Transferable Hypergraph Neural Networks via Spectral Similarity

Mikhail Hayhoe
Amgen, Inc.
contact@mhayhoe.com

Hans Riess[*]
Duke University
hans.riess@duke.edu

Michael M. Zavlanos
Duke University
michael.zavlanos@duke.edu

Victor M. Preciado
University of Pennsylvania
preciado@seas.upenn.edu

Alejandro Ribeiro
University of Pennsylvania
aribeiro@seas.upenn.edu

Abstract

Hypergraphs model higher-order interactions in complex systems, e.g., chemicals reacting only in the presence of an enzyme or rumors spreading across groups, and encompass both the notion of an undirected graph and a simplicial complex. Nonetheless, due to computational complexity, machine learning on hypergraph-structured data is notoriously challenging. In an effort to transfer hypergraph neural network models, addressing this challenge, we extend results on the transferability of Graph Neural Networks (GNNs) to design a convolutional architecture for processing signals supported on hypergraphs via GNNs, which we call Hypergraph Expansion Neural Networks (HENNs). Exploiting multiple spectrally-similar graph representations of hypergraphs, we establish bounds on the transferability error. Experimental results illustrate the importance of considering multiple graph representations in HENNs, and show promise of superior performance when transferability is required.

1 Introduction

Machine learning on graph-structured data, as well as other types of data with inherent symmetries, has greatly benefited from the perspective of signal processing. By signal processing, we mean the study of *signals*, which are defined as a mappings from an arbitrary set called the *signal domain* into a vector space whose dimensions are called *channels*. In classical signal processing, the signal domain encodes time (either discrete or continuous) and each channel represents some value that is changing over time (e.g. a voltage in an electrical circuit). However, this need not be the case. Graph Neural Networks (GNNs) are machine learning architectures for processing *graph signals* [10, 39], signals whose domain is either the vertices or edges of a graph (or both), and whose channels correspond to features of the nodes (e.g. user attributes, molecular properties). GNNs have been widely used in practice for problems ranging from text analysis [16] to recommendation systems [53] to control of multi-agent robotic systems [45], among others [50, 54]. While the specific implementation details and philosophies of GNNs vary greatly from model to model, what these architectures have in common is that relationships between entities, the nodes of the graph, are inherently pairwise. However, in both the natural and social sciences, it is reasonable—even necessary—to consider higher-order interactions. Examples of higher-order interactions can be found in the disparate fields of neuroscience [23], biochemistry [30], or physics [7]. In social science, emergent dynamics of group opinions, often, are informed by, not merely interactions between pairs of individuals (edges), but *inter-group* interactions (within groups) involving arbitrarily many individuals (hyperedges) as well as *intra-group* interactions (across groups). In an online setting, in particular, a rumor can be

[*]Equal contribution.

M. Hayhoe et al., Transferable Hypergraph Neural Networks via Spectral Similarity. *Proceedings of the Second Learning on Graphs Conference (LoG 2023)*, PMLR 231, Virtual Event, November 27–30, 2023.

spread through users who are members of multiple online microcommunities, e.g. closed forums, group chats.

Thus, higher-order relationships are represented with greater fidelity by enumerating all (possibly-overlapping) subsets of nodes which form interactions; these subsets are called *hyperedges*. Furthermore, as we will see, while interactions between nodes in the same hyperedge are argued to be linear, interactions between nodes in different hyperedges are nonlinear. Hypergraphs (or higher-order graphs) and simplicial complices are tools for representing these higher-order relationship and have seen use in many applications [42]. Simplicial complexes require the collection of node subsets (called simplices) to be closed under taking subsets [40]. Hence, simplicial complexes are a specialization of hypergraphs. Simplicial complexes, with this seemingly-benign constraint, are endowed with rich geometric and topological structure arising from *combinatorial Hodge theory* [18], a discretization of the theory of differential forms on manifolds [9]. The so-called *Hodge Laplacians* and their eigenvalues can be understood as higher-order spectral analogues of the ordinary graph Laplacian, and have seen success for higher-order graph learning [37, 25, 8, 52, 6, 3]. Unfortunately, these spectral methods do not carry over to generic hypergraphs, stymieing our geometric intuition about these structures. However, generic hypergraphs, which often present themselves in real-world applications, are the structures we wish to establish a learning framework for with transferability properties.

Building on these concepts, we introduce a hypergraph learning framework called *Hypergraph Expansion Neural Networks (HENNs)* by combining graph representations of hypergraphs. No existing approach to hypergraph signal processing, to the best of our knowledge, reconciles the spectral approach of combinatorial Hodge Theory with nonlinear potentials between nodes and hyperedges, while maintaining the ability to process both node and hyperedge signals simultaneously (see Table 3). Our approach is a middle ground between hypergraph learning algorithms that exclusively utilize a nonlinear Laplacian [36] or, equivalently, minimize a potential function [48], and hypergraph learning algorithms exploiting the linear Hodge k-Laplacian, e.g. [37]. The former approach abandons the spectral approach which has been highly successful in convolutional architectures [39]. The later restricts diffusions to those that mix signals (via the boundary and coboundary operators) supported on simplices of one dimension higher or lower. Very recently, however, attention mechanisms based on the Dirac operator [11] have been introduced to mix signals supported on simplicies of non-consecutive orders [6]. Their approach, nonetheless, requires the hypergraph to be a simplicial complex, while ours does not.

The *raison d'être*, we argue, for combining both linear and nonlinear aspects of (hyper)graph signal processing is the ability to reason about the transferability of hypergraph neural networks while, still, taking into account mixtures of signals over hyperedges of various cardinalities. Informally, transferability is a type of generalization property for graph neural networks which says that, given two graphs generated from the same distribution, the graphs should process signals in a similar manner [31]. There have been three main approaches for codifying the similarity of graphs, i.e., that graphs model similar phenomena. The first compares the original graph to a mildly perturbed version [21, 32, 27], although the notions of perturbation and the measure of transferability differ. The second approach assumes a latent space model by which similar graphs are created. For example, the nodes of similar graphs may belong to the same latent measure space with signals sampled accordingly [31], or the graphs themselves may be random and drawn from the same distribution [28]. The third approach assumes similar graphs are obtained from a graphon, which can be understood as a measurable limiting object of graphs as the number of nodes approaches infinity [33]. Transferability bounds using graphons have been explored in both the asymptotic sense [38] and the non-asymptotic sense [35, 39].

While these approaches describe transferability of GNNs across similar graphs, it remains unclear how useful they are in practice. The core concept is that GNNs are spectral operators and, hence, should be transferable across graphs with similar spectra. This *spectral similarity* is achieved as a consequence of the assumptions under which the similar graphs are obtained, generated, or sampled. Unfortunately, given two real graphs of interest, it may not be possible to assert that they are small perturbations of one another, or are sampled from the same latent space or graphon. As such, it is of great practical interest to obtain transferability bounds across *arbitrary* graphs, without any assumptions on their origin. To this end, we provide the first GNN transferability bound directly between two arbitrary graphs of the same size. The key tool in our analysis is to explicitly measure the *spectral similarity* [43, 4] of the graphs for which the transferability bound is desired.

Using our approach, we provide what is, to the best of our knowledge, the first bound on the transferability performance of neural networks that perform convolutions with signals supported on arbitrary hypergraphs. The conceptual leap, herein, is to scrutinize the spectral similarity of, not hypergraphs or simplicial representations of hypergraphs (via Hodge Theory), but *graph representations* of hypergraphs (described in Section 3) that compress the hypergraph in different, but sometimes compositional, ways. While previous results on GNN transferability which make assumptions on graph structure may not apply, by measuring the spectral similarity of the graph representations, even before any training has occurred,[2] we may apply our results.

2 Preliminaries

2.1 Graph Neural Networks

A *graph* is a tuple $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$ with a set of n nodes $\mathcal{V}$, m edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and a real edge weighting function $W : \mathcal{E} \to \mathbb{R}$. We assume throughout that $\mathcal{G}$ is undirected and connected (without loss of generality). Per usual, the graph $\mathcal{G}$ can be represented using a matrix $S \in \mathbb{R}^{n\times n}$ which respects its sparsity pattern, i.e., $[S]_{ij} = 0$ whenever $(i, j) \notin \mathcal{E}$ $(i \neq j)$ with the standard examples being the (un-) normalized adjacency matrix, (un-) normalized graph Laplacian, and the (un-) normalized random walk Laplacian. Since $\mathcal{G}$ is undirected with real edge weights, the matrix representation S is symmetric and, thus, diagonalizable, with an orthonormal eigenbasis $V = [\mathbf{v}_1, \dots, \mathbf{v}_n]$ and eigenvalue matrix $\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_n)$, where the eigenvalues are real and ordered so that $\lambda_1 \leqslant \cdots \leqslant \lambda_n$. By our assumption of connectivity, $\lambda_i = 0$ if $i = 1$ and $\lambda_i > 0$ otherwise. Node signals are data vectors $\mathbf{x} = [x_1, \dots, x_n]^\intercal \in \mathbb{R}^n$ where x_i is associated to node i, although we can readily generalize this setting to signals with multiple channels (i.e. features).

To process a graph signal $\mathbf{x}$, we use a matrix representation S as a *graph shift operator* (GSO) via the linear map $\mathbf{y} = S^k\mathbf{x}$, where the k-fold application of S represent local exchanges of information between nodes connected by a (possibly- self-intersecting) path of length k [39]. With this in mind, we define linear graph filters as polynomials on the GSO below.

Definition 1 (Graph filter)**.** A *graph convolutional filter* $H(S)$ with *filter coefficients* $\{h_k\}_{k=0}^{\infty}$ is defined as

$$H(S) \coloneqq \sum_{k=0}^{\infty} h_k S^k. \tag{1}$$

Moreover, the *graph frequency response* of the filter is

$$h(\lambda) \coloneqq \sum_{k=0}^{\infty} h_k \lambda^k. \tag{2}$$

We often consider filters with an analytic frequency response, i.e., a finite number of coefficients, so that for some K, $h_k = 0\, \forall k > K$. Moreover, we assume $|h(\lambda)| \leqslant 1$ for all $\lambda \in [\lambda_1(S), \lambda_n(S)]$ so that filters do not amplify signals (trivially satisfied via normalization of the GSO).

In order to improve the representation power of graph filters, in practice they are stacked together with pointwise nonlinearities to create a *graph neural network* (GNN), defined below.

Definition 2 (Graph neural network)**.** Graph neural networks are a cascade of L layers of graph filters, each followed by a pointwise nonlinearity. Let each layer have f_l graph signals (or *features*) $\mathbf{x}_l^1, \dots, \mathbf{x}_l^{f_l} \in \mathbb{R}^n$. At layer l, we apply $f_l f_{l-1}$ graph filters of the form $H_l^{ij}(S)$ followed by a pointwise (or elementwise) nonlinear function $\sigma : \mathbb{R} \to \mathbb{R}$ to process the f_{l-1} input features into the f_l output features via

$$\mathbf{x}_l^i = \sigma\left(\sum_{j=1}^{f_{l-1}} \sum_{k=0}^{\infty} h_{lk}^{ij} S^k \mathbf{x}_{l-1}^j\right), \quad i \in \{1, \dots, f_l\}.$$

A graph neural network is the mapping $\Phi(\mathbf{x}_0; S, H) = \mathbf{x}_L$. In this work we consider normalized Lipschitz continuous nonlinearities so that $|\sigma(x) - \sigma(y)| \leqslant |x - y|$, e.g., rectified linear unit (ReLU), sigmoid, $\tanh(\cdot)$.

[2] As we will show, certain design choices for architecture of a GNN will affect its transferability.

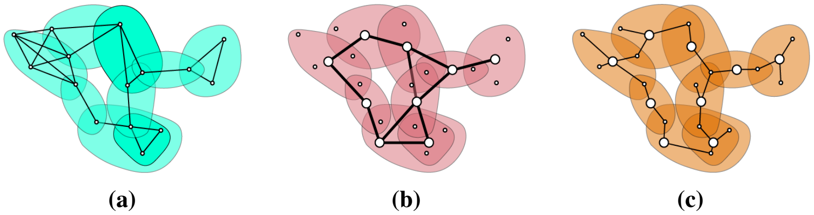
(a) (b) (c)

Figure 1: Hypergraph with graph representations: (a) clique expansion, (b) line graph, and (c) bipartite expansion.

A property of graph filters which has been shown to be valuable for stability of GNNs [21, 39] is the so-called *integral Lipschitz* condition, defined below.

Definition 3 (Integral Lipschitz filter)**.** A filter with frequency response h is *integral Lipschitz* on an interval $\mathcal{I}$ if there is some $C > 0$ such that, for all $\lambda_1, \lambda_2 \in \mathcal{I}$,

$$|h(\lambda_1) - h(\lambda_2)| \leqslant C \frac{|\lambda_1 - \lambda_2|}{|\lambda_1 + \lambda_2|/2}, \tag{3}$$

which implies that the derivative of h satisfies $|\lambda h'(\lambda)| \leqslant C$. We omit the interval $\mathcal{I}$ when it is clear from context, e.g., for filters that will be applied to some GSOs $S_1, \ldots, S_m$ we have $\mathcal{I} = \cup_{i=1}^{m} [\lambda_1(S_i), \lambda_n(S_n)]$.

Integral Lipschitz graph filters can be arbitrarily discriminative for small eigenvalues, but become effectively flat for larger eigenvalues. However, by applying nonlinearities to the output of these filters, the portion of the spectrum containing larger eigenvalues may be scattered to the lower portion. In other words, GNNs with integral Lipschitz filters can be both discriminative and stable [21]. Given some finite GSO S as well as a filter $H(S)$ with an analytic frequency response and support contained in $[\lambda_1(S), \lambda_n(S)]$, the integral Lipschitz condition is trivially satisfied with

$$C = \max \left\{ \left| \sum_{k=1}^{K} h_k k \lambda_1(S)^k \right|, \left| \sum_{k=1}^{K} h_k k \lambda_n(S)^k \right| \right\}. \tag{4}$$

Furthermore, we note that the value of the integral Lipschitz constant in a learning architecture may be affected by adding a penalty term to the loss function used for training.

3 Hypergraph Neural Networks

Table 1: Notation for a hypergraph with n nodes and m hyperedges

Symbol	Meaning
$A \in \mathbb{R}^{n \times n}$	Adjacency matrix
$B \in \mathbb{R}^{n \times m}$	Node-hyperedge incidence matrix
$D_v \in \mathbb{R}^{n \times n}$	Diagonal node degree matrix
$D_e \in \mathbb{R}^{m \times m}$	Diagonal hyperedge size matrix
$D_{ee} \in \mathbb{R}^{m \times m}$	Diagonal hyperedge intersection count matrix
$W \in \mathbb{R}^{m \times m}$	Diagonal hyperedge weight matrix

A *hypergraph*, or higher-order graph, is a tuple $\mathcal{H} = (\mathcal{V}, \mathcal{E}, W)$ with a set of n nodes $\mathcal{V}$ and a collection of m hyperedges $\mathcal{E} \subseteq 2^{\mathcal{V}}$, where $2^{\mathcal{V}}$ is the powerset of $\mathcal{V}$, i.e., the collection of all subsets of $\mathcal{V}$. Thus, the hyperedges are arbitrarily-sized sets of nodes, to which we ascribe some hyperedge weighting function $W : \mathcal{E} \to \mathbb{R}$. With an abuse of notation, we stack the hyperedge weights into a diagonal matrix W. If all hyperedges have cardinality two, we recover the ordinary definition of a graph. Hypergraphs are commonly represented as matrices via the node-hyperedge *incidence matrix* $B \in \mathbb{R}^{n \times m}$, where

$$[B]_{i,e} = \begin{cases} 1, & i \in e \\ 0, & \text{otherwise} \end{cases}, \quad i \in \{1, \ldots, n\}, \quad e \in \{1, \ldots, m\}. \tag{5}$$

The *energy* of a hypergraph node signal x on $\mathcal{H}$ (called the *total variation* in [26]) is defined as

$$Q_{\mathcal{H}}(x) \coloneqq \sum_{e\in\mathcal{E}} W(e) \max_{i,j\in e}(x_i - x_j)^2. \tag{6}$$

The *hypergraph Laplacian* is then defined as the gradient of the energy functional, i.e.,

$$\mathcal{L}_{\mathcal{H}}(x) \coloneqq \frac{1}{2}\nabla Q_{\mathcal{H}}(x). \tag{7}$$

The hypergraph energy (6) is a generalization of the graph energy of a signal [4] defined for a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}_{\mathcal{G}}, W)$ as

$$Q_{\mathcal{G}}(x) = \sum_{(i,j)\in\mathcal{E}_{\mathcal{G}}} (x_i - x_j)^2 = x^{\intercal} B B^{\intercal} x = x^{\intercal} \mathcal{L}_{\mathcal{G}} x, \tag{8}$$

where B is the node-edge incidence matrix, and $\mathcal{L}_{\mathcal{G}}$ is the graph Laplacian. If $\mathcal{H}$ is a graph, the definitions of hypergraph energy and the hypergraph Laplacian coincide with (8) and $\mathcal{L}_{\mathcal{G}}$, respectively. However, in general, $\mathcal{L}_{\mathcal{H}}$ is a nonlinear operator. As such, the resultant diffusions are not easily-approximated with any linear graph Laplacian. Indeed, in contrast to diffusion of graph signals, the hypergraph Laplacian diffuses a node signal $\mathbf{x}$ across both nodes and hyperedges.

A common technique for processing hypergraph signals is to use graph representations [20, 22, 2, 51, 17]. While many representations are used in practice [42], the most common are the clique expansion, line graph, and bipartite expansion (see Figure 1). The *clique expansion* is the graph generated by replacing each hyperedge with k nodes by a k-clique, and the *line graph* is the clique expansion of the *dual hypergraph* (wherein the roles of nodes and hyperedges are reversed). The *bipartite expansion* partitions the nodes and hyperedges, and places edges between them based on hyperedge inclusions in the hypergraph.

In principle, any choice of graph representation induces a signal processing framework on the underlying hypergraph via the GSO on the graph representation. However, in this paper, we narrow our focus on the clique expansion and line graph. To justify our choices, we note that convolution on the bipartite expansion graph is difficult to interpret due to the inherent heterogeneity of the graph node-set, although we admit polynomial filters make sense if we restrict the filter coeficients to be zero for odd powers of the GSO. In contrast, the clique expansion and line graph are homogeneous and of reasonable size; moreover, they are intimately related to the theory of simplicial complices. Indeed, since simplicial complices require closure under taking subsets, we can build a simplicial complex from a hypergraph by taking the original hyperedges and adding any missing subsets of these hyperedges as simplices. The clique expansion is, then, the 1-skeleton of this simplicial representation of the hypergraph, i.e., it includes only the 0-simplices (nodes) and 1-simplices (edges) and throws away all higher-order simplices. The line graph can also be seen as the 1-skeleton of the nerve of the cover of the nodes of the hypergraph by hyperedges [24]. Finally, we remark that the hypergraph Laplacian in (7) is a nonlinear operator that diffuses information across *both* the nodes and hyperedges and, hence, cannot be well-approximated via the linear graph Laplacian of one representation alone.

For these reasons we introduce HENN below, which combines convolutions using the clique expansion and line graph.

Definition 4 (HENN)**.** A *Hypergraph Expansion Neural Network (HENN)*, $\Phi(\cdot; S_c, \Theta_c, S_l, \Theta_l)$ is composed of cascades of clique expansion layers of the form

$$X_v^{l+1} = \sigma\left(\sum_{k=0}^{K} \left(D_v^{-1/2} B W B^{\intercal} D_v^{-1/2}\right)^k X_v^l H_v^{k,l}\right), \tag{9}$$

and line graph layers of the form

$$X_e^{l+1} = \sigma\left(\sum_{k=0}^{K} \left(D_{ee}^{-1/2} B^{\intercal} B W D_{ee}^{-1/2}\right)^k X_e^l H_e^{k,l}\right), \tag{10}$$

where D_{ee} is the hyperedge degree matrix from the line graph. The two GNN modules are composed at the interface between the last clique expansion layer and the first line graph layer via the max-pooling operation $\rho(X)_e = \max_{v\in e} X_v$. We pool between these types of layers using max-pooling according to node-hyperedge membership, i.e., based on the entries of B (see [13] for a similar, but more general pooling framework). Max-pooling was chosen as the aggregation function for node-signals into hyperedge-signals in order to align with the definition of the nonlinear hypergraph Laplacian in (7), thus conditioning the architecture with an appropriate relational inductive bias [5, 46].

3.1 Spectral similarity

We quantify the similarity of connected, undirected graphs with the same number of nodes, including graph representations of hypergraphs, by measuring the similarity of the spectra of their graph shift operators (GSOs), which we assume to be symmetric and positive semi-definite. We stress that a graph admits many different shift operators, such as the normalized graph Laplacian, and a hypergraph admits many graph representations, such as the clique expansion and line graph. While our notion of similarity will be dependent on which graph representations and/or GSOs we consider, this is an appropriate measure for similarity when signals are being processed by these particular GSOs.

Definition 5 (Spectral similarity [43])**.** The symmetric and positive semi-definite matrices $S \in \mathbb{R}^{n\times n}$ and $\tilde{S} \in \mathbb{R}^{n\times n}$ are called *ϵ-spectrally similar* if $(1-\epsilon)S \preceq \tilde{S} \preceq (1+\epsilon)S$, i.e.,

$$(1-\epsilon)x^\intercal Sx \leqslant x^\intercal \tilde{S}x \leqslant (1+\epsilon)x^\intercal Sx \quad \forall x \in \mathbb{R}^n, \tag{11}$$

which implies,

$$(1-\epsilon)\lambda_i(S) \leqslant \lambda_i(\tilde{S}) \leqslant (1+\epsilon)\lambda_i(S) \quad \forall i \in \{1,\dots,n\}. \tag{12}$$

For arbitrary graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$ with n nodes and GSOs S and $\tilde{S}$, respectively, the coefficient ϵ of spectral similarity can be computed to precision κ in time polynomial in n and $\log(1/\kappa)$ [4], for example via semi-definite programming. If the GSOs of interest are diagonally dominant (e.g., graph Laplacians), we may instead employ linear programming to greatly increase scalability [1]. Note also that the multiplicity of the eigenvalue zero must be the same (possibly both zero) for S and $\tilde{S}$, which will be the case for most GSOs of connected graphs, such as the normalized Laplacian.

The coefficient of spectral similarity, ϵ, will be the quantity by which we will provide bounds on the transferability of GNNs between two arbitrary graphs of the same size. This coefficient may *always* be measured between the GSOs of arbitrary graphs with the same number of nodes, however ϵ may, in practice, not be small. However, for a GNN to be transferable we require only that the output is close for similar graphs. Indeed, if the output of a GNN was similar when considering graphs that are not spectrally similar, the GNN would have poor ability to discriminate.

4 Transferability via spectral similarity

To claim that an architecture is transferable, we need to show that using similar graphs to process signals produces similar outputs. In this paper, we will investigate transferability of graph filters, graph neural networks, and hypergraph neural networks that use graph representations. Since these tools are permutation equivariant [21], we need not be concerned with differences in node labelings between the graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$. To that end, given some signal processing architecture Φ along with GSOs S and $\tilde{S}$, we wish to examine the quantity

$$\left\|\Phi(\tilde{S}) - \Phi(S)\right\|_{\mathcal{P}} := \min_{P\in\mathcal{P}} \max_{\mathbf{x}\in\mathbb{R}^n:\|\mathbf{x}\|_2=1} \left\|\Phi(\mathbf{x}; P^\intercal\tilde{S}P) - \Phi(\mathbf{x}; S)\right\|_2, \tag{13}$$

which is referred to as the *distance modulo permutation* [21]. Here, the set of all permutation matrices is denoted $\mathcal{P} := \{P \in \{0,1\}^{n\times n} : P\mathbf{1} = P^\intercal\mathbf{1} = \mathbf{1}\}$. Note that when Φ is linear, we may substitute the operator norm and ignore the unit-norm signal $\mathbf{x}$. If (13) is small, then the way Φ processes signals with S is similar to the way it processes signals using $\tilde{S}$, in a worst-case sense, regardless of any differences in the node labeling. In particular, if (13) is small whenever S and $\tilde{S}$ are ϵ-spectrally similar with small ϵ, then Φ has the transferability property. We formalize this intuition below.

Proposition 1. *For ϵ-spectrally similar symmetric GSOs S and $\tilde{S}$ and an integral Lipschitz filter with constant C, the operator difference modulo permutation between the filters $H(S)$ and $H(\tilde{S})$ satisfies*

$$\left\|H(\tilde{S}) - H(S)\right\|_{\mathcal{P}} \leqslant C\epsilon + \mathcal{O}(\epsilon^2).$$

Moreover, if the filter applies only one shift operation with bias so $H(S) = h_0I + h_1S$, then $\left\|H(\tilde{S}) - H(S)\right\|_{\mathcal{P}} \leqslant C\epsilon$.

Proof. See Appendix B.1. □

As discussed in Section 2.1, GNNs are cascading layers of graph filters passed through pointwise nonlinearities. Thus we arrive at our main result, which is a bound on the transferability of graph neural networks based on spectral similarity.

Theorem 1. *Given ϵ-spectrally similar GSOs S and $\tilde{S}$ and a GNN $\Phi(\cdot; S, \Theta)$ with normalized Lipschitz nonlinearities and L layers with f features, each with filters that have unit operator norm and are C-integral Lipschitz, then*

$$\left\|\Phi(\cdot; S, \Theta) - \Phi(\cdot; \tilde{S}, \Theta)\right\|_{\mathcal{P}} \leqslant CLf^L\epsilon + \mathcal{O}(\epsilon^2).$$

Proof. See Appendix B.2. □

The proof of Theorem 1 relies on the fact that the filter coeficients are integral Lipschitz so that the bound is independent of Θ. Informally, since GNNs are spectral operators, our result says that their actions on any signal $x \in \mathbb{R}^n$ for graphs with similar spectra will be similar. Indeed, with this result in hand, the task of GNN transferability is reduced to measuring the difference of graph spectra. While this bound depends on the design choices of the architecture (such as the number of features and layers), it is largely independent of the learned parameters of the GNN.[3] As a consequence, we may obtain this bound on the transferability error between arbitrary graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$ before any training has taken place. In practice, computing the spectrum of a matrix takes at most $O(n^3)$ time, and the coefficient of spectral similarity can, then, be computed. For large graphs this may be computationally expensive; thus, in Appendices C and D we provide bounds for spectral similarity in many regimes of practical interest. Thus, transferability is entirely characterized by parameters of the architecture (the integral Lipschitz constant, number of features, and number of layers), and the spectral similarity between the graphs of interest. Naturally, lesser spectral similarity results in looser transferability bounds, as does a larger integral Lipschitz constant C and more features f or convolutional layers L. However, larger C, f, and L suggest enhanced discriminability, since the GNN can produce sharper filters as C is larger, and more of those filters can be composed as f and L grow. Together, these insights sugest a trade-off between transferability and discriminability; if a GNN is indifferent to large differences in graph spectra, it cannot also process similar graphs differently. We remark that, in practice, the filters may not be normalized, the integral Lipschitz constants may differ, and each layer may have a different number of features. We explicitly compute our transferability bound in this context in Appendix B.2.

5 Transferability of Hypergraph Neural Networks

In this section we will provide what is, to the best of our knowledge, the first bound on the transferability performance of neural networks that perform convolutions with signals supported on arbitrary hypergraphs. The key herein is to consider graph expansions of hypergraphs (described in Section 3), which may have arbitrary structure; hence, previous results on GNN transferability which make assumptions on graph structure may not apply. In contrast, we may use our results by simply measuring the spectral similarity of the graph expansions of the hypergraphs of interest.

Consider a hypergraph $\mathcal{H}$ and another (similar) hypergraph $\tilde{\mathcal{H}}$. We may consider $\tilde{\mathcal{H}}$ as a perturbation of $\mathcal{H}$ (see Appendix D.1), or it may simply be a related hypergraph that models similar phenomena. Furthermore, consider any hypergraph signal processing framework $\Phi(\cdot; \{S_i, \Theta_i\}_{i=1}^r)$ comprised of graph filtering layers for r graph representations of the original hypergraph. By computing the spectral similarities of these graph representations of $\mathcal{H}$ and $\tilde{\mathcal{H}}$, the result below allows us to understand how similar the output of the hypergraph learning framework $\Phi(\cdot; \{S_i, \Theta_i\}_{i=1}^r)$ will be when applied to the GSOs $\tilde{S}_r, \ldots, \tilde{S}_r$.

Theorem 2. *Consider two hypergraphs $\mathcal{H}$ and $\tilde{\mathcal{H}}$ and r graph representations with GSOs $\{S_i\}_{i=1}^r$ and $\{\tilde{S}_i\}_{i=1}^r$, respectively, such that S_i and $\tilde{S}_i$ are ϵ_i-spectrally similar. If the hypergraph learning framework $\Phi(\cdot; \{S_i, \Theta_i\}_{i=1}^r)$ has a normalizing pooling function between graph representations with normalized Lipschitz nonlinearities, with L_i layers having f_i features for graph representation i,*

[3] The bound is independent of Θ, excepting (possibly) the filters' integral Lipschitz constant C. Indeed, if the filters satisfy $|h(\lambda)| \leqslant 1$ and the integral Lipshitz constant is constrained, the bound is truly independent of the learning parameters. The former may be achieved via normalization during training, and the latter by adding a penalty term to the training loss, such as a log-barrier function on the integral Lipschitz constant, which may be computed via (4).

each with filters that have unit operator norm and are C-integral Lipschitz, then

$$\left\| \Phi(\cdot; \{S_i, \Theta_i\}_{i=1}^r) - \Phi(\cdot; \{\tilde{S}_i, \Theta_i\}_{i=1}^r) \right\| \\ \leqslant \sum_{i=1}^{r} C L_i \epsilon_i \prod_{j=1}^{r} f_j^{L_j} + \mathcal{O}(\epsilon_1^2 + \cdots + \epsilon_r^2).$$

Proof. Similar to Theorem 1 with potentially different GSOs in each layer; see Appendix B.2. □

Since we make no assumptions on the structure of the hypergraphs and, hence, their graph representations beyond connectedness, Theorem 2 provides a transferability bound for *any* hypergraph learning framework which uses graph representations. This result also lends intuition to the formation of HENN. From the connection of the clique expansion and line graph with simplicial complices, we know these graph representations are associated with the higher-order spectral theory of Hodge Laplacians [41, 42]. Together with the results of Appendix D, this suggests that HENN will be stable to structural perturbations in the hypergraph, which is not a statement that can be asserted for the other hypergraph learning methods described earlier. Hence, for two related hypergraphs, HENN will be *both* stable and transferable if the GSOs of their clique expansions and line graphs are spectrally similar. A critical distinction between transferability of HENNs and transferability of GNNs is that, as a sufficient condition, every graph representation of $\mathcal{H}$ needs to be spectrally similar to the corresponding graph representation of $\widetilde{\mathcal{H}}$.

6 Experiments

Table 2: Cross-validated comparison & ablation test.

Model	Validation accuracy	Test accuracy
Clique Expansion	0.631 ± 0.027	0.552 ± 0.016
Line Graph	0.625 ± 0.057	0.588 ± 0.016
HGNN [20]	0.638 ± 0.024	0.580 ± 0.012
HENN (ours)	$\mathbf{0.889 \pm 0.155}$	$\mathbf{0.852 \pm 0.123}$

To evaluate the performance of HENN, we train several graph and hypergraph neural networks to solve a hyperedge source localization problem,[4] a problem of practical interest in discovering the sources of rumors that have propagated across groups. In the source localization problem, the goal is to determine the source hyperedge of a diffusion process that has been occurring over the nodes of a hypergraph for an unknown period of time. Hence, the input is a node signal, but the desired output is a hyperedge signal. This nonlinear diffusion of both node and edge signals occurs via the hypergraph Laplacian (7). To make the problem more difficult, we add both input and measurement noise to the resulting signals, so our problem also involves hypergraph signal de-noising. We perform an ablation test for HENN by comparing HENN with GNNs that use only the clique expansion or line graph of the original hypergraph, as well as a comparison with HGNN [20], showing the improvement in performance of HENN relative to approaches which diffuse the signal across *only* the nodes or hyperedges. Since none of these other approaches can natively handle both node and hyperedge signals, where appropriate, we pool all signals according to hyperedge inclusions, in the same manner as HENN. In particular, we note that all methods (including HGNN) were constrained to have normalized and integral Lipschitz filter weights to match the setting of Theorem 1.

The results of this study are presented in Table 2. Mean and standard deviations of performance metrics were computed after randomly shuffling the data 5 times. Hyperparameters were set based on the highest upper-confidence bound cross-validation score (mean plus standard deviation). HENN exhibits a 45-54$\%$ improvement in test accuracy over GNNs which use only the clique expansion or line graph, illustrating the value of combining these representations for problems that inherently involve a nonlinear diffusion process across both the nodes and hyperedges of a hypergraph. Moreover, it shows a 47% improvement over HGNN when both methods have normalized and integral Lipschitz filter weights, suggesting superior performance when transferability is desired.

[4] Please see Appendix A for more details on experimental setup.

Table 3: Comparison of hypergraph signal processing approaches which use graph representations, including the GSOs used, their connections with higher-order spectral theory, a nonlinear hypergraph Laplacian (7), simplicial complices (SCs), and whether they can process both node and hyperedge signals.

Architecture	GSO(s)	Spectral	$\mathcal{L}_H$	SCs	X_v & X_e
HENN (ours)	$D_v^{-1/2} BWB^\intercal D_v^{-1/2}$ & $D_{ee}^{-1/2} B^\intercal BW D_{ee}^{-1/2}$	✔	✔	✔	✔
HGNN [20]	$D_v^{-1/2} BWD_e^{-1} B^\intercal D_v^{-1/2}$	✔		✔	
HGNN+ [22]	$D_v^{-1} BWD_e^{-1} B^\intercal$	✔			✔
HyperAtten [2]	$D_v^{-1/2} \tilde{B} WD_e^{-1} \tilde{B}^\intercal D_v^{-1/2}$	✔			
HyperGCN [51]	$\tilde{D}_v^{-1/2} \tilde{B} WD_e^{-1} \tilde{B}^\intercal \tilde{D}_v^{-1/2}$		✔		
HNHN [17]	$D_e^{-1} B^\intercal D_v$ & $D_v^{-1} BD_e$	✔			✔
HCoN [49]	$D_v^{-1/2} BWD_e^{-1} B^\intercal D_v^{-1/2}$ & $D_v^{-1/2} BWD_e^{-1/2}$	✔			✔
SHINE [34]	$D_v^{-1/2} BWD_e^{-1} B^\intercal D_v^{-1/2}$	✔			

7 Related Work

There are several existing approaches for learning with hypergraphs via graph convolutions. We summarize their methodologies, benefits, and particular GSOs in our notation in Table 3. HGNN [20] performs convolutions using a hyperedge-normalized and weighted clique expansion of the hypergraph via the incidence matrix. HGNN+ [22] performs a convolution using the weighted and normalized incidence matrices, but the resulting convolution is interpreted as convolving the hyperedge signals via $WD_e^{-1}B^\intercal$ and then a node convolution via $D_v^{-1}B$. This process may include multi-modal nodes and/or hyperedges, in which case the separately learned embeddings are joined together (stacked, pooled, etc.). HNHN [17] uses the incidence matrices as GSOs in a two-phase approach, similarly to HGNN+ with an extra nonlinearity added between the hyperedge and node convolutions. HyperAtten [2] adapts HGNN by adding an attention module to adaptively learn the weights of the incidence matrix $\tilde{B}$, in cases where the hyperedges and nodes are from the same domain (e.g., hyperedges are formed from nearest neighbours of nodes). HyperGCN [51] builds a restricted clique expansion and performs convolutions. Instead of full cliques for each hyperedge, in each epoch, for each graph filtering layer and each hyperedge e it includes only the edge $(i, j) = \arg\max_{i,j \in e} \left\| x_i^l - x_j^l \right\|_2$ between the nodes with the largest pairwise signal difference, following (7). Edges between these two nodes and the other nodes in the hyperedge may also be included using much smaller weights. Note that the resulting GSO will depend on the node signals, of which there may be many in a dataset. HCoN [49] leverages both vertex and hyperedge information, preserves the relationships across them, and incorporates the hypergraph reconstruction error to learn informative latent representations for both vertices and hyperedges. SHINE [34] utilizes a mechanism for hyperedge attention over nodes and node attention over hyperedges and explicitly regularizes with penalty term utilizing the GSO of the clique expansion. The AllSet [13] framework for hypergraph learning utilizes a multi-set function that pools signals supported on nodes (resp. hyperedges) to signals supported on hyperedges (resp. nodes). AllSet learns these functions, assumed to be permutation-equivalent, from scratch with MLPs, while we fix a pooling function and learn filter coefficients of filters for each graph representations.

8 Conclusion

We introduced HENNs as a stitching of two GNN models, one supported on the clique expansion, the other supported on the line graph. As our primary theoretical contribution, we provided transferability bounds given the degree of spectral similarity between arbitrary graph representations as well as design considerations. We hope practitioners can apply our methods to gain computational advantages by training on a spectral sparsifications of graph representations of large hypergraphs, utilizing the transferability guarantees. In the future, we hope to explore transferability of hypergraph neural networks from the perspective of hypergraphons [33], limiting objects of hypergraphs analogus to graphons. We also plan to investigate other ways to compose GNNs supported on different graph representations.

References

[1] Amir Ali Ahmadi and Anirudha Majumdar. Dsos and sdsos optimization: more tractable alternatives to sum of squares and semidefinite optimization. *SIAM Journal on Applied Algebra and Geometry*, 3(2):193–230, 2019. 6

[2] Song Bai, Feihu Zhang, and Philip HS Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021. 5, 9

[3] Sergio Barbarossa and Stefania Sardellitti. Topological signal processing over simplicial complexes. *IEEE Transactions on Signal Processing*, 68:2992–3007, 2020. 2

[4] Joshua Batson, Daniel A Spielman, Nikhil Srivastava, and Shang-Hua Teng. Spectral sparsification of graphs: theory and algorithms. *Communications of the ACM*, 56(8):87–94, 2013. 2, 5, 6

[5] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018. 5

[6] Claudio Battiloro, Lucia Testa, Lorenzo Giusti, Stefania Sardellitti, Paolo Di Lorenzo, and Sergio Barbarossa. Simplicial attention neural networks. *arXiv preprint arXiv:2203.07485*, 2022. 2

[7] Federico Battiston, Enrico Amico, Alain Barrat, Ginestra Bianconi, Guilherme Ferraz de Arruda, Benedetta Franceschiello, Iacopo Iacopini, Sonia Kéfi, Vito Latora, Yamir Moreno, et al. The physics of higher-order interactions in complex systems. *Nature Physics*, 17(10):1093–1098, 2021. 1

[8] Cristian Bodnar, Fabrizio Frasca, Yuguang Wang, Nina Otter, Guido F Montufar, Pietro Lio, and Michael Bronstein. Weisfeiler and Lehman go topological: Message passing simplicial networks. In *International Conference on Machine Learning*, pages 1026–1037. PMLR, 2021. 2

[9] Raoul Bott, Loring W Tu, et al. *Differential forms in algebraic topology*, volume 82. Springer, 1982. 2

[10] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 1

[11] Lucille Calmon, Michael T Schaub, and Ginestra Bianconi. Higher-order signal processing with the dirac operator. In *2022 56th Asilomar Conference on Signals, Systems, and Computers*, pages 925–929. IEEE, 2022. 2

[12] Arijit Chakrabarty, Rajat Subhra Hazra, Frank den Hollander, and Matteo Sfragara. Spectra of adjacency and Laplacian matrices of inhomogeneous Erdős–Rényi random graphs. *Random matrices: Theory and applications*, 10(01):2150009, 2021. 18

[13] Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. In *International Conference on Learning Representations*, 2022. 5, 9

[14] Fan Chung. *Spectral graph theory*. American Mathematical Soc., 1997. 22

[15] Fan Chung, Linyuan Lu, and Van Vu. The spectra of random graphs with given expected degrees. *Internet Mathematics*, 1(3):257–275, 2004. 17, 18, 19

[16] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016. 1

[17] Yihe Dong, Will Sawin, and Yoshua Bengio. Hnhn: Hypergraph networks with hyperedge neurons. *ICML Graph Representation Learning and Beyond Workshop*, 2020. 5, 9

[18] Beno Eckmann. Harmonische funktionen und randwertaufgaben in einem komplex. *Commentarii Mathematici Helvetici*, 17(1):240–255, 1944. 2

[19] László Erdős, Horng-Tzer Yau, and Jun Yin. Rigidity of eigenvalues of generalized Wigner matrices. *Advances in Mathematics*, 229(3):1435–1515, 2012. 18, 19

[20] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3558–3565, 2019. 5, 8, 9

[21] Fernando Gama, Joan Bruna, and Alejandro Ribeiro. Stability properties of graph neural networks. *IEEE Transactions on Signal Processing*, 68:5680–5695, 2020. 2, 4, 6, 14, 20, 23

[22] Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. HGNN$^+$: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 5, 9

[23] Chad Giusti, Robert Ghrist, and Danielle S Bassett. Two's company, three (or more) is a simplex. *Journal of computational neuroscience*, 41(1):1–14, 2016. 1

[24] Branko Grünbaum. Nerves of simplicial complexes. *Aequationes Mathematicae*, 4(1):63–73, 1970. 5

[25] Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Vasileios Maroulas, and Xuanting Cai. Simplicial complex representation learning. In *Machine Learning on Graphs (MLoG) Workshop at 15th ACM International WSDM (2022) Conference*, 2022. 2

[26] Matthias Hein, Simon Setzer, Leonardo Jost, and Syama Sundar Rangapuram. The total variation on hypergraphs-learning on hypergraphs revisited. *Advances in Neural Information Processing Systems*, 26, 2013. 5

[27] Henry Kenlay, Dorina Thanou, and Xiaowen Dong. Interpretable stability bounds for spectral graph filters. In *International conference on machine learning*, pages 5388–5397. PMLR, 2021. 2

[28] Nicolas Keriven, Alberto Bietti, and Samuel Vaiter. Convergence and stability of graph convolutional networks on large random graphs. *Advances in Neural Information Processing Systems*, 33:21512–21523, 2020. 2, 16, 18

[29] Nicolas Keriven, Alberto Bietti, and Samuel Vaiter. On the universality of graph neural networks on large random graphs. *Advances in Neural Information Processing Systems*, 34:6960–6971, 2021. 16, 18

[30] Steffen Klamt, Utz-Uwe Haus, and Fabian Theis. Hypergraphs and cellular networks. *PLoS computational biology*, 5(5):e1000385, 2009. 1

[31] Ron Levie, Wei Huang, Lorenzo Bucci, Michael M. Bronstein, and Gitta Kutyniok. Transferability of spectral graph convolutional neural networks. *J. Mach. Learn. Res.*, 22, 2021. 2

[32] Ron Levie, Elvin Isufi, and Gitta Kutyniok. On the transferability of spectral graph filters. In *2019 13th International conference on Sampling Theory and Applications (SampTA)*, pages 1–5. IEEE, 2019. 2

[33] László Lovász. *Large networks and graph limits*, volume 60. American Mathematical Soc., 2012. 2, 9, 19

[34] Yuan Luo. Shine: Subhypergraph inductive neural network. *Advances in Neural Information Processing Systems*, 35:18779–18792, 2022. 9

[35] Sohir Maskey, Ron Levie, and Gitta Kutyniok. Transferability of graph neural networks: an extended graphon approach. *arXiv preprint arXiv:2109.10096*, 2021. 2, 16, 19

[36] Konstantin Prokopchik, Austin R Benson, and Francesco Tudisco. Nonlinear feature diffusion on hypergraphs. In *International Conference on Machine Learning*, pages 17945–17958. PMLR, 2022. 2

[37] T Mitchell Roddenberry, Nicholas Glaze, and Santiago Segarra. Principled simplicial neural networks for trajectory prediction. In *International Conference on Machine Learning*, pages 9020–9029. PMLR, 2021. 2

[38] Luana Ruiz, Luiz Chamon, and Alejandro Ribeiro. Graphon neural networks and the transferability of graph neural networks. *Advances in Neural Information Processing Systems*, 33:1702–1712, 2020. 2, 16, 18, 19

[39] Luana Ruiz, Fernando Gama, and Alejandro Ribeiro. Graph neural networks: architectures, stability, and transferability. *Proceedings of the IEEE*, 109(5):660–682, 2021. 1, 2, 3, 4

[40] Vsevolod Salnikov, Daniele Cassese, and Renaud Lambiotte. Simplicial complexes and complex systems. *European Journal of Physics*, 40(1):014001, 2018. 2

[41] Michael T Schaub, Austin R Benson, Paul Horn, Gabor Lippner, and Ali Jadbabaie. Random walks on simplicial complexes and the normalized Hodge 1-Laplacian. *SIAM Review*, 62(2):353–391, 2020. 8

[42] Michael T Schaub, Yu Zhu, Jean-Baptiste Seby, T Mitchell Roddenberry, and Santiago Segarra. Signal processing on higher-order networks: Livin'on the edge... and beyond. *Signal Processing*, 187:108149, 2021. 2, 5, 8

[43] Daniel A Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011. 2, 6

[44] Terence Tao. *Topics in random matrix theory*, volume 132. American Mathematical Soc., 2012. 18, 19, 20, 23

[45] Ekaterina Tolstaya, Fernando Gama, James Paulos, George Pappas, Vijay Kumar, and Alejandro Ribeiro. Learning decentralized controllers for robot swarms with graph neural networks. In *Conference on robot learning*, pages 671–682. PMLR, 2020. 1

[46] Petar Veličković and Charles Blundell. Neural algorithmic reasoning. *Patterns*, 2(7), 2021. 5

[47] Renato Vizuete, Federica Garin, and Paolo Frasca. The Laplacian spectrum of large graphs sampled from graphons. *IEEE Transactions on Network Science and Engineering*, 8(2):1711–1721, 2021. 19

[48] Peihao Wang, Shenghao Yang, Yunyu Liu, Zhangyang Wang, and Pan Li. Equivariant hypergraph diffusion neural operators. In *The Eleventh International Conference on Learning Representations*, 2022. 2

[49] Hanrui Wu, Yuguang Yan, and Michael Kwok-Po Ng. Hypergraph collaborative network on vertices and hyperedges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3245–3258, 2023. 9

[50] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020. 1

[51] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. HyperGCN: A new method for training graph convolutional networks on hypergraphs. *Advances in neural information processing systems*, 32, 2019. 5, 9

[52] Maosheng Yang, Elvin Isufi, and Geert Leus. Simplicial convolutional neural networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8847–8851. IEEE, 2022. 2

[53] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983, 2018. 1

[54] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020. 1

[55] Yizhe Zhu. A graphon approach to limiting spectral distributions of Wigner-type matrices. *Random Structures & Algorithms*, 56(1):251–279, 2020. 18

[56] Afra Zomorodian. Fast construction of the Vietoris-Rips complex. *Computers & Graphics*, 34(3):263–271, 2010. 13

Author Contributions

Mikhail Hayhoe contributed in writing the original draft, writing proofs, visualization, methodology, software, and validation. Hans Riess contributed in conceptualization, review and editing, project administration, methodology, and visualization. Michael M. Zavlanos contributed in funding acquisition. Victor M. Preciado contributed in supervision and funding acquisition. Alejandro Ribeiro contributed in conceptualization, supervision, and funding acquisition.

Acknowledgements

This work is supported in part by AFOSR under award #FA9550-19-1- 0169 and by ONR under agreement #N00014-18-1-2374.

A Experimental setup

The hypergraph is created by randomly sampling 500 points on a 3-dimensional torus with inner radius 1 and outer radius 2, and keeping maximal hyperedges of a Vietoris-Rips complex [56] with radius 0.4, i.e., constructing simplices between any points jointly within a 2-norm distance 0.4. At random, 10 hyperedges are selected to be the sources, making the source localization problem a 10-class classification problem.

To generate K data samples $\{(\mathbf{x}_k, y_k)\}_{k=1}^K$, we pick 10 hyperedges at random to be the sources. For each source hyperedge i, we construct a node source signal for node j:

$$x_{0,j}^i = \begin{cases} 1+z, & \text{node } i \text{ is in the source hyperedge,} \\ z, & \text{otherwise.} \end{cases}$$

where $z \sim \text{Normal}(0, 0.01)$ is independent Gaussian white noise. This noisy source signal is diffused for $t_{max} = 30$ time steps via the nonlinear hypergraph Laplacian (7) to generate the sequence $\{\mathbf{x}_t^i\}_{t=0}^{t_{max}}$. Samples are then of the form $(\mathbf{x}_t^i + \mathbf{z}, y_i)$, where we pick $\mathbf{x}_t^i$ by randomly sampling a source hyperedge i and time t and add measurement noise $\mathbf{z} \sim \text{Normal}(\mathbf{0}, 0.01 I_n)$, and the hyperedge label is $y_i = i$. We use 500 of these signals for training (including cross-validation), and 300 of these signals for testing. Note that while we may sample a signal from the same source at the same time more than once, they will not be identical due to the added independent measurement noise.

All GNNs have two graph filtering layers with one input feature per node and a fixed readout layer to select signals from the 10 candidate source hyperedges. The filters are normalized during training to ensure $|h(\lambda)| \leqslant 1$ and the integral Lipschitz constant is constrained to be less than 10 via a loss penalty term. Pooling from node to hyperedge signals is done based on node inclusion in the hyperedges. We train using `adam` with weights 0.9 and 0.999 with a learning rate of 0.0005, decay rate of 0.99 and decay period of 20. These hyperparameters, as well as others such as the number of filters and number of filter taps were chosen via 5-fold cross-validation.

B Proofs of main results

B.1 Transferability of graph filters

Proof of Proposition 1. By spectral similarity, we have that

$$(1-\epsilon)S \preceq \tilde{S} \preceq (1+\epsilon)S \tag{14}$$

$$\Rightarrow\ (1-\epsilon)\mathbf{x}^\intercal S\mathbf{x} \leqslant \mathbf{x}^\intercal \tilde{S}\mathbf{x} \leqslant (1+\epsilon)\mathbf{x}^\intercal S\mathbf{x} \quad \forall \mathbf{x} \in \mathbb{R}^n. \tag{15}$$

Hence,

$$\begin{aligned} H(\tilde{S}) - H(S) &= \sum_{k=0}^{\infty} h_k(\tilde{S}^k - S^k) = \sum_{k=1}^{\infty} h_k(\tilde{S}^k - S^k) \\ &\preceq \sum_{k=1}^{\infty} h_k(((1+\epsilon)S)^k - S^k). \end{aligned}$$

For symmetric matrices $\|A\|_{op} \leqslant \|B\|_{op}$ if $A \preceq B$. Thus, using the first-order expansion $((1+\epsilon)S)^k = (1+k\epsilon)S^k + O(\epsilon^2)$, since graph filters are permutation equivariant we have

$$\begin{aligned} \left\|H(\tilde{S}) - H(S)\right\|_{\mathcal{P}} &= \left\|H(\tilde{S}) - H(S)\right\|_{op} \\ &\leqslant \left\|\sum_{k=1}^{\infty} h_k(((1+\epsilon)S)^k - S^k)\right\|_{op} \end{aligned}$$

$$= \left\| \sum_{k=1}^{\infty} h_k((1+k\epsilon)S^k - S^k) + O(\epsilon^2) \right\|_{op}$$
$$\leqslant \left\| \sum_{k=1}^{\infty} h_k k\epsilon S^k \right\|_{op} + \mathcal{O}(\epsilon^2).$$

Next, notice that the derivative of the frequency response is $\tilde{h}'(\lambda) = \sum_{k=1}^{\infty} h_k k \lambda^{k-1}$. Thus, using the graph Fourier representation of $\mathbf{x}$,

$$\begin{aligned}\sum_{k=1}^{\infty} h_k k\epsilon S^k \mathbf{x} &= \epsilon \sum_{k=1}^{\infty} h_k k S^k \left(\sum_{i=1}^{n} \tilde{x}_i \mathbf{v}_i \right) \\ &= \epsilon \sum_{i=1}^{n} \tilde{x}_i \sum_{k=1}^{\infty} h_k k S^k \mathbf{v}_i \\ &= \epsilon \sum_{i=1}^{n} \tilde{x}_i \sum_{k=1}^{\infty} h_k k \lambda_i^k \mathbf{v}_i \\ &= \epsilon \sum_{i=1}^{n} \tilde{x}_i \lambda_i \tilde{h}'(\lambda_i) \mathbf{v}_i.\end{aligned}$$

By the integral Lipschitz assumption of the filter, $\lambda \tilde{h}'(\lambda) \leqslant C$. Moreover, since the signal is assumed to have a unit norm, $\|\mathbf{x}\|_2 = \|\tilde{\mathbf{x}}\|_2 = 1$. Thus, by orthonormality of the $\mathbf{v}_i$,

$$\begin{aligned}\left\| \sum_{k=1}^{\infty} h_k k\epsilon S^k \mathbf{x} \right\|_2^2 &= \left\| \epsilon \sum_{i=1}^{n} \tilde{x}_i \lambda_i \tilde{h}'(\lambda_i) \mathbf{v}_i \right\|_2^2 \\ &= \epsilon^2 \sum_{i=1}^{n} (\tilde{x}_i \lambda_i \tilde{h}'(\lambda_i))^2 \\ &\leqslant (C\epsilon)^2 \sum_{i=1}^{n} \tilde{x}_i^2 = (C\epsilon)^2.\end{aligned}$$

Finally, we thus have

$$\left\| H(\tilde{S}) - H(S) \right\|_{\mathcal{P}} \leqslant \left\| \sum_{k=1}^{\infty} h_k k\epsilon S^k \right\|_{op} + \mathcal{O}(\epsilon^2) \leqslant C\epsilon + \mathcal{O}(\epsilon^2),$$

as desired. Moreover, note that if we apply only one shift operation with a bias term, i.e., $H(S) = h_0 I + h_1 S$, then the first-order expansion performed earlier is exact and $\left\| H(\tilde{S}) - H(S) \right\|_{\mathcal{P}} \leqslant C\epsilon$. □

B.2 Transferability of graph neural networks (GNNs)

Proof of Theorem 1. We wish to bound the quantity $||\Phi(\cdot; S, \Theta) - \Phi(\cdot; \tilde{S}, \Theta)||_{\mathcal{P}}$, and proceed similarly to [21, Theorem 4]. At an arbitrary layer $l \in \{1, \ldots, L\}$ of $\Phi(\cdot; S, \Theta)$ with f_l features (where $l = L$ is the output layer), the (possibly hidden) node signals are some $\{\mathbf{x}_l^f\}_{f=1}^{f_l}$, where $\mathbf{x}_l^f \in \mathbb{R}^n$. Similarly, the node signals in the GNN $\Phi(\cdot; \tilde{S}, \Theta)$ at layer l are denoted by $\{\tilde{\mathbf{x}}_l^f\}_{f=1}^{f_l}$. We thus wish to bound quantities of the form $||\mathbf{x}_l^f - \tilde{\mathbf{x}}_l^f||_2$, i.e.,

$$\left\| \sigma\left(\sum_{g=1}^{f_{l-1}} H_l^{fg}(S) \mathbf{x}_{l-1}^g \right) - \sigma\left(\sum_{g=1}^{f_{l-1}} H_l^{fg}(\tilde{S}) \tilde{\mathbf{x}}_{l-1}^g \right) \right\|_2 . \tag{16}$$

Since the nonlinearities are assumed to be normalized Lipschitz, $|\sigma(x) - \sigma(y)| \leqslant |x - y|$. So, by the triangle inequality,

$$\begin{aligned}
&\left\|\mathbf{x}_l^f - \tilde{\mathbf{x}}_l^f\right\|_2 \\
&\leqslant \sum_{g=1}^{f_{l-1}} \left\|H_l^{fg}(S)\mathbf{x}_{l-1}^g - H_l^{fg}(\tilde{S})\tilde{\mathbf{x}}_{l-1}^g\right\|_2 \\
&= \sum_{g=1}^{f_{l-1}} \left\|H_l^{fg}(S)\big(\mathbf{x}_{l-1}^g - \tilde{\mathbf{x}}_{l-1}^g\big) + \Big(H_l^{fg}(S) - H_l^{fg}(\tilde{S})\Big)\tilde{\mathbf{x}}_{l-1}^g\right\|_2 \\
&\leqslant \sum_{g=1}^{f_{l-1}} \left\|H_l^{fg}(S)\right\|_{op} \left\|\mathbf{x}_{l-1}^g - \tilde{\mathbf{x}}_{l-1}^g\right\|_2 + \left\|H_l^{fg}(S) - H_l^{fg}(\tilde{S})\right\|_{op} \left\|\tilde{\mathbf{x}}_{l-1}^g\right\|_2.
\end{aligned}$$

Notice similarly that

$$\begin{aligned}
\left\|\mathbf{x}_l^{g_l}\right\|_2 &= \left\|\sigma\left(\sum_{g_{l-1}=1}^{f_{l-1}} H_l^{g_l g_{l-1}}(S)\mathbf{x}_{l-1}^{g_{l-1}}\right)\right\|_2 \\
&\leqslant \sum_{g_{l-1}=1}^{f_{l-1}} \left\|H_l^{g_l g_{l-1}}(S)\mathbf{x}_{l-1}^{g_{l-1}}\right\|_2 \\
&\leqslant \sum_{g_{l-1}=1}^{f_{l-1}} \left\|H_l^{g_l g_{l-1}}(S)\right\|_{op} \left\|\mathbf{x}_{l-1}^{g_{l-1}}\right\|_2 \\
&\leqslant \sum_{g_{l-1}=1}^{f_{l-1}} \cdots \sum_{g_0=1}^{f_0} \left\|\mathbf{x}_0^{g_0}\right\|_2 \prod_{s=1}^{l} \left\|H_s^{g_s g_{s-1}}(S)\right\|_{op}.
\end{aligned}$$

Since the GSOs S and $\tilde{S}$ are ϵ-spectrally similar, by Proposition 1 we have $||H_1^{g_l g_{l-1}}(S) - H_1^{g_l g_{l-1}}(\tilde{S})||_{op} \leqslant C_l^{g_l g_{l-1}}\epsilon + \mathcal{O}(\epsilon^2)$ for all g_l, g_{l-1}, where $C_l^{g_l g_{l-1}}$ is the integral Lipschitz constant of the filter in the l-th layer for the g_{l-1}-th input feature and g_l-th output feature. Combining these results,

$$\begin{aligned}
&\left\|\mathbf{x}_l^{g_l} - \tilde{\mathbf{x}}_l^{g_l}\right\|_2 \\
&\leqslant \sum_{g_{l-1}=1}^{f_{l-1}} \left\|H_l^{g_l g_{l-1}}(S)\right\|_{op} \left\|\mathbf{x}_{l-1}^{g_{l-1}} - \tilde{\mathbf{x}}_{l-1}^{g_{l-1}}\right\|_2 \\
&\qquad + \left(C_l^{g_l g_{l-1}}\epsilon + \mathcal{O}(\epsilon^2)\right) \left\|\tilde{\mathbf{x}}_{l-1}^g\right\|_2 \\
&\leqslant \mathcal{O}(\epsilon^2) + \sum_{g_{l-1}=1}^{f_{l-1}} \left\|H_l^{g_l g_{l-1}}(S)\right\|_{op} \left\|\mathbf{x}_{l-1}^{g_{l-1}} - \tilde{\mathbf{x}}_{l-1}^{g_{l-1}}\right\|_2 \\
&\qquad + \sum_{g_{l-2}=1}^{f_{l-2}} \cdots \sum_{g_0=1}^{f_0} \left\|\tilde{\mathbf{x}}_0^{g_0}\right\|_2 C_l^{g_l g_{l-1}}\epsilon \prod_{s=1}^{l-1} \left\|H_s^{g_s g_{s-1}}(S)\right\|_{op}.
\end{aligned}$$

This establishes a recurrence relation for $\left\|\mathbf{x}_l^{g_l} - \tilde{\mathbf{x}}_l^{g_l}\right\|_2$. Notice that for $l = 1$ and any $g_1 \in \{1, \ldots, f_1\}$,

$$\left\|\mathbf{x}_1^{g_1} - \tilde{\mathbf{x}}_1^{g_1}\right\|_2 \leqslant \sum_{g_0=1}^{f_0} C_1^{g_1 g_0}\epsilon \left\|\mathbf{x}_0^{g_0}\right\|_2 + \mathcal{O}(\epsilon^2),$$

since the input signals to both GNNs are the same and, hence, $\|\mathbf{x}_0^{g_0}\|_2 = \|\tilde{\mathbf{x}}_0^{g_0}\|_2$ and $\|\mathbf{x}_0^{g_0} - \tilde{\mathbf{x}}_0^{g_0}\|_2 = 0$ for all $g_0 \in \{1, \ldots, f_0\}$. Thus we may solve the recurrence relation above to obtain

$$\begin{aligned} &\|\mathbf{x}_l^{g_l} - \tilde{\mathbf{x}}_l^{g_l}\|_2 \\ &\leqslant \sum_{g_{l-1}=1}^{f_{l-1}} \cdots \sum_{g_0=1}^{f_0} \|\mathbf{x}_0^{g_0}\|_2 \sum_{s=1}^{l} C_s^{g_s g_{s-1}} \epsilon \prod_{\substack{t=1\\t\neq s}}^{l} \left\|H_t^{g_t g_{t-1}}(S)\right\|_{op} + \mathcal{O}(\epsilon^2). \end{aligned}$$

This inequality makes explicit the effects of the integral Lipschitz constants $C_l^{g_l g_{l-1}}$ as well as the operator norms $\left\|H_t^{g_t g_{t-1}}(S)\right\|_{op}$ of each filter, which are functions of both the (learned) filter coefficients and the spectrum of the GSO S used for training. Both quantities control the amplification of the input signals; if they are smaller, the magnitude of the difference in the signals at layer l is smaller. The output signals of the GNNs are the quantities $\{\mathbf{x}_L^f\}_{f=1}^{f_L}$ and $\{\tilde{\mathbf{x}}_L^f\}_{f=1}^{f_L}$. Hence, since we wish to bound the difference in the output of the GNNs, for an arbitrary input signal $\mathbf{x}_0 = \{\mathbf{x}_0^f\}_{f=1}^{f_0}$, we have

$$\begin{aligned} &\left\|\Phi(\mathbf{x}_0; S, \Theta) - \Phi(\mathbf{x}_0; \tilde{S}, \Theta)\right\|_2 \\ &= \sum_{g_L=0}^{f_L} \|\mathbf{x}_L^{g_L} - \tilde{\mathbf{x}}_L^{g_L}\|_2 \\ &\leqslant \sum_{g_L=0}^{f_L} \cdots \sum_{g_0=1}^{f_0} \|\mathbf{x}_0^{g_0}\|_2 \sum_{s=1}^{L} C_s^{g_s g_{s-1}} \epsilon \prod_{t=1, t\neq s}^{L} \left\|H_t^{g_t g_{t-1}}(S)\right\|_{op} \\ &\quad + \mathcal{O}(\epsilon^2). \end{aligned}$$

If the filter responses are bounded, i.e., $\left\|H_t^{g_t g_{t-1}}(S)\right\|_{op} \leqslant c$ (which can be achieved via normalization during training), all filters share an integral Lipschitz constant C (which is possible by adding a penalty to the training loss), and the input signals have unit norm (via normalization before training), then

$$\left\|\Phi(\mathbf{x}_0; S, \Theta) - \Phi(\mathbf{x}_0; \tilde{S}, \Theta)\right\|_2 \leqslant CLc^{L-1} \prod_{s=0}^{L} f_s \epsilon + \mathcal{O}(\epsilon^2).$$

The constant c can be understood as controlling the amplification or contraction of the input and hidden signals, while the number of features f_s and layers L determine how many filters are stacked together, each obeying the bound from Proposition 1, which depends on the integral Lipschitz constant C and the spectral similarity coefficient ϵ. The desired result is achieved by setting $c = 1$ and $f_s = f$ for all $s \in \{0, \ldots, L\}$. □

C Transferability between random graphs

In this section we show that two graphs of the same size drawn from an appropriate random graph distribution will be spectrally similar with a coefficient ϵ that shrinks as n grows. In other words, random graphs become *more* spectrally similar as they grow larger. By combining this notion with Theorem 1, we obtain results in agreement with works that investigate the transferability of GNNs applied to large random graphs [28, 29] and convergent graph sequences[35, 38].

For practicality and to build intuition throughout this section we will assume that all graph shift operators are normalized Laplacians. However, we stress that this is *not* a requirement of our approach, and any GSO that satisfies the conditions we provide can be used in practice. We begin with a general result which provides the conditions under which a family of random graphs will produce GSOs that grow more spectrally similar as they grow larger, with high probability.

Proposition 2. *Let $\{\mathcal{G}_n\}_{n=1}^{\infty}$ and $\{\tilde{\mathcal{G}}_n\}_{n=1}^{\infty}$ be two independent sequences of graphs drawn from the same family of random graph distributions so that $\mathcal{G}_n$, $\tilde{\mathcal{G}}_n \sim P_{\mathcal{G}}(n)$ for each n, with graph shift operators S_n and $\tilde{S}_n$, respectively, having eigenvalues $\{\lambda_i\}_{i=1}^{n}$ and $\{\tilde{\lambda}_i\}_{i=1}^{n}$. Assume the following:*

(A1) *Multiplicity of zero: the zero eigenvalue has almost surely constant multiplicity indepedent of n (potentially zero);*

(A2) *Bounded spectral gap: there exists $c > 0$ independent of n such that $|\lambda_i| \geqslant c$ almost surely for all non-zero eigenvalues;*

(A3) *Concentration: given $\epsilon_c > 0$ and $\delta_c > 0$ there exist some values γ_i, $i \in \{1 \dots, n\}$, such that for large enough n,*

$$P\left(|\lambda_i - \gamma_i| < \epsilon_c,\ \forall i \in \{1, \dots, n\}\right) > 1 - \delta_c. \tag{17}$$

Then for any $\epsilon > 0$ and $\delta > 0$, there exists $N = N(\epsilon, \delta)$ such that for any $n \geqslant N$,

$$P\Big((1-\epsilon)\lambda_i < \tilde{\lambda}_i < (1+\epsilon)\lambda_i,\ \forall i \in \{1, \dots, n\}\Big) > 1-\delta. \tag{18}$$

In other words, for appropriate graph shift operators of large random graphs whose eigenvalues concentrate, the coefficient of spectral similarity converges in probability to 0 as $n \to \infty$.

Proof. For ease of notation, fix some n and set $\lambda_i := \lambda_i(S_n)$ and $\tilde{\lambda}_i := \lambda_i(\tilde{S}_n)$. Hence, denoting $\mathcal{I} := \{i \in \{1, \dots, n\} \ :\ \lambda_i(S_n) \neq 0\}$, note that *(A1)* implies almost surely that we can order the eigenvalues of $\tilde{S}_n$ so that $\mathcal{I} = \Big\{ i \in \{1, \dots, n\} \ :\ \lambda_i(\tilde{S}_n) \neq 0 \Big\}$ and, thus, $|\lambda_i - \tilde{\lambda}_i| = 0$ for all $i \notin \mathcal{I}$. Thus, it suffices to show that $\forall \epsilon > 0, \delta > 0$ there exists some N such that for any $n \geqslant N$,

$$\begin{aligned} &P\left((1-\epsilon)\lambda_i \leqslant \tilde{\lambda}_i \leqslant (1+\epsilon)\lambda_i\ \forall i \in \mathcal{I}\right) > 1 - \delta \\ \Leftrightarrow\ &P\left(|\lambda_i - \tilde{\lambda}_i| \leqslant \epsilon|\lambda_i|\ \forall i \in \mathcal{I}\right) > 1 - \delta. \end{aligned} \tag{19}$$

By *(A2)*, the non-trivial eigenvalues of S_n and $\tilde{S}_n$ are bounded away from zero by some $c > 0$ almost surely. Thus, since $P(A) > P(B)$ if $A \supseteq B$,

$$\begin{aligned} &P\left(|\lambda_i - \tilde{\lambda}_i| \leqslant \epsilon|\lambda_i|\ \forall i \in \mathcal{I}\right) \\ &\geqslant P\left(|\lambda_i - \tilde{\lambda}_i| \leqslant c\epsilon\ \forall i \in \mathcal{I}\right) \\ &= P\left(|\lambda_i - \gamma_i - (\tilde{\lambda}_i - \gamma_i)| \leqslant c\epsilon\ \forall i \in \mathcal{I}\right) \\ &\geqslant P\left(|\lambda_i - \gamma_i| + |\tilde{\lambda}_i - \gamma_i| \leqslant c\epsilon\ \forall i \in \mathcal{I}\right) \\ &\geqslant P\Big(|\lambda_i - \gamma_i| \leqslant c\epsilon/2\ \forall i \in \mathcal{I}\Big) P\Big(|\tilde{\lambda}_i - \gamma_i| \leqslant c\epsilon/2\ \forall i \in \mathcal{I}\Big), \end{aligned}$$

where the last line follows by independence. Then by *(A3)*, for any $\epsilon > 0$, $\delta > 0$ we can choose N_1 large enough such that $P(|\lambda_i - \gamma_i| \leqslant c\epsilon/2\ \forall i \in \mathcal{I}) > \sqrt{1-\delta}$ (and similarly choose N_2 for all $\tilde{\lambda}_i$), and the result follows by picking $N = \max\{N_1, N_2\}$. □

This proposition applies to a large class of random graphs and corresponding shift operators. Assumption *(A1)* is necessary for spectral similarity to hold but is trivially satisfied by the normalized Laplacian of a connected graph. As we will show in Appendix C.1, *(A2)* and *(A3)* are satisfied by the normalized Laplacian of many families of random graphs including the Erdös-Rényi and Chung-Lu models, as well as random power-law graphs with given expected degree sequences and large enough minimum expected degree [15]. Furthermore, we will show in Appendix C.2 that graphs sampled from the same well-structured graphon will satisfy all of these conditions and, hence, become arbitrarily spectrally similar as their size grows larger. The implications of these results in terms of transferability of GNNs for random graph families is summarized below.

Theorem 3. *Let $\{\mathcal{G}_n\}_{n=1}^{\infty}$ and $\{\tilde{\mathcal{G}}_n\}_{n=1}^{\infty}$ be two independent sequences of random graphs with graph shift operators $\{S_n\}_{n=1}^{\infty}$ and $\{\tilde{S}_n\}_{n=1}^{\infty}$, respectively, such that $\mathcal{G}_n, \tilde{\mathcal{G}}_n \sim P_{\mathcal{G}}(n)$, and let $P_{\mathcal{G}}(n)$ satisfy (A1)-(A3) for all n. Then, for a sequence of GNNs $\{\Phi(\cdot; S_n, \Theta)\}_{n=1}^{\infty}$ trained on the GSOs S_n with normalized Lipschitz nonlinearities, each with the same number of layers and features, as well as integral Lipschitz filters that have unit operator norm,*

$$\left\| \Phi(\cdot; S_n, \Theta) - \Phi(\cdot; \tilde{S}_n, \Theta) \right\|_{\mathcal{P}} \to 0,$$

in probability as $n \to \infty$.

Proof. Direct result of Theorem 1 and Proposition 2. □

The statement above has far-reaching implications. For example, GNNs which use the normalized Laplacian as a GSO will exhibit *better* transferability properties as the size of the original training graph grows, with high probability. This agrees with previous results on GNN transferability in the regime of large random graphs [28, 29], as well as the regime of convergent random graph sequences [38]. In particular, these results suggests that for a family of large enough random graphs, when using the normalized Laplacian as a GSO, it suffices to train on one graph realization to achieve comparable performance across all realizations of the same size. However, Theorem 3 also presents an issue of discriminability; with high probability, the action of a GNN trained on a large random graph will look very similar to the action of that same GNN applied to a different random graph of the same family. While this may appear to be an artifact of the choice to consider normalized GSOs, we remark that these are used almost exclusively in practice for large graphs. Indeed, without normalization the graph frequency response of a filter can become unbounded as n grows.

C.1 Transferability of large random graphs

To make the results in Proposition 2 and Theorem 3 concrete, in this section we will explore specific families of random graphs that satisfy (*A1*)-(*A3*). To this end, and in order to motivate the results pertaining to concentration of eigenvalues, we will introduce distributions on the spectra of random graphs.

Definition 6 (Empirical spectral distribution)**.** The *empirical spectral distribution* (ESD) of a real symmetric matrix X, μ_X, is the atomic distribution that assigns equal mass to each of the eigenvalues of X, i.e.,

$$\mu_X \coloneqq \frac{1}{n}\sum_{i=1}^{n} \delta_{\lambda_i(X)}. \tag{20}$$

Since we wish to study the spectra of random graphs, in our setting the ESD will be a *random measure*, i.e., a random variable in the space of probability measures on $\mathbb{R}$ (see [44] for more details). We will also study a common limiting distribution of the ESD of random matrices, called Wigner's semicircle law.

Definition 7 (Wigner's semicircle law)**.** *Wigner's semicircle law* μ is the distribution with density defined on $[-1, 1]$ as

$$\mu(x) \coloneqq \frac{2}{\pi}\sqrt{1-x^2}, \tag{21}$$

and zero otherwise.

Wigner's semicircle law is to random matrices what the central limit theorem is to random variables. A result of note, called Wigner's eigenvalue rigidity theorem [19, Theorem 2.2], states that the eigenvalues of a matrix whose ESD converges to the semicricle law concentrate around specific distinct values, i.e., (*A3*) is satisfied. The exact characterization of the general class of matrices whose empirical spectral distributions converge to Wigner's semicircle law is beyond the scope of this paper (see, e.g, [19, 55]). However, many graphs of interest have corresponding shift operators that satisfy these conditions, including the Erdös-Rényi and Chung-Lu models, as well as random power law graphs with large enough minimum degree. Indeed, in [15, Theorem 6] it was shown that the eigenvalues of the normalized Laplacian of any random graph with an appropriate given expected degree distribution converge to Wigner's semicircle law. Moreover, [15, Theorem 5] provides bounds which may be used to show (*A2*) is satisfied when the minimum degree is large enough. Furthermore, in [12] it was shown that inhomogeneous Erdös-Rényi random graphs (where the connection probabilities may all be different) admit limiting distributions that are related to Wigner's semicircle law, although they may be difficult to compute in general. The implications of these results in the context of transferability of GNNs are summarized below.

Corollary 1. *Let $\{\mathcal{G}_n\}_{n=1}^{\infty}$ and $\{\tilde{\mathcal{G}}_n\}_{n=1}^{\infty}$ be two independent sequences of almost surely connected random graphs drawn from the same distribution with given expected degrees, with normalized Laplacians $\{S_n\}_{n=1}^{\infty}$ and $\{\tilde{S}_n\}_{n=1}^{\infty}$, respectively. If the minimum and average degrees are large enough, then for a sequence of GNNs $\{\Phi(\cdot; S_n, \Theta)\}_{n=1}^{\infty}$ following the conditions of Theorem 3,*

$$\left\|\Phi(\cdot; S_n, \Theta) - \Phi(\cdot; \tilde{S}_n, \Theta)\right\|_{\mathcal{P}} \to 0,$$

in probability as $n \to \infty$.

Proof. Result of [15, Theorems 5 and 6], [19, Theorem 2.2] and Theorem 3. □

It may be possible to achieve almost sure convergence above, depending on the family of random graphs under consideration [44, 19]. While this result is asymptotic in nature, it is possible to obtain convergence rates which may be informative for finite n. Concretely, in the context of Proposition 2, the eigenvalue rigidity result in [19, Theorem 2.2] suggests that for some fixed n, $\epsilon \approx O(n^{-1})$ and $\delta \approx O(n^{-c})$ for some small c. In other words, GNNs trained on GSOs of size n satisfying the conditions of Corollary 1 will have transferability bounds of the order $1/n$ with high probability.

We remark that these results also apply to the transferability of large random hypergraphs. Consider a random hypergraph with n nodes and m hyperedges where the expected degrees of the hyperedges, i.e., the number of other hyperedges that share at least one node, are given by some fixed $w_1, \ldots, w_m$ and the expected cardinality of the hyperedges are $k_1, \ldots, k_m$. In this case both the clique expansion and line graph will be random with given expected degrees and, hence, we may apply Corollary 1 to assert with high probability that the HENN trained using one such random hypergraph will be transferable to others of the same size with a bound of the order $1/n$.

C.2 Graphon Transferability

A *graphon* is a symmetric, measurable function (sometimes called a *kernel*) $W : [0,1]^2 \to [0,1]$ which can be understood as the limit of a sequence of dense undirected graphs where the node index is a continuous set [33]. While many authors explore transferability for sequences of graphs that converge to the same graphon [35, 38], we will focus on the particular related case of random graphs sampled from graphons. This allows us to make use of existing results which explore the spectrum of the normalized Laplacian of such sampled graphs [47]; however, it may be possible to obtain more general results to measure the spectral similarity of sequences of GSOs obtained from graphs converging to the same graphon.

We assume that the graphon is bounded away from zero, so that $\inf_{x,y\in[0,1]} W(x,y) > 0$. In order to generate a random graph with n nodes via the graphon W, following [47] we sample some points $x_1, \ldots, x_n \sim \text{Uniform}([0,1])$ and create the edge (i,j) with a probability $W(x_{(i)}, x_{(j)})$, where $x_{(i)}$ denotes the i-th order statistic of the sampled points. We say the resulting graph $\mathcal{G}_n$ is sampled from the graphon W. There is an explicit connection to the results of Appendix C, as it has been shown that graphs sampled from graphons in this manner have fixed expected degree distributions [47]. Indeed, [47, Lemma 3] shows that the eigenvalues of these sampled graphs concentrate, and [47, Proposition 3] bounds the spectral gap. Moreover, the graphon being bounded away from zero ensures the sampled graphs will be almost surely connected if they are large enough. Hence, all conditions of Proposition 2 are satisfied. With this in mind, we present our result on the transferability of sequences of random graphs sampled from the same graphon.

Corollary 2. *Let $\{\mathcal{G}_n\}_{n=1}^{\infty}$ and $\{\tilde{\mathcal{G}}_n\}_{n=1}^{\infty}$ be two independent sequences of almost surely connected graphs sampled from the same graphon W, which is bounded away from zero, with normalized Laplacians $\{S_n\}_{n=1}^{\infty}$ and $\{\tilde{S}_n\}_{n=1}^{\infty}$. Then, for a sequence of GNNs $\{\Phi(\cdot; S_n, \Theta)\}_{n=1}^{\infty}$ following the conditions of Theorem 3,*

$$\left\| \Phi(\cdot; S_n, \Theta) - \Phi(\cdot; \tilde{S}_n, \Theta) \right\|_{\mathcal{P}} \to 0,$$

in probability as $n \to \infty$.

Proof. Direct result of [47, Lemma 3 and Proposition 3] and Theorem 3. □

D Stability of GNNs via spectral similarity

Stability is a generalization property of graph learning architectures that is closely related to transferability. In particular, an architecture with the stability property is one for which small changes in the underlying topology or structure of the graph have a small effect on the way signals are processed. Hence, stability can be understood as a special case of transferability, where one graph is a perturbed version of another graph. However, stability is no less important, as it characterizes the robustness of a graph learning architecture to small changes in the underlying graph, which could arise for example via measurement noise. In the following sections, we will explore stability of graph neural networks by computing the spectral similarity of a graph and its perturbed version. By applying the results from Section 4 together with the similarity bounds we produce herein, we will show that graph neural networks can be stable to perturbations of the graph structure. In contrast to prior results [21], the results herein along with Appendix C.1 show that stability does not decay as the size of the graph grows.

D.1 Spectral similarity of perturbed matrices

To investigate stability, we are interested in small changes to some graph $\mathcal{G}$ that results in a perturbed version which we call $\tilde{\mathcal{G}}$. In particular, since we are processing graph signals, we will focus on the graph shift operator S of the original graph $\mathcal{G}$ and the GSO $\tilde{S}$ of the perturbed graph $\tilde{\mathcal{G}}$. We will explore two types of perturbations, both relative and additive, in order to model how a graph may be changed slightly.

Fundamentally, graph neural networks are spectral operators; the graph frequency response of a filter (2) makes this relationship explicit. Well-known results such as Weyl's inequality or the Weilandt-Hoffman inequalities [44] show that that the spectrum of a real symmetric matrix is stable to small perturbations. Thus, it stands to reason that if small perturbations of a graph lead to small perturbations in the eigenvalues, then these small perturbations should not change the graph frequency response of a filter very much. To this end, we will explore the spectral similarity of a graph and its perturbed version. In doing so, we can bound the change in the eigenvalues of a graph after a perturbation has been applied.

We study two types of perturbations herein, called *additive* and *relative* perturbations. The first and simplest type involves adding a small perturbation to the GSO S of the original graph $\mathcal{G}$, so that the GSO $\tilde{S}$ of the perturbed graph $\tilde{\mathcal{G}}$ becomes

$$\tilde{S} = S + E. \tag{22}$$

The perturbation is small in the sense of the operator norm, so that $\|E\|_{op} \leqslant \delta$. Such perturbations can be understood as adding or removing edge weight regardless of the original magnitude of the edges, and could be as extreme as adding or removing edges entirely. The other type of perturbation instead affects the edge weights in a manner that is relative to their magnitude, so that

$$\tilde{S} = S + \frac{1}{2}(SE + ES), \tag{23}$$

again with $\|E\|_{op} \leqslant \delta$ for some small $\delta > 0$. In both cases we assume that the perturbed GSO $\tilde{S}$ will be positive semi-definite. We explore the spectral similarity between the original GSOs and their perturbed versions in the following results, beginning with relative perturbations below.

Proposition 3. *Given a symmetric, positive semi-definite GSO S and its symmetric, positive semi-definite relatively perturbed version $\tilde{S} = S + \frac{1}{2}(SE + ES)$, where E is diagonalizable with $\|E\|_{op} \leqslant \delta$, the matrices are δ-spectrally similar.*

Proof. Recall by definition of ϵ-spectral similarity (11) that

$$\begin{aligned} &(1-\epsilon)S \preceq \tilde{S} \preceq (1+\epsilon)S \\ &\Leftrightarrow (1-\epsilon)\mathbf{x}^\intercal S\mathbf{x} \leqslant \mathbf{x}^\intercal \tilde{S}\mathbf{x} \leqslant (1+\epsilon)\mathbf{x}^\intercal S\mathbf{x} \quad \forall \mathbf{x} \in \mathbb{R}^n. \end{aligned}$$

Given some perturbation matrix E, we wish to determine the coefficient of spectral similarity, ϵ. Let us begin with the latter inequality,

$$\begin{aligned} \tilde{S} \preceq (1+\epsilon)S \Leftrightarrow S + \frac{1}{2}(SE+ES) &\preceq (1+\epsilon)S \\ \Leftrightarrow \epsilon S - \frac{1}{2}(SE+ES) &\succeq 0 \\ \Leftrightarrow \mathbf{x}^\intercal\left(\epsilon S - \frac{1}{2}(SE+ES)\right)\mathbf{x} &\geqslant 0 \quad \forall \mathbf{x} \in \mathbb{R}^n. \end{aligned}$$

Note that, since S is PSD, we may always choose ϵ to be large enough that $\epsilon S - \frac{1}{2}(SE + ES)$ is PSD. Let us diagonalize $E = UMU^\mathsf{T}$, and recall that $\|E\|_{op} = |\lambda_{max}(E)| \leqslant \delta$. Hence,

$$\begin{aligned}
&\mathbf{x}^\mathsf{T}\left(\epsilon S - \frac{1}{2}(SE + ES)\right)\mathbf{x} \\
&= \mathbf{x}^\mathsf{T} S((\epsilon/2)I - E/2)\mathbf{x} + \mathbf{x}^\mathsf{T}((\epsilon/2)I - E/2)S\mathbf{x} \\
&= \mathbf{x}^\mathsf{T} SU((\epsilon/2)I - M/2)U^\mathsf{T}\mathbf{x} + \mathbf{x}^\mathsf{T} U((\epsilon/2)I - M/2)U^\mathsf{T} S\mathbf{x} \\
&\geqslant \frac{1}{2}\mathbf{x}^\mathsf{T} SU(\epsilon I - \delta I)U^\mathsf{T}\mathbf{x} + \frac{1}{2}\mathbf{x}^\mathsf{T} U(\epsilon I - \delta I)U^\mathsf{T} S\mathbf{x} \\
&= \frac{1}{2}(\epsilon - \delta)\mathbf{x}^\mathsf{T} SUU^\mathsf{T}\mathbf{x} + \frac{1}{2}(\epsilon - \delta)\mathbf{x}^\mathsf{T} UU^\mathsf{T} S\mathbf{x} \\
&= (\epsilon - \delta)\mathbf{x}^\mathsf{T} S\mathbf{x},
\end{aligned}$$

with equality when $E = \delta I$. Since S is PSD, $\mathbf{x}^\mathsf{T} S\mathbf{x} \geqslant 0 \; \forall \mathbf{x} \in \mathbb{R}^n$. Thus, if $\epsilon \geqslant \delta$, then $\tilde{S} \preceq (1+\epsilon)S$. Moreover, when $E = \delta I$, $\tilde{S} = (1 + \delta)S = (1 + \epsilon)S$, and δ-similarity is tight.

Next, notice similarly that

$$\begin{aligned}
(1 - \epsilon)S \preceq \tilde{S} &\Leftrightarrow \epsilon S + \frac{1}{2}(SE + ES) \succeq 0 \\
&\Leftrightarrow \mathbf{x}^\mathsf{T}\left(\epsilon S + \frac{1}{2}(SE + ES)\right)\mathbf{x} \geqslant 0 \; \forall \mathbf{x} \in \mathbb{R}^n.
\end{aligned}$$

Then, as above,

$$\begin{aligned}
&\mathbf{x}^\mathsf{T}\left(\epsilon S + \frac{1}{2}(SE + ES)\right)\mathbf{x} \\
&= \mathbf{x}^\mathsf{T} SU((\epsilon/2)I + M/2)U^\mathsf{T}\mathbf{x} + \mathbf{x}^\mathsf{T} U((\epsilon/2)I + M/2)U^\mathsf{T} S\mathbf{x} \\
&\geqslant \frac{1}{2}\mathbf{x}^\mathsf{T} SU(\epsilon I + \lambda_{min}(E)I)U^\mathsf{T}\mathbf{x} \\
&\quad + \frac{1}{2}\mathbf{x}^\mathsf{T} U(\epsilon I + \lambda_{min}(E)I)U^\mathsf{T} S\mathbf{x} \\
&= (\epsilon + \lambda_{min}(E))\mathbf{x}^\mathsf{T} S\mathbf{x}.
\end{aligned}$$

Since $|\lambda_{min}(E)| \leqslant |\lambda_{max}(E)| = \delta$, $\epsilon \geqslant \delta$ implies $(1 - \epsilon)S \preceq \tilde{S}$. Hence if $\epsilon \geqslant \delta$, we have that $(1 - \epsilon)S \preceq S + SE + ES \preceq (1 + \epsilon)S$. □

Consider a relative perturbation via dilation where $E = \delta I$ and thus $\|E\| = \delta$, and $\tilde{S} = (1+\delta)S$. We thus observe that δ-similarity is tight, i.e., for a *general* E with $\|E\| \leqslant \delta$, we cannot hope for a better bound on the similarity coefficient. Since our results require spectral similarity to hold regardless of the structure of the perturbation E, the bound in Proposition 3 is thus tight. However, for some arbitrary relative perturbation, it may be the case that a spectral similarity coefficient $\epsilon < \delta$ suffices. Thus, this result and those that follow should be viewed as tight *worst-case* bounds on the coefficient of spectral similarity for general relative perturbations. Next, let us explore additive perturbations.

Proposition 4. *Given a symmetric, positive semi-definite GSO S and its symmetric, positive semi-definite additively perturbed version $\tilde{S} = S + E$, where $\|E\|_{op} \leqslant \delta$, if ker$(E) \subseteq$ ker(S) then the matrices are $(\delta/\bar{\lambda}(S))$-spectrally similar, where $\bar{\lambda}(S)$ is the smallest non-zero eigenvalue of S.*

Proof. As in Proposition 3, let us find the value of ϵ which satisfies the inequality

$$\begin{aligned}
\tilde{S} \preceq (1 + \epsilon)S &\Leftrightarrow S + E \preceq (1 + \epsilon)S \\
&\Leftrightarrow \epsilon S - E \succeq 0 \\
&\Leftrightarrow \mathbf{x}^\mathsf{T}(\epsilon S - E)\mathbf{x} \geqslant 0 \quad \forall \mathbf{x} \in \mathbb{R}^n.
\end{aligned}$$

For any $\mathbf{x} \in \ker(S)$, $\mathbf{x}^\intercal(\epsilon S - E)\mathbf{x} = -\mathbf{x}^\intercal E\mathbf{x}$, but by assumption $\ker(E) \subseteq \ker(S)$. Hence, regardless of ϵ, $\mathbf{x}^\intercal(\epsilon S - E)\mathbf{x} = 0$ for $\mathbf{x} \in \ker(S)$. Now, for $\mathbf{x} \notin \ker(S)$, since $\|E\|_{op} \leqslant \delta$,

$$\begin{aligned}\mathbf{x}^\intercal(\epsilon S - E)\mathbf{x} &= \epsilon\mathbf{x}^\intercal S\mathbf{x} - \mathbf{x}^\intercal E\mathbf{x} \\ &\geqslant \epsilon\mathbf{x}^\intercal S\mathbf{x} - \delta\|\mathbf{x}\|_2 \\ &\geqslant \epsilon\bar{\lambda}(S)\|\mathbf{x}\| - \delta\|\mathbf{x}\|_2 \\ &= (\epsilon\bar{\lambda}(S) - \delta)\|\mathbf{x}\|_2\,,\end{aligned}$$

where we have used the fact that $\min\{\mathbf{x}^\intercal S\mathbf{x} \mid \mathbf{x} \notin \ker(S)\} = \bar{\lambda}(S)\|\mathbf{x}\|_2$. Thus, we must have that $\epsilon \geqslant \delta/\bar{\lambda}(S)$. Now, for the second inequality,

$$(1-\epsilon)S \preceq \tilde{S} \Leftrightarrow \mathbf{x}^\intercal(\epsilon S + E)\mathbf{x} \geqslant 0 \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

Clearly for $\mathbf{x} \in \ker(S)$, $\mathbf{x}^\intercal(\epsilon S + E)\mathbf{x} = 0$. Then, for $\mathbf{x} \notin \ker(S)$,

$$\begin{aligned}\mathbf{x}^\intercal(\epsilon S + E)\mathbf{x} &= \epsilon\mathbf{x}^\intercal S\mathbf{x} + \mathbf{x}^\intercal E\mathbf{x} \\ &\geqslant (\epsilon\bar{\lambda}(S) + \lambda_{min}(E))\|\mathbf{x}\|_2\,.\end{aligned}$$

Thus, if $\epsilon \geqslant \max\{-\lambda_{min}(E)/\bar{\lambda}(S), 0\}$ then $(1-\epsilon)S \preceq \tilde{S}$. However, $\delta = |\lambda_{max}(E)| \geqslant |\lambda_{min}(E)|$. Hence, it is sufficient for $\epsilon \geqslant \delta/\bar{\lambda}(S)$ to ensure $(1-\epsilon)S \preceq S + E \preceq (1+\epsilon)S$. □

The condition $\ker(E) \subseteq \ker(S)$ mirrors *(A1)* from Proposition 2. If S is the normalized Laplacian, this condition enforces that the perturbation not cause the graph to become disconnected, which is reasonable in practice. Indeed, from a graph signal processing perspective, disconnected components have no effect on each other. Further, similarly to *(A2)*, if S is the normalized Laplacian then Cheeger's inequality states $h_{\mathcal{G}}^2/2 \leqslant \bar{\lambda}(S) \leqslant 2h_{\mathcal{G}}$, where $h_{\mathcal{G}}$ is the Cheeger constant (or conductance) of the graph $\mathcal{G}$ [14]. Hence, our result in Proposition 4 tells us that graphs with higher Cheeger constant, i.e., better-connected graphs, will be more similar to their additively perturbed versions. In other words, graphs which are well-connected will result in trained GNNs that exhibit better stability properties than more sparsely connected graphs.

In practice, perturbations to graphs can be modeled as both relative and additive perturbations. Thus, below we consider spectral similarity when both perturbations are present.

Proposition 5. *Given a symmetric, positive semi-definite GSO S, an additive perturbation matrix D such that $\|D\|_{op} \leqslant \delta_A$ and $\ker(D) \subseteq \ker(S)$, a diagonalizable relative perturbation matrix E such that $\|E\|_{op} \leqslant \delta_R$, and the symmetric, positive semi-definite perturbed GSO $\tilde{S} = S + \frac{1}{2}(SE+ES)+D$, the matrices are $(\delta_R + \delta_A/\bar{\lambda}(S))$-spectrally similar.*

Proof. Combination of Proposition 3 and Proposition 4. □

While our bounds on spectral similarity are tight for relative perturbations, they are maximal in the sense that we require them to hold for all eigenvalues, i.e., they are equivalent to $(1-\epsilon)\lambda_i(S) \leqslant \lambda_i(\tilde{S}) \leqslant (1+\epsilon)\lambda_i(S)$ for all $i \in \{1, \ldots, n\}$. However, perturbations in practice will not affect the eigenvalues in such a uniform manner, and thus the following stability bounds may not be tight for arbitrary perturbations. Moreover, for additive perturbations, in Proposition 4 the quantity $1/\bar{\lambda}(S)$ arises due to the necessity of satisfying spectral similarity for all signals $\mathbf{x} \in \mathbb{R}^n$, including the eigenvector $\bar{\mathbf{v}}$ associated with the smallest non-zero eigenvalue of S. If the signals to be used in practice are not well-aligned with $\bar{\mathbf{v}}$, we may be able to tighten this bound.

D.2 Stability via spectral similarity

To achieve our main result on stability of graph neural networks, we combine the above results on perturbations with the transferability bounds from Section 4.

Theorem 4. *Consider a symmetric, positive semi-definite GSO S and its perturbed version $\tilde{S} = S + \frac{1}{2}(SE + ES) + D$, where D is an additive perturbation matrix such that $\|D\|_{op} \leqslant \delta_A$ and $\ker(D) \subseteq \ker(S)$, and E is a diagonalizable relative perturbation matrix such that $\|E\|_{op} \leqslant \delta_R$.*

The GNN $\Phi(\cdot; S, \Theta)$ with normalized Lipschitz nonlinearities and L layers with f features, each with filters that have unit operator norm and are C-integral Lipschitz, satisfies

$$\begin{aligned} &\left\| \Phi(\cdot; S, \Theta) - \Phi(\cdot; \tilde{S}, \Theta) \right\| \\ &\qquad \leqslant CLf^L(\delta_R + \delta_A/\bar{\lambda}(S)) + \mathcal{O}((\delta_R + \delta_A)^2) \end{aligned}$$

Proof. Follows from Theorem 1 and Proposition 5. □

In contrast to previous results on the stability of graph neural networks [21], this bound has no dependence on the size of the graph n. This can be explained by the effect of the eigenvector differences which appear in the result in [21]. Via spectral similarity, we instead upper-bound the difference between the matrices (in the sense of the positive semi-definite cone), i.e., we consider $(1+\epsilon)S$ instead of $\tilde{S}$. In this way we can bound the transferability error by considering the eigenvalues, which are known to be stable to perturbations, without considering any changes whatsoever to the eigenvectors, which are known to be unstable [44]. Indeed, the results of Appendix C suggest that stability in fact *improves* as the number of nodes n grows larger. Thus, stability is entirely characterized by parameters of the architecture (the integral Lipschitz constant and number of layers), the structure of the original graph (via the smallest nonzero eigenvalue of S), and the magnitude of the perturbations that occur. Naturally, larger magnitude perturbations result in looser stability bounds, as does a larger integral Lipschitz constant C and more layers L or features f. However, larger C, L, and f suggest enhanced discriminability, since the GNN can produce sharper filters as C is larger, and more of those filters can be composed as L and f grow. A larger value of $\bar{\lambda}(S)$ when S is the normalized Laplacian suggests better connectedness and, hence, more local smoothing, which explains how it tightens the stability bound. Together, these insights pose a tradeoff between stability and discriminability; if a GNN is indifferent to large perturbations in a graph's structure, it cannot also tell very similar graphs apart.