Node Feature Kernels Increase Graph Convolutional Network Robustness

Mohamed El Amine Seddik, Changmin Wu, Johannes Lutzeyer, Michalis Vazirgiannis

▶ **To cite this version:**

Mohamed El Amine Seddik, Changmin Wu, Johannes Lutzeyer, Michalis Vazirgiannis. Node Feature Kernels Increase Graph Convolutional Network Robustness. International Conference on Artificial Intelligence and Statistics (AISTATS), Mar 2022, Online, France. hal-04447471

HAL Id: hal-04447471
https://hal.science/hal-04447471

Submitted on 8 Feb 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Node Feature Kernels Increase Graph Convolutional Network Robustness

Mohamed El Amine Seddik
Huawei, Paris, France

Changmin Wu
LIX, École Polytechnique,
Institute Polytechnique de Paris, France

Johannes F. Lutzeyer
LIX, École Polytechnique,
Institute Polytechnique de Paris, France

Michalis Vazirgiannis
LIX, École Polytechnique,
Institute Polytechnique de Paris, France;
AUEB, Athens, Greece

Abstract

The robustness of the much used Graph Convolutional Networks (GCNs) to perturbations of their input is becoming a topic of increasing importance. In this paper the *random* GCN is introduced for which a random matrix theory analysis is possible. This analysis suggests that if the graph is sufficiently perturbed, or in the extreme case random, then the GCN fails to benefit from the node features. It is furthermore observed that enhancing the message passing step in GCNs by adding the node feature kernel to the adjacency matrix of the graph structure solves this problem. An empirical study of a GCN utilised for node classification on six real datasets further confirms the theoretical findings and demonstrates that perturbations of the graph structure can result in GCNs performing significantly worse than Multi-Layer Perceptrons run on the node features alone. In practice, adding a node feature kernel to the message passing of perturbed graphs results in a significant improvement of the GCN's performance, thereby rendering it more robust to graph perturbations. Our code is publicly available at: https://github.com/ChangminWu/RobustGCN.

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).

1 INTRODUCTION

In recent years Graph Neural Networks (GNNs) have been a highly impactful model type for the analysis of graph data. This is mainly due to their dominating empirical performance and ability to process attributed graphs composed of node information and an underlying graph structure. Many GNN architectures have been proposed, successively improving on weaknesses of previous architectures (e.g. Corso et al. (2020); Hamilton et al. (2017); Xu et al. (2019)). A popular GNN architecture which has remained a benchmark throughout the past years, partly due to the simplicity of its model equation and partly due to its good performance is the Graph Convolutional Network (GCN) (Kipf and Welling, 2017). The GCN is part of a class of GNNs called message passing neural networks (Gilmer et al., 2017), where the computations are split into a message passing step in which node features are aggregated over neighbourhoods in the underlying graph structure and an update step in which node features are processed, most commonly by a Multi-Layer Perceptron (MLP).

While much work is being done in the empirical exploration of GNNs, relatively fewer advances have been made in their theoretical analysis. A major advance in the theoretical line of research was the expressivity analysis of different message passing operators performed by Xu et al. (2019) and Morris et al. (2019). This analyses inspired many researchers to further investigate the expressivity of GNNs and resulted in a multitude of new architectures being proposed (Maron et al., 2019; Dasoulas et al., 2020). Another upcoming topic in the development of GNNs is their robustness to perturbations of the underlying graph structure (Zügner and Günnemann, 2019; Sun et al., 2020).

In the presented work, we introduce the *random* GCN, in which parameters of the update step are randomly sampled from Gaussian distributions rather than trained as is commonly the case. The *random* GCN allows us to make use of several powerful random matrix theory tools to gain a theoretical understanding of the factors driving the inference obtained from the GCN model. Our most insightful hypothesis obtained in this way is that *the message passing step dilutes (or in the extreme case completely ignores) information present in the node features if the underlying graph structure is noisy (or in the extreme case completely random).* In our theoretical analysis we observe that if information of the node features is introduced to the message passing operation, then this loss of information is avoided. This leads us to hypothesise that the addition of the node feature kernel to the message passing operators in GNNs could render them more robust to noise or mispecification of the underlying graph structure.

In a second part of our presented work we test the hypotheses, obtained in our study of the *random* GCN, on the state-of-the-art GCN architecture applied to six real-world benchmark datasets. This allows us to empirically verify our theoretical insight, rendering the random features approach for theoretical analysis a promising avenue for further theoretical study of GNN architectures, and the inclusion of node feature information in the message passing step a valid method to increase the robustness of GNNs.

Our main findings may be summarised as: **(i)** We contribute both a theoretical and an empirical understanding of how graph and node feature information is processed by the GCN, and **(ii)** importantly find that the preservation of node feature information is entirely dependent on an informative underlying graph structure. **(iii)** We furthermore, propose a novel GCN message passing scheme which results in more robust inference from a GCN to structural noise.

The remainder of this paper is organised as follows. In Section 2 we introduce related literature. In Section 3 we propose the *random* GCN and analyse it using tools from random matrix theory. The theoretical insight from Section 3 is then empirically verified in Section 4, where we confirm our hypotheses on the standard GCN on six benchmark datasets and observe the robust performance of the GCN when the node feature kernel matrix is added in the message passing step.

2 RELATED WORK

There exists an extensive literature branch which studies *adversarial attack and defence strategies on graph data* in the context of GNNs summarised in Günnemann (2022), Sun et al. (2020) and Zhou et al. (2020) with the latter pointing out directly the need for the development of more robust GNNs. In this paper we present one approach to robustifying the performance of GNNs to graph perturbations. In this literature the focus often lies on specific attack strategies perturbing the graph structure in order to alter the inference obtained from a GNN, most commonly the GCN, and defence strategies which aim to develop methodology which is robust to these attacks. Recent advances in this literature include, Zügner and Günnemann (2019) proposing a meta learning approach to find optimal graph perturbations. Their perturbation mechanism is found to drastically decrease the global performance of GNNs to be in some cases worse than simple benchmarks such as logistic regression run on the node features only. Zügner and Günnemann (2020) propose an algorithm which certifies robustness of individual nodes for the GCN used for node classification under perturbations of the graph structure. In Geisler et al. (2020) and Jin et al. (2021) the message passing operator in the GCN is replaced by the Soft Mediod function and the sum of several distance based adjacency matrices, respectively, with the aim of more robust GCN performance. Jin et al. (2020) propose to learn the graph structure jointly with the GNN parameters to robustify performance and also Entezari et al. (2020) propose to alter the graph structure by using a low rank approximation of the adjacency matrix. The works of Zhu et al. (2019) and Zhang and Zitnik (2020) are most closely related to our proposition of using a node feature kernel to reweight edges in Section 3.3 as they both propose to reweight edges based on the node features. Our theoretical findings in Section 3 support this approach of more directly taking the node features into account in the aggregation scheme of GNNs to increasing their robustness.

This paper distinguishes itself from adversarial attacks and defence literature fundamentally in that we study untargeted, random graph perturbations which arise as a result of mispecification of the data or uncertainty in the recording methods of the networks. For this kind of perturbation we are able to provide both theoretical (on a toy data example) and empirical understanding, which enables us to offer a distinction between the node feature data and the graph data in networks data sets and how these different information sources are processed by a GNN architecture.

Our work is also related to the literature studying the challenges that heterophilic graphs pose for GNNs (Pei et al., 2020; Zhu et al., 2020, 2021). This literature distinguishes homophilic and heterophilic graphs, in which edges in the graph predominantly connect nodes of equal and unequal classes, respectively. Both of

these structures can be, from a theoretical standpoint, equally class-informative, it is only the structure of the class-information which varies. In our work here we consider an orthogonal problem, which is the situation of a diminishing class-structure in the graph, independent of its homo- or heterophilic nature, and the effect this diminishment has on the ability of GNNs to process the information contained in the node features.

3 ANALYSIS OF THE RANDOM GCN

In this section we present our theoretical analysis and main findings. Throughout this section $\|\cdot\|$ denotes the Euclidean (resp., spectral) norm for vectors (resp., matrices); $\|\cdot\|_F$ denotes the Frobenius norm. Specifically, we consider a *random* GCN model[1], defined as

$$\boldsymbol{\Phi} = \sigma(\tilde{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}), \tag{1}$$

where $\tilde{\boldsymbol{A}} \in \mathbb{R}^{n\times n}$ denotes the normalised adjacency operator encoding the graph structure (see (3) for its definition), $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ corresponds to the node features, $\boldsymbol{W} \in \mathbb{R}^{p\times d}$ is a random matrix with $W_{ij} \sim \mathcal{N}(0,1)$ independent and identically distributed (i.i.d.) and σ is an activation function applied entry-wise. In particular, we will study the spectral behaviour of the *Gram matrix*[2] defined as

$$\boldsymbol{G} = \frac{1}{d}\boldsymbol{\Phi}\boldsymbol{\Phi}^\intercal = \frac{1}{d}\sigma(\tilde{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W})\sigma(\boldsymbol{W}^\intercal\boldsymbol{X}^\intercal\tilde{\boldsymbol{A}}^\intercal). \tag{2}$$

To analyse $\boldsymbol{G}$ we require assumptions on the node features and graph structure.

Assumption 1 (Node features)**.** *We suppose that $\boldsymbol{X}^\intercal = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{p\times n}$, where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are independent node feature vectors, each being a sample from one of $k = 2$ distribution classes $\mathcal{C}_1$ and $\mathcal{C}_2$. We further assume that the node feature vectors $\boldsymbol{x}_i$ follow a Gaussian mixture model; Specifically, for $\boldsymbol{x}_i \in \mathcal{C}_a$, $\boldsymbol{x}_i = (-1)^a \frac{\boldsymbol{\mu}}{\sqrt{p}} + \boldsymbol{z}_i$ for some vector $\boldsymbol{\mu} \in \mathbb{R}^p$ and $\boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p/p)$.*

We stress that Assumption 1 can be relaxed to a larger class of random vectors $\boldsymbol{x} \in \mathcal{X}$, where $\mathcal{X}$ denotes any normed space, satisfying the concentration property $\mathbb{P}(|\varphi(\boldsymbol{x}) - \mathbb{E}[\varphi(\boldsymbol{x})]| > t) \leq Ce^{-(t/\sigma)^q}$ with $q \in \mathbb{R}^+$, for all 1-Lipschitz functions $\varphi : \mathcal{X} \to \mathbb{R}$. Such vectors are called *random concentrated vectors* and have the particular property to be stable by Lipschitz transformations (Louart and Couillet, 2018). The simplest example of concentrated vectors is the standard Gaussian vector $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ (Ledoux, 2005). A more complicated class of examples arises from the fact that the concentration property is stable through Lipschitz maps: if $\boldsymbol{z} \in \mathbb{R}^d$ is concentrated and $g : \mathbb{R}^d \to \mathbb{R}^p$ is 1-Lipschitz, then $g(\boldsymbol{z})$ is also concentrated. A large family of *generative models* falls under this more complicated class of examples, such as, the "fake" images generated by Generative Adversarial Networks due to these images being constructed as Lipschitz transformations of random Gaussian vectors (Seddik et al., 2020).

Now we introduce the underlying model that defines the graph structure. We assume that the adjacency matrix $\boldsymbol{A}$ of the graph is generated by a stochastic block model (Karrer and Newman, 2011).

Assumption 2 (Graph structure)**.** *We assume that the entries of $\boldsymbol{A}$ are independent (except for $A_{ii} = 1$ for all i) Bernoulli random variables with parameter $\pi_{ij} = q^2 C_{ab} \in (0,1)$ for $\boldsymbol{x}_i \in \mathcal{C}_a$ and $\boldsymbol{x}_j \in \mathcal{C}_b$. In particular, $q \in (0,1)$ represents the probability of an edge occurring between two nodes, while C_{ab} represents the probability of an edge arising between nodes in classes $\mathcal{C}_a$ and $\mathcal{C}_b$.*

Note that self-loops are implicitly added in Assumption 2, where we assume $A_{ii} = 1$ for all i. Therefore, we consider that the normalised adjacency operator in (1) is defined as

$$\tilde{\boldsymbol{A}} = \frac{1}{\sqrt{n}}\left(\boldsymbol{A} - \boldsymbol{q}\boldsymbol{q}^\intercal\right), \tag{3}$$

where $\boldsymbol{q} = q\boldsymbol{1}_n$[3]. The centring by $\boldsymbol{q}\boldsymbol{q}^\intercal$ is necessary for the eigenvectors corresponding to the extremal eigenvalues of the operator to be class informative (Li et al., 2018), i.e., the centering operation removes the uninformative eigenvector corresponding to the largest eigenvalue of the adjacency matrix, simplifying the theoretical analysis. Specifically, for our analysis in the asymptotic regime where $n \to \infty$ (see Assumption 3 subsequently), the centring with $\boldsymbol{q}\boldsymbol{q}^\intercal$ and the normalisation by $\frac{1}{\sqrt{n}}$ are required so that $\tilde{\boldsymbol{A}}$ has a bounded spectral norm asymptotically. In practice, the centring by $\boldsymbol{q}\boldsymbol{q}^\intercal$ is not feasible as it results in a dense matrix. In our experiments in Section 4, we see this discrepancy to be of little consequence in practice.

Remark 1. *Assumption 2 allows us to sample directed as well as undirected graphs. Often the spectral analysis of graphs needs to be restricted to undirected graphs, since the analysis of complex-valued spectra arising for directed graphs poses a significant challenge. We are able to include directed graphs since the Gram matrix,*

[1] In Section 4.1, we show that the performance of the large *random* GCN matches that of the vanilla GCN.

[2] $\boldsymbol{G}$ provides access to the internal functioning and performance evaluation of the random GCN.

[3] The vectors $\boldsymbol{q}$ can be consistently estimated through the degree vector $\boldsymbol{d} = \boldsymbol{A}\boldsymbol{1}_n$ as $\boldsymbol{q} \approx \boldsymbol{d}/\sqrt{\boldsymbol{d}^\intercal\boldsymbol{1}_n}$.

analysed in Section 3.1, and $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$*, analysed in Section 3.2, have real spectra even if the underlying graph structure is directed.*

3.1 Spectral Behaviour of the Gram Matrix

Let $\tilde{\boldsymbol{X}}^\intercal = [\tilde{\boldsymbol{x}}_1, \dots, \tilde{\boldsymbol{x}}_n] = \boldsymbol{X}^\intercal \tilde{\boldsymbol{A}} \in \mathbb{R}^{p\times n}$, the entries of the Gram matrix defined in (2) are given by

$$G_{ij} = \frac{1}{d}\sigma(\boldsymbol{W}^\intercal \tilde{\boldsymbol{x}}_i)^\intercal \sigma(\boldsymbol{W}^\intercal \tilde{\boldsymbol{x}}_j) = \frac{1}{d}\sum_{\ell=1}^{d} \sigma(\boldsymbol{w}_\ell^\intercal \tilde{\boldsymbol{x}}_i)\sigma(\boldsymbol{w}_\ell^\intercal \tilde{\boldsymbol{x}}_j),$$

where $\boldsymbol{w}_\ell^\intercal$ denotes the ℓ-th row of $\boldsymbol{W}^\intercal$. Since all the $\boldsymbol{w}_\ell$ follow the same distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$, taking the expectation over $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ (conditionally on $\boldsymbol{X}$ and $\boldsymbol{A}$) yields the average Gram matrix $\bar{\boldsymbol{G}}$ defined with entries

$$\bar{\boldsymbol{G}}_{ij} = \mathbb{E}_{\boldsymbol{w}|\boldsymbol{X},\boldsymbol{A}}\left[\sigma(\boldsymbol{w}^\intercal \tilde{\boldsymbol{x}}_i)\sigma(\boldsymbol{w}^\intercal \tilde{\boldsymbol{x}}_j)\right]. \tag{4}$$

In particular, in the large n, p, d limit, it has been shown in (Louart et al., 2018) that the spectrum (and largest eigenvectors) of $\boldsymbol{G}$ are fully described by $\bar{\boldsymbol{G}}$. Specifically, the *resolvent* of $\boldsymbol{G}$ defined as,

$$\boldsymbol{Q}(z) = (\boldsymbol{G} + z\boldsymbol{I}_n)^{-1}, \tag{5}$$

for $z \in \mathbb{C}_+$ (with $\Im(z) > 0$), has a *deterministic equivalent*[4] $\bar{\boldsymbol{Q}}(z)$ (conditionally on $\boldsymbol{X}$ and $\boldsymbol{A}$). In other words, for all $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ of bounded spectral and Euclidean norms, respectively, with probability one, $\frac{1}{n}\operatorname{Tr}\left(\boldsymbol{M}(\boldsymbol{Q}(z) - \bar{\boldsymbol{Q}}(z))\right) \to 0$, $\boldsymbol{u}^\intercal(\boldsymbol{Q}(z) - \bar{\boldsymbol{Q}}(z))\boldsymbol{v} \to 0$, which we will simply express using the notation $\boldsymbol{Q}(z) \leftrightarrow \bar{\boldsymbol{Q}}(z)$.

A large dimensional growth rate assumption provides the existence of $\bar{\boldsymbol{Q}}(z)$.

Assumption 3 (Growth rate)**.** *As $n \to \infty$, **1.** $p/n \to c \in (0, \infty)$ and $d/n \to r \in (0, \infty)$; **2.** $\limsup_n \|\tilde{\boldsymbol{X}}\| < \infty$[5] and $|\mathcal{C}_a|/n \to c_a \in (0, 1)$; **3.** σ is λ_σ-Lipschitz continuous with $\lambda_\sigma > 0$ constant.*

[4]Such a deterministic equivalent is a standard object within random matrix theory (Hachem et al., 2007) since it allows us to characterise the behaviour of the eigenvalues of $\boldsymbol{G}$ as well as its largest (often informative) eigenvectors. Specifically, the *spectral measure* $\mu_n = \frac{1}{n}\sum_{i=1}^n \delta_{\lambda_i(\boldsymbol{G})}$ of $\boldsymbol{G}$ (where $\lambda_i(\boldsymbol{G})$ denotes the i^{th} eigenvalue of $\boldsymbol{G}$) is related to $\boldsymbol{Q}(z)$ through the *Stieltjes transform* $q_n(z) = \int (t-z)^{-1}\mu_n(dt) = \frac{1}{n}\operatorname{Tr}(\boldsymbol{Q}(-z))$. While the eigenvector $\hat{\boldsymbol{u}}_i \in \mathbb{R}^n$ corresponding to eigenvalue $\lambda_i(\boldsymbol{G})$ is related to $\boldsymbol{Q}(z)$ through the Cauchy-integral $\hat{\boldsymbol{u}}_i\hat{\boldsymbol{u}}_i^\intercal = \frac{-1}{2\pi i}\oint_{\Gamma_i} \boldsymbol{Q}(-z)dz$ where Γ_i is a positively oriented complex contour surrounding $\lambda_i(\boldsymbol{G})$.

[5]This assumption holds if additional assumptions on the node feature mean vector $\boldsymbol{\mu}$ and the graph parameters C_{ab}, which shall be provided Assumption 5, are placed.

Under Assumption 3, we have from (Louart et al., 2018)

$$\boldsymbol{Q}(z) \leftrightarrow \bar{\boldsymbol{Q}}(z) = \left(\frac{\bar{\boldsymbol{G}}}{1+\delta_g(z)} + z\boldsymbol{I}_n\right)^{-1}, \tag{6}$$

where $\delta_g(z)$ is the unique positive solution to the fixed point equation $\delta_g(z) = \frac{1}{n}\operatorname{Tr}(\bar{\boldsymbol{G}}\bar{\boldsymbol{Q}}(z))$.

From (6), to describe the behaviour of $\boldsymbol{G}$ one needs to address the non-linearity σ in the matrix $\bar{\boldsymbol{G}}$, this is achieved by approximating $\bar{\boldsymbol{G}}$ by a more tractable form in the large n limit. An additional regularity condition on σ is needed which we formulate now.

Assumption 4 (Regularity of σ)**.** *Suppose that σ is twice differentiable with $\limsup_{n, x\in\mathbb{R}} |\sigma''(x)| < \infty$. Furthermore, for $\xi \sim \mathcal{N}(0,1)$ suppose $\mathbb{E}[\sigma(\xi)] = 0$ and $\mathbb{E}[\sigma^2(\xi)] = 1$.*

Denote the quantity $b_\sigma = \mathbb{E}[\sigma'(\xi)]$. Under Assumptions 3-4, from (Fan and Wang, 2020, Lemma F.1), the average Gram matrix $\bar{\boldsymbol{G}}$ can be approximated by the $n \times n$ matrix $\tilde{\boldsymbol{G}} = b_\sigma^2\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal + (1 - b_\sigma^2)\boldsymbol{I}_n$, since almost surely as $n \to \infty$

$$\frac{1}{n}\|\bar{\boldsymbol{G}} - \tilde{\boldsymbol{G}}\|_F^2 \to 0. \tag{7}$$

This approximation ensures in particular that $\bar{\boldsymbol{G}}$ and $\tilde{\boldsymbol{G}}$ share the same spectrum.

Remark 2. *The approximation of $\bar{\boldsymbol{G}}$ by $\tilde{\boldsymbol{G}}$ in (7) is valid when the matrix $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ is of bounded spectral norm. This will be ensured in Assumption 5 where additional assumptions are placed on our model parameters $\boldsymbol{\mu}$ and C_{ab}. Furthermore, since the node features $\boldsymbol{x}_i$ follow a Gaussian mixture model (as per Assumption 1), if $\tilde{\boldsymbol{A}}$ has a bounded spectral norm, then the matrix $\tilde{\boldsymbol{X}}$ falls under the setting of (Fan and Wang, 2020) in which the relation in (7) holds.*

Since the behaviour of the average Gram matrix $\bar{\boldsymbol{G}}$ reduces to the analysis of the spectral behaviour of the matrix $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ as per the approximation in (7), we will analyse $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ for the remainder of Section 3.

3.2 Spectral Behaviour of $\tilde{X}\tilde{X}^\intercal$

We first need further controls on the quantities $\boldsymbol{\mu}$ and C_{ab} as we describe in the following assumption.

Assumption 5. *As $n \to \infty$, **1.** $\limsup_n \|\boldsymbol{\mu}\| < \infty$; **2.** $C_{aa} = 1 + \frac{\eta_a}{\sqrt{n}}$ for $a \in \{1, 2\}$ and $C_{ab} = 1$ for $a \neq b \in \{1, 2\}$, where $\eta_a = (-1)^a\eta$ and $\limsup_n \eta < \infty$.*

Remark 3. *Assumption 5.2 defines a dense graph such that the clustering with spectral methods is not asymptotically trivial. Real-World graphs are usually sparse and fall within our theoretical analysis by considering the entry-wise multiplication of the adjacency*

matrix $\boldsymbol{A}$ by a random binary mask as is done by Zarrouk et al. (2020). Furthermore without loss of generality, we have specified $\eta_a = (-1)^a \eta$ for clarity of exposition of our theoretical results (Theorem 1), which can be generalised to different choices of the inter-class similarities (choices of η_a).

Our main result (Theorem 1) provides a deterministic equivalent for the resolvent of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ defined as

$$\boldsymbol{Q}_{\tilde{\boldsymbol{X}}}(z) = \left(\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal + z\boldsymbol{I}_n\right)^{-1}. \tag{8}$$

Theorem 1. *Define the quantities $\gamma_f = \|\boldsymbol{\mu}\|^2$, $\gamma_g = q^2\eta$, $\nu = q^2(1-q^2)$ and the matrices $\boldsymbol{U} = \begin{bmatrix}\bar{\boldsymbol{y}} & \boldsymbol{\phi}\end{bmatrix} \in \mathbb{R}^{n\times 2}$,*

$$\boldsymbol{B} = \begin{bmatrix} \gamma_g^2(\frac{\gamma_f}{c}+1) & \gamma_g(\frac{\gamma_f}{c}+1) \\ \gamma_g(\frac{\gamma_f}{c}+1) & \frac{\gamma_f}{c} \end{bmatrix}, \quad \boldsymbol{T} = \begin{bmatrix} 1 & 0 \\ 0 & \nu \end{bmatrix},$$

where $\bar{\boldsymbol{y}} = \frac{\boldsymbol{y}}{\sqrt{n}}$ (with $\boldsymbol{y} \in \{-1,1\}^n$ the vector of labels) and $\boldsymbol{\phi} = \frac{1}{\sqrt{n}}\boldsymbol{N}\bar{\boldsymbol{y}}$ with $\boldsymbol{N} \in \mathbb{R}^{n\times n}$ a random matrix having random i.i.d. entries with zero mean and variance ν. Under Assumptions 1, 2, 3 and 5, the resolvent $\boldsymbol{Q}_{\tilde{\boldsymbol{X}}}(z)$ has a deterministic[6] equivalent defined as

$$\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z) = \zeta\cdot(1+\delta_1)\left(\boldsymbol{I}_n - \zeta\boldsymbol{U}\left[\boldsymbol{B}^{-1} + \zeta\boldsymbol{T}\right]^{-1}\boldsymbol{U}^\intercal\right),$$

where $\zeta = \frac{1+\delta_2}{\nu + z(1+\delta_1)(1+\delta_2)}$ and (δ_1, δ_2) is the unique couple solution of the fixed point equations system

$$\delta_1 = \frac{1}{c}\frac{\nu(1+\delta_1)}{\nu + z(1+\delta_1)(1+\delta_2)},\ \delta_2 = \frac{\nu(1+\delta_2)}{\nu + z(1+\delta_1)(1+\delta_2)}.$$

Sketch of proof. The proof starts by determining a random equivalent of the adjacency matrix $\boldsymbol{A}$. Since A_{ij} is Bernoulli distributed (see Assumption 2) with parameter $q^2(1+(-1)^{k_i}\delta_{k_i=k_j}\eta/\sqrt{n})$ with $k_i \in \{1,2\}$ the class of node i, we may write $A_{ij} = q^2 + q^2(-1)^{k_i}\delta_{k_i=k_j}\eta/\sqrt{n} + N_{ij}$ where N_{ij} is a zero mean random variable with variance $\nu + \mathcal{O}(n^{-\frac{1}{2}})$. Hence, $\|\tilde{\boldsymbol{A}} - (q^2\eta\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \frac{1}{\sqrt{n}}\boldsymbol{N})\| \to 0$ as $n\to\infty$. Finally, exploiting standard random matrix theory tools from (Hachem et al., 2007; Louart and Couillet, 2018) provides the deterministic equivalent $\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z)$. □

In essence, Theorem 1 shows that the deterministic equivalent $\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z)$ is composed of two main terms: a diagonal matrix $\zeta\cdot(1+\delta_1)\boldsymbol{I}_n$, which describes the behaviour of the noise in the data model (both adjacency and node features), and an informative rank-2 matrix $\boldsymbol{U}\left[\boldsymbol{B}^{-1} + \zeta\boldsymbol{T}\right]^{-1}\boldsymbol{U}^\intercal$ which correlates with the vector of labels $\bar{\boldsymbol{y}}$ if the adjacency matrix and/or the node features are informative, i.e., values γ_g and γ_f, respectively, are sufficiently large. Figure 1(a) and (b) depict a histogram of the eigenvalues of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ which converges to the limiting distribution described by Theorem 1, as well as its dominant eigenvector which correlates with $\bar{\boldsymbol{y}}$. Importantly, our analysis allows us to conclude that when the graph structure is completely noisy (i.e., $\eta = 0$), the dominant eigenvector of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ is no longer aligned with $\bar{\boldsymbol{y}}$ even if the node features are informative (i.e., γ_f large) as will be clarified in Corollary 2.

Corollary 2 (Case $\eta = 0$)**.** *Recall the notation and Assumptions of Theorem 1, for $\eta = 0$ (i.e., a non-informative graph structure), $\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z)$ takes the form*

$$\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z) = \zeta\cdot(1+\delta_1)\left(\boldsymbol{I}_n - \frac{\zeta^2\gamma_f}{c + \zeta\nu\gamma_f}\boldsymbol{\phi}\boldsymbol{\phi}^\intercal\right). \tag{9}$$

And, for $\hat{\boldsymbol{y}}$ the eigenvector of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ corresponding to its largest eigenvalue, $|\bar{\boldsymbol{y}}^\intercal\hat{\boldsymbol{y}}|^2 \to_{n\to\infty} 0$.

Sketch of proof. Expression (9) follows from Theorem 1 by simply taking the limit as $\eta \to 0$. The second part of the Corollary is obtained by computing $|\bar{\boldsymbol{y}}^\intercal\hat{\boldsymbol{y}}|^2 = \frac{-1}{2i\pi}\oint_\Gamma \bar{\boldsymbol{y}}^\intercal\boldsymbol{Q}_{\tilde{\boldsymbol{X}}}(-z)\bar{\boldsymbol{y}}dz$ where Γ is a small positively oriented complex contour surrounding the largest eigenvalue of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$. Hence, using $\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z)$ as a proxy allows us to state $|\bar{\boldsymbol{y}}^\intercal\hat{\boldsymbol{y}}|^2 + \frac{1}{2i\pi}\oint_\Gamma \bar{\boldsymbol{y}}^\intercal\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(-z)\bar{\boldsymbol{y}}dz \to 0$ almost surely as $n\to\infty$. The final result is obtained by showing that $\bar{\boldsymbol{y}}^\intercal\boldsymbol{\phi}\boldsymbol{\phi}^\intercal\bar{\boldsymbol{y}}$ concentrates around its expectation with $\mathbb{E}[\bar{\boldsymbol{y}}^\intercal\boldsymbol{\phi}\boldsymbol{\phi}^\intercal\bar{\boldsymbol{y}}] = \frac{1}{n}\mathrm{Var}[\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}] = \frac{\nu}{n} \to 0$ and by evaluating $\frac{1}{2i\pi}\oint_\Gamma \zeta(-z)(1+\delta_1(-z))dz = 0$. □

The main conclusion from Corollary 2 is that when the graph structure is completely random (when $\eta = 0$, $\|\tilde{\boldsymbol{A}} - \frac{1}{\sqrt{n}}\boldsymbol{N}\| \to 0$), the largest eigenvector of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ (which is intuitively supposed to be informative) does not correlate with the labels vector $\bar{\boldsymbol{y}}$ independently of the information contained in the node features. To overcome this issue, we propose in the now following Section 3.3 to utilise node feature kernels to ensure the preservation of node feature information.

3.3 Message Passing through Node Feature Kernels

As discussed in Section 3.2, when the graph structure (in the extreme case) is completely random, the *random* GCN model fails to extract information from the node features. To make the message passing informative and thereby to robustify the GCN, we propose to consider the operator $\tilde{\boldsymbol{A}} + \bar{\boldsymbol{K}}$ instead of $\tilde{\boldsymbol{A}}$, where $\bar{\boldsymbol{K}}$ is a kernel matrix computed on the node features $\boldsymbol{X}$.

[6] The matrix $\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z)$ is not deterministic since it depends on the random vector $\boldsymbol{\phi}$. However, since we are interested in evaluating quantities of the forms $\frac{1}{n}\mathrm{Tr}(\boldsymbol{M}\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z))$ or $\boldsymbol{u}^\intercal\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z)\boldsymbol{v}$ for $\boldsymbol{M}$, $\boldsymbol{u}$ and $\boldsymbol{v}$ independent of $\boldsymbol{\phi}$, $\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z)$ has a deterministic behaviour asymptotically as $n\to\infty$.

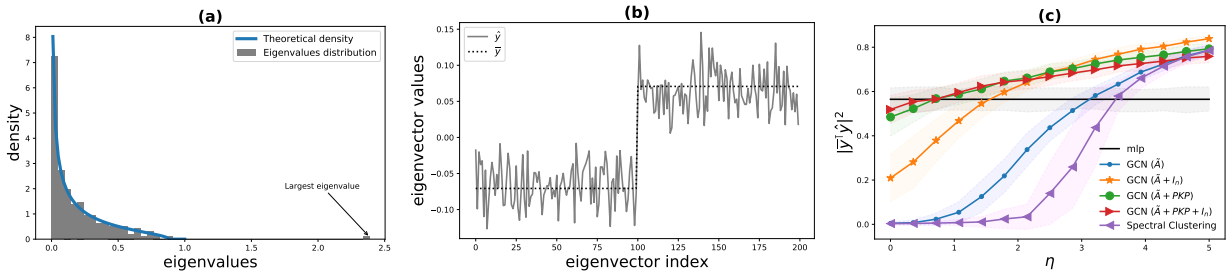

Figure 1: **(a)** Eigenvalues distribution of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ versus the theoretical density as per Theorem 1 (the theoretical density is obtained as $f(x) = \frac{1}{\pi}\lim_{\epsilon\to 0}\Im[q(x+i\epsilon)]$ where $q(z) = \frac{1}{n}\operatorname{Tr}(\bar{\boldsymbol{Q}}_{\tilde{\boldsymbol{X}}}(z))$). **(b)** Eigenvector of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ corresponding to its largest eigenvalue which correlates with $\bar{\boldsymbol{y}}$. The parameters are: $p = 1000$, $n = 200$, $q = 0.5$, $\eta = 4$ and $\boldsymbol{\mu} = [2, \mathbf{0}_{p-1}]^\intercal$. **(c)** Alignment between the largest eigenvector of $\tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^\intercal$ and the labels vector $\bar{\boldsymbol{y}}$ for different added node feature kernel message passing strategies in terms of η. The different parameters are: $p = 500$, $n = 250$, $q = 0.4$, $\boldsymbol{\mu} = [1.7, \mathbf{0}_{p-1}]^\intercal$ and the kernel matrix has entries $K_{ij} = \boldsymbol{x}_i^\intercal \boldsymbol{x}_j$, mean and std computed over 100 runs. ***The GCN with message passing operator $\tilde{\boldsymbol{A}} + \boldsymbol{PKP}$ outperforms other models when the graph structure is noisy (i.e., low values of η).***

Indeed, let $\boldsymbol{K}$ be a matrix with entries $K_{ij} = \kappa(\boldsymbol{x}_i^\intercal \boldsymbol{x}_j)$ for some smooth function $\kappa : \mathbb{R} \to \mathbb{R}$. Relying on (El Karoui et al., 2010), the kernel matrix $\boldsymbol{K}$ can be approximated in spectral norm asymptotically as $n \to \infty$ by

$$\tilde{\boldsymbol{K}} = \kappa(0)\mathbf{1}_n\mathbf{1}_n^\intercal + \kappa'(0)\left(\frac{\gamma_f}{c}\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \boldsymbol{Z}\boldsymbol{Z}^\intercal\right) + \boldsymbol{\Delta}, \quad (10)$$

where $\boldsymbol{\Delta} = \frac{\kappa''(0)}{2p}\mathbf{1}_n\mathbf{1}_n^\intercal + (\kappa(1) - \kappa(0) - \frac{\gamma_f}{c}\kappa'(0))\boldsymbol{I}_n$. Hence, considering the matrix $\tilde{\boldsymbol{A}} + \boldsymbol{PKP}$, where $\boldsymbol{P} = \boldsymbol{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\intercal$ (the centring matrix), maintains the informative nature of the message passing step (through the term $\frac{\gamma_f}{c}\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal$) even in the case where the operator $\tilde{\boldsymbol{A}}$ is not informative. Intuitively, the addition of the node feature kernel can be interpreted as considering both the originally recorded graph and a node feature similarity graph in the message passing architecture. This addition gives GNNs the necessary expressive ability to preserve information present in the node features, which is lost in the case of uninformative or noisy graph structures.

Figure 1(c) shows the performance of different message passing strategies (compared to random MLP and spectral clustering; involving only node features or adjacency matrix, respectively) which confirms the effectiveness of introducing a node feature kernel in the regime where the graph similarity is noisy (i.e., low values of η), a property which is also validated for practical GCNs as we will discuss in Section 4.

4 EXPERIMENTS

In order to validate our theoretical findings in real-world scenarios, we experiment on the node classification task using GCNs on perturbed data. In Section 4.1, we begin by justifying the use of the *random* GCN in the theoretical analysis. Then, we discuss results from experiments on both synthetic and real-world graphs involving a perturbation scheme on their edges. We show that the observed phenomena extend to deeper GCN architectures, and cases with node feature perturbations. Finally, in the case of deeper GCN models under structural perturbation, we demonstrate that our proposed method is comparable with state-of-the-art GNN models also placing a particular emphasis on the node features and can be further improved when combined with other techniques.

We work on synthetic SBM graphs aligned with Assumption 5, with the intra-community link probability being $q^2(1 + \frac{\eta}{\sqrt{n}})$ and the inter-community link probability being q^2. We vary the parameter η to generate SBM graphs with different types of community structure and keep other parameters fixed as $q = 0.5$, $p = 2500$, $n = 1600$, $\mu = (2, 0, \ldots, 0)$.

We furthermore work with six real-world datasets which often serve as node classification benchmarks. These are the three well-studied citation networks of Cora, CiteSeer and PubMed (Sen et al., 2008), an extended version of Cora (Bojchevski and Günnemann, 2018), called CoraFull, an Amazon co-purchase graph of Photo and a Co-author network from the authors of Computer Science (CS) (Shchur et al., 2018). We follow the semi-supervised node classification setting proposed by Yang et al. (2016), i.e., we use their train/valid/test split for Cora/CiteSeer/PubMed, and we randomly sample 20 nodes from each class as training set, 500 nodes as validation set and another 1000 nodes as test set for CoraFull/Photo/CS. Each experiment is repeated 10 times. Implementation details and a summary of dataset statistics can be found in the Supplementary Material Section D.

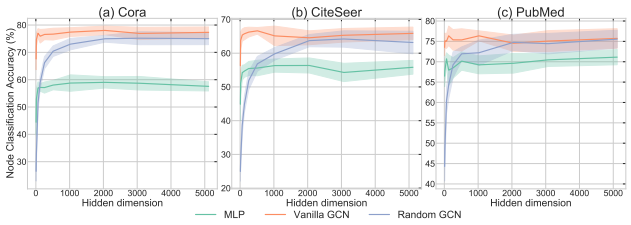

Figure 2: Performance change over the embedding dimension with different models: *random* GCN, vanilla GCN and MLP.

Perturbation Scheme The studied perturbation scheme involves both *edge-deletion* noise, where a certain amount of existing edges are randomly sampled and removed from original graph, and *edge-insertion* noise, where we add a certain amount of connections sampled from the non-existing edges in the original graph. We consider scenarios where edges are removed or added or both. The ratio of edges changed, i.e., the perturbation ratio, is denoted by α for *edge-deletion* and β for *edge-insertion*. A node feature kernel matrix is added to study its impact in practice, as shown in the following equation,

$$\boldsymbol{X}^{(i+1)} = \sigma\left((\epsilon\hat{\boldsymbol{A}} + (1-\epsilon)\mathcal{N}(\boldsymbol{K}))\boldsymbol{X}^{(i)}\boldsymbol{W}\right), \quad (11)$$

where $\hat{\boldsymbol{A}}$ is the GCN message passing operator of the perturbed graph, $\mathcal{N}(\boldsymbol{K}) = \boldsymbol{D}_{\boldsymbol{K}}^{-1/2}\boldsymbol{K}\boldsymbol{D}_{\boldsymbol{K}}^{-1/2}$ is the normalised kernel matrix built from node features ($\boldsymbol{D}_{\boldsymbol{K}} = \text{diag}(\boldsymbol{K}\boldsymbol{1}_n)$). We are degree normalising the kernel to match the graph representation of the GCN message passing operator.

Node Feature Kernels As stated in Section 3.3, we use the kernel to introduce information from the node features to the message passing structure. The choices of qualified smooth kernel functions are many. In our experiment, we perform a proof of concept using the simple linear kernel, defined as the inner product between node features $K_{ij} = \boldsymbol{x}_i^\intercal \boldsymbol{x}_j$.

Kernel Sparsification Using the full kernel matrix, where the kernel value is recorded between every pair of nodes, is both computationally costly and may incorporate redundant information. Therefore, we adopt a sparsification method using the adjacency matrix of the graph, with which (11) can be rewritten as,

$$\boldsymbol{X}^{(i+1)} = \sigma\left((\epsilon\hat{\boldsymbol{A}} + (1-\epsilon)\mathcal{N}(\boldsymbol{K}\circ\hat{\boldsymbol{A}}))\boldsymbol{X}^{(i)}\boldsymbol{W}\right), \quad (12)$$

where $\circ$ denotes Hadamard product. A consequence of this sparsification method is that the added computational cost stemming from the consideration of the node feature kernel is linear in the number of edges in the graph $|E|$, where E denotes the graph's edge set, and the node feature dimension p, i.e., of order $\mathcal{O}(|E|p)$. Our initial experiments sparsifying the kernel matrix by using a threshold below which all entries are set to zero resulted in worse performance and introduced the threshold as an extra hyperparameter. Therefore, we chose to only pursue the sparsification scheme in (12). This preliminary observation could be a result of the particular node features that are recorded in our datasets.

4.1 Experiment Analysis

Asymptotic Analysis of Random GCN To validate the practical applicability of the theoretical analysis in Section 3, we study the asymptotic behaviour of the *random* GCN, the vanilla GCN and a MLP baseline when the hidden dimension of node features grows. In Figure 2, we observe that with increasing hidden dimension, the performance of both the vanilla GCN and MLP remains stable, while the performance of *random* GCN converges to vanilla GCN's accuracy. Between hidden dimensions of 2000 and 3000 the performance of *random* GCN starts to match that of vanilla GCN. Hence, we have given an empirical indication of the conditions under which our theoretical model, the *random* GCN, and the vanilla GCN are equivalent.

Robustness to Structural Noise: Synthetic SBM We first test the performance of the proposed model on synthetic SBM graphs. Three types of SBM graph are considered, which are distinguished by the parameter η. $\eta = 0$, $\eta = 4$ and $\eta = -4$ correspond to the cases where the synthetic graph has no, a homophilic and a heterophilic community structure, respectively. We experiment on perturbation scenarios with different *egde-deletion* and *edge-insertion* ratios. For each scenario, we record the performance of the vanilla GCN as well as its performance after adding the node-feature kernel, denoted by an appendage "-k"[7] in Table 1.

When a graph has no community structure, structural perturbation has little impact and adding node-feature information boosts the performance. When the graphs are homophilic and with a clear community structure, the impact from graph-structural noise become more visible. Adding a node-feature kernel significantly improves the model's robustness against *edge-deletion* and *edge-insertion* noise and their mix. The same conclusion can be drawn on heterophilic graphs perturbed by *edge-insertion* noise.

[7] If not specified, the weight coefficient ϵ of the added kernel in graph propagation is set to 0.5.

Table 1: Performance of GCN with node-feature kernel under perturbation on synthetic SBM graphs. The best results are set to **bold** if their range of one standard deviation does not overlap with the standard deviation of their counterpart.

		SBM($q = 0.5$, $\eta = 0$)		SBM($q = 0.5$, $\eta = 4$)		SBM($q = 0.5$, $\eta = -4$)	
	(α, β)	GCN	GCN-k	GCN	GCN-k	GCN	GCN-k
	(0.0, 0.0)	50.53 ± 0.49	$\mathbf{66.36 \pm 0.81}$	$\mathbf{64.42 \pm 0.43}$	62.26 ± 1.04	$\mathbf{63.20 \pm 0.94}$	61.03 ± 1.08
Deletion	(0.2, 0.0)	51.03 ± 0.56	$\mathbf{65.44 \pm 1.07}$	58.63 ± 0.68	$\mathbf{71.57 \pm 1.42}$	$\mathbf{60.89 \pm 0.83}$	54.91 ± 1.00
	(0.5, 0.0)	49.29 ± 0.59	$\mathbf{64.14 \pm 1.01}$	60.76 ± 1.29	$\mathbf{68.80 \pm 2.04}$	58.41 ± 1.11	59.51 ± 2.47
Insertion	(0.0, 0.5)	50.57 ± 0.75	$\mathbf{68.57 \pm 1.25}$	60.49 ± 0.40	$\mathbf{68.20 \pm 1.38}$	58.82 ± 1.16	$\mathbf{63.54 \pm 0.97}$
	(0.0, 1.0)	49.19 ± 0.47	$\mathbf{59.31 \pm 0.58}$	53.67 ± 1.11	$\mathbf{66.57 \pm 1.73}$	54.87 ± 0.53	$\mathbf{60.84 \pm 0.75}$
Delet.+Insert.	(0.5, 0.5)	49.26 ± 0.59	$\mathbf{68.84 \pm 0.86}$	50.50 ± 0.37	$\mathbf{63.36 \pm 1.67}$	50.94 ± 0.86	$\mathbf{63.02 \pm 0.91}$
	(0.5, 1.0)	49.84 ± 0.69	$\mathbf{65.49 \pm 1.22}$	48.34 ± 0.22	$\mathbf{60.16 \pm 1.21}$	49.23 ± 0.45	$\mathbf{59.64 \pm 1.33}$

Table 2: Performance of GCN with and without node-feature kernel under perturbation on six real-world datasets. The format follows Table 1.

		Cora		CiteSeer		PubMed	
	(α, β)	GCN	GCN-k	GCN	GCN-k	GCN	GCN-k
	(0.0, 0.0)	$\mathbf{79.37 \pm 0.65}$	76.94 ± 0.35	67.45 ± 0.82	68.08 ± 0.91	76.04 ± 0.67	74.68 ± 0.76
Deletion	(0.2, 0.0)	76.15 ± 0.81	74.83 ± 1.24	66.79 ± 0.57	66.94 ± 0.82	$\mathbf{75.82 \pm 0.99}$	74.28 ± 0.39
	(0.5, 0.0)	72.49 ± 0.50	71.18 ± 1.00	63.53 ± 0.75	64.84 ± 1.14	73.95 ± 0.64	73.25 ± 0.75
Insertion	(0.0, 0.5)	68.57 ± 0.73	$\mathbf{73.10 \pm 1.10}$	59.85 ± 0.89	$\mathbf{66.11 \pm 1.34}$	64.18 ± 0.67	$\mathbf{72.38 \pm 0.79}$
	(0.0, 1.0)	64.14 ± 1.02	$\mathbf{73.36 \pm 0.98}$	55.39 ± 0.93	$\mathbf{64.94 \pm 0.77}$	60.56 ± 0.80	$\mathbf{71.31 \pm 0.51}$
Delet.+Insert.	(0.5, 0.5)	54.98 ± 1.13	$\mathbf{66.46 \pm 1.03}$	52.84 ± 0.68	$\mathbf{59.03 \pm 1.04}$	62.62 ± 0.72	$\mathbf{70.32 \pm 0.82}$
	(0.5, 1.0)	48.09 ± 0.88	$\mathbf{62.52 \pm 0.59}$	42.28 ± 1.07	$\mathbf{58.07 \pm 1.34}$	53.25 ± 1.57	$\mathbf{69.65 \pm 0.60}$
		CoraFull		Photo		CS	
	(α, β)	GCN	GCN-k	GCN	GCN-k	GCN	GCN-k
	(0.0, 0.0)	57.21 ± 0.84	56.88 ± 0.48	90.94 ± 0.49	90.09 ± 0.65	92.89 ± 0.41	92.63 ± 0.31
Deletion	(0.2, 0.0)	$\mathbf{57.25 \pm 0.67}$	55.56 ± 0.69	91.87 ± 0.40	92.19 ± 0.45	90.58 ± 0.48	90.89 ± 0.48
	(0.5, 0.0)	53.90 ± 0.70	54.62 ± 0.87	$\mathbf{91.10 \pm 0.40}$	87.97 ± 0.54	89.75 ± 0.60	$\mathbf{91.27 \pm 0.67}$
Insertion	(0.0, 0.5)	48.11 ± 0.89	$\mathbf{51.79 \pm 0.65}$	82.79 ± 1.43	84.18 ± 1.27	87.16 ± 0.65	$\mathbf{90.81 \pm 0.70}$
	(0.0, 1.0)	41.76 ± 1.03	$\mathbf{51.91 \pm 1.00}$	72.70 ± 6.40	$\mathbf{79.58 \pm 1.80}$	80.34 ± 0.80	$\mathbf{90.61 \pm 0.37}$
Delet.+Insert.	(0.5, 0.5)	34.70 ± 0.47	$\mathbf{46.50 \pm 0.61}$	69.70 ± 3.70	$\mathbf{74.65 \pm 2.36}$	73.75 ± 0.98	$\mathbf{87.28 \pm 0.72}$
	(0.5, 1.0)	27.50 ± 1.04	$\mathbf{43.04 \pm 0.77}$	61.13 ± 2.49	63.73 ± 5.04	66.26 ± 0.95	$\mathbf{87.51 \pm 0.58}$

Robustness to Structural Noise: Real-World datasets In this set of experiments, we study the change of model performance under the same perturbation setting on real-world datasets. Table 2 shows the results over different perturbation scenarios.

Naturally the performance decreases gradually when edges are removed or added, as we can see from each column. *Edge-insertion* noise seems to have a larger impact on the performance than the *edge-deletion* noise on these real-world datasets. But this influence can be largely compensated by adding the node feature kernel. Unlike results from synthetic SBM graphs, for *edge-deletion* noise, the addition of the kernel on real-world datasets seems to have almost no impact.

Deeper GCN architecture and Benchmark Models The previous experiments are based on a single-layer GCN model. In practice, the best-performing GCN models on these datasets often contain several message-passing layers and therefore, we want to observe whether our theoretical results can be extrapolated to the multi-layer case. We build a 4-layer GCN model and repeat our experiments on the Cora and CS datasets. The results are shown in Table 3. Results of the remaining datasets can be found in the Supplementary Material Section E. Moreover, we also consider varying the weight coefficient of the added node-feature kernel in graph propagation and compare with two models specifically proposed for deep GNN architectures: Jumping Knowledge (JK, Xu et al. (2018)) and a GCN with residual connections and an identity mapping (GCNII, Chen et al. (2020)). Although not designed to tackle the graph-structural perturbation problem that we study in this paper, the two models both utilise node feature information, which makes them reasonable baselines for our proposed method.

As we can see from Table 3, there is no fundamental change in the trends observed for the single-layer model. Adding the node-feature kernel helps robustify model performance against perturbation when edges are inserted and/or removed. The improvement is even more significant when the weight coefficient ϵ on the kernel in graph propagation is increased. Additionally, our proposed method, with a high weight on the kernel, is comparable to GCNII model and in general performs better than the JK model. However, since JK can easily be combined with our method, the node-feature kernel and JK architecture yields the best-performing model as can be seen in the rightmost column of Table 3.

Table 3: Performance of GCN with and without node-feature kernel under perturbation on deep GCN models, compared with jump knowledge and GCNII. The format follows Table 1, where in addition we underline the second best result.

		Cora					
	$(\alpha,\ \beta)$	GCN	GCN-k ($\epsilon = 0.5$)	GCN-k ($\epsilon = 0.2$)	GCN-jk	GCNII	GCN-k-jk
	(0.0, 0.0)	79.05 ± 1.36	79.67 ± 1.17	79.27 ± 1.50	80.39 ± 1.13	79.62 ± 1.24	79.74 ± 0.75
Deletion	(0.5, 0.0)	74.20 ± 1.15	72.22 ± 1.25	72.74 ± 1.23	74.97 ± 0.71	73.87 ± 0.91	74.16 ± 0.77
Insertion	(0.0, 1.0)	53.48 ± 3.49	63.20 ± 1.78	$\underline{71.24 \pm 1.04}$	65.86 ± 0.62	68.26 ± 1.15	$\mathbf{72.94 \pm 0.67}$
Delet.+Insert.	(0.5, 0.5)	47.40 ± 1.73	57.34 ± 1.53	60.77 ± 1.57	57.94 ± 0.92	$\underline{62.54 \pm 1.55}$	$\mathbf{67.11 \pm 0.93}$
		CS					
	$(\alpha,\ \beta)$	GCN	GCN-k ($\epsilon = 0.5$)	GCN-k ($\epsilon = 0.2$)	GCN-jk	GCNII	GCN-k-jk
	(0.0, 0.0)	88.44 ± 0.84	89.81 ± 0.52	91.64 ± 0.39	90.19 ± 0.59	$\mathbf{92.13 \pm 0.39}$	$\underline{91.73 \pm 0.26}$
Deletion	(0.5, 0.0)	86.68 ± 0.57	86.17 ± 1.06	88.91 ± 0.62	88.44 ± 0.69	$\underline{90.01 \pm 0.69}$	$\mathbf{91.43 \pm 0.60}$
Insertion	(0.0, 1.0)	35.84 ± 6.91	81.06 ± 3.94	88.27 ± 0.92	81.70 ± 0.63	$\underline{89.33 \pm 1.02}$	$\mathbf{91.42 \pm 0.48}$
Delet.+Insert.	(0.5, 0.5)	45.08 ± 4.82	76.27 ± 1.08	82.23 ± 1.08	73.08 ± 1.07	$\mathbf{88.66 \pm 0.70}$	$\underline{87.53 \pm 0.85}$

Table 4: Performance of GCN with and without node-feature kernel under both graph-structural and node-feature perturbation on PubMed dataset. The format follows Table 1.

		$\gamma = 0.5$		$\gamma = 1.0$		$\gamma = 5.0$	
	$(\alpha,\ \beta)$	GCN	GCN-k	GCN	GCN-k	GCN	GCN-k
	(0.0, 0.0)	72.78 ± 1.62	73.88 ± 2.04	68.40 ± 2.95	66.82 ± 1.81	42.43 ± 2.95	39.14 ± 2.6
Deletion	(0.5, 0.0)	71.61 ± 1.31	70.33 ± 2.38	63.81 ± 3.43	64.93 ± 2.14	40.41 ± 1.92	38.11 ± 1.99
Insertion	(0.0, 1.0)	58.23 ± 1.79	$\mathbf{68.14 \pm 1.67}$	55.37 ± 3.33	$\mathbf{63.35 \pm 2.30}$	38.25 ± 1.93	38.72 ± 1.93

Node feature noise We now study how node feature noise interacts with our proposed model. We perform the experiments with the same perturbation setting on the PubMed dataset. But add Gaussian noise $\mathcal{N}(\mathbf{0}, \gamma\ \mathrm{diag}(\sigma_i))$ to the node features where σ_i is the estimated variance of the i^{th} feature and γ is a scaling parameter. The results are recorded in Table 4. We observe that only when the node feature noise is five times larger than the graph structure noise ($\gamma = 5$), the addition of the node feature kernel stops to benefit the model performance. Our previous findings still hold if the node feature noise is reasonably small ($\gamma \leq 1$).

5 CONCLUSION

We have introduced the *random* GCN, which we analysed theoretically using random matrix theory. Our analysis allowed us to conclude that *perturbations of the graph structure strongly influence the performance of the GCN regardless of the information contained in the node features.* For stochastic blockmodel graphs the presence of community structure (and the degree to which this structure is present) is required (beneficial) for a message passing scheme which leads to eigenvectors of the message passing operator's Gram matrix that align with the node labels. These conclusions were confirmed in multiple experiments with the standard GCN architecture on synthetic and real-world datasets. On both synthetic and real-world data we observe *the introduction of a node feature kernel to the GCN's message passing scheme to significantly improve the performance of the GCN in the presence of a noisy graph structure.*

Acknowledgements

The work of Dr. Johannes Lutzeyer and Prof. Michalis Vazirgiannis is supported by the ANR chair AML-HELAS (ANR-CHIA-0020-01).

References

Bojchevski, A. and Günnemann, S. (2018), Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking, *in* '6th International Conference on Learning Representations (ICLR)', pp. 1–13.

Chen, M., Wei, Z., Huang, Z., Ding, B. and Li, Y. (2020), Simple and deep graph convolutional networks, *in* 'International Conference on Machine Learning', PMLR, pp. 1725–1735.

Corso, G., Cavalleri, L., Beaini, D., Liò, P. and Veličković, P. (2020), 'Principal neighbourhood aggregation for graph nets', *arXiv:2004.05718* .

Dasoulas, G., Dos Santos, L., Scaman, K. and Virmaux, A. (2020), Coloring graph neural networks for node disambiguation, *in* 'Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)', pp. 2126–2132.

El Karoui, N. et al. (2010), 'The spectrum of kernel random matrices', *Annals of statistics* pp. 1–50.

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', pp. 169–177.

Fan, Z. and Wang, Z. (2020), 'Spectra of the conjugate kernel and neural tangent kernel for linear-width neural networks', *arXiv:2005.11879* .

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

Geisler, S., Zügner, D. and Günnemann, S. (2020), 'Reliable graph neural networks via robust aggregation', *Advances in Neural Information Processing Systems* **33**, 13272–13284.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. and Dahl, G. E. (2017), Neural message passing for quantum chemistry, *in* 'Proceedings of the 34th International Conference on Machine Learning (ICML)', pp. 1263 – 1272.

Günnemann, S. (2022), Graph neural networks: Adversarial robustness, *in* L. Wu, P. Cui, J. Pei and L. Zhao, eds, 'Graph Neural Networks: Foundations, Frontiers, and Applications', Springer, chapter 8, pp. 149–176.

Hachem, W., Loubaton, P., Najim, J. et al. (2007), 'Deterministic equivalents for certain functionals of large random matrices', *The Annals of Applied Probability* pp. 875–930.

Hamilton, W. L., Ying, R. and Leskovec, J. (2017), Inductive representation learning on large graphs, *in* 'Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)', Curran Associates Inc., pp. 1025 – 1035.

Jin, M., Chang, H., Zhu, W. and Sojoudi, S. (2021), Power up! robust graph convolutional network via graph powering, *in* 'AAAI Conference on Artificial Intelligence (AAAI)'.

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

Karrer, B. and Newman, M. E. (2011), 'Stochastic blockmodels and community structure in networks', *Physical review E* p. 016107.

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

Kipf, T. N. and Welling, M. (2017), Semi-supervised classification with graph convolutional networks, *in* '5th International Conference on Learning Representations (ICLR)'.

Ledoux, M. (2005), *The concentration of measure phenomenon*, number 89, American Mathematical Soc.

Li, Q., Han, Z. and Wu, X.-M. (2018), Deeper insights into graph convolutional networks for semi-supervised learning, *in* 'Thirty-Second AAAI conference on artificial intelligence'.

Louart, C. and Couillet, R. (2018), 'Concentration of measure and large random matrices with an application to sample covariance matrices', *arXiv:1805.08295* .

Louart, C., Liao, Z., Couillet, R. et al. (2018), 'A random matrix approach to neural networks', *The Annals of Applied Probability* pp. 1190–1248.

Maron, H., Ben-Hamu, H., Serviansky, H. and Lipman, Y. (2019), Provably powerful graph networks, *in* 'Advances in Neural Information Processing Systems (NeurIPS)', pp. 2156 – 2167.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G. and Grohe, M. (2019), Weisfeiler and leman go neural: Higher-order graph neural networks, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence', pp. 4602 – 4609.

Pei, H., Wei, B., Chang, K. C.-C., Lei, Y. and Yang, B. (2020), Geom-gcn: Geometric graph convolutional networks, *in* 'International Conference on Learning Representations (ICLR)'.

Seddik, M. E. A., Louart, C., Tamaazousti, M. and Couillet, R. (2020), Random matrix theory proves that deep learning representations of GAN-data behave as Gaussian mixtures, *in* 'Proceedings of the 37th International Conference on Machine Learning (ICML)', PMLR, pp. 8573–8582.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B. and Eliassi-Rad, T. (2008), 'Collective classification in network data', *AI Magazine* p. 93.

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

Silverstein, J. W. and Bai, Z. (1995), 'On the empirical distribution of eigenvalues of a class of large dimensional random matrices', *Journal of Multivariate analysis* **54**(2), 175–192.

Sun, L., Dou, Y., Yang, C., Wang, J., Yu, P. S. and Li, B. (2020), 'Adversarial attack and defense on graph data: A survey', *arXiv:1812.10528* .

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y. (2018), Graph attention networks, *in* '6th International Conference on Learning Representations (ICLR)'.

Xu, K., Hu, W., Leskovec, J. and Jegelka, S. (2019), How powerful are graph neural networks?, *in* '7th

International Conference on Learning Representations (ICLR)'.

Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i. and Jegelka, S. (2018), Representation learning on graphs with jumping knowledge networks, *in* 'International Conference on Machine Learning', PMLR, pp. 5453–5462.

Yang, Z., Cohen, W. and Salakhudinov, R. (2016), Revisiting semi-supervised learning with graph embeddings, *in* 'Proceedings of The 33rd International Conference on Machine Learning (ICML)', PMLR, pp. 40–48.

Zarrouk, T., Couillet, R., Chatelain, F. and Le Bihan, N. (2020), Performance-complexity trade-off in large dimensional statistics, *in* '2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)', IEEE, pp. 1–6.

Zhang, X. and Zitnik, M. (2020), 'Gnnguard: Defending graph neural networks against adversarial attacks', *Advances in Neural Information Processing Systems* **33**, 9263–9275.

Zhou, Y., Zheng, H. and Huang, X. (2020), 'Graph neural networks: Taxonomy, advances and trends', *arXiv:2012.08752* .

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', pp. 1399–1407.

Zhu, J., Rossi, R. A., Rao, A., Mai, T., Lipka, N., Ahmed, N. K. and Koutra, D. (2021), 'Graph neural networks with heterophily', *Proceedings of the AAAI Conference on Artificial Intelligence* pp. 11168–11176.

Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L. and Koutra, D. (2020), 'Beyond homophily in graph neural networks: Current limitations and effective designs', *Advances in Neural Information Processing Systems* **33**, 7793–7804.

Zügner, D. and Günnemann, S. (2019), Adversarial attacks on graph neural networks via meta learning, *in* '7th International Conference on Machine Learning (ICLR)'.

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', pp. 1656–1665.

Supplementary Material: Node Feature Kernels Increase Graph Convolutional Network Robustness

A RANDOM MATRIX THEORY BACKGROUND

We begin by recalling several random matrix theory tools that are needed to establish our main results. First, we recall a fundamental result (Theorem 3) from Louart and Couillet (2018) which provides a deterministic equivalent for the *resolvent* of a *sample covariance* matrix.

Theorem 3 (Deterministic equivalent for sample covariance matrices Louart and Couillet (2018))**.** *Let $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ such that $\|\boldsymbol{M}\boldsymbol{M}^\intercal\| < \infty$[8] w.r.t. n and $\tilde{\boldsymbol{Z}} \in \mathbb{R}^{n\times p}$ some random matrix with i.i.d. entries having zero mean, unit variance and a finite forth order moment. In the limit $n \to \infty$ with $p/n \to c \in (0,\infty)$, the resolvent $\boldsymbol{Q}(z) = \left(\frac{1}{p}\boldsymbol{M}\tilde{\boldsymbol{Z}}\tilde{\boldsymbol{Z}}^\intercal\boldsymbol{M}^\intercal + z\boldsymbol{I}_n\right)^{-1}$ for $z \in \mathbb{C}$ with $\Im(z) > 0$, admits a deterministic equivalent $\bar{\boldsymbol{Q}}(z)$ defined as*

$$\bar{\boldsymbol{Q}}(z) = \left(\frac{\boldsymbol{M}\boldsymbol{M}^\intercal}{1+\delta(z)} + z\boldsymbol{I}_n\right)^{-1},$$

where $\delta(z)$ is the unique solution to the fixed point equation $\delta(z) = \frac{1}{p}\operatorname{Tr}\left(\boldsymbol{M}^\intercal\bar{\boldsymbol{Q}}(z)\boldsymbol{M}\right)$.

Proof. Denote $\boldsymbol{a}_i = \boldsymbol{M}\tilde{\boldsymbol{z}}_i$, hence

$$\boldsymbol{Q}(z) = \left(\frac{1}{p}\sum_{i=1}^n \boldsymbol{a}_i\boldsymbol{a}_i^\intercal + z\boldsymbol{I}_n\right)^{-1} = \boldsymbol{Q}_{-i} - \frac{\boldsymbol{Q}_{-i}\frac{1}{p}\boldsymbol{a}_i\boldsymbol{a}_i^\intercal\boldsymbol{Q}_{-i}}{1+\frac{1}{p}\boldsymbol{a}_i^\intercal\boldsymbol{Q}_{-i}\boldsymbol{a}_i},$$

where $\boldsymbol{Q}_{-i} = \left(\frac{1}{p}\sum_{j\neq i}^n \boldsymbol{a}_j\boldsymbol{a}_j^\intercal + z\boldsymbol{I}_n\right)^{-1}$, and we also have

$$\boldsymbol{Q}(z)\boldsymbol{a}_i = \frac{\boldsymbol{Q}_{-i}\boldsymbol{a}_i}{1+\frac{1}{p}\boldsymbol{a}_i^\intercal\boldsymbol{Q}_{-i}\boldsymbol{a}_i}.$$

A deterministic equivalent for $\boldsymbol{Q}(z)$ (which approximates $\mathbb{E}[\boldsymbol{Q}(z)]$) is of the form $\bar{\boldsymbol{Q}}(z) = (\boldsymbol{F} + z\boldsymbol{I}_n)^{-1}$ for some deterministic matrix $\boldsymbol{F}$, by computing the difference $\bar{\boldsymbol{Q}}(z) - \mathbb{E}[\boldsymbol{Q}(z)]$ using the above identities, we obtain

$$\begin{aligned}\bar{\boldsymbol{Q}}(z) - \mathbb{E}[\boldsymbol{Q}(z)] = \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\boldsymbol{Q}_{-i}\left(\frac{\boldsymbol{a}_i\boldsymbol{a}_i^\intercal}{1+\frac{1}{p}\boldsymbol{a}_i^\intercal\boldsymbol{Q}_{-i}\boldsymbol{a}_i} - \boldsymbol{F}\right)\bar{\boldsymbol{Q}}(z)\right] \\ + \frac{1}{n^2}\sum_{i=1}^n \mathbb{E}\left[\boldsymbol{Q}_{-i}(z)\boldsymbol{a}_i\boldsymbol{a}_i^\intercal\boldsymbol{Q}_{-i}(z)\boldsymbol{F}\bar{\boldsymbol{Q}}(z)\right].\end{aligned}$$

It can be shown that the matrix $\frac{1}{n^2}\sum_{i=1}^n \mathbb{E}\left[\boldsymbol{Q}_{-i}(z)\boldsymbol{a}_i\boldsymbol{a}_i^\intercal\boldsymbol{Q}_{-i}(z)\boldsymbol{F}\bar{\boldsymbol{Q}}(z)\right]$ has a vanishing operator norm as $n \to \infty$. Therefore, $\boldsymbol{F}$ can be taken as

$$\boldsymbol{F} = \frac{\mathbb{E}\left[\boldsymbol{a}_i\boldsymbol{a}_i^\intercal\right]}{1+\mathbb{E}\left[\frac{1}{p}\boldsymbol{a}_i^\intercal\boldsymbol{Q}_{-i}\boldsymbol{a}_i\right]} = \frac{\boldsymbol{M}\boldsymbol{M}^\intercal}{1+\frac{1}{p}\operatorname{Tr}(\boldsymbol{M}^\intercal\mathbb{E}\left[\boldsymbol{Q}_{-i}\right]\boldsymbol{M})}$$

Which provides the defined deterministic equivalent in Theorem 3. □

Another useful result which is known as the perturbation lemma (Silverstein and Bai, 1995) is also needed here.

[8] $\|\boldsymbol{M}\boldsymbol{M}^\intercal\|$ remains constant as n goes to infinity.

Lemma 4 (Perturbation lemma Silverstein and Bai (1995)). *Let $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}$ some symmetric matrices, $\boldsymbol{u} \in \mathbb{R}^n$, $\gamma \in \mathbb{R}$ and $z \in \mathbb{C}$ with $\Im(z) > 0$, then*

$$\left|\operatorname{Tr}\left(\boldsymbol{A}(\boldsymbol{B}+\gamma\boldsymbol{u}\boldsymbol{u}^\intercal + z\boldsymbol{I}_n)^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{A}(\boldsymbol{B}+z\boldsymbol{I}_n)^{-1}\right)\right| \leq \frac{\|\boldsymbol{A}\|}{|\Im(z)|}.$$

In particular, for $\boldsymbol{A} = \frac{1}{n}\boldsymbol{I}_n$, we have $\frac{1}{n}\operatorname{Tr}(\boldsymbol{B}+\gamma\boldsymbol{u}\boldsymbol{u}^\intercal + z\boldsymbol{I}_n)^{-1} = \frac{1}{n}\operatorname{Tr}(\boldsymbol{B}+z\boldsymbol{I}_n)^{-1} + \mathcal{O}(n^{-1})$, which shows that the spectral measure of $\boldsymbol{B}+\gamma\boldsymbol{u}\boldsymbol{u}^\intercal$ is asymptotically close to that of $\boldsymbol{B}$ in the large n limit.

Finally, we will need the Woodbury matrix identity from the following Lemma.

Lemma 5 (Woodbury identity). *Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{k\times k}$ invertible and $\boldsymbol{U} \in \mathbb{R}^{n\times k}$, then*

$$(\boldsymbol{A}+\boldsymbol{U}\boldsymbol{B}\boldsymbol{U}^\intercal)^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}\left(\boldsymbol{B}^{-1}+\boldsymbol{U}^\intercal\boldsymbol{A}^{-1}\boldsymbol{U}\right)^{-1}\boldsymbol{U}^\intercal\boldsymbol{A}^{-1}.$$

B PROOF OF THEOREM 1

The proof starts by establishing a random equivalent for the normalised Adjacency operator given by $\tilde{\boldsymbol{A}} = \frac{1}{\sqrt{n}}(\boldsymbol{A} - \boldsymbol{q}\boldsymbol{q}^\intercal)$. By Assumptions 2, 3 and 5, we have straightforwardly that, almost surely

$$\left\|\tilde{\boldsymbol{A}} - \left(q^2\eta\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \frac{1}{\sqrt{n}}\boldsymbol{N}\right)\right\| \to 0, \tag{13}$$

where $\boldsymbol{N}$ is a random matrix with i.i.d. entries of zero mean and variance $\nu = q^2(1-q^2)$. Besides, since $\mathbb{E}[\boldsymbol{X}\boldsymbol{X}^\intercal] = \frac{\|\boldsymbol{\mu}\|^2}{c}\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \boldsymbol{I}_n$, letting $\gamma_f = \|\boldsymbol{\mu}\|^2$ and $\gamma_g = q^2\eta$, we consider the equivalent multiplicative model for $\boldsymbol{Y}$ defined as

$$\boldsymbol{Y} = \left(\gamma_g\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \frac{1}{\sqrt{n}}\boldsymbol{N}\right)\left(\frac{\gamma_f}{c}\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \boldsymbol{I}_n\right)^{\frac{1}{2}}\boldsymbol{Z},$$

where $\boldsymbol{Z}$ is random matrix with i.i.d. entries of zero mean and variance $\frac{1}{p}$. Conditionally on $\boldsymbol{N}$, applying Theorem 3 for $\boldsymbol{M} = \left(\gamma_g\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \frac{1}{\sqrt{n}}\boldsymbol{N}\right)\left(\frac{\gamma_f}{c}\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \boldsymbol{I}_n\right)^{\frac{1}{2}}$, a deterministic equivalent of $\boldsymbol{Q}_{\boldsymbol{Y}}(z) = (\boldsymbol{Y}\boldsymbol{Y}^\intercal + z\boldsymbol{I}_n)^{-1}$ is given by

$$\bar{\boldsymbol{Q}}_{\boldsymbol{Y}|\boldsymbol{N}}(z) = \left(\frac{\left(\gamma_g\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \frac{1}{\sqrt{n}}\boldsymbol{N}\right)\left(\frac{\gamma_f}{c}\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \boldsymbol{I}_n\right)\left(\gamma_g\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}^\intercal + \frac{1}{\sqrt{n}}\boldsymbol{N}\right)}{1+\delta_1(z)} + z\boldsymbol{I}_n\right)^{-1} \tag{14}$$

$$= \left(\frac{\boldsymbol{U}\boldsymbol{B}\boldsymbol{U}^\intercal + \frac{1}{n}\boldsymbol{N}\boldsymbol{N}^\intercal}{1+\delta_1(z)} + z\boldsymbol{I}_n\right)^{-1}, \tag{15}$$

where

$$\boldsymbol{U} = [\bar{\boldsymbol{y}}, \boldsymbol{\phi}] \in \mathbb{R}^{n\times 2}, \quad \boldsymbol{B} = \begin{bmatrix} \gamma_2^2\left(\frac{\gamma_1}{c}+1\right) & \left(\frac{\gamma_1}{c}+1\right)\gamma_2 \\ \left(\frac{\gamma_1}{c}+1\right)\gamma_2 & \frac{\gamma_1}{c} \end{bmatrix} \tag{16}$$

with $\boldsymbol{\phi} = \frac{1}{\sqrt{n}}\boldsymbol{N}\bar{\boldsymbol{y}}$, $\bar{\boldsymbol{y}} = \boldsymbol{y}/\sqrt{n}$ and $\delta_1(z) = \frac{1}{p}\operatorname{Tr}\left(\left(\boldsymbol{U}\boldsymbol{B}\boldsymbol{U}^\intercal + \frac{1}{n}\boldsymbol{N}\boldsymbol{N}^\intercal\right)\bar{\boldsymbol{Q}}_{\boldsymbol{Y}|\boldsymbol{N}}(z)\right)$. Applying Lemma 4, $\delta_1(z)$ is simply the solution to $\delta_1(z) = \frac{1}{p}\operatorname{Tr}\left(\frac{1}{n}\boldsymbol{N}\boldsymbol{N}^\intercal\bar{\boldsymbol{Q}}_{\boldsymbol{Y}|\boldsymbol{N}}(z)\right)$. Moreover, defining the matrix $\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}|\boldsymbol{N}}(z) = \left(\frac{\frac{1}{n}\boldsymbol{N}\boldsymbol{N}^\intercal}{1+\delta_1(z)} + z\boldsymbol{I}_n\right)^{-1}$, we have by Lemma 5

$$\bar{\boldsymbol{Q}}_{\boldsymbol{Y}|\boldsymbol{N}}(z) = \bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}|\boldsymbol{N}}(z) - \bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}|\boldsymbol{N}}(z)\boldsymbol{U}\left((1+\delta_1(z))\boldsymbol{B}^{-1} + \boldsymbol{U}^\intercal\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}|\boldsymbol{N}}(z)\boldsymbol{U}\right)^{-1}\boldsymbol{U}^\intercal\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}|\boldsymbol{N}}(z). \tag{17}$$

Again by Theorem 3, a deterministic equivalent of $\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}|\boldsymbol{N}}(z)$ is given by

$$\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}}(z) = \left(\frac{\nu\boldsymbol{I}_n}{(1+\delta_1(z))(1+\delta_2(z))} + z\boldsymbol{I}_n\right)^{-1} = \frac{(1+\delta_1(z))(1+\delta_2(z))}{\nu + z(1+\delta_1(z))(1+\delta_2(z))}\boldsymbol{I}_n, \tag{18}$$

where $\delta_2(z)$ is the unique solution to $\delta_2(z) = \frac{1}{n}\operatorname{Tr}\left(\frac{\nu\boldsymbol{I}_n}{(1+\delta_1(z))}\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}}(z)\right) = \frac{\nu(1+\delta_2(z))}{\nu+z(1+\delta_1(z))(1+\delta_2(z))}$, and similarly $\delta_1(z)$ satisfies $\delta_1(z) = \frac{1}{c}\frac{\nu(1+\delta_1(z))}{\nu+z(1+\delta_1(z))(1+\delta_2(z))}$. Therefore, replacing $\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}|\boldsymbol{N}}(z)$ in (17) by its deterministic equivalent $\bar{\boldsymbol{Q}}^{\boldsymbol{B}}_{\boldsymbol{Y}}(z)$ and since $\boldsymbol{U}^\intercal\boldsymbol{U} \to \boldsymbol{T}$ almost surely, provides the final result of Theorem 3.4.

Table 5: Statistics of the 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
CORA-FULL	8710	18703	62421	67
PHOTO	745	7487	119043	8
CS	6805	18333	81894	15

C Proof of Corollary 2

Following the same procedure as in Section B, when $\eta = 0$, a deterministic equivalent for $\boldsymbol{Q}_{\boldsymbol{Y}}(z)$ takes the form

$$\bar{\boldsymbol{Q}}_{\boldsymbol{Y}}(z) = \zeta(z)(1+\delta_1(z))\left(\boldsymbol{I}_n - \frac{\zeta^2(z)\gamma_f}{c+\nu\gamma_f\zeta(z)}\boldsymbol{\phi}\boldsymbol{\phi}^\intercal\right). \tag{19}$$

By definition of the deterministic equivalent, we have almost surely

$$|\bar{\boldsymbol{y}}^\intercal\hat{\boldsymbol{y}}|^2 = \frac{-1}{2\pi i}\oint_\Gamma \bar{\boldsymbol{y}}^\intercal\boldsymbol{Q}_{\boldsymbol{Y}}(-z)\bar{\boldsymbol{y}}dz \to_{n\to\infty} \frac{-1}{2\pi i}\oint_\Gamma \bar{\boldsymbol{y}}^\intercal\bar{\boldsymbol{Q}}_{\boldsymbol{Y}}(-z)\bar{\boldsymbol{y}}dz. \tag{20}$$

Hence, we need to evaluate the Cauchy-integral $\frac{-1}{2\pi i}\oint_\Gamma \bar{\boldsymbol{y}}^\intercal\bar{\boldsymbol{Q}}_{\boldsymbol{Y}}(-z)\bar{\boldsymbol{y}}dz$. In particular, the quadratic form $\bar{\boldsymbol{y}}^\intercal\bar{\boldsymbol{Q}}_{\boldsymbol{Y}}(z)\bar{\boldsymbol{y}}$ evaluates as

$$\bar{\boldsymbol{y}}^\intercal\bar{\boldsymbol{Q}}_{\boldsymbol{Y}}(z)\bar{\boldsymbol{y}} = \zeta(z)(1+\delta_1(z))\left(1 - \frac{\zeta^2(z)\gamma_f}{c+\nu\gamma_f\zeta(z)}\bar{\boldsymbol{y}}^\intercal\boldsymbol{\phi}\boldsymbol{\phi}^\intercal\bar{\boldsymbol{y}}\right) \to_{n\to\infty} \zeta(z)(1+\delta_1(z)).$$

Indeed, since the mapping $\boldsymbol{X} \mapsto \frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{X}\bar{\boldsymbol{y}}$ is $\frac{1}{\sqrt{n}}$-Lipschitz transformation w.r.t. the Frobenius norm $\|\cdot\|_F$, then we have the concentration inequality, for all $t \geq 0$

$$\mathbb{P}\left(\left|\frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}} - \mathbb{E}\left[\frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}\right]\right| > t\right) \leq C\, e^{-(\sqrt{n}\,t/\nu)^2}, \tag{21}$$

for some constant $C \geq 0$ independent of n. In particular, since $\mathbb{E}\left[\frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}\right] = 0$, we have

$$\mathbb{P}\left(\left|\left(\frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}\right)^2 - \mathbb{E}\left[\left(\frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}\right)^2\right]\right| > t\right) \leq C\, e^{-\frac{\sqrt{n}\,t}{2\nu}}, \tag{22}$$

which shows that $\bar{\boldsymbol{y}}^\intercal\boldsymbol{\phi}\boldsymbol{\phi}^\intercal\bar{\boldsymbol{y}} = \left(\frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}\right)^2$ concentrates around its mean value, with

$$\mathbb{E}\left[\left(\frac{1}{\sqrt{n}}\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}\right)^2\right] = \frac{1}{n}\mathrm{Var}\left[\bar{\boldsymbol{y}}^\intercal\boldsymbol{N}\bar{\boldsymbol{y}}\right] = \frac{1}{n}\sum_{i,j=1}^n \bar{y}_i^2\bar{y}_j^2\mathrm{Var}[N_{ij}] = \frac{\|\bar{\boldsymbol{y}}\|^4\,\nu}{n} \to 0.$$

Therefore, $\bar{\boldsymbol{y}}^\intercal\boldsymbol{\phi}\boldsymbol{\phi}^\intercal\bar{\boldsymbol{y}} \to 0$ almost surely as $n \to \infty$. The final step consists in evaluating the integral $\frac{1}{2\pi i}\oint_\Gamma \zeta(-z)(1+\delta_1(-z))dz = 0$ since the function $z \mapsto \zeta(-z)(1+\delta_1(-z))$ does not have singularities on the contour Γ. Indeed, this integral corresponds to the only noise case from the data model (i.e., $\boldsymbol{Y} = \frac{1}{\sqrt{n}}\boldsymbol{N}\boldsymbol{Z}$).

D DATASETS AND IMPLEMENTATION DETAILS

Summary statistics of the datasets we used are shown in Table 5. As mentioned in the main paper, we experiment on citation networks of Cora/CiteSeer/Pubmed/CoraFull (Sen et al., 2008; Shchur et al., 2018), as well as Amazon co-purchase networks of Photo and Co-author network of authors from Computer Science (CS) domain (Shchur et al., 2018). For train/valid/test splits, we follow the public split on Cora/CiteSeer/PubMed and construct the

Table 6: Performance of the GCN with and without node-feature kernel under perturbation on six real-world datasets with **multiple train/valid/test splits**. The format follows Table 1 in the main paper.

		Cora		CiteSeer		PubMed	
	(α, β)	GCN	GCN-k	GCN	GCN-k	GCN	GCN-k
Deletion	(0.0, 0.0)	76.25 ± 2.32	75.48 ± 1.87	66.31 ± 2.04	66.53 ± 2.10	74.80 ± 2.07	76.93 ± 2.20
	(0.2, 0.0)	74.82 ± 1.68	73.26 ± 1.88	64.96 ± 1.57	64.27 ± 2.34	75.14 ± 2.06	74.21 ± 2.84
	(0.5, 0.0)	70.78 ± 1.53	70.80 ± 1.83	63.14 ± 1.49	62.74 ± 1.80	74.17 ± 1.75	74.11 ± 2.06
Insertion	(0.0, 0.5)	67.01 ± 2.11	$\mathbf{71.96 \pm 1.79}$	57.35 ± 2.02	$\mathbf{62.19 \pm 1.68}$	63.70 ± 2.95	$\mathbf{71.62 \pm 2.01}$
	(0.0, 1.0)	58.92 ± 2.29	$\mathbf{68.50 \pm 1.45}$	50.53 ± 2.13	$\mathbf{59.60 \pm 2.04}$	57.07 ± 1.94	$\mathbf{69.56 \pm 1.92}$
		CoraFull		Photo		CS	
	(α, β)	GCN	GCN-k	GCN	GCN-k	GCN	GCN-k
Deletion	(0.0, 0.0)	58.42 ± 2.12	57.64 ± 1.45	91.48 ± 0.94	91.05 ± 1.28	92.09 ± 0.98	92.62 ± 0.92
	(0.2, 0.0)	56.36 ± 1.70	57.18 ± 1.68	90.51 ± 1.29	91.30 ± 1.22	91.37 ± 1.26	91.79 ± 0.91
	(0.5, 0.0)	54.58 ± 1.33	53.44 ± 1.47	90.07 ± 1.63	90.16 ± 1.10	89.78 ± 1.04	91.08 ± 1.27
Insertion	(0.0, 0.5)	48.53 ± 1.58	$\mathbf{53.55 \pm 1.44}$	81.77 ± 2.31	83.52 ± 1.43	88.19 ± 0.96	$\mathbf{91.25 \pm 1.00}$
	(0.0, 1.0)	43.18 ± 1.73	$\mathbf{50.77 \pm 1.89}$	75.21 ± 4.30	79.02 ± 3.04	83.83 ± 1.60	$\mathbf{90.32 \pm 0.89}$

train/valid/test set on CoraFull/Photo/CS by randomly sampling 20 nodes from each class to form the training set and 500/1000 nodes respectively from the rest to form the validation and test set, as proposed in Yang et al. (2016).

In line with our theoretical analysis, the main GCN architecture on which we experimented in this paper is a single-layer GCN (one iteration of message passing and update) stacked with a Multi-Layer Perception (MLP). The objective of the experiments is to validate our theoretical hypotheses and experiment with the robustness of GCN models under graph structure perturbation. We also study empirically to what extent the validated hypotheses extrapolate to scenarios where deeper GCN architectures with multiple layers of graph propagation are used and/or node features are also perturbed. In comparison to the state-of-the-art models, in particular to those which also place particular emphasis on the node features, we demonstrate that our proposed method has superior or comparable performance and can be further improved when combined with other techniques.

All the experiments are performed using the Adam optimiser (Kingma and Ba, 2014) and the same set of hyper-parameters, with learning rate being 1e-2, number of epochs being 200 and hidden feature dimension being 128. We repeat each experiment 10 times and report the resulting means and standard deviations to accurately report the impact of random initialisation.

We have made our implementation publicly available online[9]. It is built upon the open source library *PyTorch Geometric* (PyG) under MIT license (Fey and Lenssen, 2019). The experiments are run on a Intel(R) Xeon(R) W-2123 processor with 64GB ram and a NVIDIA GeForce RTX 2080Ti GPU with 12GB ram.

E ADDITIONAL EXPERIMENTS

E.1 Multiple Splits

Shchur et al. (2018) argue that different train/valid/test splits of datasets may have a non-negligible impact on the performance of GNN models for the node classification task. To investigate the influence of different splits on our hypothesis, we construct train/valid/test split for each dataset following Yang et al. (2016) over 10 random seeds. As each experiment is repeated 10 times, a total of 100 results is obtained for a specific setting, i.e., specific α, β or ϵ. Similar to the results of one split, we report the mean and standard deviations of the 100 results in Table 6. Although the standard deviation increases since we introduce more variation in the multi-split setting, the general trend remains the same, as shown in Table 6. Our conclusion drawn in the main paper still holds: *perturbations of the graph structure strongly influence the performance of the GCN and adding a node feature kernel can robustify the GCN against such perturbations.*

E.2 Models beyond the GCN

We also study the behaviour of our proposed method in a more general Message-Passing Neural Network (MPNN) setting beyond the GCN. We choose three characteristic models, which are GIN (Xu et al., 2019), GraphSage (Hamilton et al., 2017) and GAT (Veličković et al., 2018) models, and observe their performance under graph

[9]https://github.com/ChangminWu/RobustGCN

Table 7: Performance of the GIN/GraphSage/GAT with and without node-feature kernel under perturbation on three citation datasets. The format follows Table 1 in the main paper.

		Cora					
	(α, β)	GIN	GIN-k	Sage	Sage-k	GAT	GAT-k
Deletion	(0.0, 0.0)	78.56 ± 1.12	78.59 ± 0.8	75.94 ± 0.25	$\mathbf{77.12 \pm 0.65}$	78.20 ± 0.54	78.55 ± 0.69
	(0.2, 0.0)	76.61 ± 0.58	76.73 ± 1.79	73.87 ± 0.92	75.29 ± 1.25	75.54 ± 0.77	$\mathbf{77.01 \pm 0.30}$
	(0.5, 0.0)	71.31 ± 1.06	70.57 ± 0.63	67.35 ± 0.92	$\mathbf{71.98 \pm 0.99}$	71.25 ± 0.89	72.37 ± 0.74
Insertion	(0.0, 0.5)	71.08 ± 1.08	72.19 ± 0.87	70.24 ± 1.01	71.42 ± 1.23	67.45 ± 1.21	$\mathbf{72.13 \pm 0.53}$
	(0.0, 1.0)	66.21 ± 1.52	67.96 ± 0.67	66.0 ± 1.23	$\mathbf{70.57 \pm 0.95}$	62.14 ± 1.46	$\mathbf{67.68 \pm 0.89}$
Delet.+Insert.	(0.5, 0.5)	56.82 ± 1.35	$\mathbf{62.41 \pm 1.07}$	61.86 ± 1.32	63.61 ± 0.94	53.82 ± 0.86	$\mathbf{61.04 \pm 1.24}$
	(0.5, 1.0)	51.20 ± 2.04	$\mathbf{59.28 \pm 0.97}$	60.56 ± 0.81	$\mathbf{64.13 \pm 0.52}$	46.97 ± 1.15	$\mathbf{52.13 \pm 1.36}$
		CiteSeer					
	(α, β)	GIN	GIN-k	Sage	Sage-k	GAT	GAT-k
Deletion	(0.0, 0.0)	66.24 ± 0.89	66.96 ± 0.86	67.74 ± 0.85	68.97 ± 0.60	68.19 ± 0.92	67.82 ± 0.87
	(0.2, 0.0)	62.24 ± 1.19	$\mathbf{66.70 \pm 1.44}$	65.97 ± 1.06	66.22 ± 0.78	66.31 ± 0.93	66.90 ± 0.94
	(0.5, 0.0)	62.01 ± 1.01	62.30 ± 1.24	63.24 ± 0.59	64.97 ± 1.03	63.61 ± 1.03	65.43 ± 0.83
Insertion	(0.0, 0.5)	58.90 ± 1.31	$\mathbf{64.75 \pm 1.49}$	64.71 ± 0.94	66.21 ± 0.72	59.44 ± 1.34	$\mathbf{62.61 \pm 1.30}$
	(0.0, 1.0)	54.61 ± 1.28	$\mathbf{59.25 \pm 0.99}$	62.08 ± 0.99	63.04 ± 1.07	50.34 ± 0.99	$\mathbf{58.46 \pm 0.98}$
Delet.+Insert.	(0.5, 0.5)	50.46 ± 2.02	$\mathbf{57.9 \pm 1.51}$	57.88 ± 1.35	$\mathbf{63.56 \pm 0.67}$	50.75 ± 1.21	$\mathbf{55.81 \pm 1.04}$
	(0.5, 1.0)	43.98 ± 1.55	$\mathbf{48.4 \pm 1.47}$	58.51 ± 1.13	59.52 ± 1.07	43.56 ± 1.19	$\mathbf{50.17 \pm 1.35}$
		PubMed					
	(α, β)	GIN	GIN-k	Sage	Sage-k	GAT	GAT-k
Deletion	(0.0, 0.0)	77.13 ± 0.36	77.02 ± 0.58	76.44 ± 0.59	77.09 ± 0.64	76.08 ± 0.81	76.63 ± 0.47
	(0.2, 0.0)	75.96 ± 0.47	75.45 ± 0.44	75.38 ± 0.34	75.83 ± 0.83	75.93 ± 0.46	76.17 ± 0.55
	(0.5, 0.0)	74.29 ± 0.79	$\mathbf{76.09 \pm 0.45}$	72.10 ± 1.60	$\mathbf{76.24 \pm 0.49}$	73.46 ± 0.45	$\mathbf{76.05 \pm 0.51}$
Insertion	(0.0, 0.5)	71.85 ± 0.79	73.63 ± 1.10	73.48 ± 0.48	74.18 ± 0.43	67.42 ± 0.63	$\mathbf{69.80 \pm 0.69}$
	(0.0, 1.0)	64.27 ± 1.60	$\mathbf{69.61 \pm 1.3}$	74.37 ± 0.72	74.44 ± 0.32	64.73 ± 0.98	65.38 ± 0.73
Delet.+Insert.	(0.5, 0.5)	64.65 ± 1.07	$\mathbf{68.24 \pm 0.61}$	72.59 ± 0.61	$\mathbf{74.87 \pm 0.37}$	62.66 ± 0.81	$\mathbf{66.51 \pm 0.9}$
	(0.5, 1.0)	59.12 ± 1.19	$\mathbf{63.20 \pm 1.41}$	73.77 ± 0.37	72.64 ± 0.45	58.58 ± 0.80	$\mathbf{63.60 \pm 1.27}$

structural perturbation with node feature kernel (our proposed method) on three citation datasets. The models are also implemented as a single graph-propagation layer followed by a MLP readout, as was the case for the GCN. Results are gathered in Table 7. Similar to the results of the GCN, we can observe from Table 7 that on every model and every dataset, adding a node-feature kernel in the graph propagation helps to robustify the model performance against structural noises, in particular when edges are added. This empirical evidence demonstrates the versatility of our proposed method for a general MPNN model.

E.3 Deeper GCN architecture and Benchmark Models

In this set of experiments, we observe to what extent the conclusions drawn in our theoretical analysis carry over to deeper GCN architectures, and how our proposed model performs against similar methods in the deeper architecture setting. We add three extra message passing layer to the previous single-layer model and repeat our experiments on this deeper model as well as two state-of-the-art methods, the Jumping Knowledge (JK) and the GCN with residual connections and an identity mapping (GCNII), which are specifically designed for deeper models and take also advantage of the node feature information. Part of the results have already been shown in Section 4.1 of the main paper. In Table 8 we show the results of the remaining four datasets, which agree with the trends observed in the main paper.

E.4 Node Feature Noise

We now provide further experiment results in Figure 3 studying how node feature noise interacts with our proposed model. We observe that only when the node feature noise is five times larger than the graph structure noise, it starts to overshadow the benefit of adding the node feature kernel, as we can see from the pink curves, with-kernel (dashed or dotted line) performs worse than without-kernel (solid line). Otherwise, the performance of our kernel is robust and matches the observed trend repeatedly demonstrated in the previous experiments.

Table 8: Performance of the GCN with and without node-feature kernel under perturbation on deep GCN models, compared with jump knowledge and GCNII. The format follows Table 1 in the main paper, where in addition we underline the second best result.

		CiteSeer					
	$(\alpha,\ \beta)$	GCN	GCN-k ($\epsilon = 0.5$)	GCN-k ($\epsilon = 0.2$)	GCN-jk	GCNII	GCN-k-jk
	(0.0, 0.0)	66.72 ± 1.93	66.76 ± 0.78	67.75 ± 1.39	68.70 ± 1.06	67.59 ± 0.81	69.10 ± 0.92
Deletion	(0.5, 0.0)	62.68 ± 1.39	64.52 ± 2.03	62.72 ± 1.65	$\mathbf{65.49 \pm 0.94}$	61.91 ± 1.52	63.82 ± 2.01
Insertion	(0.0, 1.0)	45.14 ± 1.89	48.39 ± 1.88	56.69 ± 1.91	53.26 ± 1.44	57.41 ± 1.67	$\mathbf{62.83 \pm 1.06}$
Delet.+Insert.	(0.5, 0.5)	39.68 ± 1.96	48.26 ± 1.91	51.10 ± 1.26	49.50 ± 0.86	53.19 ± 1.58	$\mathbf{58.12 \pm 1.02}$
		PubMed					
	$(\alpha,\ \beta)$	GCN	GCN-k ($\epsilon = 0.5$)	GCN-k ($\epsilon = 0.2$)	GCN-jk	GCNII	GCN-k-jk
	(0.0, 0.0)	76.50 ± 0.71	77.82 ± 0.86	77.80 ± 0.92	77.36 ± 0.74	78.14 ± 0.87	77.56 ± 1.62
Deletion	(0.5, 0.0)	73.92 ± 0.64	74.53 ± 0.97	$\mathbf{76.19 \pm 0.73}$	73.00 ± 0.93	75.00 ± 0.65	75.60 ± 1.09
Insertion	(0.0, 1.0)	46.98 ± 2.87	68.29 ± 7.93	72.42 ± 1.11	65.01 ± 1.70	72.86 ± 0.75	$\mathbf{73.20 \pm 0.89}$
Delet.+Insert.	(0.5, 0.5)	62.35 ± 1.01	65.19 ± 1.78	$\mathbf{70.79 \pm 1.08}$	65.07 ± 0.83	69.06 ± 0.67	66.28 ± 1.37
		CoraFull					
	$(\alpha,\ \beta)$	GCN	GCN-k ($\epsilon = 0.5$)	GCN-k ($\epsilon = 0.2$)	GCN-jk	GCNII	GCN-k-jk
	(0.0, 0.0)	49.33 ± 0.61	55.27 ± 0.97	52.38 ± 1.62	$\mathbf{58.56 \pm 1.20}$	56.83 ± 0.67	56.10 ± 0.89
Deletion	(0.5, 0.0)	46.80 ± 1.35	45.33 ± 1.37	46.71 ± 1.12	51.42 ± 1.48	51.57 ± 1.03	$\mathbf{51.76 \pm 1.32}$
Insertion	(0.0, 1.0)	3.52 ± 0.76	9.60 ± 2.62	19.59 ± 2.42	42.07 ± 0.92	48.79 ± 1.28	$\mathbf{52.00 \pm 0.87}$
Delet.+Insert.	(0.5, 0.5)	7.10 ± 1.60	12.98 ± 1.56	22.06 ± 1.27	38.74 ± 0.86	41.27 ± 1.31	$\mathbf{45.74 \pm 1.22}$
		Photo					
	$(\alpha,\ \beta)$	GCN	GCN-k ($\epsilon = 0.5$)	GCN-k ($\epsilon = 0.2$)	GCN-jk	GCNII	GCN-k-jk
	(0.0, 0.0)	87.67 ± 1.59	89.59 ± 0.63	88.64 ± 1.19	91.82 ± 0.58	92.05 ± 0.88	$\mathbf{92.42 \pm 0.58}$
Deletion	(0.5, 0.0)	88.04 ± 1.02	88.81 ± 0.33	87.23 ± 1.20	90.44 ± 1.07	87.65 ± 1.85	$\mathbf{91.45 \pm 0.52}$
Insertion	(0.0, 1.0)	27.75 ± 3.39	29.6 ± 3.22	24.05 ± 2.56	77.54 ± 3.44	$\mathbf{85.68 \pm 1.94}$	81.89 ± 3.24
Delet.+Insert.	(0.5, 0.5)	31.27 ± 4.58	27.65 ± 6.01	29.32 ± 4.17	75.38 ± 5.41	$\mathbf{87.32 \pm 1.11}$	73.57 ± 5.92

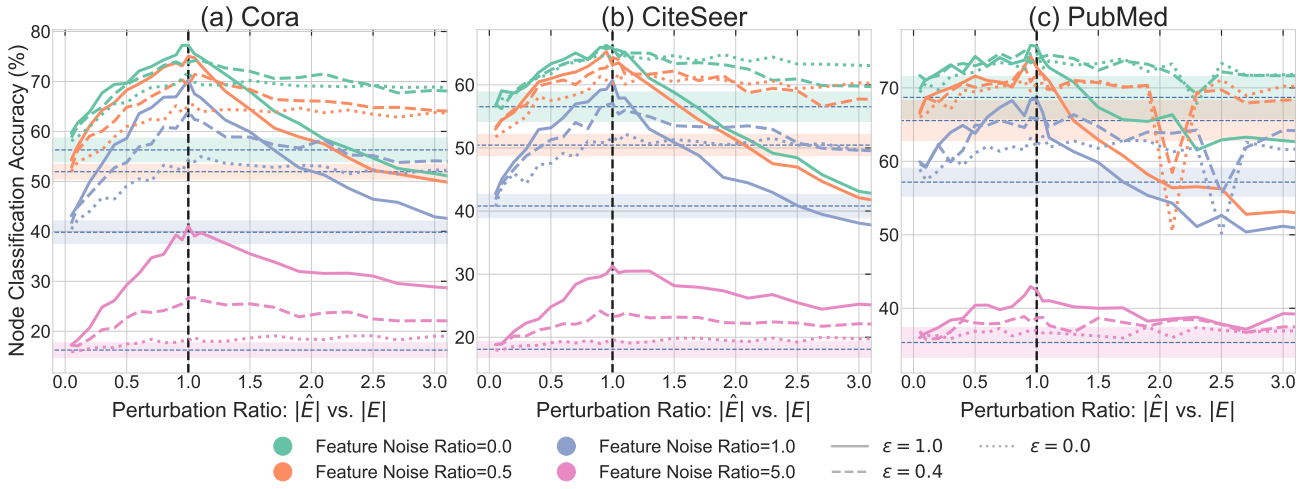

Figure 3: Experiment Results with the co-appearance of graph structural noise and node feature noise on three citation datasets. The vertical line at a rate of 1 represents the performance on the unperturbed graph. On its left is the *edge deletion* case, with rate less than 1, where the most perturbed case corresponds to rate 0; on the right is the *edge insertion* case, where the perturbations grow with the rate. Different colours represent the extent of node feature perturbation.