A Simple and Yet Fairly Effective Defense for Graph Neural Networks

Sofiane Enadir [1] **Yassine Abbahaddou** [2] **Michalis Vazirgiannis** [1 2] **Henrik Boström** [1]

Abstract

Graph neural networks (GNNs) have become the standard approach for performing machine learning on graphs. However, concerns have been raised regarding their vulnerability to small adversarial perturbations. Existing defense methods suffer from high time complexity and can negatively impact the model's performance on clean graphs. In this paper, we propose NoisyGCN, a defense method that injects noise into the GCN architecture. We derive a mathematical upper bound linking GCN's robustness to noise injection, establishing our method's effectiveness. Through empirical evaluations on the node classification task, we demonstrate superior or comparable performance to existing methods while minimizing the added time complexity.

1. Introduction

In recent years, graphs have garnered significant attention due to their natural occurrence in various fields, including social networks, chemo- and bio-informatics. The abundance of graph-structured data has necessitated the development of machine learning algorithms capable of operating on graphs. One such powerful technique is the graph neural network (GNN), which has emerged as a valuable tool for learning representations of nodes and graphs. Many GNNs belong to the family of Message Passing Neural Networks (MPNNs) (Gilmer et al., 2017), such as Graph Isomorphism Networks (GIN) (Xu et al., 2019b) and Graph Convolutional Networks (GCN) (Kipf & Welling, 2017). GNNs have proven successful in tackling numerous real-world problems. For example, in the field of chemistry, considerable attention has been devoted to employing deep learning systems based on graphs for drug screening and design, where molecules are represented as graphs (Kearnes et al., 2016). Furthermore, these methods have demonstrated efficacy in predicting protein functions modeled as graphs (You et al., 2021) and session-based recommendation systems (Wu et al., 2019b).

Recently, deep learning architectures, particularly in computer vision related tasks, have been shown to give unreliable predictions when subject to small perturbations to the input (Goodfellow et al., 2014). These perturbations, commonly referred to as Adversarial attacks, impose limitations on the practicality of neural networks for real-world problems. Notably, numerous investigations (Dai et al., 2018a; Zügner et al., 2018; Günnemann, 2022) have highlighted that graph neural networks (GNNs) are also prone to such adversarial attacks. By introducing slight structural or node feature-based perturbations, an attacker can successfully manipulate the model's predictions, thus posing a substantial threat to the reliability of GNNs. This vulnerability is particularly concerning in safety-critical applications like finance and healthcare. Consequently, it is imperative to delve into the vulnerability of these models by investigating and proposing new adversarial attacks and defenses. Numerous studies have been dedicated to developing techniques that mitigate the potential effects of perturbations and enhance the robustness of Message Passing Neural Networks (MPNNs). These proposed methods encompass strategies such as augmenting training data with adversarial examples and retraining the model (Feng et al., 2019), fortifying the robustness of pre-trained GNNs (Zhang & Zitnik, 2020), and more recently introducing robustness certificates (Schuchardt et al., 2021). While some of these defense methods have exhibited success in countering adversarial perturbations, they often entail a high level of complexity due to their underlying architecture. In many cases, the time complexity of these methods tends to increase heavily with the graph's size. Furthermore, a significant portion of these approaches requires extensive adaptations to the internal architecture posing challenge to integrate into different models.

Recently, there has been growing interest in utilizing adversarial weight perturbation as a means to improve the generalization capabilities of graph neural networks (GNNs) (Wu et al., 2023). In line with this perspective, the present study explores a defense strategy, denoted NoisyGCN, that leverages randomization by introducing random noise within the network architecture. We begin by establishing a mathematical formulation to the robustness of GNNs against adver-

[1]EECS, KTH Royal Institute of Technology, Stockholm, Sweden [2]LIX, Ecole Polytechnique, IP Paris, France. Correspondence to: Sofiane ENNADIR <ennadir@kth.se>.

2^{nd} AdvML Frontiers workshop at 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

sarial attacks targeting both the structure and features of the graphs. We afterwards theoretically analyze the effect of randomization in enhancing the robustness of GNNs. We note that while we will be focusing on the Graph Convolutional Networks (GCN), we consider that our analysis is model agnostic and can be applied to different architectures. Finally, we empirically evaluate the proposed perturbation defense for its effectiveness against various adversarial attack methods in comparison to other available defense approaches on commonly used real-world benchmark datasets.

Our main contributions can be summarized in the following points:

- We provide a mathematical formalization of adversarial attacks on GNNs and propose our general framework NoisyGCN for defending against structural perturbations through noise injection.
- We derive an upper bound, from a theoretical point, that demonstrates the effectiveness of our proposed framework in enhancing the robustness of GCN-based classifiers.
- We conduct extensive evaluations of our theoretical findings on the node classification task using various benchmark datasets. Our model is compared to several state-of-the-art defense methods, and in the majority of cases, our proposed framework demonstrates superior or comparable performance while minimizing the time complexity.

2. Related Work

In recent years, there has been significant interest in the field of attacking machine learning models (Goodfellow et al., 2014; Ren et al., 2020). While most research has focused on image-based attacks, there has been a growing body of work exploring attacks in discrete spaces, particularly in the context of graphs. However, the discrete nature of graphs poses unique challenges when applying attack methods from other domains. Similar to attacks on images, existing graph-based attack methods typically approach the problem as a search task, aiming to discover the closest adversarial perturbation to a given input graph. This approach has led to the development of various attack strategies. For example, Nettack (Zügner et al., 2018) introduced a targeted attack method that perturbs both the graph structure and node features using a greedy optimization algorithm. Building upon this, Mettack (Zügner & Günnemann, 2019) formulated the problem as a bi-level optimization task and employed meta-gradients to solve it. Zhan & Pei (2021) expanded this work by proposing a black-box gradient attack algorithm to overcome several limitations of the original work. Alternatively, Dai et al. (2018b) proposed the use of Reinforcement Learning to solve the search problem and generate adversarial attacks, offering a different perspective on tackling the challenge.

From another perspective, given the limitations of the aforementioned methods in terms of theoretical guarantees, there has been a growing interest in exploring robustness certificates (Zügner & Günnemann, 2019; Bojchevski & Günnemann, 2019) as a promising direction to address adversarial attacks by providing attack-independent guarantees regarding the stability of model predictions. For instance, (Bojchevski et al., 2020) introduced the use of randomized smoothing techniques to offer highly scalable model-agnostic certificates for graphs. Their approach provides a robustness guarantee that is independent of the attack method employed. Furthermore, (Jin et al., 2020) proposed robustness certificates specifically for GCN-based graph classification in the presence of topological perturbations. These certificates consider both local and global budgets, enabling a comprehensive analysis of the model's stability.

Recently, defending against adversarial attacks through the injection of noise into the architecture has emerged as a promising approach in the field of Computer Vision. Several studies (Pinot et al., 2019; Liu et al., 2018; Rakin et al., 2018) have showed that noise injection can enhance the robustness of networks against adversarial perturbations. In line with this, and in the context of GNNs, the work (Wu et al., 2023) investigated the effect of injecting noise, specifically adversarial weight perturbation, on improving the generalization of models. The findings of this study demonstrate that these perturbations effectively mitigate the vanishing-gradient issue and lead to significant enhancements in generalization performance. In our work, we extend upon these insights by considering the application of noise injection schemes to enhance the robustness of graph neural networks (GNNs) against adversarial attacks.

3. Preliminaries

Before continuing with our contribution, we begin by introducing notation and some fundamental concepts.

Notation and Problem Setup. Let $G = (V, E)$ be a graph where V is its set of vertices and E its set of edges. We will denote by $n = |V|$ and $m = |E|$ the number of vertices and number of edges, respectively. Let $\mathcal{N}(v)$ denote the set of neighbors of a node $v \in V$, i. e., $\mathcal{N}(v) = \{u \colon (v, u) \in E\}$. The degree of a node is equal to its number of neighbors, i. e., equal to $|\mathcal{N}(v)|$ for a node $v \in V$. A graph is commonly represented by its adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ which encodes edge information. The (i, j)-th element of the adjacency matrix is equal to the weight of the edge between the i-th and j-th node of the graph and a weight of 0 in case the edge does not exist. In some settings, the nodes of a graph might

be annotated with feature vectors. We use $\mathbf{X} \in \mathbb{R}^{n \times K}$ to denote the node features where K is the feature dimensionality. The feature of the i-th node of the graph corresponds to the i-th row of $\mathbf{X}$. In a node classification setting, we consider a graph G, represented by its adjacency matrix A and its node attribute matrix X. Formally, given a set of labeled $V_L \subset V$, where nodes are assigned exactly one class in $\mathcal{C} = \{y_1, y_2, \ldots, y_c\} \subset \mathcal{Y}$, the goal is to learn a function f_θ, which maps each node $v \in V$ to exactly one of the c classes in $\mathcal{C}$ while minimizing a classification loss (Cross entropy for example).

GNNs. A GNN model consists of a series of neighborhood aggregation layers which use the graph structure and the nodes' feature vectors from the previous layer to generate new representations for the nodes. Specifically, GNNs update nodes' feature vectors by aggregating local neighborhood information. Suppose we have a GNN model that contains T neighborhood aggregation layers. Let also $\mathbf{h}_v^{(0)}$ denote the initial feature vector of node v, i. e., the row of matrix $\mathbf{X}$ that corresponds to node v. At each iteration ($t > 0$), the hidden state $\mathbf{h}_v^{(t)}$ of a node v is updated as follows:

$$\mathbf{a}_v^{(t)} = \text{AGGREGATE}^{(t)}\Big(\big\{\mathbf{h}_u^{(t-1)} : u \in \mathcal{N}(v)\big\}\Big);$$
$$\mathbf{h}_v^{(t)} = \text{COMBINE}^{(t)}\Big(\mathbf{h}_v^{(t-1)}, \mathbf{a}_v^{(t)}\Big),$$

where AGGREGATE is a permutation invariant function that maps the feature vectors of the neighbors of a node v to an aggregated vector. This aggregated vector is passed along with the previous representation of v (i. e., $\mathbf{h}_v^{(t-1)}$) to the COMBINE function which combines those two vectors and produces the new representation of v.

4. Proposed Approach

In this section, we provide a mathematical formalization of robustness specifically tailored for Graph Neural Networks (GNNs). Subsequently, we investigate the impact of noise injection on the robustness of GNNs. Throughout our analysis, without loss of generality, we will focus on the semi-supervised node classification task as a representative scenario. Let us consider the following three metric spaces, the graph space associated with the adjacency matrices $(\mathcal{A}, \|\cdot\|_{\mathcal{A}})$, the feature space associated with the node feature attributes $(\mathcal{X}, \|\cdot\|_{\mathcal{X}})$ and the label space $(\mathcal{Y}, \|\cdot\|_{\mathcal{Y}})$. We finally consider an underlying probability distribution $\mathcal{D}$ defined on $(\mathcal{A}, \mathcal{X}, \mathcal{Y})$.

4.1. Graph Adversarial Attacks

Let's consider a trained victim classifier $f : (\mathcal{A}, \mathcal{X}) \to \mathcal{Y}$. Let $(A, X) \in (\mathcal{A}, \mathcal{X})$ be an input graph with its associated label vectors $y \in \mathcal{Y}$, such that $f(A, X) = y$. The objective of an adversarial attack is to generate a perturbed graph, represented by the adjacency matrix $\tilde{A}$ and the corresponding features $\tilde{X}$, which are only slightly different from the original input (A, X), resulting in a predicted class for $(\tilde{A}, \tilde{X})$ that differs from the predicted class for (A, X). The effectiveness of the adversarial attack relies on defining a similarity measure between the input graph and the adversarially generated graph. We hence introduce a distance metric that considers both the graph structure and its associated features:

$$d([A, X], [\tilde{A}, \tilde{X}]) = \min_{P \in \Pi}\{\|A - P\tilde{A}P^T\|_2 + \|X - P\tilde{X}\|_2\}, \quad (1)$$

with Π being the set of permutation matrices. It is important to note that for graphs without attributed features, the distance defined in Equation 1 corresponds to the widely used graph edit distance. This distance metric measures the similarity between two graphs by quantifying the minimum number of edge modifications required to transform one graph into another, while considering graph isomorphism. Building on this distance, we formulate the adversarial task as the search for a perturbed attributed graph $(\tilde{A}, \tilde{X})$ within a defined budget ϵ, such that $f(\tilde{A}, \tilde{X}) = \tilde{y} \neq y$, while satisfying $d([A, X], [\tilde{A}, \tilde{X}]) < \epsilon$. Additionally, as the attacker typically lacks access to the true labels, we consider an adversarial graph attack successful when $f(\tilde{A}, \tilde{X}) \neq f(A, X)$. From this perspective, we can define a quantity related to the vulnerability of a GNN to adversarial attacks within a certain budget ϵ as the following:

$$Adv_\epsilon[f] = \mathbb{P}_{(A,X)\sim\mathcal{D}_{\mathcal{A},\mathcal{X}}}[(\tilde{A}, \tilde{X}) \in B([A, X]; \epsilon) : d_{\mathcal{Y}}(f(\tilde{A}, \tilde{X}), f(A, X)) > 0], \quad (2)$$

with $B(A, X; \epsilon) = \{(\tilde{A}, \tilde{X}) : d([A, X], [\tilde{A}, \tilde{X}]) < \epsilon\}$ for any budget $\epsilon \geq 0$ and $d_{\mathcal{Y}}$ can be any defined distance in the measurable output space $\mathcal{Y}$. We can consider a GNN as robust when its vulnerability, as formulated in 2 is upper-bounded, we can consequently introduce the following robustness definition:

Definition 4.1. (Adversarial Robustness). Let d be a graph distance on the measurable sets $\mathcal{A}, \mathcal{X}$. The graph-based function $f : (\mathcal{A}, \mathcal{X}) \to \mathcal{Y}$ is said to be $d - (\epsilon, \gamma)$ robust if $Adv_\epsilon[f] \leq \gamma$.

We note that in the previous definition, we consider that a perturbed graph for which an attacker successfully flip one node's prediction is considered as a valid attack. In other setting that are less sensitive and in which the previous assumption doesn't hold, this definition can be easily generalized by adding an extra hyper-parameter to control the confidence level.

4.2. Effect of Noise Injection

In our study, we want to investigate the effect of noise addition in term of defending against adversarial attacks.

Specifically, we consider injecting noise sampled from a predefined distribution. We will hence, and similar to (Pinot et al., 2019), consider that our victim model f is a probabilistic mapping where an output is obtained by sampling from the mapping. Accordingly, we will consider the Kullback–Leibler (KL) divergence in our robustness quantification as introduced in Equation 2.

Our analysis will focus on the widely used Graph Convolutional Neural Network (GCN) (Kipf & Welling, 2017) within the broader context of GNNs. As introduced in Section 3, we can write an iteration of the iterative process of GCN as follows:

$$\Phi^{(\ell)} = \phi^{(\ell)}(\tilde{A}\Phi^{(\ell-1)}W^{(\ell)}), \tag{3}$$

where $\Phi^{(\ell)}$ represents the hidden state in the ℓ-th GCN layer with $\Phi^{(0)}$ corresponding to the initial node features $X \in \mathbb{R}^{n\times K}$, $W^{(\ell)} \in \mathbb{R}^{p\times e}$ is the weight matrix in the ℓ-th layer, e is the embedding dimension and $\phi^{(\ell)}$ is a 1-Lipschitz continuous non-linear activation function. Moreover, $\tilde{A} \in \mathbb{R}^{n\times n}$ is the normalized adjacency matrix $\tilde{A} = D^{-1/2}AD^{-1/2}$.

In the remaining of our analysis, we consider our victim model to be a GCN-based classifier utilizing 1-Lipschitz continuous activation functions, such as the Hyperbolic Tangent. Our focus will be on the injection of noise sampled from a centered Gaussian distribution ($\mathcal{N}(0, I)$) with a scaling parameter controlling its covariance matrix. The work (Wu et al., 2023) has shown that injecting noise at each layer can lead to a collapse in the model's generalization. As a result, we will restrict the introduction of noise to specific layers. Under these assumptions, our victim model can be expressed as $f(.) = \Phi^l \circ \ldots \circ \Phi^{i+1}(\Phi^i \circ \ldots \circ \Phi^1(.) + T)$, where T represents the Gaussian random variable.

Theorem 4.2. *Let f denote a graph-based function composed of 2 GCN layers, where the weight matrix of the i-th layer is denoted by $W^{(i)}$. We assume that f is based on 1-Lipschitz continuous activation functions. We consider injecting noise drawn from a centered Gaussian with a scaling parameter β. When subject to structural perturbations of the input graph (A, X), with a budget ϵ, the classifier f is d-(ϵ, γ) robust (with respect to Definition 4.1) with:*

$$\gamma = \frac{2(\|W^{(2)}\|\|W^{(1)}\|\|X\|\epsilon)^2}{\beta}.$$

Theorem 4.2 formulate an upper bound on GCN's robustness and establishes the connection between noise injection and defending against adversarial attacks based on structural perturbations with a predefined neighborhood and budget ϵ. A tighter upper bound intuitively signifies a higher level of robustness in the targeted victim model. Therefore, based on the results derived from the theorem, controlling the injected noise using the β parameter can effectively enhance the model's robustness. However, it is important to exercise caution when increasing the injected noise as it can potentially compromise the model's performance. Hence, striking a balance between defending against adversarial attacks and preserving the model's clean accuracy becomes crucial. Furthermore, although the previous theorem primarily focuses on a 2-layer GCN-based graph classifier known for its benchmark accuracy across diverse datasets, the results can be easily extended to graph classifiers with L layers. The proof of Theorem 4.2 is provided in Appendix A.

4.3. On the Complexity and Advantages of our Approach

Many existing defense methods suffer from a significant increase in complexity as the input graph size grows, making them challenging to apply in practical scenarios. For example, GNNGuard (Zhang & Zitnik, 2020) involves computing neighbor importance estimation, which has a complexity of $\mathcal{O}(e\times \mid E \mid)$, where e denotes the embedding dimension and $\mid E \mid$ represents the graph size. In contrast, our proposed approach based on noise injection in the architecture is advantageous due to its minimal complexity, requiring only sampling from a distribution. Additionally, unlike many existing methods, our approach does not compromise the performance of the underlying GCN when applied to clean, non-attacked graphs, as it will be demonstrated in our experimental results.

5. Experimental Results

This section focuses on empirically validating our theoretical findings by evaluating the performance of the proposed approach on real-world benchmarks. We begin by outlining the experimental settings employed in our study, followed by a comprehensive analysis and discussion of the obtained results. Through our experimental evaluation, we aim to address two key aspects: firstly, the effectiveness of our method in defending against adversarial attacks, particularly structural perturbations, and secondly, its capability to maintain the model's accuracy and performance, specially when tested on clean/non-attacked input graphs.

5.1. Experimental Setup

In our analysis, we focus on node classification where we use the citation networks Cora, CiteSeer and PubMed (Sen et al., 2008). Further information about the used datasets and implementation details are provided in Appendix B. For all the experiments, the baseline models consisted of a 2-layer GCN-based classifier combined with a Multi-Layer Perception (MLP) as a readout. This choice aimed to ensure a fair evaluation of the models' robustness within an iso-architectural framework. For our proposed approach, we have to chosen to inject noise after the first message passing step. The experiments were conducted using the

Table 1. Classification accuracy (± standard deviation) of the models on different benchmark node classification dataset before ("Clean") and after the attack application. The higher the accuracy (in %) the better the model. The best accuracy in each setting and each dataset is typeset in **bold**.

Attack	Dataset	Budget	GCN	GNNGuard	GCN-Jaccard	RGCN	GCN-SVD	NoisyGCN
Mettack	Cora	Clean	82.2 ± 0.3	77.5 ± 0.7	80.8 ± 0.4	83.5 ± 0.3	63.2 ± 2.4	**82.4 ± 0.5**
		Budget (5%)	79.2 ± 1.2	75.8 ± 0.6	77.9 ± 0.8	78.3 ± 0.6	61.3 ± 0.7	**81.2 ± 0.7**
		Budget (10%)	73.2 ±1.1	74.7 ± 0.4	**76.7 ± 0.7**	70.7 ± 0.8	63.8 ± 1.8	74.4 ± 0.6
	CiteSeer	Clean	71.8 ± 0.3	68.9 ± 1.5	71.2 ± 0.7	**72.2 ± 0.3**	63.7 ± 1.2	71.9 ± 0.2
		Budget (5%)	69.4 ± 2.1	69.9 ± 1.1	70.3 ± 2.3	70.6 ± 0.7	63.9 ± 0.8	**72.2 ± 0.6**
		Budget (10%)	67.5 ± 1.3	**70.0 ± 1.5**	67.2 ± 2.1	67.3 ± 0.4	65.0 ± 1.9	68.6 ± 0.3
	PubMed	Clean	84.9 ± 0.4	84.5 ± 0.6	84.9 ± 0.5	**85.2 ± 0.8**	81.0 ± 0.6	84.9 ± 0.8
		Budget (5%)	79.0 ± 0.2	**84.2 ± 0.9**	79.6 ± 0.3	81.1 ± 0.7	81.0 ± 0.2	81.8 ± 0.4
		Budget (10%)	64.5 ± 1.2	**84.1 ± 0.3**	67.4 ± 1.1	65.0 ± 0.4	81.0 ± 0.2	73.3 ± 0.6
PGD	Cora	Clean	82.2 ± 0.3	77.5 ± 0.7	80.8 ± 0.4	83.5 ± 0.3	63.2 ± 2.4	**82.4 ± 0.5**
		Budget (5%)	74.9 ± 0.8	71.0 ± 1.0	73.9 ± 0.8	75.8 ± 0.9	55.9 ± 0.6	76.6 ± 0.3
		Budget (10%)	71.8 ± 1.2	69.9 ± 1.6	72.2 ± 1.4	72.4 ± 1.8	55.3 ± 0.9	73.4 ± 0.5
	CiteSeer	Clean	71.8 ± 0.3	68.9 ± 1.5	71.2 ± 0.7	**72.2 ± 0.3**	63.7 ± 1.2	71.9 ± 0.2
		Budget (5%)	61.8 ± 1.1	57.9 ± 2.8	62.9 ± 1.5	58.1 ± 2.2	51.7 ± 1.3	**64.5 ± 1.2**
		Budget (10%)	60.5 ± 0.7	58.2 ± 3.8	61.3 ± 0.7	56.2 ± 0.8	50.5 ± 0.3	**62.2 ± 1.0**
	PubMed	Clean	84.9 ± 0.4	84.5 ± 0.6	84.9 ± 0.5	**85.2 ± 0.8**	81.0 ± 0.6	84.9 ± 0.8
		Budget (5%)	75.9 ± 0.8	75.3 ± 0.4	76.1 ± 0.7	78.5 ± 0.8	67.7 ± 1.5	**76.2 ± 0.7**
		Budget (10%)	64.8 ± 0.6	**70.7 ± 0.9**	64.7 ± 1.2	65.6 ± 0.9	67.5 ± 1.7	65.2 ± 1.1
DICE	Cora	Clean	82.2 ± 0.3	77.5 ± 0.7	80.8 ± 0.4	83.5 ± 0.3	63.2 ± 2.4	**82.4 ± 0.5**
		Budget (5%)	81.3 ± 0.8	76.4 ± 0.4	79.6 ± 0.6	81.9 ± 0.6	61.3 ± 1.3	**82.5 ± 0.8**
		Budget (10%)	79.5 ± 0.9	76.6 ± 0.5	78.6 ± 0.8	80.0 ± 0.6	61.5 ± 1.5	**80.5 ± 0.6**
	CiteSeer	Clean	71.8 ± 0.3	68.9 ± 1.5	71.2 ± 0.7	**72.2 ± 0.3**	63.7 ± 1.2	71.9 ± 0.2
		Budget (5%)	69.9 ± 0.3	68.5 ± 1.6	70.9 ± 0.4	69.3 ± 0.5	65.4 ± 1.6	**70.8 ± 0.3**
		Budget (10%)	68.2 ± 1.7	69.9 ± 1.5	69.9 ± 0.6	67.8 ± 1.1	65.1 ± 1.5	**70.4 ± 0.8**
	PubMed	Clean	84.9 ± 0.4	84.5 ± 0.6	84.9 ± 0.5	**85.2 ± 0.8**	81.0 ± 0.6	84.9 ± 0.8
		Budget (5%)	83.4 ± 0.6	**84.0 ± 0.8**	83.4 ± 0.7	83.8 ± 0.6	81.5 ± 0.8	83.6 ± 0.9
		Budget (10%)	81.7 ± 1.1	**83.6 ± 1.0**	81.8 ± 0.5	82.4 ± 0.8	81.4 ± 0.5	82.1 ± 2.3

Adam optimizer (Kingma & Ba, 2014) and standardized hyperparameters, including a learning rate of 1e-2, 300 epochs, and a hidden feature dimension of 16. To reduce the impact of random initialization, we repeated each experiment 10 times and used the train/validation/test splits provided with the datasets (Yang et al., 2016).

Attacks. We use three main global structural based adversarial attacks: **(i)** We first consider the optimization-based formulation of the adversarial task Mettack with the "Meta-Self" training strategy. **(ii)** We afterwards consider another optimization based adversarial attack based on Proximal Gradient Descent (PGD) (Xu et al., 2019a). **(iii)** We finally consider DICE (Zügner & Günnemann, 2019). For all these attacks, we tested and considered two perturbation budgets $\Delta = 5\%$ and $\Delta = 10\%$.

Baseline Models. We additionally conducted a comprehensive empirical evaluation by comparing our proposed NoisyGCN against five baseline defense algorithms that focus on structural perturbations: GNN-Jaccard (Wu et al., 2019a), RobustGCN (Zhu et al., 2019), GNN-SVD (Entezari et al., 2020), and GNNGuard (Zhang & Zitnik, 2020).

5.2. Experimental Results

Table 1 presents the average node classification accuracies for the GCN, the GNNGUard, GCN-Jaccard, RGCN, GCN-SVD and the proposed approach, NoisyGCN.

The empirical findings reveal that, in the absence of attacks, the proposed approach demonstrates comparable accuracy to the classical GCN, and in some cases, it even improves the model's generalization and performance, as studied by prior research (Wu et al., 2023). Importantly, these results affirm that our approach does not compromise the perfor-

mance of the underlying network, addressing our second research question. This is particularly significant as real-world scenarios often involve uncertain knowledge regarding potential malicious perturbations on the input graph. Hence, it is crucial that an effective defense strategy does not diminish the predictive capabilities of the model, while simultaneously enhancing its robustness. From another perspective, the results indicate that our proposed approach, NoisyGCN, performs on par with or even surpasses state-of-the-art defense baselines in several instances. Notably, it demonstrates greater efficiency when subjected to the "DICE" attack framework. Moreover, we observe consistent outperformance compared to GCN-SVD, GCN-Jaccard, and RGCN, while also exhibiting competitive performance against the highly performant GNNGuard. It is important to highlight that despite similar performance to GNNGuard, our approach offers significantly reduced complexity in term of operation and time. The complete time analysis study is provided in Appendix C.

6. Conclusion

In this study, we introduce NoisyGCN, a cost-effective and highly effective defense method for Graph Neural Networks (GNNs). Through a theoretical analysis, we establish a clear connection between noise injection in the victim model's architecture and improved robustness. Our proposed method offers the key advantage of minimal added complexity while delivering strong defense performance. Experimental comparisons on diverse real-world datasets demonstrate that NoisyGCN achieves similar or superior performance compared to the standard GCN and existing defense methods. While our focus was on GCN in this study, our theoretical analysis can be extended to other GNNs architectures, which we plan to explore in future work.

Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computation (through GPU) was enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at Alvis partially funded by the Swedish Research Council through grant agreement no. "2022-06725" and was performed using HPC resources from GENCI-IDRIS (Grant 2023-AD010613410R1).

References

Bojchevski, A. and Günnemann, S. Certifiable robustness to graph perturbations, 2019. URL https://arxiv.org/abs/1910.14356.

Bojchevski, A., Klicpera, J., and Günnemann, S. Efficient robustness certificates for discrete data: Sparsity-aware randomized smoothing for graphs, images and more, 2020. URL https://arxiv.org/abs/2008.12952.

Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J., and Song, L. Adversarial Attack on Graph Structured Data. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1115–1124, 2018a.

Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J., and Song, L. Adversarial Attack on Graph Structured Data. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1115–1124, 2018b.

Entezari, N., Al-Sayouri, S. A., Darvishzadeh, A., and Papalexakis, E. E. 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*, WSDM '20, pp. 169–177, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368223. doi: 10.1145/3336191.3371789. URL https://doi.org/10.1145/3336191.3371789.

Feng, F., He, X., Tang, J., and Chua, T.-S. Graph adversarial training: Dynamically regularizing based on graph structure, 2019. URL https://arxiv.org/abs/1902.08226.

Fey, M. and Lenssen, J. E. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry, 2017. URL https://arxiv.org/abs/1704.01212.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples, 2014. URL https://arxiv.org/abs/1412.6572.

Günnemann, S. Graph neural networks: Adversarial robustness. In *Graph Neural Networks: Foundations, Frontiers, and Applications*, pp. 149–176. Springer, 2022.

Jin, H., Shi, Z., Peruri, V. J. S. A., and Zhang, X. Certified robustness of graph convolution networks for graph classification under topological attacks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 8463–8474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/609a199881ca4ba9c95688235cd6ac5c-Paper.pdf.

Kearnes, S., McCloskey, K., Berndl, M., Pande, V., and Riley, P. Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980.

Kipf, T. N. and Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, 2017.

Liu, X., Cheng, M., Zhang, H., and Hsieh, C.-J. Towards robust neural networks via random self-ensemble, 2018.

Pinot, R., Meunier, L., Araujo, A., Kashima, H., Yger, F., Gouy-Pailler, C., and Atif, J. Theoretical evidence for adversarial robustness through randomization, 2019.

Rakin, A. S., He, Z., and Fan, D. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack, 2018.

Ren, K., Zheng, T., Qin, Z., and Liu, X. Adversarial attacks and defenses in deep learning. *Engineering*, 6(3):346–360, 2020. ISSN 2095-8099. doi: https://doi.org/10.1016/j.eng.2019.12.012. URL https://www.sciencedirect.com/science/article/pii/S209580991930503X.

Schuchardt, J., Bojchevski, A., Gasteiger, J., and Günnemann, S. Collective robustness certificates: Exploiting interdependence in graph neural networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ULQdiUTHe3y.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. Collective classification in network data. *AI Magazine*, 29 (3):93, Sep. 2008. doi: 10.1609/aimag.v29i3.2157. URL https://ojs.aaai.org/index.php/aimagazine/article/view/2157.

Wu, H., Wang, C., Tyshetskiy, Y., Docherty, A., Lu, K., and Zhu, L. Adversarial examples for graph data: Deep insights into attack and defense. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 4816–4823. International Joint Conferences on Artificial Intelligence Organization, 7 2019a. doi: 10.24963/ijcai.2019/669. URL https://doi.org/10.24963/ijcai.2019/669.

Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., and Tan, T. Session-based Recommendation with Graph Neural Networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pp. 346–353, 2019b.

Wu, Y., Bojchevski, A., and Huang, H. Adversarial weight perturbation improves generalization in graph neural networks, 2023.

Xu, K., Chen, H., Liu, S., Chen, P.-Y., Weng, T.-W., Hong, M., and Lin, X. Topology attack and defense for graph neural networks: An optimization perspective. 2019a. doi: 10.48550/ARXIV.1906.04214. URL https://arxiv.org/abs/1906.04214.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How Powerful are Graph Neural Networks? In *7th International Conference on Learning Representations*, 2019b.

Yang, Z., Cohen, W. W., and Salakhutdinov, R. Revisiting semi-supervised learning with graph embeddings, 2016. URL https://arxiv.org/abs/1603.08861.

You, R., Yao, S., Mamitsuka, H., and Zhu, S. DeepGraphGO: graph neural network for large-scale, multi-species protein function prediction. *Bioinformatics*, 37 (Supplement$_1$) : $i262 - -i271, 072021. ISSN1367 - 4803. doi : .$ URL https://doi.org/10.1093/bioinformatics/btab270.

Zhan, H. and Pei, X. Black-box Gradient Attack on Graph Neural Networks: Deeper Insights in Graph-based Attack and Defense. *arXiv preprint arXiv:2104.15061*, 2021.

Zhang, X. and Zitnik, M. Gnnguard: Defending graph neural networks against adversarial attacks, 2020. URL https://arxiv.org/abs/2006.08149.

Zhu, D., Zhang, Z., Cui, P., and Zhu, W. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '19, pp. 1399–1407, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. 10.1145/3292500.3330851. URL https://doi.org/10.1145/3292500.3330851.

Zügner, D. and Günnemann, S. Adversarial attacks on graph neural networks via meta learning. In *7th International Conference on Learning Representations*, 2019.

Zügner, D., Akbarnejad, A., and Günnemann, S. Adversarial Attacks on Neural Networks for Graph Data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2847–2856, 2018.

Zügner, D. and Günnemann, S. Certifiable robustness and robust training for graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery &amp Data Mining*. ACM, jul 2019. 10.1145/3292500.3330905. URL https://doi.org/10.1145%2F3292500.3330905.

Zügner, D., Akbarnejad, A., and Günnemann, S. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, jul 2018.

A. Proof Of Theorem 4.2

Theorem Let f denote a graph-based function composed of 2 GCN layers, where the weight matrix of the i-th layer is denoted by $W^{(i)}$. We assume that f is based on 1-Lipschitz continuous activation functions. We consider injecting noise drawn from a centered Gaussian with a scaling parameter β. When subject to structural perturbations of the input graph (A, X), with a budget ϵ, the classifier f is d-(ϵ, γ) robust (with respect to Definition 4.1) with:

$$\gamma = \frac{2(\|W^{(2)}\|\|W^{(1)}\|\|X\|\epsilon)^2}{\beta}.$$

Proof: We consider f to be a 2-Layers GCN-based classifier with 1-Lipschitz continuous activation functions. Let A' the produced perturbed adjacency matrix. We consider injecting noise that is sampled from a Gaussian distribution $\mathcal{N}(0, I)$ with a scaling parameter β. We start therefore by considering the general case of a centered Gaussian with a matrix parameter Σ.

Let's consider the general case of KL divergence which is the Renyi Divergence with parameter λ. Our victim model can be transformed into a probabilistic mapping using the Dirac distribution. Based on the general formula of the sum (convolution) of two random variables, we have the following:

$$\begin{aligned}
d_{R,\lambda}(f(A,X), f(A',X)) &= \frac{1}{\lambda-1}\log\int_{\mathbb{R}^e}(p_G * \delta_{f(A,X)})^{\lambda}(p_G * \delta_{f(A',X)})^{(1-\lambda)}d\mu \\
&= \frac{1}{\lambda-1}\log\int_{\mathbb{R}^e}\frac{e^{-1/2[\lambda(z-f(A,X))^T\Sigma^{-1}(z-f(A,X))+(1-\lambda)(z-f(A',X))^T\Sigma^{-1}(z-f(A',X))]}}{(2\pi)^{\frac{n}{2}}\mid\Sigma\mid^{\frac{1}{2}}} \\
&= \frac{\lambda-1}{2}[f(A,X)-f(A',X)]^T(\Sigma^{-1})[f(A,X)-f(A',X)] \\
&\leq \frac{\lambda}{2}\sigma_{max}(\Sigma^{-1})\|f(A,X)-f(A',X)\|^2
\end{aligned}$$

From this result, we can deduce the KL divergence:

$$\begin{aligned}
d(f(A,X), f(A',X)) &= \lim_{\lambda\to 1} d_{R,\lambda}(f(A,X), f(A',X)) &(4)\\
&= \frac{1}{2}\sigma_{max}(\Sigma^{-1})\|f(A,X)-f(A',X)\|^2 &(5)
\end{aligned}$$

In what follows, we study the effect of input perturbation which is reflected by the quantity $\|f(A,X)-f(A',X)\|$ in the previous result:

$$\begin{aligned}
\|f(A,X), f(A',X)\| &= \|\phi^{(2)}(\tilde{A}\Phi^{(1)}(A,X)W^{(2)}) - \phi^{(2)}(\tilde{A'}\Phi^{(1)}(A',X)W^{(2)})\| \\
&\leq \|\tilde{A}\Phi^{(1)}(A,X)W^{(2)} - \tilde{A'}\Phi^{(1)}(A',X)W^{(2)}\| \\
&\leq \|W^{(2)}\|\|\tilde{A}\Phi^{(1)}(A,X) - \tilde{A}\Phi^{(1)}(A',X) + \tilde{A}\Phi^{(1)}(A',X) - \tilde{A'}\Phi^{(1)}(A',X)\| \\
&\leq \|W^{(2)}\|\|\tilde{A}[\Phi^{(1)}(A,X) - \Phi^{(1)}(A',X)] + \Phi^{(1)}(A',X)[\tilde{A}-\tilde{A'}]\|
\end{aligned}$$

We also have the following:

$$\begin{aligned}
\|\Phi^{(1)}(A,X) - \Phi^{(1)}(A',X)\| &= \|\phi^{(1)}(\tilde{A}XW^{(1)}) - \phi^{(1)}(\tilde{A'}XW^{(1)})\| \\
&\leq \|\tilde{A}XW^{(1)} - \tilde{A'}XW^{(1)}\| \\
&\leq \|W^{(1)}\|\|X\|\epsilon
\end{aligned}$$

From the previous result and using the triangular inequality, we have the following:

$$\begin{aligned}
\|W^{(2)}\|\|\tilde{A}[\Phi^{(1)}(A,X) - \Phi^{(1)}(A',X)] + \Phi^{(1)}(A',X)[\tilde{A}-\tilde{A'}]\| &\leq \|W^{(2)}\|[\|W^{(1)}\|\|X\|\epsilon + \|W^{(1)}\|\|X\|\epsilon] \\
&= 2\|W^{(2)}\|\|W^{(1)}\|\|X\|\epsilon
\end{aligned}$$

We finally conclude that:

$$d(f(A,X),f(A',X)) = \frac{1}{2\beta}\|f(A,X) - f(A',X)\|^2$$
$$= \frac{2(\|W^{(2)}\|\|W^{(1)}\|\|X\|\epsilon)^2}{\beta}$$

B. Datasets and Implementation Details

Characteristics and information about the datasets utilized in the node classification part of the study are presented in Table 2. As outlined in the main paper, we conduct experiments the citation networks Cora, CiteSeer, and PubMed (Sen et al., 2008). For these benchmarks, we adhere to the train/valid/test splits provided by the datasets.

Table 2. Statistics of the node classification datasets used in our experiments.

DATASET	#FEATURES	#NODES	#EDGES	#CLASSES
CORA	1433	2708	5208	7
CITESEER	3703	3327	4552	6
PUBMED	500	19717	44338	3

B.1. Implementation Details

Our implementation is built using the open-source library *PyTorch Geometric* (PyG) under the MIT license (Fey & Lenssen, 2019). We leveraged the publicly available implementation of the different benchmarks from their available repositories : From GNNGuard [1], for RGCN, GCN-Jaccard and GCN-SVD, we used the implementation from the DeepRobust package. Note that we additionally utilized the PyTorch DeepRobust package[2] to implement the adversarial attacks used in this study. The experiments have been run on both a Tesla V100 GPU.

C. Time Complexity Analysis

Our proposed method NoisyGCN, based on noise injection, offers a clear advantage in term of its required complexity. It only requires sampling from a pre-defined distribution during each forward pass, independent of the graph size. Therefore, its complexity does not increase with the graph size, unlike other baselines. For example, GNNGuard, the state-of-the-art defense method, involves computing neighbor importance estimation, which results in complexity that scales with the input graph. We empirically validate this observation by comparing the training time complexity of each baseline.

Table 3. Mean training time analysis (in s) of the NoisyGCN in comparison to other baselines.

DATASET	GNNGUARD	GCN-JACCARD	RGCN	GCN-SVD	NOISYGCN
CORA	14.19	1.56	0.73	0.47	0.52
CITESEER	16.65	1.18	0.68	0.46	0.48
PUBMED	695.31	10.24	12.18	3.65	0.89

The analysis of training time, as presented in Table 3, highlights the distinct time complexities observed between NoisyGCN and the other baseline methods. Specifically, there is a notable disparity in training time complexity between NoisyGCN and GNNGuard. While GCN-SVD exhibits a comparable time complexity to our approach, the superior defense capabilities of NoisyGCN, as demonstrated in Table 1, differentiate it from GCN-SVD. Furthermore, the results obtained on the PubMed dataset affirm the motivation outlined in our paper, illustrating that the majority of existing methods impose a complexity burden when dealing with large graphs.

[1] https://github.com/mims-harvard/GNNGuard
[2] https://github.com/DSE-MSU/DeepRobust