Generalization of Graph Neural Networks is Robust to Model Mismatch

Zhiyang Wang[1], **Juan Cerviño**[2], **Alejandro Ribeiro**[1]

[1]University of Pennsylvania
[2]Massachusetts Institute of Technology
zhiyangw@seas.upenn.edu, jcervino@mit.edu, aribeiro@seas.upenn.edu

Abstract

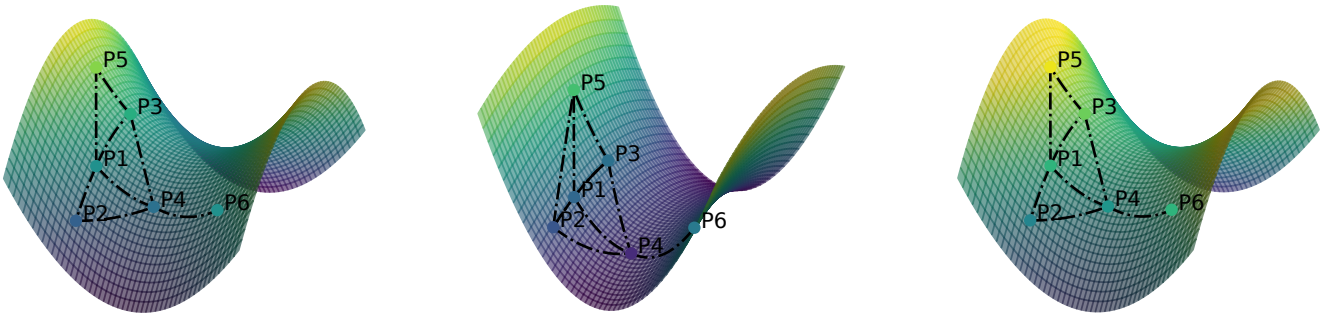
Graph neural networks (GNNs) have demonstrated their effectiveness in various tasks supported by their generalization capabilities. However, the current analysis of GNN generalization relies on the assumption that training and testing data are independent and identically distributed (i.i.d). This imposes limitations on the cases where a model mismatch exists when generating testing data. In this paper, we examine GNNs that operate on geometric graphs generated from manifold models, explicitly focusing on scenarios where there is a mismatch between manifold models generating training and testing data. Our analysis reveals the robustness of the GNN generalization in the presence of such model mismatch. This indicates that GNNs trained on graphs generated from a manifold can still generalize well to unseen nodes and graphs generated from a mismatched manifold. We attribute this mismatch to both node feature perturbations and edge perturbations within the generated graph. Our findings indicate that the generalization gap decreases as the number of nodes grows in the training graph while increasing with larger manifold dimension as well as larger mismatch. Importantly, we observe a trade-off between the generalization of GNNs and the capability to discriminate high-frequency components when facing a model mismatch. The most important practical consequence of this analysis is to shed light on the filter design of generalizable GNNs robust to model mismatch. We verify our theoretical findings with experiments on multiple real-world datasets.

Introduction

Graph Neural Networks (GNNs) (Sandryhaila and Moura 2013; Kipf and Welling 2017; Gama et al. 2019), as a deep learning model on graphs, have been unarguably one of the most recognizable architectures when processing graph-structured data. GNNs have achieved notable performances in numerous applications, such as recommendation systems (Wu et al. 2022), protein structure predictions (Yin et al. 2023), and multi-agent robotic control (Gosrich et al. 2022). These outstanding results of GNNs depend on their empirical performances when *predicting* over unseen testing data. This is evaluated in theory with *statistical generalization analysis*, which quantifies the difference between the *empirical risk* (i.e. training error) and the *statistical risk* (i.e. testing error) in deep learning theory (Kawaguchi, Kaelbling, and Bengio 2022). Recent works have focused on proving the generalization bounds of GNNs without any dependence on the underlying model responsible for generating the graph data (Scarselli, Tsoi, and Hagenbuchner 2018; Garg, Jegelka, and Jaakkola 2020; Verma and Zhang 2019). Generalization analysis on graph classification is studied in a series of works when graphs are drawn from random limit models (Ruiz, Chamon, and Ribeiro 2023; Maskey et al. 2022; Maskey, Kutyniok, and Levie 2024; Levie 2024). In (Wang, Cervino, and Ribeiro 2024), the authors study the generalization of GNNs over graphs generated from an underlying manifold on both node and graph levels. These works assume that the training and testing graphs are generated from the same underlying model. In practice, there are inevitable scenarios with generative model mismatch between testing and training graphs (Li et al. 2022). Hence, it is of crucial to demonstrate the generalization ability of GNNs remains robust to generative model mismatch. This would provide a promising assurance that GNNs can maintain outstanding generalizable performance even in noisy environments.

The model mismatch may stem from disturbances during the graph generation process from the manifold. Moreover, the underlying manifold model is prone to undergo alternations and fluctuations in practical situations. While we take into account the underlying model mismatches, they can be interpreted as perturbations within the generated graph domain. Figure 1a displays the original manifold and a graph derived from it. Figure 1b illustrates a mismatched manifold, which leads to edge perturbations in the generated graph. Figure 1c shows the interpretation of perturbed manifold function values, resulting in node feature perturbations in the generated graph.

We prove that the generalization of GNNs is robust to model mismatches based on the convergence of GNNs on generated graphs to neural networks on the manifold model (Wang, Ruiz, and Ribeiro 2024a), combined with the stability of the Manifold Neural Networks (MNNs) under manifold deformations (Wang, Ruiz, and Ribeiro 2024b). Implementing low-pass and integral Lipschitz continuous filters (Definition 2) allows us to bound the generalization gap under model mismatches. This bound decreases with the number of nodes in the training graph and increases with both



(a) Manifold $\mathcal{M}$ (b) Mismatched Manifold $\mathcal{M}'$ (c) Perturbed manifold function f'

Figure 1: Example of model mismatch. (a) The original manifold with a generated graph based on sampled points $(P1, \cdots, P6)$. (b) The mismatched manifold with the sampled points also shifted, resulting in a perturbed graph. (c) The manifold mismatch can be seen as the perturbation of manifold function values, which leads to perturbed node features on the generated graph.

the mismatch size and the manifold dimension. The key insight from the bound is that a more robust generalization necessitates the cost of failing to discriminate high spectral components in the graph data.

Our main contributions are as follows:

1. We analyze the generalization of the GNNs when there exists a manifold model mismatch between the training data and testing data.
2. We determine the manifold mismatch as perturbations on the generated graphs, both as node feature perturbations and edge perturbations.
3. We propose that implementing continuity restrictions on the filters composing the GNN ensures the robust generalization in both node and graph classification tasks.
4. We observe a trade-off between robust generalization and discriminability through the generalization bound.

We conduct experiments on real-world datasets to validate our theoretical findings.

Related Works

In-Distribution Generalization of GNNs

Existing works on the in-distribution generalization of GNNs fall in node and graph level tasks. For node classification tasks of GNNs, there are works providing a generalization bound of GNNs based on a Vapnik-Chervonenkis dimension (Scarselli, Tsoi, and Hagenbuchner 2018), algorithmic stability analysis (Verma and Zhang 2019; Zhou and Wang 2021), PAC-Bayesian analysis (Ma, Deng, and Mei 2021) and Rademacher complexity (Esser, Chennuru Vankadara, and Ghoshdastidar 2021). For graph classification tasks of GNNs, the authors prove the generalization bound via Rademacher complexity (Garg, Jegelka, and Jaakkola 2020) and PAC-Bayes analysis (Liao, Urtasun, and Zemel 2020; Ju et al. 2023). The authors consider a continuous graph limit model to analyze the generalization of GNNs on graph classification in (Maskey et al. 2022; Maskey, Kutyniok, and Levie 2024; Levie 2024). In (Wang, Cerviño, and Ribeiro 2024a), the authors prove the generalization of GNNs on graphs sampled from a manifold both for node and graph classification tasks. These works are considered only in the in-distribution case where the training and testing data are sampled from the same distribution.

Out-of-Distribution Generalization of GNNs

Several works have extensively addressed the out-of-distribution generalization of GNNs with graph enhancement methods. The authors in (Tian et al. 2024) propose a domain generalization framework for node-level tasks on graphs to address distribution shifts in node attribute distribution and graphic topology. In (Fan et al. 2023), the authors study the out-of-distribution generalization of GNNs on graph-level tasks with a causal representation learning framework. In (Li et al. 2022) the authors handle graph distribution shifts in complex and heterogeneous situations of GNNs with a nonlinear graph representation decorrelation method. The authors in (Yehudai et al. 2021) propose a size generalization analysis of GNNs correlated to the discrepancy between local distributions of graphs. In our novel approach, we conceptualize the generative model mismatch as a distribution shift. Our findings can be further generalized into a theoretical framework for GNNs in manifold domain shift scenarios. In such cases, the generalization gap is directly proportional to the distance between the manifold models. This extension is discussed in more detail within the supplementary material.

GNNs and Manifold Neural Networks

Geometric deep learning has been proposed in (Bronstein et al. 2017) with neural network architectures on manifolds. The authors in (Monti et al. 2017) and (Chakraborty et al. 2020) provide neural network architectures for manifold-valued data. In (Wang, Ruiz, and Ribeiro 2024b) and (Wang, Ruiz, and Ribeiro 2022), the authors define convolutional operation over manifolds and see the manifold convolution as a generalization of graph convolution, which establishes

the limit of neural networks on large-scale graphs as manifold neural networks (MNNs). The authors in (Wang, Ruiz, and Ribeiro 2024a) further establish the relationship between GNNs and MNNs with non-asymptotic convergence results for different graph constructions.

Neural Networks on Manifolds

Suppose there is a d-dimensional embedded Riemannian manifold $\mathcal{M} \subset \mathbb{R}^{\mathsf{M}}$ which is compact, smooth and differentiable. A measure μ over $\mathcal{M}$ with density function $\rho : \mathcal{M} \to (0, \infty)$, which is assumed to be bounded as $0 < \rho_{min} \leq \rho(x) \leq \rho_{max} < \infty$ for each $x \in \mathcal{M}$. Data supported over the manifold is defined as a scalar function $f : \mathcal{M} \to \mathbb{R}$ (Wang, Ruiz, and Ribeiro 2024b) mapping datum value $f(x)$ to each point $x \in \mathcal{M}$. The manifold with density ρ is endowed with a weighted Laplace operator (Grigor'yan 2006), which generalizes the Laplace-Beltrami operator as

$$\mathcal{L}f = -\frac{1}{2\rho}\text{div}(\rho^2 \nabla f), \tag{1}$$

where div denotes the divergence operator of $\mathcal{M}$ and ∇ denotes the gradient operator of $\mathcal{M}$ (Bronstein et al. 2017).

We consider square-integrable functions over $\mathcal{M}$, denoted as use $L^2(\mathcal{M})$. The inner product of functions $f, g \in L^2(\mathcal{M})$ is defined as

$$\langle f, g \rangle_{\mathcal{M}} = \int_{\mathcal{M}} f(x) g(x) \text{d}\mu(x), \tag{2}$$

while the L^2 norm is defined as $\|f\|^2_{\mathcal{M}} = \langle f, f \rangle_{\mathcal{M}}$.

Manifold convolution is defined by aggregating the heat diffusion process over $\mathcal{M}$ with operator $\mathcal{L}$ (Wang, Ruiz, and Ribeiro 2022, 2024b). With the input manifold function $f \in L^2(\mathcal{M})$, the manifold convolution $\mathbf{h}(\mathcal{L})$ is

$$g(x) = \mathbf{h}(\mathcal{L}) f(x) = \sum_{k=0}^{K-1} h_k e^{-k\mathcal{L}} f(x). \tag{3}$$

Considering the frequency representation, the Laplace operator $\mathcal{L}$ has real, positive and discrete eigenvalues $\{\lambda_i\}_{i=1}^{\infty}$ as the Laplace operator is self-adjoint and positive-semidefinite. The eigenfunction associated with each eigenvalue is denoted as $\boldsymbol{\phi}_i$, i.e. $\mathcal{L}\boldsymbol{\phi}_i = \lambda_i \boldsymbol{\phi}_i$. The eigenvalues are ordered as $0 = \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \ldots$, and the eigenfunctions are orthonormal and form the eigenbasis of $L^2(\mathcal{M})$. When mapping a manifold function onto one of the eigenbasis $\boldsymbol{\phi}_i$, we have the spectral component as $[\hat{f}]_i = \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}}$. The manifold convolution can be written in the spectral domain point-wisely as

$$[\hat{g}]_i = \sum_{k=0}^{K-1} h_k e^{-k\lambda_i} [\hat{f}]_i. \tag{4}$$

Hence, the frequency response of manifold filter is given by $\hat{h}(\lambda) = \sum_{k=0}^{K-1} h_k e^{-k\lambda}$, depending only on the filter coefficients h_k and eigenvalues of $\mathcal{L}$ when $\lambda = \lambda_i$.

Manifold neural networks (MNNs) are built by cascading layers consisting of a bank of manifold filters and a point-wise nonlinearity function $\sigma : \mathbb{R} \to \mathbb{R}$, with the output of each layer written as

$$f_l(x) = \sigma\Big(\mathbf{h}_l(\mathcal{L}) f_{l-1}(x)\Big). \tag{5}$$

To represent the MNN succinctly, we group all learnable parameters, and denote the mapping based on input manifold function $f \in L^2(\mathcal{M})$ to predict target function $g \in L^2(\mathcal{M})$ as $\boldsymbol{\Phi}(\mathbf{H}, \mathcal{L}, f)$, where $\mathbf{H} \in \mathcal{H} \subset \mathbb{R}^P$ is a filter parameter set of the manifold filters. A positive loss function is denoted as $\ell(\boldsymbol{\Phi}(\mathbf{H}, \mathcal{L}, f), g)$ to measure the estimation performance.

Geometric Graph Neural Networks

Suppose we can access a discrete set of sampled points over manifold $\mathcal{M}$. A graph $\mathbf{G}$ is generated based on a set of N i.i.d. randomly sampled points $X_N = \{x_N^1, x_N^2, \cdots, x_N^N\}$ according to measure μ over $\mathcal{M}$. Seeing these N sampled points as nodes, edges connect every pair of nodes (x_N^i, x_N^j) is connected with weight value $[\mathbf{W}_N]_{ij}, \mathbf{W}_N \in \mathbb{R}^{N \times N}$ determined by a function of their Euclidean distance $\|x_N^i - x_N^j\|$ (Calder and Trillos 2022), explicitly written as

$$[\mathbf{W}_N]_{ij} = \frac{\alpha_d}{(d+2)N\epsilon^{d+2}} \mathbf{1}_{[0,1]}\left(\frac{\|x_N^i - x_N^j\|}{\epsilon}\right), \tag{6}$$

where α_d is the volume of the d-dimensional Euclidean unit ball and $\mathbf{1}$ represents an indicator function. Based on this, the graph Laplacian can be calculated as $\mathbf{L}_N = \text{diag}(\mathbf{W}_N \mathbf{1}) - \mathbf{W}_N$. On this generated graph, graph data values are sampled from the functions over manifold $\mathcal{M}$ (Wang, Ruiz, and Ribeiro 2022; Chew, Needell, and Perlmutter 2023). Consider the input and target functions $f, g \in L^2(\mathcal{M})$, the sampled input and target functions $\mathbf{x}, \mathbf{y} \in L^2(X_N)$ over graph $\mathbf{G}$ can be written as

$$[\mathbf{x}]_i = f(x_N^i), \quad [\mathbf{y}]_i = g(x_N^i). \tag{7}$$

A convolutional filter on graph $\mathbf{G}$ can be extended from manifold convolution defined in (3) by replacing the Laplace operator with the graph Laplacian as

$$\mathbf{y} = \mathbf{h}(\mathbf{L})\mathbf{x} = \sum_{k=0}^{K-1} h_k e^{-k\mathbf{L}} \mathbf{x}. \tag{8}$$

We observe that this formation is accordant with the definition of the graph convolution (Gama et al. 2019) with graph shift operator as $e^{-\mathbf{L}}$. Replace $\mathbf{L}$ with eigendecomposition $\mathbf{L} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^H$, where $\mathbf{V}$ is the eigenvector matrix and $\boldsymbol{\Lambda}$ is a diagonal matrix with eigenvalues of $\mathbf{L}$ as the entries. The spectral representation of this filter on $\mathbf{G}$ is

$$\mathbf{V}^H \mathbf{h}(\mathbf{L}_\tau)\mathbf{x} = \sum_{k=1}^{K-1} h_k e^{-\Lambda k} \mathbf{V}^H \mathbf{x} = \hat{h}(\boldsymbol{\Lambda})\mathbf{V}^H \mathbf{x}. \tag{9}$$

This analogously leads to a frequency response of this graph convolution, i.e. $\hat{h}(\lambda) = \sum_{k=0}^{K-1} h_k \lambda^k$, relating the input and output spectral components point-wisely.

Neural networks on $\mathbf{G}$ is composed of layers consisting of graph filters and point-wise nonlinearity σ, written as

$$\mathbf{x}_l = \sigma\Big(\mathbf{h}_l(\mathbf{L})\mathbf{x}_{l-1}\Big). \tag{10}$$

The mapping from input graph data $\mathbf{x} \in L^2(X_N)$ to predict $\mathbf{y} \in L^2(X_N)$ is denoted as $\boldsymbol{\Phi}(\mathbf{H}, \mathbf{L}, \mathbf{x})$ with $\mathbf{H} \in \mathcal{H} \subset \mathbb{R}^P$ which is trained to minimize the loss $\ell(\boldsymbol{\Phi}(\mathbf{H}, \mathbf{L}, \mathbf{x}), \mathbf{y})$.

Generalization of GNNs to Model Mismatch

Manifold Model Mismatch

A mismatched model of manifold $\mathcal{M}$ is denoted as $\mathcal{M}^\tau$ where τ maps each point $x \in \mathcal{M}$ to a displaced $\tau(x) \in \mathcal{M}^\tau$. We restrict this mismatch function class τ as it preserves the properties of $\mathcal{M}$ and denote the curvature distance between x and the displaced $\tau(x)$ as $\text{dist}(x, \tau(x))$. The mismatch τ induces a tangent map $\tau_{*,x} : T_x\mathcal{M} \to T_{\tau(x)}\mathcal{M}$ which is a linear map between the tangent spaces (Tu 2011). With the coordinate description over $\mathcal{M}$, the tangent map $\tau_{*,x}$ can be exactly represented by the Jacobian matrix $J_x(\tau)$.

The manifold model mismatch can be seen as deformations to input manifold functions as shown in Figure 1c.

$$\mathcal{L}f(\tau(x)) = \mathcal{L}f'(x), \qquad x \in \mathcal{M}. \tag{11}$$

While the model mismatch can also be understood as deformations to the Laplacian operator as shown in Figure 1b.

$$\mathcal{L}f(\tau(x)) = \mathcal{L}_\tau f(x), \qquad x \in \mathcal{M}. \tag{12}$$

Generalization of GNNs on node-level

The generalization analysis is restricted to a finite-dimensional subset of $L^2(\mathcal{M})$, i.e. the functions over the manifold are bandlimited as defined in Definition 1.

Definition 1 *Function $f \in L^2(\mathcal{M})$ is bandlimited if there exists some $\lambda > 0$ such that for all eigenpairs $\{\lambda_i, \boldsymbol{\phi}_i\}_{i=1}^\infty$ of the weighted Laplacian $\mathcal{L}$ when $\lambda_i > \lambda$, we have $\langle f, \boldsymbol{\phi}_i \rangle_\mathcal{M} = 0$.*

Assumption 1 *(Lipschitz target function) The target function g is Lipschitz continuous, i.e., $|g(x) - g(y)| \leq C_g \text{dist}(x, y)$, for all $x, y \in \mathcal{M}$.*

We further assume that the filters in MNN $\boldsymbol{\Phi}(\mathbf{H}, \mathcal{L}, \cdot)$ and GNN $\boldsymbol{\Phi}(\mathbf{H}, \mathbf{L}, \cdot)$ are low-pass and integral Lipschitz filters as defined in Definition 2. We note that this is a mild assumption as high frequency components on the graph/manifold implies that there exist functions with large variations in adjacent entries. This naturally generates instabilities that are more difficult to learn.

Definition 2 *A filter is a low-pass and integral Lipschitz filter if its frequency response satisfies*

$$\left|\hat{h}(\lambda)\right| = \mathcal{O}\left(\lambda^{-d}\right), \quad \lambda \to \infty, \tag{13}$$

$$\left|\hat{h}'(\lambda)\right| \leq C_L \lambda^{-d-1}, \quad \lambda \in (0, \infty). \tag{14}$$

with d denoted as the dimension of manifold $\mathcal{M}$ and C_L a positive continuity constant.

We note that with a smaller C_L, the filter function tends to be smoother especially in the high-spectrum domain. The filters fail to discriminate different high spectral components with similar frequency responses given to them.

In the neural network architectures, we consider assumptions on both the nonlinear activation function and the loss function as presented in Assumption 2 and 3 respectively. We note that these are reasonable assumptions as most activations (e.g. ReLU, modulus, sigmoid) and loss functions (e.g. L1 regression, Huber loss, quantile loss).

Assumption 2 *(Normalized Lipschitz activation functions) The activation function σ is normalized Lipschitz continuous, i.e., $|\sigma(a) - \sigma(b)| \leq |a - b|$, with $\sigma(0) = 0$.*

Assumption 3 *(Normalized Lipschitz loss function) The loss function ℓ is normalized Lipschitz continuous, i.e., $|\ell(y_i, y) - \ell(y_j, y)| \leq |y_i - y_j|$, with $\ell(y, y) = 0$.*

The generalization gap is evaluated between the *empirical risk* over the discrete graph model and the *statistical risk* over the manifold model, which has been studied in both node-level and graph level in previous works when no model mismatch is considered (Wang, Cerviño, and Ribeiro 2024b,a).

Suppose the training graph $\mathbf{G}$ is generated from a manifold $\mathcal{M}$. The empirical risk that we train the GNN to minimize is therefore defined as

$$R_\mathbf{G}(\mathbf{H}) = \frac{1}{N} \sum_{i=1}^N \ell\left([\boldsymbol{\Phi}(\mathbf{H}, \mathbf{L}, \mathbf{x})]_i, [\mathbf{y}]_i\right). \tag{15}$$

The statistical risk over mismatched manifold $\mathcal{M}_\tau$ is

$$R_{\mathcal{M}^\tau}(\mathbf{H}) = \int_{\mathcal{M}^\tau} \ell\left(\boldsymbol{\Phi}(\mathbf{H}, \mathcal{L}_\tau, f)(x), g(x)\right) \mathrm{d}\mu_\tau(x). \tag{16}$$

The generalization gap under model mismatch is

$$GA_\tau = \sup_{\mathbf{H} \in \mathcal{H}} \left|R_{\mathcal{M}^\tau}(\mathbf{H}) - R_\mathbf{G}(\mathbf{H})\right|. \tag{17}$$

Theorem 1 *Suppose a neural network is equipped with filters defined in Definition 2 and normalized nonlinearites (Assumption 2). The neural network is operated on a graph $\mathbf{G}$ generated according to (6) from $\mathcal{M}$ and a mismatched manifold $\mathcal{M}^\tau$ with a bandlimited (Definition 1) input manifold function. Suppose the mismatch $\tau : \mathcal{M} \to \mathcal{M}$ satisfies $\text{dist}(x, \tau(x)) \leq \gamma$ and $\|J_x(\tau) - I\|_F \leq \gamma$ for all $x \in \mathcal{M}$. The generalization of neural network trained on $\mathbf{G}$ to minimize a normalized Lipschitz loss function (Assumption 3) holds in probability at least $1 - \delta$ that*

$$GA_\tau \leq C_1 \frac{\epsilon}{\sqrt{N}} + C_2 \frac{\sqrt{\log(1/\delta)}}{N} + C_3 \left(\frac{\log N}{N}\right)^{\frac{1}{d}} + C_4 \gamma, \tag{18}$$

with $\epsilon \sim \left(\frac{\log(C/\delta)}{N}\right)^{\frac{1}{d+4}}$, C_1 scaling with C_L, C_4 scaling with C_L and C_g. C_1 and C_3 depend on the geometry of $\mathcal{M}$.

Remark 1 This conclusion is ready to extend to multi-layer and multi-feature neural network architectures, as the neural network is cascaded by layers of filters and nonlinearities. The generalization error propagates across layers, leading to an increase in the generalization gap of multi-layer and multi-feature GNNs with the size of the architecture, which we will further verify in simulations.

Theorem 1 indicates that the generalization of GNNs is robust to model mismatches, with the generalization gap decreasing with the number of nodes N in the training graph. As more points are sampled from the manifold, the generated graph can better approximate the underlying manifold. This leads to a better generalization of the GNN to predict the unseen points over the manifold. The upper bound also

increases with the dimension of the underlying manifold d, as higher dimension indicates higher model complexity. Specifically, the generalization gap increases with the mismatch size γ as the testing manifold is shifted more from the original manifold.

Remark 2 It is important to note that there is a trade-off involved in designing GNNs: achieving better generalization often comes at the cost of reduced discriminability. Specifically, using a smaller C_L leads to improved generalization and robustness by reducing generalization error. While this leads to smoother filter functions, which limits the GNN ability to discriminate between different spectral components. This reduced discriminability can negatively impact prediction performance. In essence, we can enhance better and more robust generalization capabilities of GNNs, but this improvement necessitates a compromise in their discriminative power.

Extension to Graph Classification

The generalization ability of GNN can be extended from the node level to the graph level, which indicates the manifold classification with approximated graph classification.

Suppose we have manifolds $\{\mathcal{M}_k^{\tau_k}\}_{k=1}^K$ with dimension d_k under a mismatch τ_k. Manifold $\mathcal{M}_k^{\tau_k}$ is labeled with $y_k \in \mathbb{R}$. The manifolds are smooth, compact, differentiable, and embedded in $\mathbb{R}^{\mathsf{M}}$ with measure $\mu_{\tau_k,k}$. Each manifold $\mathcal{M}_k^{\tau_k}$ is equipped with a weighted Laplace operator $\mathcal{L}_{\tau_k,k}$. Assume that we can access N_k randomly sampled points according to μ_k over each manifold $\mathcal{M}_k$. Graphs generated based on these sampled points are denoted as $\{\mathbf{G}_k\}_{k=1}^K$ with graph Laplacians $\{\mathbf{L}_k\}_{k=1}^K$. A GNN $\mathbf{\Phi}(\mathbf{H}, \mathbf{L}_\cdot, \mathbf{x}_\cdot)$ is trained on these graphs with $\mathbf{x}_k$ denoted as the input data on graphs sampled from the data on manifolds $f_k \in L^2(\mathcal{M}_k)$. The output of the GNN is set as the average of the output values over all the nodes while the output of MNN $\mathbf{\Phi}(\mathbf{H}, \mathcal{L}_{\tau_\cdot,\cdot}, f_\cdot)$ is the averaged value over the mismatched manifold. Loss function ℓ evaluates the performance of GNN and MNN by comparing the difference between the output label and the target label. The empirical risk that the GNN is trained to minimize is defined as

$$R_{\mathbf{G}}(\mathbf{H}) = \sum_{k=1}^{K} \ell\left(\frac{1}{N_k}\sum_{i=1}^{N_k}[\mathbf{\Phi}(\mathbf{H}, \mathbf{L}_k, \mathbf{x}_k)]_i, y_k\right). \quad (19)$$

The statistical risk is defined based on the statistical MNN output and the target label over mismatched models as

$$R_{\mathcal{M}^\tau}(\mathbf{H}) = \sum_{k=1}^{K} \ell\left(\int_{\mathcal{M}_k} \mathbf{\Phi}(\mathbf{H}, \mathcal{L}_{\tau_k,k}, f_k)(x)\mathrm{d}\mu_{\tau_k}(x), y_k\right). \quad (20)$$

The generalization gap is therefore

$$GA_\tau = \sup_{\mathbf{H}\in\mathcal{H}} \left|R_{\mathcal{M}^\tau}(\mathbf{H}) - R_{\mathbf{G}}(\mathbf{H})\right|. \quad (21)$$

Theorem 2 *Suppose the GNN and MNN with filters defined in Definition 2 and the input manifold functions are bandlimited (Definition 1). Suppose the mismatches $\tau_k : \mathcal{M}_k \to \mathcal{M}_k$ where dist$(x, \tau_k(x)) \leq \gamma$ and $\|J_x(\tau_k) - I\|_F \leq \gamma$ for all $x \in \mathcal{M}_k$ for $k = 1, 2, \cdots K$. Under Assumptions 2 and 3 it holds in probability at least $1 - \delta$ that*

$$GA_\tau \leq \sum_{k=1}^{K}\left(\frac{C_1}{\sqrt{N_k}}\epsilon_k + C_2\frac{\sqrt{\log(1/\delta)}}{N_k}\right) + C_3\sum_{k=1}^{K}\left(\frac{\log N_k}{N_k}\right)^{\frac{1}{d_k}} + KC_4\gamma, \quad (22)$$

with $\epsilon_k \sim \left(\frac{\log(C/\delta)}{N_k}\right)^{\frac{1}{d_k+4}}$, C_1 and C_4 scaling with C_L. C_1 and C_3 depend on the geometry of $\mathcal{M}$.

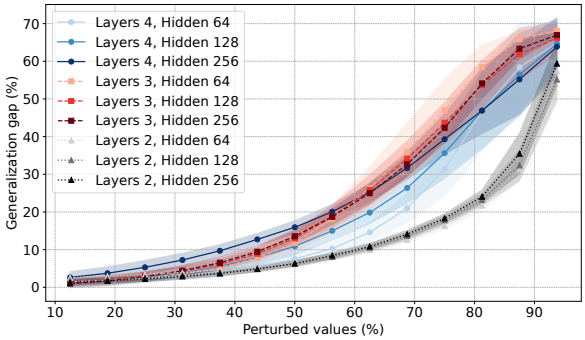
Theorem 2 indicates that a graph with large enough points sampled from each underlying manifold can approximately predict the label for mismathed manifolds. This shows that GNN trained over these generated graphs can generalize to classify unseen graphs generated from mismatched manifold models. The generalization gap on the graph level also decreases with the number of sampled points over each manifold. The generalization gap increases with the dimensions of the manifolds and the size of deformations. A similar trade-off phenomenon can also be observed.

Experiments

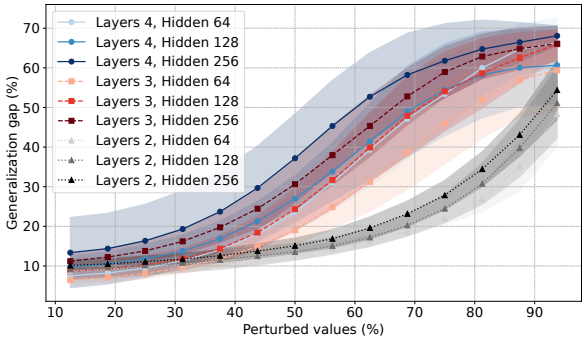
Node Classification with Arxiv Dataset

In this section, we showcase the generalization properties of a trained GNN on a real-world dataset, OGBN-Arxiv (Wang et al. 2020). The graph has $169,343$ nodes and $1,166,243$ edges, representing the citation network between computer science arXiv papers. The node features are 128 dimensional embeddings of the title and abstract of each paper (Mikolov et al. 2013). The objective is to predict which of the 40 categories the paper belongs to. We consider two types of perturbations to model the impact of the underlying manifold model mismatch – node and edge perturbations.

In all cases, we train the GNN on the original graph, and we evaluate it on the perturbed graph. The generalization gap is measured by the difference between the training accuracy and the perturbed testing accuracy. For node perturbations, we randomly zero out a percentage of the 128 dimensional embeddings for all the points in the dataset. Theoretically, that corresponds to modifying the underlying scalar function f (see equation (11)). For the edge perturbation, we randomly remove a percentage of the existing edges in the graph. This corresponds to perturbing the Laplace operator $\mathcal{L}$ (see equation (12)). We run these experiments for a varying number of nodes, by partitioning the training set in $\{1, 2, 4, 8, ...512, 1024\}$ partitions. We train the GNN with the *cross-entropy* loss, for 1000 epochs, using 0.005 learning rate, ADAM optimizer (Kingma and Ba 2014), and no weight decay with `ReLU` non-linearity. The purpose of the experiments is to show that GNN with more layers and hidden units has a larger generalization gap under the same perturbation level. We further verify the relationship between the generalization gaps and the logarithm of the number of nodes in the graphs as indicated in Theorem 1.

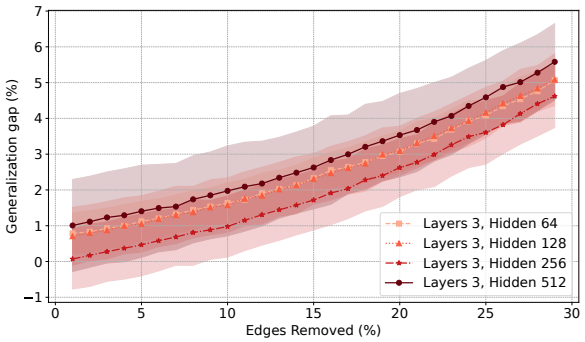


(a) 90941 nodes

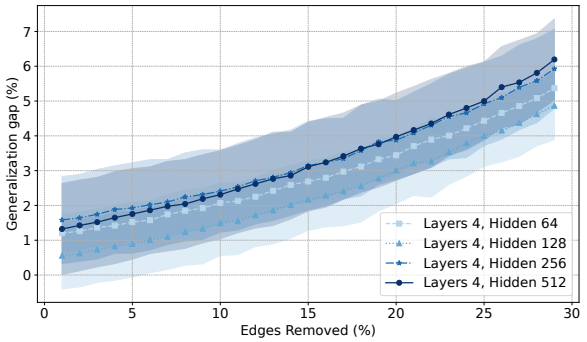


(b) 2841 nodes

Figure 2: Generalization gap as a function of the percentage of perturbed feature values in node feature perturbation.



(a) 3 Layered GNN



(b) 4 Layered GNN

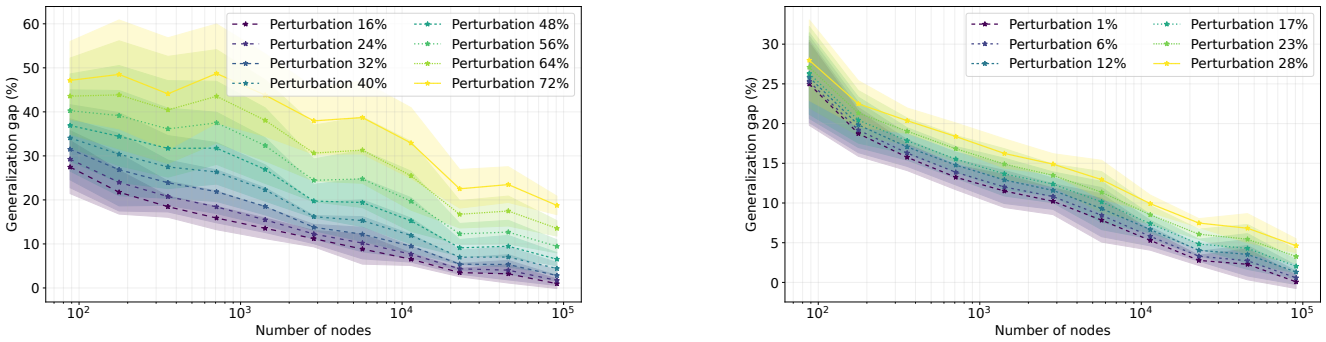
Figure 3: Generalization gap as a function of the percentage of perturbed edges values in node removal perturbation.

In Figure 2 we can see the generalization gap as a function of the perturbation level for different GNN architectures. In these two figures, we confirm that GNNs with fewer layers and fewer hidden units are more robust to changes in the underlying manifold. This can be seen as the black lines (2 layers), are below both the red (3 layers) and the blue lines(4 layers). Intuitively, this can be attributed to overfitting the training set, given that more capacity in the GNN translates into a better training set accuracy. We also showcase that if the number of nodes in the training set is larger, the GNN is more robust to changes on the graph, given that the generalization gap increases more slowly in the GNN trained with 90941 nodes (Figure 2a), than in the one trained with 2841 nodes (Figure 2b). In Figure 3, we plot the generalization gap as a function of the perturbation magnitude for a GNN with 3 (Figure 3a), and 4 layers (Figure 3b). In both cases, the GNN with the largest number of hidden units is at the top, thus indicating that a larger generalization gap as indicated in Remark 1.

Figure 4 shows the generalization gaps of GNNs under node feature perturbations and edge perturbations decrease approximately linearly with the number of nodes in the logarithmic scale while increasing with the perturbation sizes. In Figure 4a and 4b, we show the generalization gap of a 3 layered GNN with 256 hidden units as a function of the number of nodes in the training set under node feature perturbation and edge removal perturbation, respectively. Each color in the figure represents a different perturbation level, going from less perturbation (16% darker blue) to higher perturbation (72% lighter yellow). As can be seen in the plot, the GNNs generalization degrades as the perturbation level increases. Aligning with our theory, the generalization gap is linear with respect to the logarithm of the number of nodes. This is also true when the perturbation level increases.

Graph Classification with ModelNet Dataset

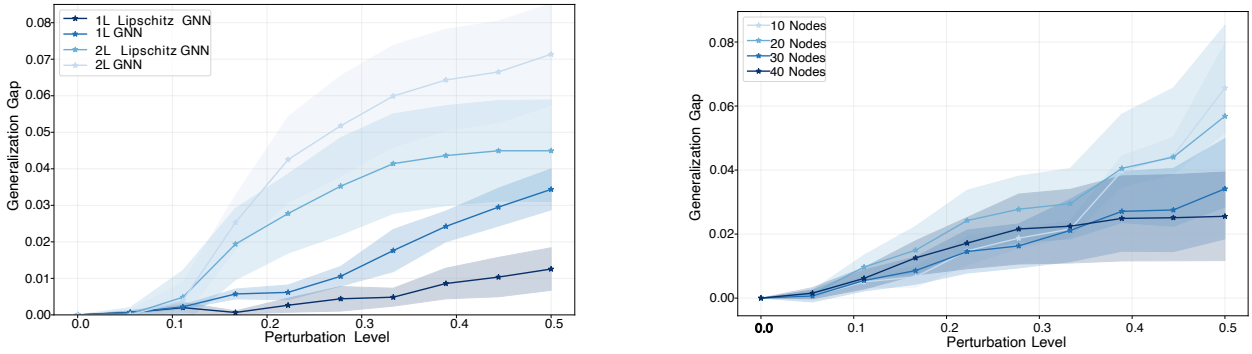
We evaluate the generalization gap of GNNs on graph level with the ModelNet10 dataset (Wu et al. 2015). The dataset contains 3991 meshed CAD models from 10 categories for training and 908 models for testing. For each model, discrete points are uniformly randomly sampled from all points of the model to form the graphs. Each point is characterized by the 3D coordinates as features. Each node in the graph can be modeled as the sampling point and each edge weight is constructed based on the distance between each pair of nodes. The input graph features are set as the coordinates of each point, and the weights of the edges are calculated based on the Euclidean distance between the points.



(a) Node Feature Perturbation

(b) Edge Removal Perturbation

Figure 4: Generalization gap for edge and node perturbation for the Arxiv dataset for a 3 layered, 256 feature GNN.



(a) GNN with different architectures.

(b) GNN on different number of nodes.

Figure 5: Generalization gap as a function of number of nodes in the Point Cloud Classification.

The mismatched point cloud model adds a Gaussian random variable with mean γ and variance 2γ to each coordinate of every sampled point. This can be seen as the underlying manifold mismatch. We calculate the generalization gap by training GNNs on graphs with $N = 40$ sampled points, and plotting the differences between the testing accuracy on the trained graphs sampled from normal point clouds and the testing accuracy on the graphs sampled from deformed point clouds with perturbation level γ. We implement Graph Neural Networks (GNN) with 1 and 2 layers with a single layer containing $F_0 = 3$ input features which are the 3d coordinates of each point, $F_1 = 64$ output features and $K = 5$ filter taps. While the architectures with 2 layers has another layer with $F_2 = 32$ features and 5 filter taps. We use the `ReLU` as nonlinearity. All architectures also include a linear readout layer mapping the final output features to a binary scalar that estimates the classification. Figure 5a shows the generalization gaps for GNNs with or without Lipschitz continuity assumptions of the graph filters. We observe that the continuity assumptions imposed on the graph filters help to improve the generalization capabilities as Theorem 2 shows. Figure 5b shows the results of GNNs on different number of nodes. We can see that the generalization gaps of GNNs scale with the size of the GNN architectures and scale with the size of mismatch. Figure 5b also shows that the generalization gap decreases with the number of nodes.

Conclusion

We showed that the robustness of GNN generalization to model mismatch from a manifold perspective. We focused on graphs derived from manifolds, and we established that GNNs equipped with low-pass and integral Lipschitz graph filters exhibit robust generalization. This generalization extends to previously unseen nodes and graphs derived from mismatched manifolds. Notably, we identified a tradeoff between the robust generalization and discriminability of GNNs. We validate our results both on node-level and graph-level classification tasks with real-world datasets.

Acknowledgments

This work is supported by Theorinet Simons.

References

Bronstein, M. M.; Bruna, J.; LeCun, Y.; Szlam, A.; and Vandergheynst, P. 2017. Geometric deep learning: going beyond

euclidean data. *IEEE Signal Processing Magazine*, 34(4): 18–42.

Calder, J.; and Trillos, N. G. 2022. Improved spectral convergence rates for graph Laplacians on ε-graphs and k-NN graphs. *Applied and Computational Harmonic Analysis*, 60: 123–175.

Chakraborty, R.; Bouza, J.; Manton, J. H.; and Vemuri, B. C. 2020. Manifoldnet: A deep neural network for manifold-valued data with applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(2): 799–810.

Chew, J. A.; Needell, D.; and Perlmutter, M. 2023. A convergence rate for manifold neural networks. In *2023 International Conference on Sampling Theory and Applications (SampTA)*, 1–5. IEEE.

Esser, P.; Chennuru Vankadara, L.; and Ghoshdastidar, D. 2021. Learning theory can (sometimes) explain generalisation in graph neural networks. *Advances in Neural Information Processing Systems*, 34: 27043–27056.

Fan, S.; Wang, X.; Shi, C.; Cui, P.; and Wang, B. 2023. Generalizing graph neural networks on out-of-distribution graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Gama, F.; Marques, A. G.; Leus, G.; and Ribeiro, A. 2019. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing*, 67(4): 1034–1049.

Garg, V.; Jegelka, S.; and Jaakkola, T. 2020. Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning*, 3419–3430. PMLR.

Gosrich, W.; Mayya, S.; Li, R.; Paulos, J.; Yim, M.; Ribeiro, A.; and Kumar, V. 2022. Coverage control in multi-robot systems via graph neural networks. In *2022 International Conference on Robotics and Automation (ICRA)*, 8787–8793. IEEE.

Grigor'yan, A. 2006. Heat kernels on weighted manifolds and applications. *Cont. Math*, 398(2006): 93–191.

Ju, H.; Li, D.; Sharma, A.; and Zhang, H. R. 2023. Generalization in graph neural networks: Improved pac-bayesian bounds on graph diffusion. In *International Conference on Artificial Intelligence and Statistics*, 6314–6341. PMLR.

Kawaguchi, K.; Kaelbling, L. P.; and Bengio, Y. 2022. Generalization in Deep Learning. *In Mathematical Aspects of Deep Learning. Cambridge University Press*.

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 *International Conference on Learning Representations*.

Levie, R. 2024. A graphon-signal analysis of graph neural networks. *Advances in Neural Information Processing Systems*, 36.

Li, H.; Wang, X.; Zhang, Z.; and Zhu, W. 2022. Ood-gnn: Out-of-distribution generalized graph neural network. *IEEE Transactions on Knowledge and Data Engineering*, 35(7): 7328–7340.

Liao, R.; Urtasun, R.; and Zemel, R. 2020. A PAC-Bayesian Approach to Generalization Bounds for Graph Neural Networks. In *International Conference on Learning Representations*.

Ma, J.; Deng, J.; and Mei, Q. 2021. Subgroup generalization and fairness of graph neural networks. *Advances in Neural Information Processing Systems*, 34: 1048–1061.

Maskey, S.; Kutyniok, G.; and Levie, R. 2024. Generalization Bounds for Message Passing Networks on Mixture of Graphons. *arXiv preprint arXiv:2404.03473*.

Maskey, S.; Levie, R.; Lee, Y.; and Kutyniok, G. 2022. Generalization analysis of message passing neural networks on large random graphs. *Advances in neural information processing systems*, 35: 4805–4817.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed Representations of Words and Phrases and their Compositionality. In Burges, C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Monti, F.; Boscaini, D.; Masci, J.; Rodola, E.; Svoboda, J.; and Bronstein, M. M. 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5115–5124.

Ruiz, L.; Chamon, L. F.; and Ribeiro, A. 2023. Transferability properties of graph neural networks. *IEEE Transactions on Signal Processing*.

Sandryhaila, A.; and Moura, J. M. 2013. Discrete signal processing on graphs. *IEEE transactions on signal processing*, 61(7): 1644–1656.

Scarselli, F.; Tsoi, A. C.; and Hagenbuchner, M. 2018. The vapnik–chervonenkis dimension of graph and recursive neural networks. *Neural Networks*, 108: 248–259.

Tian, Q.; Wang, W.; Zhao, C.; Shao, M.; Zhang, W.; and Li, D. 2024. Graphs Generalization under Distribution Shifts. *arXiv preprint arXiv:2403.16334*.

Tu, L. W. 2011. *An introduction to manifolds*. Springer.

Verma, S.; and Zhang, Z.-L. 2019. Stability and generalization of graph convolutional neural networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1539–1548.

Wang, K.; Shen, Z.; Huang, C.; Wu, C.-H.; Dong, Y.; and Kanakia, A. 2020. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 1(1): 396–413.

Wang, Z.; Cervino, J.; and Ribeiro, A. 2024. A Manifold Perspective on the Statistical Generalization of Graph Neural Networks. *arXiv preprint arXiv:2406.05225*.

Wang, Z.; Cerviño, J.; and Ribeiro, A. 2024a. Generalization of Geometric Graph Neural Networks. *submitted to Asilomar Conference on Signals, Systems, and Computers 2024.*

Wang, Z.; Cerviño, J.; and Ribeiro, A. 2024b. A Manifold Perspective on the Statistical Generalization of Graph Neural Networks. *arXiv preprint arXiv:2406.05225*.

Wang, Z.; Ruiz, L.; and Ribeiro, A. 2022. Convolutional neural networks on manifolds: From graphs and back. In *2022 56th Asilomar Conference on Signals, Systems, and Computers*, 356–360. IEEE.

Wang, Z.; Ruiz, L.; and Ribeiro, A. 2024a. Geometric Graph Filters and Neural Networks: Limit Properties and Discriminability Trade-offs. *IEEE Transactions on Signal Processing*.

Wang, Z.; Ruiz, L.; and Ribeiro, A. 2024b. Stability to Deformations of Manifold Filters and Manifold Neural Networks. *IEEE Transactions on Signal Processing*, 1–15.

Wu, S.; Sun, F.; Zhang, W.; Xie, X.; and Cui, B. 2022. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5): 1–37.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1912–1920.

Yehudai, G.; Fetaya, E.; Meirom, E.; Chechik, G.; and Maron, H. 2021. From local structures to size generalization in graph neural networks. In *International Conference on Machine Learning*, 11975–11986. PMLR.

Yin, N.; Shen, L.; Wang, M.; Lan, L.; Ma, Z.; Chen, C.; Hua, X.-S.; and Luo, X. 2023. Coco: A coupled contrastive framework for unsupervised domain adaptive graph classification. In *International Conference on Machine Learning*, 40040–40053. PMLR.

Zhou, X.; and Wang, H. 2021. The generalization error of graph convolutional networks may enlarge with more layers. *Neurocomputing*, 424: 97–106.