | 40.1 | 36.3 | 36.3 | 33.5 |

Table 3: Misclassification rates (%) of CE-PGD attack against robust training model versus (smaller is better) different $\epsilon$ (%) on Cora dataset. Here $\epsilon = 0$ in training means natural model and $\epsilon = 0$ in attack means unperturbed topology.

## 7 Conclusion

In this paper, we first introduce an edge perturbation based topology attack framework that overcomes the difficulty of attacking discrete graph structure data from a first-order optimization perspective. Our extensive experiments show that with only a fraction of edges changed, we are able to compromise state-of-the-art graph neural networks model noticeably. Additionally, we propose an adversarial training framework to improve the robustness of GNN models based on our attack methods. Experiments on different datasets show that our method is able to improve the GNN model's robustness against both gradient based and greedy search based attack methods without classification performance drop on original graph. We believe that this paper provides potential means for theoretical study and improvement of the robustness of deep learning models on graph data.

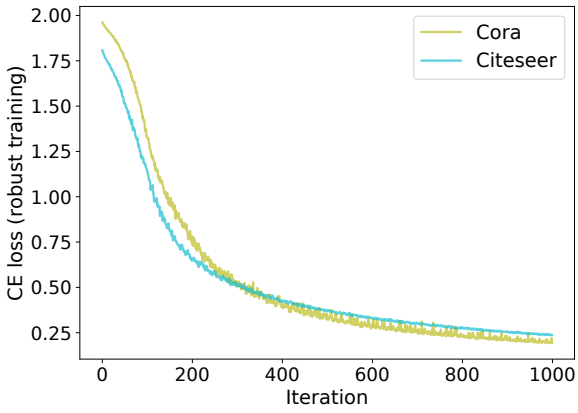


Figure 2: Robust training loss on Cora and Citeseer datasets.

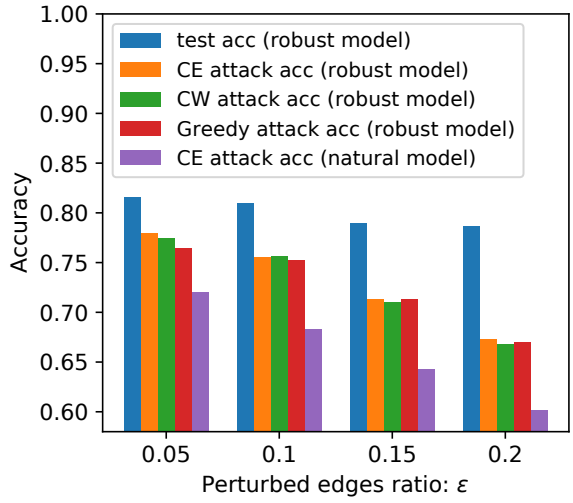


Figure 3: Test accuracy of robust model (no attack), CE-PGD attack against robust model, CW-PGD attack against robust model, Greedy attack against robust model and CE-PGD attack against natural model for different $\epsilon$ used in robust training and test on Cora dataset.

## Acknowledgments

This work is supported by Air Force Research Laboratory FA8750-18-2-0058 and the MIT-IBM Watson AI Lab.

## References


[Bojcheski and Günnemann, 2018] Aleksandar Bojcheski and Stephan Günnemann. Adversarial attacks on node embeddings. *arXiv preprint arXiv:1809.01093*, 2018.

[Boyd and Vandenberghe, 2004] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[Carlini and Wagner, 2017] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[Carlini and Wagner, 2018] N. Carlini and D. Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 1–7. IEEE, 2018.

[Chen *et al.*, 2017] R. S. Chen, B. Lucier, Y. Singer, and V. Syrgkanis. Robust optimization for non-convex objectives. In *NeurIPS*, pages 4705–4714, 2017.

[Chen *et al.*, 2018a] Hongge Chen, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, and Cho-Jui Hsieh. Attacking visual language grounding with adversarial examples: A case study on neural image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2587–2597, 2018.

[Chen *et al.*, 2018b] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. EAD: elastic-net attacks to deep neural networks via adversarial examples. *AAAI*, 2018.

[Dai *et al.*, 2018] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song. Adversarial attack on graph structured data. *arXiv preprint arXiv:1806.02371*, 2018.

[Goodfellow *et al.*, 2015] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

[Kipf and Welling, 2016] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Kurakin *et al.*, 2017] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *ICLR*, 2017.

[Liu *et al.*, 2015] S. Liu, S. Kar, M. Fardad, and P. K. Varshney. Sparsity-aware sensor collaboration for linear coherent estimation. *IEEE Transactions on Signal Processing*, 63(10):2582–2596, 2015.

[Liu *et al.*, 2016] S. Liu, S. P. Chepuri, M. Fardad, E. Maşazade, G. Leus, and P. K. Varshney. Sensor selection for estimation with correlated measurement noise. *IEEE Transactions on Signal Processing*, 64(13):3509–3522, 2016.

[Lu *et al.*, 2019a] S. Lu, R. Singh, X. Chen, Y. Chen, and M. Hong. Understand the dynamics of GANs via primal-dual optimization, 2019.

[Lu *et al.*, 2019b] S. Lu, I. Tsaknakis, and M. Hong. Block alternating optimization for non-convex min-max problems: Algorithms and applications in signal processing and communications. In *ICASSP*, 2019.

[Madry *et al.*, 2017] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[Qian *et al.*, 2018] Qi Qian, Shenghuo Zhu, Jiasheng Tang, Rong Jin, Baigui Sun, and Hao Li. Robust optimization over multiple domains. *CoRR*, abs/1805.07588, 2018.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[Veličković *et al.*, 2017] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[Waniek *et al.*, 2018] Marcin Waniek, Tomasz P Michalak, Michael J Wooldridge, and Talal Rahwan. Hiding individuals and communities in a social network. *Nature Human Behaviour*, 2(2):139, 2018.

[Xu *et al.*, 2019a] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[Xu *et al.*, 2019b] Kaidi Xu, Sijia Liu, Gaoyuan Zhang, Mengshu Sun, Pu Zhao, Quanfu Fan, Chuang Gan, and Xue Lin. Interpreting adversarial examples by activation promotion and suppression. *arXiv preprint arXiv:1904.02057*, 2019.

[Xu *et al.*, 2019c] Kaidi Xu, Sijia Liu, Pu Zhao, Pin-Yu Chen, Huan Zhang, Quanfu Fan, Deniz Erdogmus, Yanzhi Wang, and Xue Lin. Structured adversarial attack: Towards general implementation and better interpretability. In *International Conference on Learning Representations*, 2019.

[Zügner *et al.*, 2018] D. Zügner, A. Akbarnejad, and S. Günnemann. Adversarial attacks on neural networks for graph data. In *KDD*. ACM, 2018.

[Zügner and Günnemann, 2019] Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *ICLR*, 2019.