Not All Low-Pass Filters are Robust in Graph Convolutional Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph Convolutional Networks (GCNs) are promising deep learning approaches in the representation learning for graph-structured data. However, despite the proliferation of such methods, it is known that they are vulnerable to carefully crafted adversarial attacks on the graph structure. In this paper, we first conduct an adversarial vulnerability analysis based on matrix perturbation theory. We prove that the low-frequency components of the symmetric normalized Laplacian, which is usually used as the convolutional filter in GCNs, could be more robust against structural perturbations when their eigenvalues fall into a robust interval. Our results indicate that not all low-frequency components are robust to adversarial attacks and provide a deeper understanding of the relationship between graph spectrum and robustness of GCNs. Motivated by the theory, we present GCN-LFR, a general robust co-training paradigm for GCN-based models, that encourages transferring the robustness of low-frequency components with an auxiliary neural network. To this end, GCN-LFR could enhance the robustness of various kinds of GCN-based models against poisoning structural attacks in a plug-and-play manner. Extensive experiments across five benchmark datasets and five GCN-based models also confirm that GCN-LFR is resistant to the adversarial attacks without compromising on performance in the benign situation.

1 Introduction

Graph Convolutional Networks (GCNs) elaborate the expressive power of deep learning from grid-like data to graph-structured data and have achieved remarkable success in a wide variety of domains [6, 5, 10, 21, 19]. Just like CNNs, modern GCNs could promisingly learn both the local and global structural patterns of graphs through designed convolutions. However, the vulnerability of GCNs against adversarial attacks has been revealed recently [52, 9, 7]. The lack of robustness arouses concerns on applying GCNs in a variety of fields pertaining to security and privacy. Adversarial attacks on graphs aim to degrade the ability of representation learning of GCNs, and fool them to make wrong decisions by perturbing either node features or graph structures. Given the complexity of the underlying structural information and the ease of operation in practice [36], the majority of literature focuses on the adversarial attacks on structures by inserting/removing/rewiring adversarial edges, *i.e.*, structural attacks. In other words, structural attacks on graphs during the training stage of GCNs with node features unchanged becomes a commonly considered setting by attackers [17]. Therefore, it is of emerging importance to enhance the robustness of GCNs against structural attacks, which is also the aim of this work.

For a better understanding of the structural attacks, it naturally leads to the question: *Do the adversarial edges post equal influences on the graph spectrum*? Here, graph spectrum plays a significant role in Graph Signal Processing (GSP) [31, 28] and graph learning. Empirical observations

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

from recent papers suggest the answer could be "no". We can observe that the perturbations resulting from structural attacks express an implicit trend on the graph spectrum [11]. As the toy example shown in Figure 1, low-frequency components (far left) tend to be more robust in comparison with high-frequency ones (far right). Similarly, [23] observes that the low-pass filters seem to be more stable than high-pass filters when we randomly add/remove edges. These observations point to a promising direction for developing principled defense approaches based on the theoretical analysis of graph spectrum.

Contributions. To mitigate the vulnerability of GCNs against adversarial attacks, in this paper, we aim to bridge the gap between the spectral analysis and the robustness of GCNs under structural perturbations. Specifically, for the symmetric normalized Laplacian, which is commonly chosen as the graph filter in GCNs, we prove that the low-frequency components could be more robust against both one-edge and multi-edge perturbations when their eigenvalues fall into a robust interval. Interestingly, this result reveals that *not all low-frequency components are robust to adversarial attacks*.

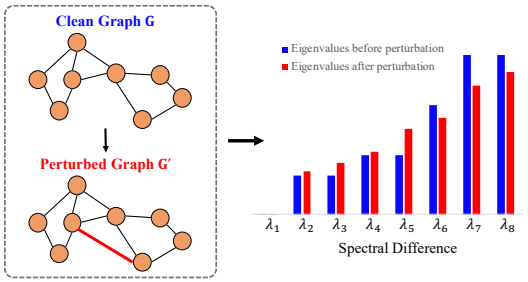

Figure 1: Observation. After adversarial attacks on the structure of graphs, the perturbations on the low-frequency components (far left) are smaller than that in the high-frequency ones (far right).

Considering the benefit from robust interval in low frequencies, we further propose GCN-LFR (**L**ow-**F**requency based **R**egularization), a general robust co-training paradigm that can transfer the robustness from the eligible low-frequency components. GCN-LFR is a general defender that can work with any GCN-based model against various training-time attacks. In particular, GCN-LFR regularizes the training process of any given GCN with the robust information from an auxiliary regularization net $\mathcal{M}_{\text{LFR}}$, which injects a set of learnable parameters to imitate the robust interval for low-frequency components. Moreover, instead of jointly training two branches, we design an alternative training scheme to accelerate the training process. We observe that the alternative training scheme also benefits the final performance in practice.

In experiments, to demonstrate the flexibility of GCN-LFR, we integrate GCN-LFR with five popular GCN-based models and compare GCN-LFR with four state-of-the-art defenders across five benchmark datasets under a variety of settings, including one-edge targeted, multi-edge targeted, and non-targeted attacks. Extensive results demonstrate that GCN-LFR consistently outperforms the defending baselines under all settings. Remarkably, we also show that GCN-LFR successfully enhances the robustness of GCNs without deduction of performance on benign graphs. It further reveals the broad applicability and relevance of GCN-LFR in the area of deep graph learning.

2 Related Works

Adversarial attacks on graphs. Adversarial attacks on graph neural networks have drawn unprecedented attention from researchers recently [33, 17, 35]. Based on the stage where the attack happens, adversarial attacks on graph-structured data can be divided into 1) *poisoning attacks* (*e.g.*, Nettack [52]) that perturb the graph in the training stage and 2) *evasion attacks* (*e.g.*, RL-S2V [9]) that perturb the graph at testing. A series of studies integrates various of techniques to construct persuasive adversarial samples, such as gradient based methods [41, 45], meta-learning [53], matrix perturbation theory [2], graph signal processing [7] and reinforcement learning [9, 34].

Defenses on graphs. Other than the adversarial attacks on graphs, researchers are also concerned about the robustness of graph models. Generally, we can category the existing methods into two families: spatial-based and spectral-based defense. 1) *Spatial-based defense*. GCN-Jaccard [41] proposes to examine fake edges in the step of preprocessing by utilizing similarity metrics. RGCN [50], Pro-GNN [18], and VPN [16] focus on proposing new variants of GCNs that can effectively defend against attacks on VanillaGCN. GNNGUARD [47] detects the adverse effects existing in the relationship between the graph structure and node features by neighbor importance estimation. Low-Pass [39] considers low-pass message passing from neighborhoods, which is similar to the attention mechanism in GNNGUARD to defense structural attacks. 2) *Spectral-based defense*. GCN-SVD [11] empirically

verifies that only the high-rank singular components of the graph are affected by Nettack, and preprocesses the poisoned graph via SVD with pure structural information for defense. In this paper, we first explore the theoretical connection between robustness and graph spectrum, and propose GCN-LFR to bring the robustness information from eligible low-frequency components in the spectral domain. The proposed framework GCN-LFR is a general defender and can protect various GCN-based models from adversarial attacks.

3 Preliminaries

We consider an undirected attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$, where $|\mathcal{V}| = N$ is the set of N nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of E edges, and $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ is the associated feature matrix, in which $\boldsymbol{x}_u \in \mathbb{R}^M$ is the M-dimensional node feature for node $u \in \mathcal{V}$. We utilize $\mathcal{N}_u$ to denote the neighbors of node u. We denote the adjacency matrix as $\boldsymbol{A} \in \mathbb{R}^{N \times N}$, where $\boldsymbol{A}_{uv} \in \{0, 1\}$ indicates whether an edge $e_{uv} \in \mathcal{E}$ exists. The node degree matrix is defined as $\boldsymbol{D} = \text{diag}(d_1, \cdots, d_n)$, where $d_u = \sum_v \boldsymbol{A}_{uv}$ represents the degree of node u. An adversarial attacker deliberately perturbs edges in $\mathcal{G}$ and results in the poisoned version of $\mathcal{G}$, which we denote as $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \boldsymbol{X})$, and $\boldsymbol{A}'$ is the perturbed version of $\boldsymbol{A}$. Let $\boldsymbol{\mathcal{L}} = \boldsymbol{D}^{-1/2}(\boldsymbol{D} - \boldsymbol{A})\boldsymbol{D}^{-1/2}$ be the symmetric normalized graph Laplacian. $\boldsymbol{\mathcal{L}}$ has eigenvalues ranging from 0 to 2. $\hat{\boldsymbol{A}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{I}_n - \boldsymbol{\mathcal{L}}$ is the symmetric normalized adjacency matrix, whose eigenvalues range from -1 to 1. Specifically, the *low-frequency components* refer to the ones of $\boldsymbol{\mathcal{L}}$ with eigenvalues in $[0, 1)$ (of $\hat{\boldsymbol{A}}$ with eigenvalues in $(0, 1]$), and the *high-frequency* ones refer to that of $\boldsymbol{\mathcal{L}}$ with eigenvalues in $(1, 2]$ (of $\hat{\boldsymbol{A}}$ with eigenvalues in $[-1, 0)$).

3.1 Graph Convolutional Networks

VanillaGCN can be viewed as the spectral convolution based on the Fourier transform on graphs with first-order Chebyshev polynomial filter $g(\boldsymbol{\mathcal{L}})$. The layer-wise update rule is given as:

$$\text{feature transformation}: \boldsymbol{H}'^{(l)} = \boldsymbol{H}^{(l)}\boldsymbol{\Theta}, \quad \text{graph convolution}: \boldsymbol{H}^{(l+1)} = \sigma(g(\boldsymbol{\mathcal{L}})\boldsymbol{H}'^{(l)}), \tag{1}$$

where $\boldsymbol{H}^{(l+1)} \in \mathbb{R}^{N \times q^{(l+1)}}$ is the output from hidden layer l with input as $\boldsymbol{H}^{(l)} \in \mathbb{R}^{N \times q^{(l)}}$. $q^{(l)}$ refers to the dimension of output at layer l. For the input layer, we have $\boldsymbol{H}^{(0)} = \boldsymbol{X}$. $\sigma(\cdot)$ refers to the activation function, such as ReLU. $\boldsymbol{\Theta}^l \in \mathbb{R}^{q^{(l)} \times q^{(l+1)}}$ refers to the weight matrix. In Eq.(1), different filter $g(\boldsymbol{\mathcal{L}})$ leads to different GCNs. For example, VanillaGCN along with its variants [46, 40, 13, 8] usually use the symmetric normalized adjacency matrix $\hat{\boldsymbol{A}} = \boldsymbol{I}_n - \boldsymbol{\mathcal{L}}$ as $g(\boldsymbol{\mathcal{L}})$. Therefore, we mainly focus on the analysis of filter $\hat{\boldsymbol{A}}$, and try to find out the connection between the robustness of GCNs and the perturbation on $\hat{\boldsymbol{A}}$. To ease the notation, we denote $\boldsymbol{U}$ and $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \cdots, \lambda_N)$ as the eigen-pairs of $\hat{\boldsymbol{A}}$ in the following sections unless otherwise indicated.

3.2 Poisoning Structural Attacks

Poisoning structural attacks towards graph-structured data focus on corrupting the topology during training stage [17]. Specifically, the attacker is able to *delete* or *insert* a small number of edges on the benign graph to generate a perturbed graph $\mathcal{G}'$ with adjacency matrix $\boldsymbol{A}'$. Through this manipulation, the attacker intends to fool the prediction of GCNs and damage the performance of the downstream learning tasks. Formally, given a fixed budget of perturbation Δ on edges, *i.e.*, $||\boldsymbol{A}' - \boldsymbol{A}||_0 \leq 2\Delta$, the adversarial attack on a GCN-based model $\mathcal{M}$ can be formulated as a bi-level optimization problem [2, 7]:

$$\underset{\boldsymbol{A}' \in \mathcal{P}_\Delta^{\mathcal{G}}}{\text{argmin}}\ \mathcal{L}_{\text{attack}}(\mathcal{M}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta}^*), \boldsymbol{y}) \quad \text{s.t.} \quad \boldsymbol{\Theta}^* = \underset{\boldsymbol{\Theta}}{\text{argmin}}\ \mathcal{L}_{\text{GCN}}(\mathcal{M}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta}), \boldsymbol{y}),$$

where $\mathcal{P}_\Delta^{\mathcal{G}}$ denotes the possible perturbations within budget Δ. $\boldsymbol{y}$ is the ground-truth label. $\mathcal{L}_{\text{attack}}$ and $\mathcal{L}_{\text{GCN}}$ denote the loss of adversarial attack and model $\mathcal{M}$, respectively. $\mathcal{M}(\boldsymbol{A}', \boldsymbol{X}; \Theta^*)$ refers to the prediction of model $\mathcal{M}$ on the poisoned graph $\mathcal{G}'$, where the weights $\boldsymbol{\Theta}^*$ keep trained to optimal.

We denote $\mathcal{T}$ as the set of target nodes aimed by the attacker. We consider the following two types of attacks: 1) **Targeted attacks.** The attacker aims to destroy prediction for a specific node u by manipulating the adjacent edges of u [52, 41, 7], and $\mathcal{T} = \{u\}$ in this situation. 2) **Non-targeted attacks.** The attacker aims to degrade the overall classification performance but does not care about which node is being targeted [53, 45]. Under this setting, $\mathcal{T} = \mathcal{V}_{\text{test}}$, where $\mathcal{V}_{\text{test}}$ is the full test set.

4 Methodology

4.1 Problem Formulation

In a poisoning attack, the training data from $\mathcal{G}$ is perturbed before fed into a GCN model $\mathcal{M}$, so that the performance of $\mathcal{M}$ can be degraded. Therefore, we formulate the defense problem as:

Problem 1. *Given a perturbed graph $\mathcal{G}'$ generated by a poisoning attacker, the aim of defense on target set $\mathcal{T}$ is to produce a more robust model $\mathcal{M}^r$ to optimize:*

$$\min_{\boldsymbol{\Theta}^{r*}} \sum_{u \in \mathcal{T}} \|\mathcal{M}_u^r(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta}^{r*}) - \mathcal{M}_u(\boldsymbol{A}, \boldsymbol{X}; \boldsymbol{\Theta}^*)\|, \tag{2}$$

where $\mathcal{M}_u^r(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta}^{r*}) = \hat{\boldsymbol{y}}_u^r$ refers to the prediction of node u, when $\mathcal{M}^r$ is trained on the poisoned graph $\mathcal{G}'$ with optimal $\boldsymbol{\Theta}^{r*}$. $\mathcal{M}_u(\boldsymbol{A}, \boldsymbol{X}; \boldsymbol{\Theta}^*) = \hat{\boldsymbol{y}}_u$ is a hypothetical prediction that $\mathcal{M}$ would make if the clean graph $\mathcal{G}$ is accessible.

In Eq.(2), we aim to force $\hat{\boldsymbol{y}}_u^r$ to approximate $\hat{\boldsymbol{y}}_u$, then the prediction $\hat{\boldsymbol{y}}_u$ in the benign situation can be recovered. However, the clean graph $\mathcal{G}$ is almost impossible to be accessed under the setting of adversarial attacks, which implies that we can not obtain $\mathcal{M}'$ by directly optimizing Eq.(2). To tackle this problem, we begin with investigating the potential impacts of structural attacks by examining the variability between $\boldsymbol{A}$ and $\boldsymbol{A}'$.

4.2 Spectral View of Structural Attacks

Following [11, 12], we observe the eigenvalues of $\hat{\boldsymbol{A}}$ after structural attacks as shown in Figure 1.

Observation 1. Under structural attacks, the high-valued eigenvalues (low-frequency components) of $\hat{\boldsymbol{A}}$ are likely to be more *robust* than the low-valued eigenvalues (high-frequency components).

With this empirical inspiration, GCN-SVD [11] directly preprocesses the poisoned graph and retains the top low-frequency components for defense. However, we argue that this justification is inaccurate. The relationship between robustness and graph spectrum is more complex than empirical observation. In the following, we provide an in-depth analysis of structural perturbations from the spectral domain.

4.2.1 One-edge Perturbation

We first consider the targeted attacks with one-edge perturbation, where $\mathcal{T} = \{u\}$ and $\Delta = 1$. Note that under one-edge perturbation, *i.e.*, we conduct a single edge flip e_{uv} on $\boldsymbol{A}$, $\Delta\boldsymbol{A}$ is a matrix with only 2 non-zero elements, namely $\Delta\boldsymbol{A}_{uv} = \Delta\boldsymbol{A}_{vu} = 1 - 2\boldsymbol{A}_{uv}$. Following [27], we assume that the eigenvector matrix $\boldsymbol{U}$ satisfies the following assumption:

Assumption 1. $\boldsymbol{U}$ has an orthonormal basis $(\boldsymbol{u}_y)_{y \in [N]}$ that consists of non-negative vectors, where N is the number of nodes in the graph.

Then, we introduce the Lemma 1 to describe the differences of eigenvalues after structural attacks.

Lemma 1. *Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$ and one edge e_{uv} to be perturbed, the change of y-th eigenvalue after perturbation is formulated as*

$$\Delta\lambda_y = \begin{cases} 2\boldsymbol{u}_{yu} \cdot \boldsymbol{u}_{yv} - \lambda_y(\boldsymbol{u}_{yu}^2 + \boldsymbol{u}_{yv}^2) & \text{if } \mathcal{E} \cup \{e_{uv}\} \\ -2\boldsymbol{u}_{yu} \cdot \boldsymbol{u}_{yv} + \lambda_y(\boldsymbol{u}_{yu}^2 + \boldsymbol{u}_{yv}^2) & \text{if } \mathcal{E} \setminus \{e_{uv}\}, \end{cases}$$

where $\boldsymbol{u}_{yu}$ refers to the u-th element of eigenvector $\boldsymbol{u}_y$.

The proof is available in the Appendix. With the help of Lemma 1, we can establish the relationship between robustness of GCNs and the change on each eigenvalue $|\Delta\lambda_y|$ resulting from perturbations. Intuitively, smaller changes on eigenvalues indicate stronger robustness of GCNs against attacks.

Theorem 1 (**Low-frequency**). *Assume that $\boldsymbol{u}_{au}^2 + \boldsymbol{u}_{av}^2 \neq 0$, $\boldsymbol{u}_{bu}^2 + \boldsymbol{u}_{bv}^2 \neq 0$, and*

$$(\boldsymbol{u}_{au} + \boldsymbol{u}_{av})^2 > (\boldsymbol{u}_{bu} - \boldsymbol{u}_{bv})^2. \tag{3}$$

Then, there exists a pair of high-frequency $\lambda_a \in [-1, 0)$ and low-frequency $\lambda_b \in (0, 1]$, so that $|\Delta\lambda_a| > |\Delta\lambda_b|$ always holds if λ_a and λ_b satisfy

$$\max\big(0, \frac{d_b - d_a + c_a\lambda_a}{c_b}\big) < \lambda_b < 1; \quad -1 < \lambda_a < \min\big(\frac{d_a - c_b + d_b}{c_a}, 0\big), \tag{4}$$

where $c_a = \boldsymbol{u}_{au}^2 + \boldsymbol{u}_{av}^2$, $d_a = 2\boldsymbol{u}_{au} \cdot \boldsymbol{u}_{av}$, $c_b = \boldsymbol{u}_{bu}^2 + \boldsymbol{u}_{bv}^2$, and $d_b = 2\boldsymbol{u}_{bu} \cdot \boldsymbol{u}_{bv}$.

The proof of Theorem 1 is available in the Appendix. From Theorem 1, we summarize two remarks.

Remark 1. The top low-frequency components (*i.e.*, components are selected from the values closer to 1) could be more robust than the top high-frequency ones (*i.e.*, components are selected from the values closer to -1) against one-edge perturbation under the condition from Inequality 3, which is consistent with Observation 1. However, it is still vulnerable to such attacks that perturb the edges satisfying $(\boldsymbol{u}_{au} + \boldsymbol{u}_{av})^2 \leq (\boldsymbol{u}_{bu} - \boldsymbol{u}_{bv})^2$. This is contrary to the justification from GCN-SVD [11] and implies that not all low-frequency components are robust.

Remark 2. We do not consider two special situations that $\boldsymbol{u}_{au}^2 = \boldsymbol{u}_{av}^2 = 0$ or $\boldsymbol{u}_{bu}^2 = \boldsymbol{u}_{bv}^2 = 0$, because they conclude that $\Delta\lambda_a = 0$ or $\Delta\lambda_b = 0$, respectively. This implies no perturbation happens on λ_a or λ_b, which is rare in practice and out of discussion.

4.2.2 Robust Interval of Low-Frequency

As Remark 1 from Theorem 1, Inequality 3 still makes the low-frequency components of $\hat{\boldsymbol{A}}$ suffer from certain attacks. To further get rid of this condition, we derive a new bound for low-frequency λ_b by removing the dependency on λ_a in Inequality 4 here, which reveals the *robust interval*:

Theorem 2 (Robust interval). *Assume that $\boldsymbol{u}_{au}^2 + \boldsymbol{u}_{av}^2 \neq 0$ and $\boldsymbol{u}_{bu}^2 + \boldsymbol{u}_{bv}^2 \neq 0$. There exists low-frequency λ_b, so that $|\Delta\lambda_a| > |\Delta\lambda_b|$ always holds for any $-1 < \lambda_a < 0$ if λ_b satisfies*

$$\max\big(0, \frac{d_b - d_a + c_a\lambda_a}{c_b}\big) < \lambda_b < \min\big(\frac{d_b + d_a - c_a\lambda_a}{c_b}, 1\big).$$

The proof is available in the Appendix. Theorem 2 indicates the interval of λ_b that is able for defense, which we denote as the *robust interval* of λ_b. In other words, when the low-frequencies fall into the robust interval, they are always more robust than the high-frequencies under one-edge perturbation.

4.2.3 Non-targeted Perturbation

In this part, we consider a more complex and practical setting, the multi-edge non-targeted perturbation, where $\mathcal{T} = \mathcal{V}_{\text{test}}$ and $\Delta > 1$. The perturbation $\mathcal{P}$ on each node $u \in \mathcal{T}$ can be treated as multi-edge targeted attack. Note that the multi-edge targeted perturbation is a special case when $\mathcal{T} = \{u\}$. Specifically, we discuss the type of attacks that either consecutively inserting or deleting edges on the clean graph, since every state after perturbations can be decomposed into consecutively inserting then removing edges. In this situation, the attacker directly manipulates the adjacent edges of u. Based on Theorem 1, we obtain the following results:

Corollary 1 (Low-frequency). *We decompose non-targeted attacks as two steps: 1) selecting target u from $\mathcal{T}$, and 2) conducting perturbations on u as multi-edge targeted attack. Assume p adversarial edges are either consecutively inserted or deleted on node u as set $\mathcal{P}_u$,* i.e.*, $\mathcal{E} \cup \mathcal{P}_u$ or $\mathcal{E} \setminus \mathcal{P}_u$. With the same assumptions as in Theorem 1, there exists a pair of high-frequency λ_a and low-frequency λ_b so that $|\sum_{u\in\mathcal{T}}\sum_{v\in\mathcal{P}_u}\Delta\lambda_{auv}| > |\sum_{u\in\mathcal{T}}\sum_{v\in\mathcal{P}_u}\Delta\lambda_{buv}|$ always holds if λ_a and λ_b satisfy*

$$\max_{v\in\mathcal{P}_u, u\in\mathcal{T}}\big(0, \frac{d_{buv} - d_{auv} + c_{auv}\lambda_a}{c_{buv}}\big) < \lambda_b < 1;\ -1 < \lambda_a < \min_{v\in\mathcal{P}_u, u\in\mathcal{T}}\big(\frac{d_{auv} - c_{buv} + d_{buv}}{c_{auv}}, 0\big),$$

where $c_{auv} = \boldsymbol{u}_{au}^2 + \boldsymbol{u}_{av}^2$, $d_{auv} = 2\boldsymbol{u}_{au}\cdot\boldsymbol{u}_{av}$, $c_{buv} = \boldsymbol{u}_{bu}^2 + \boldsymbol{u}_{bv}^2$, and $d_{buv} = 2\boldsymbol{u}_{bu}\cdot\boldsymbol{u}_{bv}$.

Then we can have the robust interval under non-targeted perturbations with the help of Corollary 1 as:

Corollary 2 (Robust interval). *With same assumptions as in Theorem 2, $|\sum_{u\in\mathcal{T}}\sum_{v\in\mathcal{P}_u}\Delta\lambda_{auv}| > |\sum_{u\in\mathcal{T}}\sum_{v\in\mathcal{P}_u}\Delta\lambda_{buv}|$ always holds for any $-1 < \lambda_a < 0$ if λ_b satisfies*

$$\max_{v\in\mathcal{P}_u, u\in\mathcal{T}}\big(0, \frac{d_{buv} - d_{auv} + c_{auv}\lambda_a}{c_{buv}}\big) < \lambda_b < \min_{v\in\mathcal{P}_u, u\in\mathcal{T}}\big(\frac{d_{buv} + d_{auv} - c_{auv}\lambda_a}{c_{buv}}, 1\big).$$

The proofs for Corollary 1 and Corollary 2 can be found in the Appendix.

Remark 3. Under non-targeted perturbation, one could come to the same remarks as Theorem 1 when consecutively inserting or deleting edges from the clean graph. The mixed situation will be left as a future work. Due to space constraints, please refer to Corollary 3 and Corollary 4 in the Appendix for details upon multi-edge targeted attack.

To summarize, we provide the theoretical justifications about the robustness of low-frequency components under the structural attacks in this section. These results imply that we can enhance the robustness of GCNs by utilizing the robust interval of low frequencies in $\hat{\boldsymbol{A}}'$ of the poisoned graph.

4.3 GCN-LFR: General Robust Training Paradigm

Inspired by the theoretical findings in Section 4.2, it is natural to migrate the information from low-frequency components within robust interval to enhance the robustness of model $\mathcal{M}$. In this vein, we present GCN-LFR (**L**ow-**F**requency based **R**egularization), which introduces an auxiliary network $\mathcal{M}_{\text{LFR}}$ to extract the low-frequency information within robust interval to regularize $\boldsymbol{\Theta}$ during the training procedure via parameter sharing [14]. Figure 2 depicts the whole framework of GCN-LFR.

4.3.1 Auxiliary Regularization Net $\mathcal{M}_{\text{LFR}}$

The effectiveness of the parameter sharing based regularization is demonstrated by co-training an auxiliary neural network on various of Deep Neural Networks [1, 38]. In GCN-LFR, we propose our auxiliary regularization net $\mathcal{M}_{\text{LFR}}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta})$ by explicitly designing the graph convolutional filter from Eq.(1) with the robust low-frequency components. However, explicitly computing the robustness interval is infeasible before the structural attack. To alleviate this chicken or the egg style causality dilemma, we employ a set of learnable parameters $\boldsymbol{F}$ as filters to learn the robust interval for the low-frequency components during training. Overall, the layer-wise update rule for $\mathcal{M}_{\text{LFR}}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta}, \boldsymbol{F})$ is:

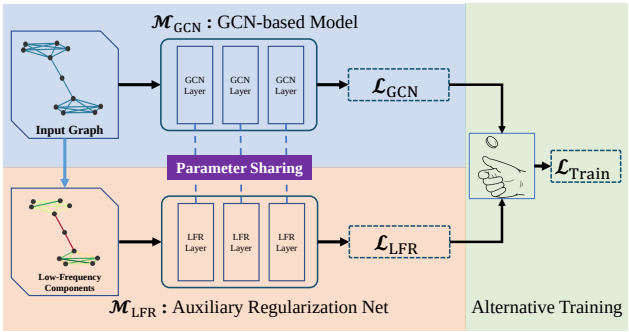

Figure 2: Overview of GCN-LFR. For any given GCN-based model $\mathcal{M}$, we enhance its robustness by co-training an auxiliary net $\mathcal{M}_{\text{LFR}}$ that utilizes only the low-frequency components as regularization.

$$\text{feature transformation}: \boldsymbol{H}'^{(l)} = \boldsymbol{H}^{(l)}\boldsymbol{\Theta}, \quad \text{graph convolution}: \boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{U}'_{\text{low}}\boldsymbol{F}\boldsymbol{U}'^{\top}_{\text{low}}\boldsymbol{H}'^{(l)}),$$

where $\boldsymbol{U}'_{\text{low}}$ are the top-k low-frequency eigenvectors from poisoned graph $\mathcal{G}'$, and the parameters $\boldsymbol{\Theta}$ will be co-trained and shared to the original GCN model $\mathcal{M}_{\text{GCN}}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta})$. $\boldsymbol{F} = \text{diag}(f_1, \cdots, f_k)$ is a diagonal matrix with k parameters as the graph filter, which can be learned in an end-to-end manner. Aside from the flexibility of learned $\boldsymbol{F}$ falling into the robust interval, $\boldsymbol{F}$ can also enlarge the capacity of $\mathcal{M}_{\text{LFR}}$ by making the learned embeddings $\boldsymbol{H}$ generalize into the subspace span by $\boldsymbol{U}'_{\text{low}}$.

4.3.2 Co-training Framework

Consider a learning task on graph $\mathcal{G}$, and a given GCN-based model $\mathcal{M}_{\text{GCN}}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta})$ parametrized by $\boldsymbol{\Theta}$ and poisoned by $\boldsymbol{A}'$ is utilized to deal with this task. We denote the loss function of $\mathcal{M}_{\text{GCN}}$ as $\mathcal{L}_{\text{GCN}} = \mathcal{L}(\mathcal{M}_{\text{GCN}}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta}), \boldsymbol{y})$. Instead of minimizing $\mathcal{L}_{\text{GCN}}$ alone, we propose to co-train it with the same supervised loss on an auxiliary regularization net $\mathcal{L}_{\text{LFR}} = \mathcal{L}(\mathcal{M}_{\text{LFR}}(\boldsymbol{A}', \boldsymbol{X}; \boldsymbol{\Theta}, \boldsymbol{F}), \boldsymbol{y})$. In this way, we are able to compel the original model to learn a more reliable representation from the robust low-frequency components:

$$\mathcal{L}_{\text{total}} = (1-\alpha)\mathcal{L}_{\text{GCN}} + \alpha\mathcal{L}_{\text{LFR}}, \tag{5}$$

where $\alpha \in [0, 1]$ is the weight coefficient for balancing the contribution of two losses. The transformation of the robust information from low-frequency components is achieved by regularizing the training of $\mathcal{L}_{\text{GCN}}$ with the shared parameters $\boldsymbol{\Theta}$.

Instead of jointed optimizing $\mathcal{L}_{\text{GCN}}$ and $\mathcal{L}_{\text{LFR}}$, we exploit the idea of alternative training [38] to optimize the two losses. Specifically, at each epoch, we randomly choose one of the two losses in GCN-LFR to optimize with a probability determined by α. To put it another way, we sample a random value $z \in U(0, 1)$, and α acts as a threshold that decides which loss would be optimized at each epoch. The thorough description of algorithm is depicted in the Appendix.

As compared to joint training, this alternative training scheme can reduce the computational cost and accelerate the training process. Furthermore, in practice, we observe that the generalization ability of model is also improved according to the alternative training strategy.

Table 1: Defense performance (multi-class classification accuracy) against one-edge targeted attacks under Nettack-One. **Note:** the accuracy is 1 on clean graph since only the correctly classified nodes are targeted.

Model	Dataset	Attacked	GCN-Jaccard	GCN-SVD	GNNGUARD	GCN-LFR
VanillaGCN [21]	Cora	0.350	0.525	0.675	0.800	**0.875**
	Citeseer	0.425	0.450	0.625	0.725	**0.775**
	Pubmed	0.375	0.400	0.600	0.775	**0.825**
	Coauthor CS	0.650	0.675	0.725	0.825	**0.900**
	Amazon Photo	0.500	0.525	0.625	0.625	**0.750**
JK-Net [46]	Cora	0.350	0.575	0.675	0.775	**0.825**
	Citeseer	0.450	0.650	0.550	0.750	**0.800**
	Pubmed	0.350	0.375	0.625	0.800	**0.850**
	Coauthor CS	0.600	0.625	0.675	0.725	**0.775**
	Amazon Photo	0.550	0.575	0.675	0.675	**0.800**
SGC [40]	Cora	0.425	0.675	0.550	0.750	**0.825**
	Citeseer	0.475	0.600	0.650	0.725	**0.750**
	Pubmed	0.300	0.475	0.650	0.750	**0.800**
	Coauthor CS	0.600	0.650	0.725	0.750	**0.775**
	Amazon Photo	0.550	0.575	0.600	0.675	**0.725**
Graph-U-Net [13]	Cora	0.375	0.525	0.650	0.725	**0.800**
	Citeseer	0.275	0.350	0.575	0.750	**0.775**
	Pubmed	0.475	0.550	0.675	0.700	**0.750**
	Coauthor CS	0.625	0.600	0.700	0.775	**0.850**
	Amazon Photo	0.500	0.525	0.650	0.725	**0.825**
GCNII [8]	Cora	0.475	0.650	0.575	0.825	**0.900**
	Citeseer	0.550	0.600	0.625	0.725	**0.750**
	Pubmed	0.400	0.475	0.675	0.750	**0.775**
	Coauthor CS	0.625	0.650	0.700	0.800	**0.825**
	Amazon Photo	0.550	0.425	0.600	0.675	**0.775**

4.3.3 Complexity Analysis

We discuss the complexity of GCN-LFR here by the light example of VanillaGCN as backbone. For an input graph with N nodes, E edges, and M as feature dimension on each node, the complexity of a solo VanillaGCN is $O(EM + NM^2)$ [43]. Then, with the help of Lanczos algorithm [22], the time complexity of calculating the top-k eigen-pairs is $O(ckN)$, where c is the average number of non-zero entries in a row of $\hat{\boldsymbol{A}}$. Thus the overall complexity of GCN-LFR on VanillaGCN is $O(EM + N(M^2 + ck))$. In a nutshell, the extra complexity introduced by GCN-LFR during co-training is not a burden, especially when the graph is sparse, as it is in most real-world applications.

5 Experiments

5.1 Experimental Setup

Datasets. We mainly focus on five node classification benchmark datasets under the semi-supervised setting: 1) three citation networks, Cora [26], Citeseer and Pubmed [29], 2) a coauthor network Coauthor CS [30], and 3) a co-purchase network Amazon Photo [25]. More details, including dataset description and statistical information, are deferred to the Appendix.

Adversarial samples generation. We evaluate our model under three kinds of adversarial attack settings from the existing studies: *targeted attack with one-edge perturbation* (Nettack-One [52]), *targeted attack with multi-edge perturbation* (Nettack-Multi [52]), and *non-targeted attack* (Metattack [53]). Details upon the three adversarial attack settings can be found in the Appendix.

Target GCN-based models. We apply GCN-LFR to five GCN-basd models to extensively evaluate its defense performance under three attack settings. The five models are: VanillaGCN [21], JK-Net [46], SGC [40], Graph-U-Net [13], and GCNII [8].

Defense baselines. We compare GCN-LFR with four SOTA general graph defenders with their available public implementations. GCN-Jaccard [41] and GCN-SVD [11] are two preprocessing based defending methods. GNNGUARD [47] and Pro-GNN [18] are two spatial-based approaches. Details upon hyperparameters and architectures can be referred to in the Appendix.

5.2 Defense Results Against Targeted and Non-Targeted Attacks

Targeted attack with one-edge perturbation. We first examine the robustness of GCN-LFR against targeted one-edge perturbation under Nettack-One and report the results in Table 1. The column

"Attacked" shows the performance without any defense. From this column, we can find that Nettack-One can successfully damage the performance of various models even with only a single edge flip. However, GCN-LFR can dramatically defend given GCNs against Nettack-One and consistently outperforms state-of-the-art baselines. Remarkably, we observe that the low-rank preprocessing approach GCN-SVD is not very effective, which implies that not all low-frequency components are robust, and our strategy by transferring robust knowledge with learnable $\boldsymbol{F}$ is beneficial.

Table 2: Defense performance (multi-class classification accuracy) against multi-edge targeted attacks under Nettack-Multi. **Note:** the accuracy is 1 on clean graph since only the correctly classified nodes are targeted.

Model	Dataset	Attacked	GCN-Jaccard	GCN-SVD	GNNGUARD	GCN-LFR
GCNII [8]	Cora	0.125	0.325	0.525	0.575	**0.600**
	Citeseer	0.175	0.425	0.575	0.700	**0.725**
	Pubmed	0.150	0.325	0.625	0.700	**0.725**
	Coauthor CS	0.125	0.350	0.625	0.675	**0.725**
	Amazon Photo	0.150	0.525	0.825	0.800	**0.850**

Table 3: Defense performance (multi-class classification accuracy) against non-targeted attacks under Metattack.

Model	Dataset	Attacked	GCN-Jaccard	GCN-SVD	GNNGUARD	Pro-GNN	GCN-LFR
VanillaGCN [21]	Cora	0.701	0.760	0.717	0.764	0.773	**0.777**
	Citeseer	0.675	0.680	0.664	0.687	**0.693**	0.690
	Pubmed	0.807	0.845	0.823	0.848	0.852	**0.855**
	Coauthor CS	0.836	0.865	0.852	0.868	-	**0.875**
	Amazon Photo	0.854	0.877	0.862	0.882	-	**0.887**

Targeted attack with multi-edge perturbation. Next, we evaluate the defending performance of targeted attacks with multi-edge perturbation. Due to space limitations, we only report the defending performance on the newest variant, GCNII, in Table 2 and leave the full results in the Appendix. It is not surprising that GCNII suffers more serious performance degradation under Nettack-Multi than that of Nettack-One. However, GCN-LFR remains the best defender among all baselines even when the amount of perturbed edges increases. We also observe that GCN-LFR achieves better defense under one-edge perturbation than multi-edge. We suppose this is due to the wider robust interval in Theorem 2, since the bounds are tighter when iterating over all candidate edges in Corollary 4.

Non-targeted attack. To validate the defending performance of GCN-LFR against non-targeted attacks, we utilize Metattack to conduct adversarial edges globally. We also include the comparison with an additional baseline Pro-GNN under this setting on small datasets, since Pro-GNN is extremely time-consuming and it could not finish within 24 hours under the above targeted attack setting on all datasets as well as large datasets for non-targeted attacks. Table 3 demonstrates the results on VanillaGCN for all datasets. In Table 3, we can observe that the defending performance of GCN-LFR is also comparable with all baselines.

Table 4: Classification accuracy on benign (*i.e.*, non-attacked) datasets with and without GCN-LFR.

Bengin dataset	Cora		Citeseer		Pubmed		Coauthor CS		Amazon Photo	
w or w/o LFR	w/o	w	w/o	w	w/o	w	w/o	w	w/o	w
VanillaGCN	0.837	0.835	0.721	0.701	0.863	0.858	0.907	0.911	0.921	0.925
JK-Net	0.820	0.827	0.705	0.693	0.854	0.836	0.927	0.891	0.929	0.896
SGC	0.825	0.819	0.719	0.701	0.859	0.861	0.928	0.897	0.929	0.927
Graph-U-Net	0.825	0.838	0.697	0.688	0.835	0.842	0.926	0.907	0.925	0.918
GCNII	0.812	0.813	0.695	0.698	0.864	0.838	0.931	0.904	0.934	0.927

5.3 Ablation Studies

Performance of GCN-LFR on clean datasets. Aside from the defense against adversarial attacks, a successful defender needs not to affect the performance on clean datasets. Table 4 shows the performance comparison of five GCN models with and without regularization net $\mathcal{M}_{\text{LFR}}$ on five benign (*i.e.*, non-attacked) datasets. As shown in Table 4, GCN-LFR achieves a quite competitive classification accuracy and even better in some cases. This indicates that GCN-LFR does not harm the performance of GCNs when no attack happens while successfully dealing with the poisoned graphs.

In-depth analysis on $\mathcal{M}_{\text{LFR}}$ net. To justify the necessity of each component of GCN-LFR, we implement several variants of GCN-LFR with VanillaGCN and evaluate the performance on the benign Cora dataset. Figure 3 shows the comparison among GCN-LFR and other variants on the training and validation accuracy. GCN and LFR represent training with only VanillaGCN and $\mathcal{M}_{\text{LFR}}$ net, respectively. GCN-LFR$_{\text{Fix}}$ is a variant of GCN-LFR that replaces the learnable filter $\boldsymbol{F}$ with the

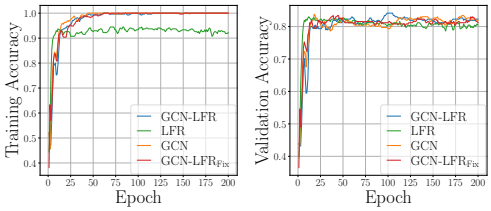

Figure 3: The training accuracy and validation accuracy comparison among GCN-LFR, two components of GCN-LFR, *i.e.*, VanillaGCN and LFR, and a variant of GCN-LFR, namely GCN-LFR$_{\text{Fix}}$.

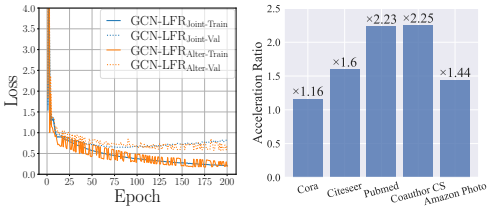

Figure 4: **Left:** the training and validation loss comparison between GCN-LFR$_{\text{Alter}}$ on Pubmed with backbone VanillaGCN. **Right:** the acceleration ratio of GCN-LFR$_{\text{Alter}}$ over GCN-LFR$_{\text{Joint}}$.

truncated eigenvalues $\boldsymbol{\Lambda}'_{\text{low}}$. We have the following observations: 1) LFR net achieves lower training accuracy and suffers from severe underfitting issue. It indicates that training with only low-frequency components would lose information despite their robustness. 2) GCN-LFR achieves competitive performance against GCN. It re-confirms the results in Section 5.3 that GCN-LFR does not harm performance in the absence of attack. 3) GCN-LFR achieves substantially better validation accuracy than that of GCN-LFR$_{\text{Fix}}$. It implies that the trainable graph filter $\boldsymbol{F}$ is more flexible and can aid the generalization of learned embeddings in the subspace spanned by $\boldsymbol{U}'_{\text{low}}$.

Joint optimization vs. alternate optimization on GCN-LFR. We investigate the benefits of alternate optimization of GCN-LFR (*i.e.*, GCN-LFR$_{\text{Alter}}$). We denote the joint optimization of GCN-LFR as GCN-LFR$_{\text{Joint}}$. Figure 4 provides the results of comparison. The left-side of Figure 4 demonstrates the training and validation accuracy on Pubmed dataset. VanillaGCN is used as backbone here. We can observe that GCN-LFR$_{\text{Alter}}$ achieves better training and validation performance compared with GCN-LFR$_{\text{Joint}}$. Furthermore, the right-side of Figure 4 shows the acceleration ratio of GCN-LFR$_{\text{Alter}}$ over GCN-LFR$_{\text{Joint}}$ in terms of training time on five datasets. Acceleration ratio denotes the quotient obtained by dividing the time-consumed of GCN-LFR$_{\text{Joint}}$ by that of GCN-LFR$_{\text{Alter}}$. We can find that GCN-LFR$_{\text{Alter}}$ is more efficient by achieving an average of $\times 1.74$ times acceleration, since GCN-LFR$_{\text{Alter}}$ only needs to optimize one model while GCN-LFR$_{\text{Joint}}$ needs to handle two models at each epoch. It verifies that GCN-LFR$_{\text{Alter}}$ enjoys both effectiveness and efficiency.

Eigenvalues vs. learned $\boldsymbol{F}$ in the robust interval. To evaluate whether the proposed theoretical analysis is reliable, we check how many λ_b and the trained $\boldsymbol{F}$ fall into the robust interval on all datasets here. We report the results of top 25% smallest of eigenvalues as λ_b on the same 40 target nodes after one-edge perturbation in Figure 5. We can find that, though by choosing a considerably larger proportion as low-frequency, we still have an average of around 40% of λ_b falling into the robust interval. Further, the learnable $\boldsymbol{F}$ has more values falling into the robust interval after training, which is consistent with our observation in Figure 3. It also verifies that $\boldsymbol{F}$ could potentially learn the frequencies within the robust interval for low-frequency components and lead to a more robust model.

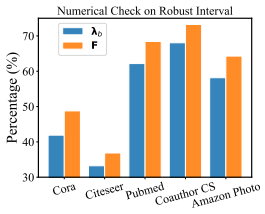

Figure 5: Numerical check on the robust interval.

6 Conclusion and Limitations

In pursuit of developing defending methods for the adversarial attacks on GCNs, we present the first vulnerability study between the graph spectrum and the robustness behaviour of GCNs, which indicates that the low-frequency components in graphs could be more robust than the high-frequency ones under certain conditions. Moreover, we further verify a robust interval of the low-frequency components, which implies that not all low-frequency components are robust unless they fall within the robust interval. Guided by this theory, we propose a general robust training paradigm, dubbed GCN-LFR, that suggests training an auxiliary network jointly with the original model through parameter sharing to transfer the robustness of low-frequency components. Extensive experimental verifications demonstrate the resistance of GCN-LFR when being exposed to adversarial attacks.

Currently, we only consider the purification in the context of structural perturbations. Given that adversarial perturbations will occur at the level of features and nodes, we would like to expand the proposed architecture in the future to protect against various forms of adversarial attacks. Meanwhile, we make every effort to include a thorough experimental study. Further results on possible datasets and baselines will be added as a future work. Societal impacts are discussed in the Appendix.

References

[1] Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.

[2] Bojchevski, A. and Günnemann, S. (2019a). Adversarial attacks on node embeddings via graph poisoning. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, Proceedings of Machine Learning Research. PMLR.

[3] Bojchevski, A. and Günnemann, S. (2019b). Adversarial attacks on node embeddings via graph poisoning. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, Proceedings of Machine Learning Research. PMLR.

[4] Bojchevski, A. and Günnemann, S. (2019c). Certifiable robustness to graph perturbations. In *Neural Information Processing Systems, NeurIPS*.

[5] Boscaini, D., Masci, J., Melzi, S., Bronstein, M. M., Castellani, U., and Vandergheynst, P. (2015). Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. In *Computer Graphics Forum*, volume 34, pages 13–23. Wiley Online Library.

[6] Bruna, J., Zaremba, W., Szlam, A., and Lecun, Y. (2014). Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, pages http–openreview.

[7] Chang, H., Rong, Y., Xu, T., Huang, W., Zhang, H., Cui, P., Zhu, W., and Huang, J. (2020). A restricted black-box adversarial framework towards attacking graph embedding models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3389–3396.

[8] Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. (2020). Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, pages 1725–1735. PMLR.

[9] Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J., and Song, L. (2018). Adversarial attack on graph structured data. *international conference on machine learning*, pages 1115–1124.

[10] Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. *neural information processing systems*, pages 3844–3852.

[11] Entezari, N., Al-Sayouri, S. A., Darvishzadeh, A., and Papalexakis, E. E. (2020). All you need is low (rank) defending against adversarial attacks on graphs. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 169–177.

[12] Gama, F., Bruna, J., and Ribeiro, A. (2020). Stability properties of graph neural networks. *IEEE Transactions on Signal Processing*, 68:5680–5695.

[13] Gao, H. and Ji, S. (2019). Graph u-nets. In *International Conference on Machine Learning*, pages 2083–2092.

[14] Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

[15] Horn, R. A. and Johnson, C. R. (2012). *Matrix analysis*. Cambridge university press.

[16] Jin, M., Chang, H., Zhu, W., and Sojoudi, S. (2021). Power up! robust graph convolutional network via graph powering. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[17] Jin, W., Li, Y., Xu, H., Wang, Y., and Tang, J. (2020a). Adversarial attacks and defenses on graphs: A review and empirical study. *arXiv preprint arXiv:2003.00653*.

[18] Jin, W., Ma, Y., Liu, X., Tang, X., Wang, S., and Tang, J. (2020b). Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 66–74.

[19] Khasanova, R. and Frossard, P. (2017). Graph-based isometry invariant representation learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1847–1856. JMLR. org.

[20] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[21] Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *international conference on learning representations*.

[22] Lanczos, C. (1950). *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA.

[23] Levie, R., Huang, W., Bucci, L., Bronstein, M. M., and Kutyniok, G. (2019). Transferability of spectral graph convolutional neural networks. *arXiv preprint arXiv:1907.12972*.

[24] Li, Y., Jin, W., Xu, H., and Tang, J. (2020). Deeprobust: A pytorch library for adversarial attacks and defenses. *arXiv preprint arXiv:2005.06149*.

[25] McAuley, J., Targett, C., Shi, Q., and Van Den Hengel, A. (2015). Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52.

[26] McCallum, A. K., Nigam, K., Rennie, J., and Seymore, K. (2000). Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163.

[27] Oono, K. and Suzuki, T. (2020). Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*.

[28] Ortega, A., Frossard, P., Kovačević, J., Moura, J. M., and Vandergheynst, P. (2018). Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828.

[29] Sen, P., Namata, G. M., Bilgic, M., Getoor, L., Gallagher, B., and Eliassi-Rad, T. (2008). Collective classification in network data. *Ai Magazine*, 29(3):93–106.

[30] Shchur, O., Mumme, M., Bojchevski, A., and Günnemann, S. (2018). Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*.

[31] Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98.

[32] Stewart, G. W. and Sun, J.-g. (1990). Matrix perturbation theory.

[33] Sun, L., Wang, J., Yu, P. S., and Li, B. (2018). Adversarial attack and defense on graph data: A survey. *arXiv preprint arXiv:1812.10528*.

[34] Sun, Y., Wang, S., Tang, X., Hsieh, T.-Y., and Honavar, V. (2020). Non-target-specific node injection attacks on graph neural networks: A hierarchical reinforcement learning approach. In *Proc. WWW*, volume 3.

[35] Tang, X., Li, Y., Sun, Y., Yao, H., Mitra, P., and Wang, S. (2020). Transferring robustness for graph neural network against poisoning attacks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 600–608.

[36] Tong, H., Prakash, B. A., Eliassi-Rad, T., Faloutsos, M., and Faloutsos, C. (2012). Gelling, and melting, large graphs by edge manipulation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 245–254.

[37] Tramer, F., Carlini, N., Brendel, W., and Madry, A. (2020). On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*.

[38] Verma, V., Qu, M., Kawaguchi, K., Lamb, A., Bengio, Y., Kannala, J., and Tang, J. (2019). Graphmix: Improved training of gnns for semi-supervised learning. *arXiv e-prints*, pages arXiv–1909.

[39] Wang, Y., Liu, S., Yoon, M., Lamba, H., Wang, W., Faloutsos, C., and Hooi, B. (2020). Provably robust node classification via low-pass message passing. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 621–630. IEEE.

[40] Wu, F., Zhang, T., Souza Jr., A. H., Fifty, C., Yu, T., and Weinberger, K. Q. (2019a). Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*.

[41] Wu, H., Wang, C., Tyshetskiy, Y., Docherty, A., Lu, K., and Zhu, L. (2019b). Adversarial examples for graph data: Deep insights into attack and defense. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4816–4823. AAAI Press.

[42] Wu, T., Ren, H., Li, P., and Leskovec, J. (2020a). Graph information bottleneck. *Advances in Neural Information Processing Systems*, 33.

[43] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. (2020b). A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*.

[44] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. (2019c). A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*.

[45] Xu, K., Chen, H., Liu, S., Chen, P.-Y., Weng, T.-W., Hong, M., and Lin, X. (2019). Topology attack and defense for graph neural networks: an optimization perspective. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3961–3967. AAAI Press.

[46] Xu, K., Li, C., Tian, Y., Sonobe, T., ichi Kawarabayashi, K., and Jegelka, S. (2018). Representation learning on graphs with jumping knowledge networks. *international conference on machine learning*, pages 5449–5458.

[47] Zhang, X. and Zitnik, M. (2020). Gnnguard: Defending graph neural networks against adversarial attacks. *Advances in Neural Information Processing Systems*, 33.

[48] Zhang, Z., Cui, P., and Zhu, W. (2020). Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

[49] Zhu, D., Cui, P., Zhang, Z., Pei, J., and Zhu, W. (2018). High-order proximity preserved embedding for dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 30:2134–2144.

[50] Zhu, D., Zhang, Z., Cui, P., and Zhu, W. (2019). Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1399–1407.

[51] Zügner, D. and Günnemann, S. (2020). Certifiable robustness of graph convolutional networks under structure perturbations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1656–1665.

[52] Zügner, D., Akbarnejad, A., and Günnemann, S. (2018). 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.

[53] Zügner, D. and Günnemann, S. (2019). Adversarial attacks on graph neural networks via meta learning. *international conference on learning representations*.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Error bars are not reported because it would be too computationally expensive for large datasets.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No] We were unable to find the license for the assets we used.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] The datasets are public benchmarks.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] The datasets are public benchmarks.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]