



HAL
open science

Higher-Order GNNs Meet Efficiency: Sparse Sobolev Graph Neural Networks

Jhony Giraldo, Aref Einizade, Andjela Todorovic, Jhon Castro-Correa, Mohsen Badiey, Thierry Bouwmans, Fragkiskos D. Malliaros

► To cite this version:

Jhony Giraldo, Aref Einizade, Andjela Todorovic, Jhon Castro-Correa, Mohsen Badiey, et al.. Higher-Order GNNs Meet Efficiency: Sparse Sobolev Graph Neural Networks. IEEE Transactions on Signal and Information Processing over Networks, 2024, 11, pp.11-22. <10.1109/TSIPN.2024.3503416>. <hal-04801077>

HAL Id: hal-04801077

<https://hal.science/hal-04801077v1>

Submitted on 17 Jul 2025

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

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



Distributed under a Creative Commons CC BY 4.0 - Attribution - International License

Higher-Order GNNs Meet Efficiency: Sparse Sobolev Graph Neural Networks

Jhony H. Giraldo, Aref Einizade, Andjela Todorovic, Jhon A. Castro-Correa, Mohsen Badiey, Thierry Bouwmans, and Fragkiskos D. Malliaros

Abstract—Graph Neural Networks (GNNs) have shown great promise in modeling relationships between nodes in a graph, but capturing higher-order relationships remains a challenge for large-scale networks. Previous studies have primarily attempted to utilize the information from higher-order neighbors in the graph, involving the incorporation of powers of the shift operator, such as the graph Laplacian or adjacency matrix. This approach comes with a trade-off in terms of increased computational and memory demands. Relying on graph spectral theory, we make a fundamental observation: *the regular and the Hadamard power of the Laplacian matrix behave similarly in the spectrum*. This observation has significant implications for capturing higher-order information in GNNs for various tasks such as node classification and semi-supervised learning. Consequently, we propose a novel graph convolutional operator based on the sparse Sobolev norm of graph signals. Our approach, known as Sparse Sobolev GNN (S2-GNN), employs Hadamard products between matrices to maintain the sparsity level in graph representations. S2-GNN utilizes a cascade of filters with increasing Hadamard powers to generate a diverse set of functions. We theoretically analyze the stability of S2-GNN to show the robustness of the model against possible graph perturbations. We also conduct a comprehensive evaluation of S2-GNN across various graph mining, semi-supervised node classification, and computer vision tasks. In particular use cases, our algorithm demonstrates competitive performance compared to state-of-the-art GNNs in terms of performance and running time.

Index Terms—Graph neural networks, sparse graph convolutions, higher-order convolutions, graph spectrum, Sobolev norm

I. INTRODUCTION

GRAPH representation learning and its applications have garnered significant attention in recent years [1]–[4]. Notably, Graph Neural Networks (GNNs) have emerged as a powerful extension of Convolutional Neural Networks (CNNs) for modeling non-Euclidean data as graphs. GNNs have

Jhony H. Giraldo, Aref Einizade, and Andjela Todorovic are with the LTCI, Télécom Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France e-mail: jhony.giraldo@telecom-paris.fr, aref.einizade@telecom-paris.fr, andela.todorovic@ip-paris.fr.

Jhon A. Castro-Correa and Mohsen Badiey are with the Department of Electrical and Computer Engineering, University of Delaware, Newark, DE, USA. E-mail: jcastro@udel.edu, badiey@udel.edu.

Thierry Bouwmans is with the laboratoire MIA, Mathématiques, Image et Applications, La Rochelle Université, 17000 La Rochelle, France e-mail: tbouwman@univ-lr.fr.

Fragkiskos D. Malliaros is with Université Paris-Saclay, CentraleSupélec, Inria, Centre for Visual Computing (CVN), Gif-Sur-Yvette, France. E-mail: fragkiskos.malliaros@centralesupelec.fr.

This work was supported in part by the ANR (French National Research Agency) under the JCJC project GraphIA (ANR-20-CE23-0009-01), by the Office of Naval Research ONR (Grant No. N00014-21-1-2760), and by the center Hi! PARIS.

been successfully applied in various domains, including semi-supervised learning [3], clustering [5], point cloud semantic segmentation [6], misinformation detection [7], and molecular modeling [8], among others.

Most GNNs update their node embeddings by performing specific operations within the neighborhood of each node through message passing [9]. However, this updating rule has limitations when it comes to capturing higher-order¹ relationships between nodes since it only leverages information from the immediate neighbors (1-hop) of each vertex. To address this limitation, previous methods in GNNs have attempted to capture higher-order connections by incorporating powers of the sparse shift operator of the graph, such as the adjacency or Laplacian matrix [2], [10]. Nevertheless, these methods encounter computational and memory issues, attributed either to the densification of the shift operator [10] or a bottleneck during the implementation of recursive graph diffusion operations [2].

This work explores a somewhat radical departure from prior approaches in higher-order graph convolutions for GNNs. We observe that the eigenvalues of the higher-order Laplacian matrix in a weighted graph exhibit similar behavior when subjected to regular and Hadamard power operations (the Hadamard power is also known as element-wise power). To establish the precise relationship between the spectra of the regular and sparse higher-order Laplacian matrix, we rely on tools from spectral graph theory [11] and the Schur product theorem [12]. This fundamental observation enables us to design accurate and efficient sparse graph convolutions for GNNs. More specifically, we propose a novel approach called Sparse Sobolev GNN (S2-GNN), which employs a cascade of sparse higher-order filtering operations. This allows for the computation of a more diverse set of functions at each layer. To achieve this goal, we introduce a new sparse Sobolev norm, drawing inspiration from the literature of Graph Signal Processing (GSP) [13], [14].

In S2-GNN, the information of the higher-order filtering operations is aggregated using either a linear combination or a Multi-Layer Perceptron (MLP) fusion layer. We conduct thorough evaluations of S2-GNN in various semi-supervised learning tasks across several domains, including tissue phenotyping in colon cancer histology images [15], text classification of news [16], activity recognition with sensors [17], and recognition of spoken letters [18]. Furthermore, we test

¹We use the term “higher-order” to refer to higher-order hop information (more than one hop).

our algorithm in node classification tasks using benchmark networks commonly employed in the literature [3], [19]. S2-GNN demonstrates superior or competitive performance compared to a broad range of GNN methods that employ first-order or higher-order graph convolutions [2], [3], [10], various types of multi-head attentions [4], [20], and graph transformer architectures [21].

In this work, we build upon and improve our preliminary study [22]. Specifically, we provide a more comprehensive theoretical explanation, along with an extensive experimental evaluation and in-depth analysis of novel findings. Additionally, we add a rigorous theoretical study about the stability properties of S2-GNN against possible graph perturbations, which is an important and practical issue in real-world scenarios [23], [24]. Furthermore, we empirically observe that utilizing smooth-learned graphs [25], [26] improves the performance of both S2-GNN and baseline GNN methods compared to the commonly used k -Nearest Neighbors (k -NN) technique for graph construction. Smoother graphs exhibit better results as they promote homophily within the graph.

The main contributions of this work can be summarized as follows:

- 1) We show that the spectrum of higher-order weighted graphs, using the Laplacian matrix, exhibits similar behavior to their sparse counterparts. This finding enables the design of GNN architectures that deliver superior or competitive performance while maintaining efficiency.
- 2) We propose S2-GNN, a novel GNN architecture that utilizes a cascade of higher-order sparse filters. S2-GNN effectively leverages sparse higher-order operations without excessively increasing complexity.
- 3) We rigorously analyze the stability properties of S2-GNN, shedding light on its robustness against possible graph perturbations.
- 4) We conduct extensive experimental evaluations on multiple publicly available benchmark datasets and compare S2-GNN against nine state-of-the-art GNNs. Our algorithm outperforms or achieves competitive results compared to prior methods.

The remainder of the paper is organized as follows: Section II presents an overview of the related work, while Section III provides an introduction to the preliminary concepts and foundations relevant to this study. Section IV provides a detailed explanation of the proposed S2-GNN model and its theoretical stability analysis. In Section V, we present the experimental framework, results, and discussion. Finally, Section VI offers the concluding remarks of the paper.

II. RELATED WORK

The study of graphs is a well-established field in machine learning and mathematics [11], [27]. With the emergence of GNNs in 2014, motivated by the success of CNNs in regular-structured data like images, a wide range of GNN models have been proposed to learn representations of graph-structured data. Bruna *et al.* proposed the first modern GNN by extending the convolutional operator of CNNs to graphs [1]. Defferrard *et al.* incorporated concepts from GSP to propose localized

spectral filtering [2], while Kipf and Welling approximated the spectral filtering operation to enable efficient Graph Convolution Networks (GCNs) [3]. These works demonstrated the potential of GNNs and have served as inspiration for the development of various GNN models.

Veličković *et al.* [4] proposed Graph Attention Networks (GATs), which employ an attention mechanism to learn different weights for different nodes in the graph. This enables GATs to effectively model complex relationships between nodes, albeit with increased computational complexity. Subsequent works have further explored attention mechanisms in GNNs, such as [20], [28].

The Simple Graph Convolution (SGC) model was introduced by Wu *et al.* [29], which simplifies the GCN by removing the non-linear activation functions, resulting in improved efficiency but reduced expressivity. Later, Chiang *et al.* [30] proposed ClusterGCN, a scalable GNN model that leverages graph clustering to enhance efficiency on large-scale graphs. By partitioning the graph into clusters and applying GNNs to each cluster in parallel, ClusterGCN can learn representations of large-scale graphs. However, ClusterGCN sacrifices global graph information and the exchange of messages between clusters. Frasca *et al.* proposed Scalable Inception Graph Neural Networks (SIGN) [10], which leverage higher-order relationships in graphs to improve expressivity at the expense of scalability. SIGN computes the powers of the adjacency matrix, making it unsuitable for large-scale problems due to its complexity.

More recently, graph transformers have emerged as an alternative to GNNs for learning graph-structured data. Some graph transformers in the literature, similar to GATs, employ attention mechanisms computed in the neighborhood of each node, as shown in [21]. Another approach in graph transformers is to compute attention mechanisms (with or without positional embeddings) by leveraging fully connected graphs [31]. However, the latter approach becomes computationally prohibitive for large-scale applications, as the complexity grows quadratically with the number of nodes in the graph. For a comprehensive review of GNNs, we recommend referring to the survey by Zhang *et al.* [9].

Previous studies have focused on improving scalability at the expense of expressivity [29], [30] or vice versa [2], [10]. Although GAT, its derivatives, and Transformers have been explored, they are not optimal solutions to this problem due to the high computational burden of the attention mechanism and the multi-head function. As a result, outside the GNN community, the GCN model [3] is the preferred choice due to its simplicity and high efficiency. In this paper, we propose an accurate and efficient GNN architecture based on higher-order sparse convolutions. Our algorithm improves accuracy while maintaining low computational and memory footprints.

III. PRELIMINARIES

A. Mathematical Notation

In this work, calligraphic letters such as \mathcal{V} denote sets, and $|\mathcal{V}|$ represents the cardinality of the set. Uppercase boldface letters such as \mathbf{A} represent matrices, while lowercase boldface letters such as \mathbf{x} denote vectors. \mathbf{I} denotes the identity

matrix, and $\mathbf{1}$ represents a vector of ones with appropriate dimensions. The superscripts $(\cdot)^T$ and $(\cdot)^H$ correspond to transposition and Hermitian transpose, respectively. $\text{diag}(\mathbf{x})$ denotes a diagonal matrix with entries given by the vector $\mathbf{x} = [x_1, x_2, \dots, x_n]^T \in \mathbb{R}^N$. $\text{tr}(\cdot)$ is the trace of a matrix. The ℓ_2 -norm of a vector is denoted by $\|\cdot\|_2$, the Frobenius norm is denoted by $\|\cdot\|_F$, and the entry-wise norm-1 of a matrix is denoted by $\|\cdot\|_{1,1}$. The symbols \circ and \otimes represent the Hadamard and Kronecker products between matrices, respectively.

B. Graph Signals

A graph is a mathematical entity represented as $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ is the set of N nodes and $\mathcal{E} \subseteq \{(i, j) \mid i, j \in \mathcal{V} \text{ and } i \neq j\}$ is the set of edges between nodes i and j . The weighted adjacency matrix of the graph is denoted as $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $\mathbf{A}(i, j) = a_{i,j} \in \mathbb{R}_+$ represents the weight of the edge (i, j) , and $\mathbf{A}(i, j) = 0 \forall (i, j) \notin \mathcal{E}$. For undirected graphs, \mathbf{A} is symmetric. A graph signal is a function $x : \mathcal{V} \rightarrow \mathbb{R}$ and is represented as $\mathbf{x} \in \mathbb{R}^N$, where $\mathbf{x}(i)$ is the graph signal evaluated on the i th node. The degree matrix of G is a diagonal matrix given by $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1})$. Different definitions exist for the Laplacian matrix, including the combinatorial Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$, and the symmetric normalized Laplacian $\mathbf{\Delta} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ [32]. The Laplacian matrix is a positive semi-definite matrix for undirected graphs, with eigenvalues² $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_N$ and corresponding eigenvectors $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N\}$. The adjacency and Laplacian matrices, as well as their normalized forms, are examples of the shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$ in GSP.

The graph Fourier basis of \mathbf{L} is given by the spectral decomposition $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ [32], where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N]$ and $\mathbf{\Lambda} = \text{diag}([\lambda_1, \lambda_2, \dots, \lambda_N]^T)$. The Graph Fourier Transform (GFT) $\hat{\mathbf{x}}$ of the graph signal \mathbf{x} is defined as $\hat{\mathbf{x}} = \mathbf{U}^T\mathbf{x}$, while the inverse GFT is given by $\mathbf{x} = \mathbf{U}\hat{\mathbf{x}}$ [32]. Although we use the spectral definitions of graphs to establish the relationship between the spectrum of the higher-order Laplacian and its sparse counterpart in this work, the eigenvalue decomposition is not required to implement S2-GNN.

C. Graph Convolutional Filters

The concept of graph convolutional filters was introduced in the literature of GSP [33].

Definition 1. Given a set of parameters $\mathbf{q} = [q_0, q_1, \dots, q_K]^T$ and a graph signal $\mathbf{x} \in \mathbb{R}^N$, a graph convolutional filter of order K is a linear mapping $Q : \mathbb{R}^N \rightarrow \mathbb{R}^N$ comprising a linear combination of K shifted signals:

$$Q(\mathbf{x}) = \sum_{k=0}^K q_k \mathbf{S}^k \mathbf{x} = \mathbf{Q}(\mathbf{S})\mathbf{x}, \quad (1)$$

where $\mathbf{Q}(\mathbf{S}) = \sum_{k=0}^K q_k \mathbf{S}^k$ is the $N \times N$ polynomial filtering matrix.

The output signal $Q(\mathbf{x})$ at node i is a linear combination of signal values located within up to K -hops away from node i

² $\lambda_N \leq 2$ in the case of the symmetric normalized Laplacian $\mathbf{\Delta}$.

given by $q_0\mathbf{x}(i) + q_1[\mathbf{S}\mathbf{x}](i) + \dots + q_K[\mathbf{S}^K\mathbf{x}](i)$, where $[\mathbf{S}^k\mathbf{x}](i)$ is the i th value of the vector $\mathbf{S}^k\mathbf{x}$.

IV. SPARSE SOBOLEV GRAPH NEURAL NETWORK

Figure 1 provides an overview of our S2-GNN algorithm. S2-GNN utilizes either a k -Nearest Neighbors (k -NN) method or the procedure described in [25] to establish a graph representation for a specific problem when the structure is unavailable. In cases where the structure is already known, step a) in Fig. 1 is not required. Consequently, we employ the proposed GNN architecture to address the primary problem at hand³. Within S2-GNN, a modified sparse Sobolev term serves as the shift operator in the propagation function of the GNN.

A. Sobolev Norm

The Sobolev norm in GSP has been used as a regularization term to solve problems in 1) video processing [34], 2) modeling of infectious diseases [35], and 3) interpolation of graph signals [13], [14].

Definition 2 (Pesenson [13]). For fixed parameters $\epsilon \geq 0$ and $\rho \in \mathbb{R}$, the Sobolev norm is defined as $\|\mathbf{x}\|_{\rho, \epsilon} \triangleq \|(\mathbf{L} + \epsilon\mathbf{I})^{\rho/2}\mathbf{x}\|$. When \mathbf{L} is symmetric, we can rewrite $\|\mathbf{x}\|_{\rho, \epsilon}^2$ as follows:

$$\|\mathbf{x}\|_{\rho, \epsilon}^2 = \mathbf{x}^T(\mathbf{L} + \epsilon\mathbf{I})^\rho\mathbf{x}. \quad (2)$$

We divide the analysis of (2) into two steps: 1) when $\epsilon = 0$, and 2) when $\rho = 1$. For $\epsilon = 0$ in (2) and considering that \mathbf{U} is orthonormal, we obtain:

$$\|\mathbf{x}\|_{\rho, 0}^2 = \mathbf{x}^T\mathbf{L}^\rho\mathbf{x} = \mathbf{x}^T\mathbf{U}\mathbf{\Lambda}^\rho\mathbf{U}^T\mathbf{x} = \hat{\mathbf{x}}^T\mathbf{\Lambda}^\rho\hat{\mathbf{x}} = \sum_{i=1}^N \hat{\mathbf{x}}^2(i)\lambda_i^\rho. \quad (3)$$

Notice that the spectral components $\hat{\mathbf{x}}(i)$ are penalized with powers of the eigenvalues λ_i^ρ of \mathbf{L} . Since the eigenvalues are ordered in increasing order, the higher frequencies of $\hat{\mathbf{x}}$ are penalized more than the lower frequencies when $\rho = 1$, leading to a smooth function in G . For $\rho > 1$, the GFT $\hat{\mathbf{x}}$ is penalized with a more diverse set of eigenvalues. Similarly, we can analyze the adjacency matrix \mathbf{A} using the eigenvalue decomposition $\mathbf{A}^\rho = (\mathbf{V}\mathbf{\Sigma}\mathbf{V}^H)^\rho = \mathbf{V}\mathbf{\Sigma}^\rho\mathbf{V}^H$, where \mathbf{V} is the matrix of eigenvectors, and $\mathbf{\Sigma}$ is the matrix of eigenvalues of \mathbf{A} . For \mathbf{A} , the GFT can be defined as $\hat{\mathbf{x}} = \mathbf{V}^H\mathbf{x}$.

For the second analysis of (2), when $\rho = 1$, we have:

$$\|\mathbf{x}\|_{1, \epsilon}^2 = \mathbf{x}^T(\mathbf{L} + \epsilon\mathbf{I})\mathbf{x}. \quad (4)$$

The term $(\mathbf{L} + \epsilon\mathbf{I})$ in (4) is associated with a better condition number⁴ than using \mathbf{L} alone. Better condition numbers are associated with faster convergence rates in gradient descent methods, as demonstrated in [14]. For the Laplacian matrix \mathbf{L} , we know that $\kappa(\mathbf{L}) = \frac{|\lambda_{\max}(\mathbf{L})|}{|\lambda_{\min}(\mathbf{L})|} \approx \frac{\lambda_{\max}(\mathbf{L})}{0} \rightarrow \infty$, where $\kappa(\mathbf{L})$ is the condition number of \mathbf{L} , $\lambda_{\max}(\mathbf{L})$ and $\lambda_{\min}(\mathbf{L})$ are the maximum and minimum eigenvalues of \mathbf{L} , respectively. Since $\kappa(\mathbf{L}) \rightarrow \infty$, relying solely on the Laplacian matrix can result in a poorly conditioned problem, particularly when solving an

³In this paper, we focus on the node and some graph classification tasks. However, our framework can be easily extended to other problem domains, such as link prediction.

⁴The condition number $\kappa(\mathbf{L})$ associated with the square matrix \mathbf{L} is a measure of how well- or ill-conditioned is the inversion of \mathbf{L} .

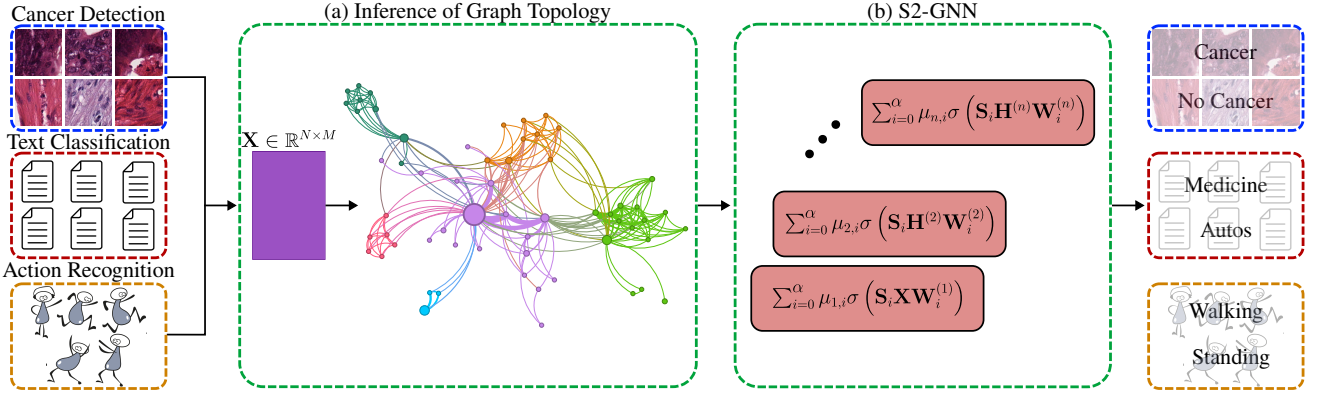


Fig. 1: The pipeline of our S2-GNN algorithm. S2-GNN can be used in a broad range of data such as images, text, and videos, among others. However, the step of mapping the original dataset to the data matrix $\mathbf{X} \in \mathbb{R}^{N \times M}$ could be different in each case. Our framework is composed of (a) inference of the graph topology and (b) the S2-GNN architecture.

optimization problem [14]. On the other hand, for the Sobolev term, we observe that:

$$\mathbf{L} + \epsilon \mathbf{I} = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^T + \epsilon \mathbf{I} = \mathbf{U} (\mathbf{\Lambda} + \epsilon \mathbf{I}) \mathbf{U}^T. \quad (5)$$

Therefore, $\lambda_{\min}(\mathbf{L} + \epsilon \mathbf{I}) = \epsilon$, *i.e.*, $\mathbf{L} + \epsilon \mathbf{I}$ is positive definite ($\mathbf{L} + \epsilon \mathbf{I} > 0$) for $\epsilon > 0$, and:

$$\kappa(\mathbf{L} + \epsilon \mathbf{I}) = \frac{|\lambda_{\max}(\mathbf{L} + \epsilon \mathbf{I})|}{|\lambda_{\min}(\mathbf{L} + \epsilon \mathbf{I})|} = \frac{\lambda_{\max}(\mathbf{L}) + \epsilon}{\epsilon} < \kappa(\mathbf{L}); \quad \forall \epsilon > 0. \quad (6)$$

Namely, $\mathbf{L} + \epsilon \mathbf{I}$ has a better condition number than \mathbf{L} . The significance of a better condition number in the context of GNNs might not be immediately apparent since the inverses of the Laplacian or adjacency matrices are not required for performing the propagation rules. However, several studies have highlighted the adverse effects of poorly conditioned matrices. For instance, Kipf and Welling [3] employed a renormalization trick $(\mathbf{A} + \mathbf{I})$ in their filtering operation to avoid exploding/vanishing gradients. Similarly, Wu *et al.* [29] demonstrated that adding the identity matrix to \mathbf{A} shrinks the graph spectral domain, resulting in a low-pass-type filter.

The previous theoretical analysis shows the benefits of the Sobolev norm in two aspects: 1) the computation of diverse frequencies in (3), and 2) the better condition number in (6).

B. Sparse Sobolev Norm

The use of \mathbf{L} , \mathbf{A} , or their normalized forms in GNNs is computationally efficient due to their sparsity. This allows us to perform a small number of sparse matrix operations, which is advantageous for computational efficiency. However, when considering the Sobolev norm, the term $(\mathbf{L} + \epsilon \mathbf{I})^\rho$ can rapidly become a dense matrix for large values of ρ , leading to computational and memory issues as illustrated in Fig. 2. To address this problem, we employ a sparse Sobolev norm that preserves the sparsity level, ensuring computational feasibility and efficient memory usage.

Definition 3. Let $\mathbf{L} \in \mathbb{R}^{N \times N}$ be the Laplacian matrix of a graph G . For fixed parameters $\epsilon \geq 0$ and $\rho \in \mathbb{N}$, the sparse Sobolev term for GNNs is introduced as the ρ Hadamard

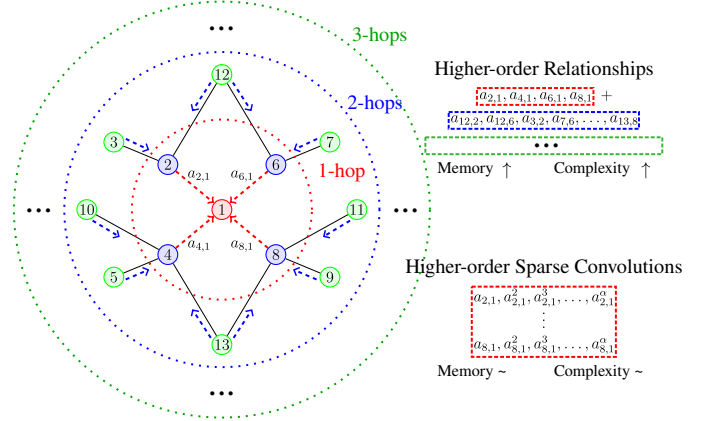


Fig. 2: Process of using higher-order relationships in GNNs vs. higher-order sparse convolutions.

multiplications of $(\mathbf{L} + \epsilon \mathbf{I})$ (also known as the Hadamard powers) such that:

$$(\mathbf{L} + \epsilon \mathbf{I})^{(\rho)} = \underbrace{(\mathbf{L} + \epsilon \mathbf{I}) \circ (\mathbf{L} + \epsilon \mathbf{I}) \circ \dots \circ (\mathbf{L} + \epsilon \mathbf{I})}_{\rho \text{ times}}. \quad (7)$$

For example, $(\mathbf{L} + \epsilon \mathbf{I})^{(2)} = (\mathbf{L} + \epsilon \mathbf{I}) \circ (\mathbf{L} + \epsilon \mathbf{I})$. Thus, the sparse Sobolev norm is given by:

$$\|\mathbf{x}\|_{(\rho), \epsilon} \triangleq \|(\mathbf{L} + \epsilon \mathbf{I})^{(\rho/2)} \mathbf{x}\|. \quad (8)$$

Let $\langle \mathbf{x}, \mathbf{y} \rangle_{(\rho), \epsilon} = \mathbf{x}^T (\mathbf{L} + \epsilon \mathbf{I})^{(\rho)} \mathbf{y}$ be the inner product between two graph signals \mathbf{x} and \mathbf{y} that induces the associated sparse Sobolev norm.

Theorem 1. The sparse Sobolev norm $\|\mathbf{x}\|_{(\rho), \epsilon} \triangleq \|(\mathbf{L} + \epsilon \mathbf{I})^{(\rho/2)} \mathbf{x}\|$ satisfies the basic properties of vector norms for $\epsilon > 0$ (for $\epsilon = 0$, we obtain a semi-norm).

Proof: See Appendix A.

The sparse Sobolev term in (7) possesses the property of maintaining the same sparsity level as $\mathbf{L} + \epsilon \mathbf{I}$ for any value of ρ . It is worth noting that $(\mathbf{L} + \epsilon \mathbf{I})^\rho$ is equivalent to the sparse Sobolev term under two conditions: 1) restricting ρ to be in \mathbb{N} , and 2) replacing the conventional matrix multiplication with

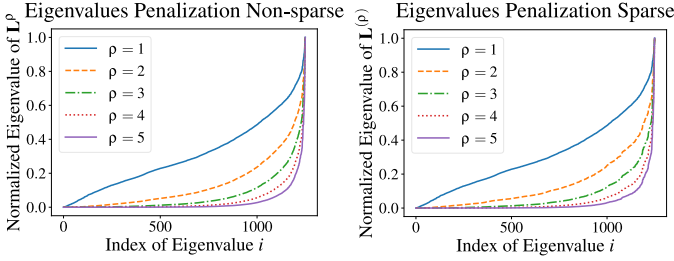


Fig. 3: Eigenvalues penalization for the non-sparse and sparse matrix multiplications of the combinatorial Laplacian matrix.

the Hadamard product. While the theoretical properties of the Sobolev norm in (3) and (6) do not directly extend to its sparse counterpart, we can gain some theoretical insights by leveraging the concepts of Kronecker products and the Schur product theorem [12].

Theorem 2. Let \mathbf{L} be any Laplacian matrix of a graph with eigenvalue decomposition $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$. We observe that:

$$\mathbf{L}^{(\rho)} = \mathbf{P}_{(\rho)}^T (\mathbf{U} \otimes \mathbf{U}_{(\rho-1)}) (\mathbf{\Lambda} \otimes \mathbf{\Lambda}_{(\rho-1)}) (\mathbf{U}^T \otimes \mathbf{U}_{(\rho-1)}^T) \mathbf{P}_{(\rho)}, \quad (9)$$

where $\mathbf{U}_{(\rho-1)}$ and $\mathbf{\Lambda}_{(\rho-1)}$ are respectively the matrices of eigenvectors and eigenvalues of $\mathbf{L}^{(\rho-1)}$, $\mathbf{P}_{(\rho)} \in \{0, 1\}^{N^2 \times N}$ is a partial permutation matrix, and:

$$\mathbf{L}^{(2)} = \mathbf{P}_N^T (\mathbf{U} \otimes \mathbf{U}) (\mathbf{\Lambda} \otimes \mathbf{\Lambda}) (\mathbf{U}^T \otimes \mathbf{U}^T) \mathbf{P}_N. \quad (10)$$

Proof: See Appendix B.

Theorem 2 provides the closed-form solution for the spectrum of the Hadamard power of the Laplacian matrix. It reveals that the spectrum of the Hadamard product is a compressed representation of the Kronecker product of its spectral components. In the case of the sparse Sobolev term utilized in S2-GNN, denoted by $(\mathbf{L} + \epsilon \mathbf{I})^{(\rho)}$, the spectral components of the graph undergo changes for each value of ρ , as shown in (9).

By applying the Schur product theorem, which states that the Hadamard product between two positive definite matrices is also positive definite, we can analyze the condition number of the Hadamard powers [12]. We know that $(\mathbf{L} + \epsilon \mathbf{I})^{(\rho)} \succ 0$; $\forall \epsilon > 0$ since $(\mathbf{L} + \epsilon \mathbf{I}) \succ 0$; $\forall \epsilon > 0$, and therefore $\kappa((\mathbf{L} + \epsilon \mathbf{I})^{(\rho)}) < \infty$. For the adjacency matrix \mathbf{A} , its eigenvalues lie within the interval $[-d, d]$, where d is the maximum degree of the vertices in G (see Theorem 8.5 in [36]). Therefore, we can bound the eigenvalues of \mathbf{A} into the interval $[-1, 1]$ by normalizing the adjacency matrix such that $\mathbf{A}_N = \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}$. Thus, we can establish that $\mathbf{A}_N + \epsilon \mathbf{I} \succ 0$; $\forall \epsilon > 1$, and subsequently, $(\mathbf{A}_N + \epsilon \mathbf{I})^{(\rho)} \succ 0$; $\forall \epsilon > 1$. We can assert that the theoretical developments presented in Section IV-A for the sparse Sobolev norm hold to some extent, including a more diverse set of frequencies and a better condition number. Figure 3 depicts five normalized eigenvalue penalizations for \mathbf{L}^ρ (non-sparse) and $\mathbf{L}^{(\rho)}$ (sparse), revealing that the normalized spectrum of \mathbf{L}^ρ and $\mathbf{L}^{(\rho)}$ exhibit a high degree of similarity.

C. Graph Sparse Sobolev Layer

In the S2-GNN architecture, we introduce the propagation rule using the graph convolution filter in Definition 1. Let $\mathbf{S} \in$

$\mathbb{R}^{N \times N}$ be the shift operator of a graph, and let $\mathbf{H}^{(l)} \in \mathbb{R}^{N \times H_l}$ be the output of the previous layer of the GNN (matrix of node embeddings), such that $\mathbf{H}^{(1)} = \mathbf{X} \in \mathbb{R}^{N \times M}$ is an M -dimensional graph signal (matrix of features). The propagation rule of the ρ th branch of S2-GNN is defined as:

$$\mathbf{B}_\rho^{(l+1)} = \sigma \left(\mathbf{S}_\rho \mathbf{H}^{(l)} \mathbf{W}_\rho^{(l)} \right), \quad (11)$$

where $\sigma(\cdot)$ is some activation function, $\mathbf{S}_\rho = \mathbf{S}^{(\rho)}$, and $\mathbf{W}_\rho^{(l)} \in \mathbb{R}^{H_l \times H_{l+1}}$ is the matrix of trainable weights in the ρ th branch of the layer l . The matrix $\mathbf{W}_\rho^{(l)}$ corresponds to a Multi-Layer Perceptron (MLP) with H_{l+1} hidden units. In practice, we typically include a bias term in the MLP; however, we have omitted it here for simplicity. Notice that (11) is an approximation of the graph convolutional filter in Definition 1 when substituting the regular matrix power by the Hadamard power, $q_i = 1$ for $i = \rho$, $q_i = 0$; $\forall i \neq \rho$, the input signal is a H_l -dimensional graph signal, and adding an MLP at the end of the filtering process.

The S2-GNN architecture consists of multiple layers, where each layer involves computing a cascade of $\alpha + 1$ propagation rules (11) with increasing Hadamard powers. Finally, the outputs from all branches are combined using some fusion mechanism, such as a linear combination layer or an MLP fusion technique.

Linear combination layer: Let $\mathbf{B}_\rho^{(l+1)} \in \mathbb{R}^{N \times H_{l+1}}$ be the output of the ρ th branch in the layer l of S2-GNN. The output of the l th layer of S2-GNN using the linear combination layer is given by:

$$\mathbf{H}^{(l+1)} = \sum_{i=0}^{\alpha} \mu_{l,i} \mathbf{B}_i^{(l+1)}, \quad (12)$$

where $\mu_{l,i}$ is the learnable weight associated with $\mathbf{B}_i^{(l+1)}$, and we consider $\mathbf{B}_0^{(l+1)} = \sigma \left(\mathbf{I} \mathbf{H}^{(l)} \mathbf{W}_0^{(l)} \right) = \sigma \left(\mathbf{H}^{(l)} \mathbf{W}_0^{(l)} \right)$.

MLP fusion: Let $\mathbf{B}_\rho^{(l+1)} \in \mathbb{R}^{N \times H_{l+1}}$ be the output of the ρ th branch in the layer l of S2-GNN. The output of the l th layer of S2-GNN using the MLP fusion is given by:

$$\mathbf{H}^{(l+1)} = \left[\mathbf{B}_0^{(l+1)}, \mathbf{B}_1^{(l+1)}, \dots, \mathbf{B}_\alpha^{(l+1)} \right] \mathbf{W}_{\text{MLP}}^{(l)}, \quad (13)$$

where $\left[\mathbf{B}_0^{(l+1)}, \mathbf{B}_1^{(l+1)}, \dots, \mathbf{B}_\alpha^{(l+1)} \right]$ is a concatenation of the matrices $\mathbf{B}_i^{(l+1)} \forall 0 \leq i \leq \alpha$, $\mathbf{B}_0^{(l+1)} = \sigma \left(\mathbf{H}^{(l)} \mathbf{W}_0^{(l)} \right)$, and $\mathbf{W}_{\text{MLP}}^{(l)} \in \mathbb{R}^{(\alpha+1)H_{l+1} \times H_{l+1}}$ is the matrix of learnable weights to combine the $\alpha + 1$ branches.

It is worth noting that a layer of S2-GNN, as described in (12) or (13), can be viewed as a mapping $H : \mathbb{R}^{N \times H_l} \rightarrow \mathbb{R}^{N \times H_{l+1}}$. In practice, the choice of the fusion mechanism is determined by a hyperparameter optimization algorithm that selects the most suitable mechanism for a given dataset. Other techniques, like attention-based fusion methods [37] could also be applied into our model. However, such techniques could increase the overall complexity of S2-GNN due to the computation of the attention mechanism.

1) Re-normalization Trick and Sobolev Term: In practice, we utilize a normalized Sobolev version of the adjacency matrix as the shift operator for computing our propagation rule. Specifically, we set $\mathbf{S}_{(\rho)} = \bar{\mathbf{D}}_\rho^{-\frac{1}{2}} \bar{\mathbf{A}}_\rho \bar{\mathbf{D}}_\rho^{-\frac{1}{2}}$, where $\bar{\mathbf{A}}_\rho =$

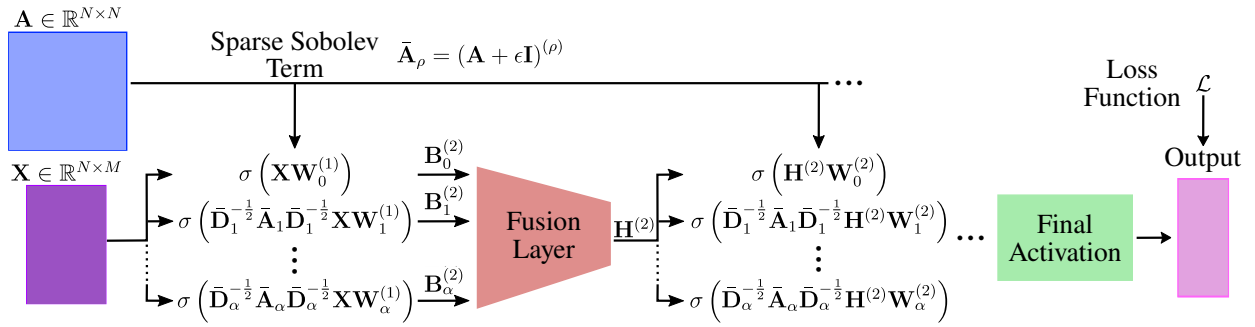


Fig. 4: Basic configuration of our S2-GNN architecture with n layers and α branches per layer.

$(\mathbf{A} + \epsilon \mathbf{I})^{(\rho)}$ represents the ρ th sparse Sobolev term of \mathbf{A} and $\bar{\mathbf{D}}_\rho$ denotes the degree matrix of $\bar{\mathbf{A}}_\rho$. This operation can be interpreted from a message-passing perspective, where we aggregate the embeddings from the 1-hop neighborhood with weights proportional to the edge attributes, node degrees, and the Hadamard power involved in that operation. Unlike the GCN operation, our approach incorporates the Hadamard power term. Our operation does not compute attention coefficients like GAT or graph Transformers, but it does learn the importance of each branch through the fusion layer, either by using a linear combination or an MLP operation.

Figure 4 illustrates the basic configuration of S2-GNN, serving as the core for constructing more complex architectures based on our filtering operation. Additional components such as batch normalization and residual connections can be added. Notice that our graph convolution is efficiently computed since the term $\bar{\mathbf{D}}_\rho^{-\frac{1}{2}} \bar{\mathbf{A}}_\rho \bar{\mathbf{D}}_\rho^{-\frac{1}{2}}$; $\forall \rho \in \{1, 2, \dots, \alpha\}$ remains the same across all layers (allowing for offline computation), and these terms remain sparse for any value of ρ (given the sparsity of \mathbf{A}). S2-GNN can employ conventional activation functions like ReLU or softmax, as well as typical loss functions \mathcal{L} suited to the specific task. The basic architecture of S2-GNN is defined by the number of filters α in each layer, the parameter ϵ , the number of hidden units in each $\mathbf{W}_\rho^{(l)}$, and the number of layers n . When constructing weighted graphs with Gaussian kernels, the edge weights fall within the interval $[0, 1]$. As a result, large values of ρ could lead to $\bar{\mathbf{A}}_\rho = \mathbf{0}$, and the diagonal elements of $\bar{\mathbf{D}}_\rho^{-\frac{1}{2}}$ may become ∞ . Similarly, when α is very large, the resulting architectures become wide with a significant number of parameters, making it desirable to maintain a reasonable value for α . In our experiments, we evaluate S2-GNN architectures up to $\alpha = 6$.

D. Relationship with Graph Convolutional Network and Computational Complexity

Kipf and Welling [3] proposed one of the most successful yet simple GNNs, called Graph Convolutional Networks (GCNs), using the following propagation rule:

$$\bar{\mathbf{H}}^{(l+1)} = \sigma(\bar{\mathbf{D}}^{-\frac{1}{2}} \bar{\mathbf{A}} \bar{\mathbf{D}}^{-\frac{1}{2}} \bar{\mathbf{H}}^{(l)} \mathbf{W}^{(l)}). \quad (14)$$

Here $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\tilde{\mathbf{D}}$ represents the degree matrix of $\tilde{\mathbf{A}}$, $\bar{\mathbf{H}}^{(l)} \in \mathbb{R}^{N \times H_l}$ denotes the output of the previous layer l , where $\bar{\mathbf{H}}^{(1)} = \mathbf{X} \in \mathbb{R}^{N \times M}$ is matrix of input features, and

$\mathbf{W}^{(l)} \in \mathbb{R}^{H_l \times H_{l+1}}$ is the matrix of trainable weights in layer l . The motivation behind the propagation rule in (14) is based on the first-order approximation of localized spectral filters on graphs [2]. We can observe that GCN is an approximation of the graph convolutional filter in Definition 1, where $q_1 = 1$, $q_i = 0$; $\forall i \neq 1$, the input signal is an H_l -dimensional graph signal, and an MLP is added at the end of the filtering process. Notice that our S2-GNN layer is a generalization of the GCN layer when only the branch $\rho = 1$ is used in (11) without the fusion layer, and the shift operator is the normalized adjacency matrix.

The computational complexity of a GCN with n layers is $\mathcal{O}(n|\mathcal{E}|)$. The GCN model is widely used in practice due to its simplicity, good performance, and low computational complexity. For S2-GNN, the computational complexity of each branch remains $\mathcal{O}(|\mathcal{E}|)$ since our sparse Sobolev term maintains the same sparsity level in the graph representation for any value of ρ . When considering α branches, the complexity becomes $\mathcal{O}(\alpha|\mathcal{E}| + 2\alpha)$ per layer, taking into account the linear combination layer and the branch for $\rho = 0$. With n layers, the computational complexity of our architecture is $\mathcal{O}(n(\alpha|\mathcal{E}| + 2\alpha)) = \mathcal{O}(n\alpha|\mathcal{E}| + 2n\alpha)$. Therefore, S2-GNN achieves a computational complexity similar to GCN while delivering better performance on several benchmark datasets. It is worth noting that the exact complexity of both methods, S2-GNN and GCN, also depends on factors such as the feature dimension, the hidden units, and the number of nodes in the graph, which have been omitted for simplicity and clarity.

E. Theoretical Stability Analysis

Here, we aim to study the effect of graph perturbations on the prediction performance of the proposed network. Similarly, we analyze the stability [23] of S2-GNN against the existence of erroneous edges. Particularly, we consider the deviation of the adjacency matrix \mathbf{A} with the error (or perturbation) matrix \mathbf{E} as follows:

$$\hat{\mathbf{A}} = \mathbf{A} + \mathbf{E}; \quad \mathbf{E} \in \mathcal{P}, \quad (15)$$

$$\mathcal{P} = \{\mathbf{E} \in \mathbb{R}^{N \times N} \mid \mathbf{E} = \mathbf{E}^T, |e_{i,j}| \leq a_{i,j}; \forall i, j, e_{i,i} = 0; \forall i\}.$$

Note that the symmetry of the error matrix \mathbf{E} stems from the underlying graph being undirected. The conditions on \mathcal{P} imply that: 1) the perturbations cannot cause negative edge weights, and 2) we do not consider edge perturbations causing self-loops.

For the sake of simplicity and without loss of generality, we consider the true and perturbed output of a simplified [29] version of the S2-GNN (SS2-GNN), which can be stated as:

$$\begin{aligned} \mathbf{Y} &= \sigma \left(\mathbf{L}^{(\rho)} \mathbf{X} \mathbf{W} \right), \\ \hat{\mathbf{Y}} &= \sigma \left(\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \hat{\mathbf{W}} \right), \end{aligned} \quad (16)$$

where $\hat{\mathbf{L}}$ is the perturbed Laplacian matrix. The next theorem obtains an upper bound on the distance between the true and perturbed outputs (*i.e.*, $d(\mathbf{Y}, \hat{\mathbf{Y}}) = \|\mathbf{Y} - \hat{\mathbf{Y}}\|$) for the SS2-GNN in the case of having true and perturbed versions of the graph and graph filter coefficients. First, we need to mention the definition of being Lipschitz for a general function $\sigma(\cdot)$ in \mathbb{R} .

Definition 4. We call a function $\sigma(\cdot)$ as Lipschitz if there exists a positive constant ϕ such that:

$$\forall x_1, x_2 \in \mathbb{R} : |\sigma(x_1) - \sigma(x_2)| \leq \phi |x_1 - x_2|. \quad (17)$$

Theorem 3. With the assumptions of the non-linear function $\sigma(\cdot)$ being Lipschitz, the node feature matrix \mathbf{X} is column-wisely normalized as $\|\mathbf{X}\|_F = \sqrt{v}$, $\|\mathbf{W} - \hat{\mathbf{W}}\| \leq \delta_{\mathbf{W}}$, $\|\mathbf{E}\| \leq \xi$, and $\|\mathbf{D}_E\| \leq \xi_D$, where \mathbf{D}_E is the degree matrix of \mathbf{E} , the difference between the true and perturbed outputs of an SS2-GNN is bounded by:

$$\begin{aligned} d(\mathbf{Y}, \hat{\mathbf{Y}}) &= \left\| \sigma \left(\mathbf{L}^{(\rho)} \mathbf{X} \mathbf{W} \right) - \sigma \left(\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \hat{\mathbf{W}} \right) \right\| \\ &\leq \sqrt{v} \left[(d^{max} + \epsilon)^\rho + \|\hat{\mathbf{A}}^{(\rho)}\| \right] \delta_{\mathbf{W}} + \mathcal{O}(\xi^2) + \mathcal{O}(\xi_D^2) \\ &\quad + \rho \sqrt{v} \left[\eta \|\mathbf{E}\| + (d^{max} + \epsilon)^{\rho-1} d_e^{max} \right] \|\mathbf{W}\|, \end{aligned} \quad (18)$$

where $\eta \triangleq \min(r_1(\mathbf{A}^{(\rho-1)}), c_1(\mathbf{A}^{(\rho-1)}))$, $r_1(\cdot)$ and $c_1(\cdot)$ denote the row and column of the involved matrix with highest norm-2 among the other rows and columns, respectively.

Proof: See Appendix C.

The obtained bound in (18) reveals interesting aspects of the S2-GNN stability against possible perturbations. Precisely, this bound relies on the Hadamard order (ρ), which implies that a high degree of Hadamard multiplications could make the network more susceptible to perturbations. Besides, the absolute dependency of this bound on $\|\mathbf{E}\|$, ξ , and ξ_D , which are respectively representations of the perturbation (noise) power, depicts the effect of perturbation amplitudes on the network stability. The existence of $\|\mathbf{W}\|$ and $\delta_{\mathbf{W}}$ emphasizes the choice of the network size and selection of optimization approach (*e.g.*, stochastic gradient descent) for reaching the suboptimal solution for $\|\mathbf{W}\|$, which has a direct impact on the distance (*i.e.*, $\delta_{\mathbf{W}}$) from the true values of \mathbf{W} . Also, as can be seen in (18), an unnecessarily high value of ϵ could have considerably negative effects on the network robustness, especially in the case of high Hadamard order (ρ). Finally, due to the existence of η and more importantly, d^{max} , in the upper bound (18), the smaller values for d^{max} (*i.e.*, sparse graphs) help in stability and robustness of the network. Therefore, the proposed network can effectively benefit from the sparse underlying structures, which are widely found in real-world

applications. We empirically validate the stability analysis of S2-GNN in Appendix D.

V. EXPERIMENTAL EVALUATION

S2-GNN is compared to nine state-of-the-art methods in GNNs. These methods include Chebyshev graph convolutions (Cheby) [2], Graph Convolutional Networks (GCN) [3], Graph Attention Networks (GAT) [4], Simple Graph Convolution (SGC) [29], ClusterGCN [30], Scalable Inception Graph Neural network (SIGN) [10], SuperGAT [28], a graph Transformer [21], and GAT version 2 (GATv2) [20].

A. Datasets

We evaluate the performance of S2-GNN on multiple datasets, which include tissue phenotyping in colon cancer histology images [15], text classification of news [16], activity recognition using sensors [17], recognition of spoken letters [18], as well as the commonly used node classification benchmarks Cora [38], Citeseer [39], Pubmed [40], and OGBN-proteins [19]. These diverse datasets provide a comprehensive evaluation of S2-GNN's performance across different domains. **The colorectal cancer histology** dataset comes from ten anonymized tissue slides from the University Medical Center Mannheim [15]. The dataset consists of eight classes: tumor epithelium, simple stroma, complex stroma, immune cells, debris, normal mucosal glands, adipose tissue, and background (no tissue). This work does not explore the feature representation, and only uses Local Binary Patterns (LBP) to represent each image. We explore two partitions of this dataset: 1) Cancer Binary (Cancer-B) considers only tumor epithelium and simple stroma, and 2) Cancer Multi-class (Cancer-M) tests all classes.

The 20 Newsgroups dataset (20News) contains approximately 20,000 news articles categorized into 20 classes [16]. In this work, we use a subset of ten categories: computer graphics, windows operative system, IBM hardware, MAC hardware, autos, motorcycles, cryptography, electronics, medicine, and space. Each news document is represented using the Term Frequency - Inverse Document Frequency (TFIDF) as described in [41].

The Human Activity Recognition dataset (HAR) comprises recordings of 30 individuals engaged in various activities of daily living [17]. The participants wore waist-mounted smartphones with embedded sensors, and the dataset consists of six activity classes: standing, sitting, laying down, walking, walking downstairs, and walking upstairs. The recordings are represented as in [17].

The Isolated Letter recognition dataset (Isolet) includes sound recordings from the 26 letters of the English alphabet spoken by 30 speakers. Isolet contains 6,238 samples represented as 617-dimensional feature vectors. The feature representation method is explained in [18].

The Cora, Citeseer, and Pubmed datasets are well-known citation networks. The datasets consist of sparse bag-of-words feature vectors for each document (node) and a list of citation links between documents. In this work, we treat the citation links as undirected edges as in [3]. To fulfill the requirement of

TABLE I: Statistics of the datasets in the experimental framework of this work.

	Cancer-B	Cancer-M	20News	HAR	Isolet	Cora	Citeseer	Pubmed	OGBN-proteins
Nodes	1, 250	5, 000	3, 000	3, 000	3, 000	2, 485	2, 120	19, 717	30, 000
Edges*	25, 734	101, 439	62, 399	87, 032	43, 330	205, 928	183, 572	1, 723, 360	13, 251, 798
Edges†	19, 163	81, 462	47, 496	61, 400	31, 130	NA	NA	NA	NA
Features	38	38	2, 000	561	617	1, 433	3, 703	500	8
Classes	2	8	10	6	26	7	6	3	112
Undirected	✓	✓	✓	✓	✓	✓	✓	✓	✓

* This number of edges corresponds to the k -NN method for the constructed graphs, and the outputs of the GDC method for the citation networks.

† This number of edges corresponds to the method in [25] for the constructed graphs.

weighted graphs, as mentioned in Section III, we preprocess the *unweighted* citation networks using the Graph Diffusion Convolution (GDC) method [42].

The Open Graph Benchmark Proteins dataset (OGBN-Proteins) is a comprehensive graph-based dataset designed for protein-protein interaction networks [19]. The dataset features more than 132,000 nodes (proteins) and 39 million edges, reflecting a diverse range of biological relationships. Each node in the graph is associated with 8-dimensional feature vectors, capturing essential biological characteristics of the proteins. The primary task of the dataset is to predict protein functions, with each protein potentially linked to one or more of the 112 function categories. Due to computational constraints, the scope of the data used in the experiments is reduced to approximately 25% of the full dataset, which equates to around 30,000 nodes. This subset selection ensures manageable computational demands while still providing a representative sample for analysis.

B. Experiments

For the Cancer-B, Cancer-M, 20News, HAR, and Isolet datasets, we generate graphs using two methods: k -Nearest Neighbors (k -NN) and the algorithm described in [25]. In our results, we refer to the graphs generated by the latter method as *learned graphs*. Regarding Cora, Citeseer, and Pubmed datasets, we consider the largest connected component of the graph following common practices [42]. OGBN-proteins is a dataset that comprises a single weighted graph, where edges represent the confidence level of interactions between proteins, eliminating the necessity for preprocessing steps like GDC. This simplifies the data preparation phase and allows for a more direct application of the model. Table I presents a comprehensive summary of the statistics for all the graphs used in the study. To optimize the hyperparameters of citation networks, we employ a random search procedure that maximizes the average accuracy on the validation sets. For OGBN-proteins, we empirically obtain the hyperparameters, focusing on the critical importance of learning rate and hidden channels over other hyperparameters. We split the datasets into training, validation, and testing sets. Initially, we divide the data into a development set and a test set, ensuring that the test set is not used during the hyperparameter optimization process. Following the experimental framework of [42], we optimize the hyperparameters for each dataset-preprocessing combination separately using a random search over five data splits, with multiple repetitions for enhanced robustness. For

the Cora, Citeseer, and Pubmed datasets, the development set consists of 1,500 nodes, while the remaining nodes are used for testing. Similarly, the train set includes 20 nodes from each class, and the remaining nodes are utilized for validation. As for the other datasets, we employ a 10/45/45 split, meaning that 10% of the nodes are assigned for training, 45% for validation, and 45% for testing. Finally, we report average accuracies on the test set accompanied by 95% confidence intervals calculated via bootstrapping with 1,000 samples, with each model being tested using 50 different seeds to ensure robustness and reliability of the results.

C. Implementation Details

S2-GNN and the state-of-the-art methods are implemented using PyTorch Geometric (PyG) [43]. The learned graph methods are implemented using the GSP toolbox [44]. The value of k in the k -NN method is set to 30. Rectified Linear Unit (ReLU) and log-softmax are used as activation functions in all methods. The hyperparameters such as the number of GNN layers, hidden units, learning rate, weight decay, and dropout rate are optimized for each method using the validation set in each experiment. For Cheby we optimize the Chebyshev filter size K . For SIGN we optimize the number of powers. For attention-based techniques like GAT and Transformers, we optimize the number of heads. For S2-GNN we optimize α , ϵ , and whether the fusion mechanism is a linear combination layer or an MLP fusion as introduced in (12) or (13). The hyperparameter search space for each method is defined as follows: 1) learning rate $lr \in [0.005, 0.02]$, 2) weight decay $wd \in [0.0001, 0.001]$, 3) hidden units of each graph convolutional layer $hu \in \{16, 32, 64\}$, 4) dropout $d \in [0.3, 0.7]$, 5) the number of layers $L \in \{2, 3, 4, 5\}$, 6) Chebyshev filter size $K \in \{1, 2, 3\}$, 7) heads of attention mechanism $hd \in \{1, 2, \dots, 6\}$, 8) powers for SIGN $pw \in \{1, 2, 3\}$, 9) $\alpha \in \{1, 2, \dots, 6\}$, 10) $\epsilon \in [0.5, 2]$, 11) fusion layer $fl \in \{\text{Linear}, \text{MLP}\}$. For the GDC method applied to Cora, Citeseer, and Pubmed datasets, personalized PageRank is used with a teleport probability of 0.05, and the top 128 entries with the highest mass per column are selected to sparsify the output of GDC. The code and constructed graphs will be publicly available upon publication under the MIT license⁵.

D. Results and Discussion

Table II provides a summary of the comparison between S2-GNN and state-of-the-art methods in terms of Accuracy for

⁵<https://github.com/jhonygiraldo/S2-GNN>

TABLE II: Accuracy (in %) for the state-of-the-art methods and our S2-GNN algorithm in five datasets, inferring the graphs with k -NN or the method for learning smooth graphs in [25].

Model	k -NN Graphs					Learned Graphs				
	Cancer-B	Cancer-M	20News	HAR	Isolet	Cancer-B	Cancer-M	20News	HAR	Isolet
Cheby [2]	82.55 \pm 4.95	59.52 \pm 1.42	69.53 \pm 1.47	85.64 \pm 2.92	71.52 \pm 1.42	87.64 \pm 3.69	57.50 \pm 2.12	72.49\pm0.82	87.07 \pm 3.57	59.14 \pm 2.50
GCN [3]	71.31 \pm 5.72	50.87 \pm 2.39	63.08 \pm 2.36	64.28 \pm 4.90	63.61 \pm 1.76	80.50 \pm 4.77	52.45 \pm 3.00	69.83 \pm 1.25	82.85 \pm 3.00	71.27 \pm 2.01
GAT [4]	68.67 \pm 5.24	48.87 \pm 1.94	48.90 \pm 2.17	70.13 \pm 3.65	58.93 \pm 2.03	77.69 \pm 4.52	51.23 \pm 2.04	54.30 \pm 2.13	75.45 \pm 3.97	58.78 \pm 1.75
SGC [29]	72.00 \pm 4.56	42.22 \pm 1.88	54.24 \pm 2.40	39.39 \pm 2.72	39.90 \pm 2.05	70.82 \pm 4.87	43.97 \pm 1.97	72.13 \pm 1.31	38.22 \pm 2.71	43.21 \pm 1.85
ClusterGCN [30]	83.03 \pm 4.43	56.85 \pm 1.25	67.95 \pm 1.55	72.67 \pm 4.70	59.05 \pm 3.20	85.63 \pm 4.26	58.58 \pm 1.76	62.12 \pm 1.59	65.98 \pm 4.45	69.54 \pm 2.37
SIGN [10]	89.67\pm0.39	62.45\pm0.17	72.34\pm0.25	91.58\pm0.24	84.11\pm0.24	90.30\pm0.38	64.99\pm0.20	74.10\pm0.30	92.88\pm0.23	84.17\pm0.27
SuperGAT [28]	69.54 \pm 5.09	40.04 \pm 1.84	54.40 \pm 1.95	72.66 \pm 3.76	60.57 \pm 2.17	78.15 \pm 4.42	52.73 \pm 2.07	59.53 \pm 1.96	74.93 \pm 4.00	61.49 \pm 1.68
Transformer [21]	74.23 \pm 5.65	54.20 \pm 1.70	59.06 \pm 2.75	79.66 \pm 3.34	65.68 \pm 2.42	79.23 \pm 4.83	52.07 \pm 2.21	56.19 \pm 2.83	78.30 \pm 2.81	65.36 \pm 2.18
GATv2 [20]	62.89 \pm 5.21	47.15 \pm 2.07	46.56 \pm 2.04	73.28 \pm 4.02	64.59 \pm 1.40	79.28 \pm 4.01	45.80 \pm 1.78	51.17 \pm 2.14	75.09 \pm 4.15	66.46 \pm 1.87
S2-GNN (ours)	93.39\pm0.41	68.48\pm0.28	72.09\pm0.35	93.92\pm0.22	86.57\pm0.27	93.50\pm0.39	68.88\pm0.25	72.19 \pm 0.41	94.50\pm0.19	86.61\pm0.29

The best and second-best performing methods on each dataset are shown in **red** and **blue**, respectively.

TABLE III: Accuracy (in %) comparison in the citation networks using the graph diffusion convolution and OGBN-proteins.

Model	Cora	Citeseer	Pubmed	OGBN-proteins
Cheby [2]	80.79\pm1.43	69.41\pm1.09	78.25\pm1.57	63.55\pm2.89
GCN [3]	70.22 \pm 2.91	58.85 \pm 2.88	56.64 \pm 5.81	58.86 \pm 7.63
GAT [4]	60.29 \pm 4.25	56.18 \pm 2.76	56.57 \pm 5.11	41.58 \pm 4.42
SGC [29]	64.23 \pm 3.66	56.74 \pm 2.82	70.35 \pm 2.89	56.87 \pm 6.42
ClusterGCN [30]	75.95 \pm 2.53	63.93 \pm 1.96	59.24 \pm 6.31	50.01 \pm 0.02
SuperGAT [28]	65.71 \pm 3.69	55.80 \pm 3.27	OOM	58.59 \pm 0.79
Transformer [21]	66.86 \pm 3.47	62.85 \pm 2.19	61.83 \pm 5.46	39.75 \pm 4.76
GATv2 [20]	62.16 \pm 3.29	53.24 \pm 2.25	56.39 \pm 5.53	50.06 \pm 0.13
S2-GNN (ours)	82.47\pm0.42	68.95\pm0.55	77.64\pm1.05	70.26\pm4.08

OOM: Out Of Memory in a GPU Nvidia A100 40GB.

TABLE IV: Sparsity (%) for the non-sparse Sobolev term $(\mathbf{L} + \epsilon\mathbf{I})^\rho$ for several values of ρ .

Power	Cancer-B	Cancer-M	20News	HAR	Isolet
$\rho = 1$	96.63	99.17	98.58	98.03	99.01
$\rho = 2$	80.33	94.05	65.65	82.29	92.73
$\rho = 3$	53.82	82.48	0.615	59.74	75.92
$\rho = 4$	29.32	67.67	0	46.05	46.19
$\rho = 5$	12.46	52.55	0	35.07	20.41

Zero sparsity percentage means that the matrix is completely dense.

Cancer-B, Cancer-M, 20News, HAR, and Isolet. In most cases, S2-GNN demonstrates the best performance, except for the 20News dataset. Both Cheby and SIGN methods, which rely on higher-order graph convolutions, also achieve competitive results. However, the disadvantage of these non-sparse higher-order methods is that the graph matrix representations (either the adjacency or Laplacian matrices) become denser with higher powers of the shift operator. S2-GNN achieves better performances while maintaining sparsity in the graph matrix representations and providing a good confidence interval in the results. Furthermore, Table II highlights that learned graphs on the right side generally outperform the k -NN graphs on the left for most methods (Appendix E presents a study in the homophily level of k -NN and learned graphs). The performance of the learned graphs is consistently better, with only a few specific cases where the k -NN graphs perform better. However, these instances are rare exceptions.

Table III provides the results of S2-GNN and state-of-the-art methods on the citation network datasets, after being pre-processed with the GDC method, and the OGBN-proteins

dataset. S2-GNN achieves the best performance in Cora and OGBN-proteins datasets and very competitive performance in Citeseer and Pubmed datasets. Tables II and III collectively demonstrate that S2-GNN exhibits strong performance in cases where we construct graphs for the downstream task (using either k -NN or smooth graphs) as well as when the graph is naturally provided by the task, as seen in the citation networks. We provide additional experiments on graph classification tasks in the Appendix F.

Table IV shows the sparsity percentage of the non-sparse Sobolev term $(\mathbf{L} + \epsilon\mathbf{I})^\rho$ for different values of ρ . We observe that the non-sparse filter loses sparsity quickly for higher values of ρ , resulting in increased memory and computational complexity since more edges are considered for message passing.

Memory consumption and inference time. We thoroughly analyze the memory consumption and inference time of S2-GNN with synthetic graphs. To this end, we first build underlying graphs using Erdős-Rényi (ER) models with varying number of nodes $N \in \{500, 1000, 5000, 10000\}$ and edge probabilities $p_{ER} \in \{0.03, 0.04, 0.05, 0.06\}$. Therefore, by feeding an identical initial feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F}$, where $F = 16$, we plotted the averaged memory usage (in terms of GB) and inference times (in seconds) over 30 random seeds in Figure 5. We observe that S2-GNN is among the most efficient models in terms of memory and time complexity when the number of nodes increases and the sparsity of the graphs decreases.

E. Ablation Study

We conduct an ablation study to investigate whether the performance improvement in S2-GNN is attributed to the additional parameters introduced by having multiple branches or the Hadamard operation in the graph filter. To explore this, we create a similar model to S2-GNN but exclude the Hadamard powers in each branch. Instead, we use the identity operation in the first branch and the term $\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ in the remaining branches. By performing hyperparameter optimization as outlined in Section V-B, we obtain the results summarized in Table V. The findings suggest that, for nearly all datasets, incorporating the Hadamard powers using either k -NN or learned graphs leads to better performance compared to simply adding extra parameters to the GNN architecture.

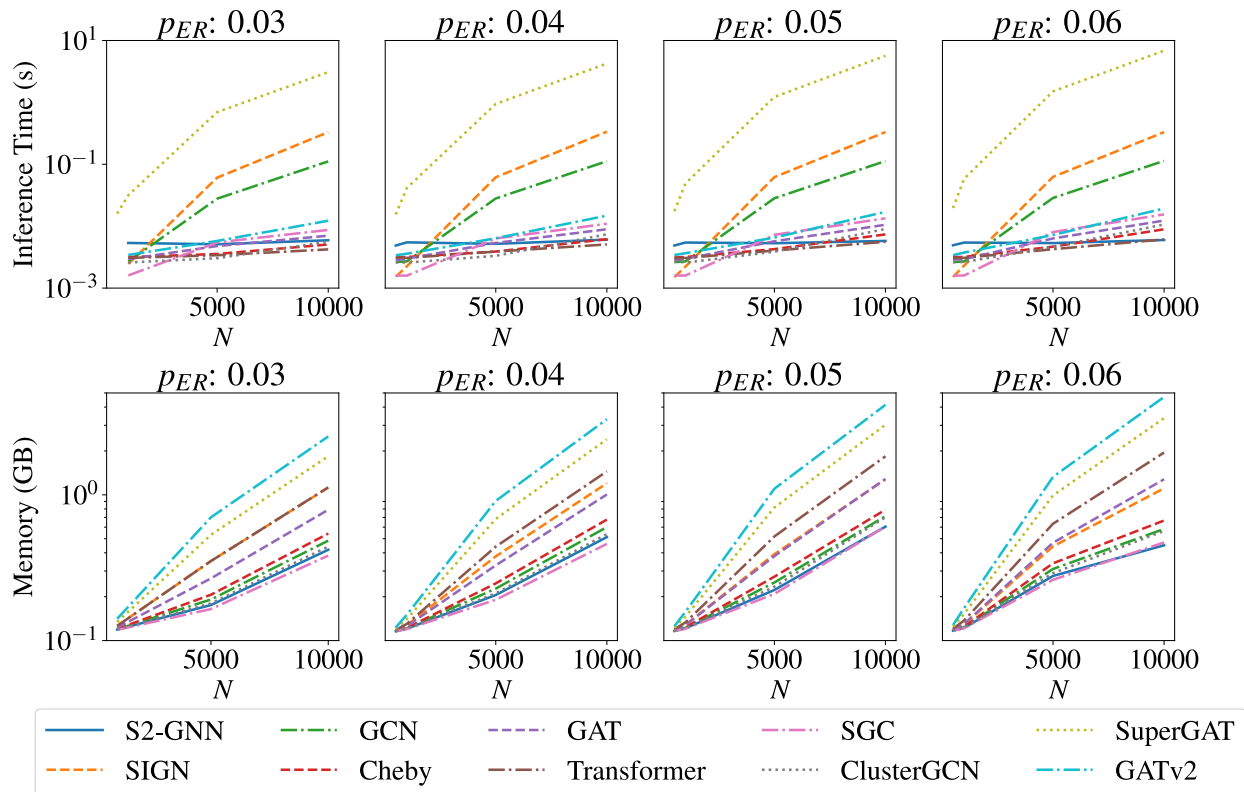


Fig. 5: Comparison of inference times (in seconds) and memory consumption (in terms of GB) on the underlying ER graphs with varying number of nodes $N \in \{500, 1000, 5000, 10000\}$ and edge probabilities $p_{ER} \in \{0.03, 0.04, 0.05, 0.06\}$.

TABLE V: Ablation study in the selection of Hadamard power for the filtering operation.

Hadamard Power	k -NN Graphs					Learned Graphs				
	Cancer-B	Cancer-M	20News	HAR	Isolet	Cancer-B	Cancer-M	20News	HAR	Isolet
\times	91.20 \pm 0.50	66.29 \pm 0.35	72.08 \pm 0.37	93.08 \pm 0.34	85.66 \pm 0.31	88.49 \pm 3.67	67.97 \pm 0.24	72.63 \pm 0.31	93.82 \pm 0.25	86.44 \pm 0.32
\checkmark	93.39 \pm 0.41	68.48 \pm 0.28	72.09 \pm 0.35	93.92 \pm 0.22	86.57 \pm 0.27	93.50 \pm 0.39	68.88 \pm 0.25	72.19 \pm 0.41	94.50 \pm 0.19	86.61 \pm 0.29

The best results on each dataset are shown in **bold**.

TABLE VI: Ablation study regarding regular versus sparse Sobolev norm.

Sparse Norm	k -NN Graphs					Learned Graphs				
	Cancer-B	Cancer-M	20News	HAR	Isolet	Cancer-B	Cancer-M	20News	HAR	Isolet
\times	87.05 \pm 4.25	64.75 \pm 0.34	72.29 \pm 0.34	92.80 \pm 0.30	85.81 \pm 0.35	92.92 \pm 0.48	66.41 \pm 0.37	73.02 \pm 0.41	93.32 \pm 0.35	86.25 \pm 0.30
\checkmark	93.39 \pm 0.41	68.48 \pm 0.28	72.09 \pm 0.35	93.92 \pm 0.22	86.57 \pm 0.27	93.50 \pm 0.39	68.88 \pm 0.25	72.19 \pm 0.41	94.50 \pm 0.19	86.61 \pm 0.29

This implies that the Hadamard operation plays a crucial role in enhancing the model’s effectiveness.

Another ablation experiment studies the usage of the regular and sparse Sobolev norms in S2-GNN. Table VI shows the comparison results, where we observe the sparse norm provides better results on all datasets except on 20News.

Sensibility analysis of ϵ . Figure 6 shows a sensibility analysis of the hyperparameter $\epsilon \in [0, 2]$ (with standard deviations) for different datasets using k -NN graphs. We observe that ϵ has a relatively important impact on the Cancer-M dataset. For the rest of the datasets, S2-GNN is relatively robust to the choice of ϵ . We note that the sparsity level of the Cancer-M dataset is the highest among the compared datasets. Therefore, there is a higher chance for the sparsest graph of Cancer-M to have an ill-condition Laplacian and be improved by the Sobolev coefficient ϵ , as discussed in (6).

F. Limitations

One of the main limitations of S2-GNN is its reliance on weighted graphs as input when using the adjacency matrix. This requirement can pose a limitation in certain scenarios where only unweighted graphs are available or when the weighted graph construction process is challenging or unreliable. Additionally, compared to traditional GCN, S2-GNN introduces three additional hyperparameters: α , ϵ , and the fusion layer. The inclusion of these hyperparameters increases the complexity of the model and makes the hyperparameter optimization process more challenging compared to GCN. Finding an optimal set of hyperparameters for S2-GNN through random search becomes more difficult due to the expanded search space. These limitations emphasize the importance of careful consideration and understanding of the graph structure and hyperparameter selection when applying

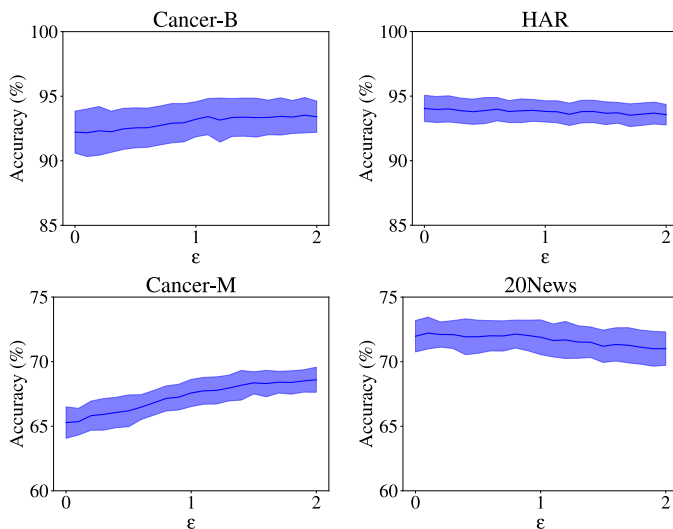


Fig. 6: Mean and standard deviation for the accuracy in different datasets for the sensibility of the hyperparameter ϵ .

S2-GNN in practical scenarios.

VI. CONCLUSION

In this work, we introduced a new higher-order sparse graph filter for GNNs. We proposed a novel sparse Sobolev norm inspired by the GSP literature and used Hadamard powers to compute efficient graph filters. Sections IV-A, IV-B, and IV-E provide theoretical insights into our filtering operation. To capture higher-order dependencies in GNNs, we introduced the S2-GNN model. This model incorporates a cascade of increasing sparse filtering operations on each layer, along with a fusion layer that selects the most effective operations. By leveraging these mechanisms, S2-GNN effectively captures and utilizes higher-order information in graph-structured data. In addition, we explored the inference of smooth graphs as an alternative when the underlying graph topology is unavailable for a given problem. Our observations indicate that smooth inferred graphs improve the performance of baseline GNN architectures by promoting homophily. We thoroughly evaluated S2-GNN in several datasets in a semi-supervised regimen. S2-GNN achieves competitive performance compared to state-of-the-art GNNs across all tasks while maintaining considerably lower complexity than the best-performing methods.

This work opens several research directions. The first direction is the extension of S2-GNN to the setting of ClusterGCN [30]. This extension would explore the possibility of removing the explicit need for inter-cluster communication, potentially simplifying the model architecture. Another promising avenue is the study of shift operators that do not require explicit weighted graphs. This direction would address the limitation of S2-GNN, which requires weighted graphs as input. By developing alternative shift operators, we can broaden the applicability of the model and overcome this constraint. Furthermore, neural architecture search can be studied in the context of S2-GNN to approach other problems such as graph classification and link prediction, while addressing the challenge of hyperparameter search in S2-GNN.

REFERENCES

- [1] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *ICLR*, 2014.
- [2] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NeurIPS*, 2016.
- [3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.
- [4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.
- [5] A. Duval and F. Malliaros, "Higher-order clustering and pooling for graph neural networks," in *ACM CIKM*, 2022.
- [6] G. Li, M. Muller, A. Thabet, and B. Ghanem, "DeepGCNs: Can GCNs go as deep as CNNs?" in *IEEE/CVF ICCV*, 2019.
- [7] A. Benamira, B. Devillers, E. Lesot, A. K. Ray, M. Saadi, and F. D. Malliaros, "Semi-supervised learning and graph neural networks for fake news detection," in *IEEE/ACM ASONAM*, 2019.
- [8] A. Duval, V. Schmidt, A. H. Garcia, S. Miret, F. D. Malliaros, Y. Bengio, and D. Rolnick, "FAENet: frame averaging equivariant GNN for materials modeling," in *ICML*, 2023.
- [9] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE T-KDE*, vol. 34, no. 1, pp. 249–270, 2022.
- [10] F. Frasca, E. Rossi, D. Eynard, B. Chamberlain, M. M. Bronstein, and F. Monti, "SIGN: Scalable inception graph neural networks," in *ICML-W*, 2020.
- [11] F. R. K. Chung, *Spectral graph theory*. American Mathematical Society, 1997, no. 92.
- [12] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.
- [13] I. Pesenson, "Variational splines and Paley–Wiener spaces on combinatorial graphs," *Constructive Approximation*, vol. 29, no. 1, pp. 1–21, 2009.
- [14] J. H. Giraldo, A. Mahmood, B. Garcia-Garcia, D. Thanou, and T. Bouwmans, "Reconstruction of time-varying graph signals via Sobolev smoothness," *IEEE T-SIPN*, vol. 8, pp. 201–214, 2022.
- [15] J. N. Kather, C.-A. Weis, F. Bianconi, S. M. Melchers, L. R. Schad, T. Gaiser, A. Marx, and F. G. Zöllner, "Multi-class texture analysis in colorectal cancer histology," *Scientific Reports*, vol. 6, p. 27988, 2016.
- [16] K. Lang, "NewsWeeder: Learning to filter netnews," in *JMLR*, 1995.
- [17] D. Anguita *et al.*, "A public domain dataset for human activity recognition using smartphones," in *ESANN*, 2013.
- [18] M. Fianty and R. Cole, "Spoken letter recognition," in *NeurIPS*, 1991.
- [19] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," in *NeurIPS*, 2020.
- [20] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?" in *ICLR*, 2022.
- [21] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked label prediction: Unified message passing model for semi-supervised classification," in *IJCAI*, 2021.
- [22] J. H. Giraldo, S. Javed, A. Mahmood, F. D. Malliaros, and T. Bouwmans, "Higher-order sparse convolutions in graph neural networks," in *IEEE ICASSP*, 2023.
- [23] F. Gama, J. Bruna, and A. Ribeiro, "Stability properties of graph neural networks," *IEEE T-SP*, vol. 68, pp. 5680–5695, 2020.
- [24] A. Parada-Mayorga, Z. Wang, F. Gama, and A. Ribeiro, "Stability of aggregation graph neural networks," *IEEE T-SIPN*, 2023.
- [25] V. Kalofolias and N. Perraudin, "Large scale graph learning from smooth signals," in *ICLR*, 2019.
- [26] X. Pu, S. L. Chau, X. Dong, and D. Sejdinovic, "Kernel-based graph learning from smooth signals: A functional viewpoint," *IEEE T-SIPN*, vol. 7, pp. 192–207, 2021.
- [27] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE SPM*, vol. 34, no. 4, pp. 18–42, 2017.
- [28] D. Kim and A. Oh, "How to find your friendly neighborhood: Graph attention design with self-supervision," in *ICLR*, 2021.
- [29] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *ICML*, 2019.
- [30] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks," in *ACM SIGKDD*, 2019.
- [31] D. Kreuzer, D. Beaini, W. Hamilton, V. Létourneau, and P. Tossou, "Rethinking graph transformers with spectral attention," in *NeurIPS*, 2021.

- [32] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.
- [33] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE T-SP*, vol. 61, no. 7, pp. 1644–1656, 2013.
- [34] J. H. Giraldo, S. Javed, and T. Bouwmans, "Graph moving object segmentation," *IEEE T-PAMI*, vol. 44, no. 5, pp. 2485–2503, 2022.
- [35] J. H. Giraldo and T. Bouwmans, "On the minimization of Sobolev norms of time-varying graph signals: Estimation of new Coronavirus disease 2019 cases," in *IEEE MLSP*, 2020.
- [36] B. Nica, *A brief introduction to spectral graph theory*. European Mathematical Society Publishing House, 2018.
- [37] Z. Peng, H. Liu, Y. Jia, and J. Hou, "Attention-driven graph clustering network," in *ACM MM*, 2021.
- [38] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of internet portals with machine learning," *Information Retrieval*, vol. 3, no. 2, pp. 127–163, 2000.
- [39] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Magazine*, vol. 29, no. 3, pp. 93–93, 2008.
- [40] G. Namata, B. London, L. Getoor, B. Huang, and U. Edu, "Query-driven active surveying for collective classification," in *International Workshop on Mining and Learning with Graphs*, 2012.
- [41] A. Gadde, A. Anis, and A. Ortega, "Active semi-supervised learning using sampling theory for graph signals," in *ACM KDD*, 2014.
- [42] J. Klicpera, S. Weissenberger, and S. Günnemann, "Diffusion improves graph learning," in *NeurIPS*, 2019.
- [43] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR-W*, 2019.
- [44] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *arXiv preprint arXiv:1408.5781*, 2014.
- [45] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE SPM*, vol. 30, no. 3, pp. 83–98, 2013.
- [46] G. Visick, "A quantitative version of the observation that the Hadamard product is a principal submatrix of the Kronecker product," *LAA*, vol. 304, no. 1-3, pp. 45–68, 2000.
- [47] X. Zhan, "Inequalities for the singular values of Hadamard products," *SIAM Journal on Matrix Analysis and Applications*, vol. 18, no. 4, pp. 1093–1095, 1997.
- [48] H. Pei, B. Wei, K. C.-C. Chang, Y. Lei, and B. Yang, "Geom-GCN: Geometric graph convolutional networks," in *ICLR*, 2020.
- [49] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "TUDataset: A collection of benchmark datasets for learning with graphs," in *ICML-W*, 2020.

APPENDIX A
PROOF OF THE THEOREM 1

Proof. 1) For the triangle inequality we have that:

$$\begin{aligned} \|\mathbf{x} + \mathbf{y}\|_{(\rho),\epsilon}^2 &= \langle \mathbf{x} + \mathbf{y}, \mathbf{y} + \mathbf{x} \rangle_{(\rho),\epsilon} = \|(\mathbf{L} + \epsilon \mathbf{I})^{(\rho/2)}(\mathbf{x} + \mathbf{y})\|^2 \\ &\rightarrow \|\mathbf{x} + \mathbf{y}\|_{(\rho),\epsilon}^2 = \|\mathbf{x}\|_{(\rho),\epsilon}^2 + \|\mathbf{y}\|_{(\rho),\epsilon}^2 + 2\langle \mathbf{x}, \mathbf{y} \rangle_{(\rho),\epsilon}. \end{aligned} \quad (19)$$

Using the Cauchy–Schwarz inequality we obtain:

$$\begin{aligned} \|\mathbf{x} + \mathbf{y}\|_{(\rho),\epsilon}^2 &\leq \|\mathbf{x}\|_{(\rho),\epsilon}^2 + 2\|\mathbf{x}\|_{(\rho),\epsilon}\|\mathbf{y}\|_{(\rho),\epsilon} + \|\mathbf{y}\|_{(\rho),\epsilon}^2 \\ &\rightarrow \|\mathbf{x} + \mathbf{y}\|_{(\rho),\epsilon}^2 \leq (\|\mathbf{x}\|_{(\rho),\epsilon} + \|\mathbf{y}\|_{(\rho),\epsilon})^2. \end{aligned} \quad (20)$$

2) Let s be a scalar, then for the absolute homogeneity property we have:

$$\|s\mathbf{x}\|_{(\rho),\epsilon} = \|(\mathbf{L} + \epsilon \mathbf{I})^{(\rho/2)}s\mathbf{x}\| = |s|\|(\mathbf{L} + \epsilon \mathbf{I})^{(\rho/2)}\mathbf{x}\| = |s|\|\mathbf{x}\|_{(\rho),\epsilon}. \quad (21)$$

3) For the positive definiteness property, we rely on the Schur product theorem, which states that the Hadamard product between two positive definite matrices is also positive definite [12]. When $\epsilon = 0$ and $\rho = 1$, the sparse Sobolev norm reduces to the graph Laplacian quadratic form [45]. In this case, $\|\mathbf{x}\|_{(1),0}^2 = \mathbf{x}^T \mathbf{L} \mathbf{x} = 0 \iff \mathbf{x} = \tau \mathbf{1}$ (with τ some constant value). As a result, $\|\mathbf{x}\|_{(\rho),\epsilon}$ becomes a semi-norm because it satisfies only the first two conditions of vector norms, *i.e.*, the triangle inequality and absolute homogeneity. However, for $\epsilon > 0$, we have $\mathbf{L} + \epsilon \mathbf{I} \succ 0$ according to (6), and then $(\mathbf{L} + \epsilon \mathbf{I})^{(\rho)} \succ 0$ by using the Schur product theorem. As a result:

$$\|\mathbf{x}\|_{(\rho),\epsilon}^2 = \mathbf{x}^T (\mathbf{L} + \epsilon \mathbf{I})^{(\rho)} \mathbf{x} = 0 \iff \mathbf{x} = 0; \forall \epsilon > 0. \quad (22)$$

As a consequence, the sparse Sobolev norm satisfies the properties of vector norms for all $\epsilon > 0$ according to (20), (21), and (22). Similarly, the sparse Sobolev norm definition satisfies the properties of semi-norms for $\epsilon = 0$ according to (20) and (21). \square

APPENDIX B
PROOF OF THEOREM 2

Proof. We can compute the spectral decomposition of the Kronecker product between matrices using concepts from product graphs. The spectrum of the Kronecker product of multiple graphs is obtained by taking the Kronecker products of their respective spectral components. For instance, the spectral decomposition of $\mathbf{L} \otimes \mathbf{L}$ can be expressed as:

$$\mathbf{L} \otimes \mathbf{L} = (\mathbf{U} \otimes \mathbf{U})(\mathbf{\Lambda} \otimes \mathbf{\Lambda})(\mathbf{U}^T \otimes \mathbf{U}^T). \quad (23)$$

Similarly, there is a relationship between the Kronecker and the Hadamard products as follows:

Theorem 4. (Visick [46]) For $\mathbf{S}, \mathbf{T} \in \mathbb{R}^{n \times m}$, we have:

$$\mathbf{S} \circ \mathbf{T} = \mathbf{P}_n^T (\mathbf{S} \otimes \mathbf{T}) \mathbf{P}_m, \quad (24)$$

where $\mathbf{P}_n \in \{0, 1\}^{n^2 \times n}$ and $\mathbf{P}_m \in \{0, 1\}^{m^2 \times m}$ are partial permutation matrices. If $\mathbf{S}, \mathbf{T} \in \mathbb{R}^{n \times n}$ are square matrices, we have that $\mathbf{S} \circ \mathbf{T} = \mathbf{P}_n^T (\mathbf{S} \otimes \mathbf{T}) \mathbf{P}_n$.

Proof: see [46].

Therefore, by using (23) and Theorem 4, we can derive a general form of the spectrum of the Hadamard product (specifically, for the Hadamard power of order 2) as follows:

$$\mathbf{L} \circ \mathbf{L} = \mathbf{L}^{(2)} = \mathbf{P}_N^T (\mathbf{U} \otimes \mathbf{U})(\mathbf{\Lambda} \otimes \mathbf{\Lambda})(\mathbf{U}^T \otimes \mathbf{U}^T) \mathbf{P}_N. \quad (25)$$

We can then conclude that the spectrum of the Hadamard power of order ρ is such that:

$$\mathbf{L}^{(\rho)} = \mathbf{P}_{(\rho)}^T (\mathbf{U} \otimes \mathbf{U}_{(\rho-1)})(\mathbf{\Lambda} \otimes \mathbf{\Lambda}_{(\rho-1)})(\mathbf{U}^T \otimes \mathbf{U}_{(\rho-1)}^T) \mathbf{P}_{(\rho)}, \quad (26)$$

where $\mathbf{U}_{(\rho-1)}$ and $\mathbf{\Lambda}_{(\rho-1)}$ are respectively the matrices of eigenvectors and eigenvalues of $\mathbf{L}^{(\rho-1)}$, and $\mathbf{P}_{(\rho)} \in \{0, 1\}^{N^2 \times N}$ is a partial permutation matrix for $\mathbf{L}^{(\rho)}$. \square

APPENDIX C
PROOF OF THEOREM 3

Proof. Firstly, we express the perturbed Laplacian matrix $\hat{\mathbf{L}}$ based on the perturbation model in (15) as:

$$\hat{\mathbf{L}} = \hat{\mathbf{D}} - \hat{\mathbf{A}} = (\mathbf{D} + \mathbf{D}_E) - (\mathbf{A} + \mathbf{E}), \quad (27)$$

where

$$\mathbf{D}_E = \text{diag}(\mathbf{E}\mathbf{1}) = \text{diag}(d_{e_1}, \dots, d_{e_N}). \quad (28)$$

With the notation of

$$\hat{\mathbf{L}}^{(\rho)} = \underbrace{(\hat{\mathbf{L}} + \epsilon \mathbf{I}) \circ \dots \circ (\hat{\mathbf{L}} + \epsilon \mathbf{I})}_{\rho \text{ times}}, \quad (29)$$

the next lemma expresses the elements of $\mathbf{L}^{(\rho)}$ and $\hat{\mathbf{L}}^{(\rho)}$, *i.e.*, $\{(\mathbf{L}^{(\rho)})_{ij}\}_{i,j=1}^N$ and $\{(\hat{\mathbf{L}}^{(\rho)})_{ij}\}_{i,j=1}^N$, respectively, based on the edge weights and node degrees as follows:

Lemma 1. The elements of $\mathbf{L}^{(\rho)}$ and $\hat{\mathbf{L}}^{(\rho)}$, *i.e.*, $\{(\mathbf{L}^{(\rho)})_{ij}\}_{i,j=1}^N$ and $\{(\hat{\mathbf{L}}^{(\rho)})_{ij}\}_{i,j=1}^N$, can be written as:

$$\begin{aligned} \forall i \neq j: (\mathbf{L}^{(\rho)})_{ij} &= (-1)^\rho a_{ij}^\rho; (\hat{\mathbf{L}}^{(\rho)})_{ij} = (-1)^\rho (a_{ij} + e_{ij})^\rho \\ (\mathbf{L}^{(\rho)})_{ii} &= (d_i + \epsilon)^\rho; (\hat{\mathbf{L}}^{(\rho)})_{ii} = (d_i + d_{e_i} + \epsilon)^\rho. \end{aligned} \quad (30)$$

Proof. Using the convention (29) and zero-diagonality of \mathbf{A} , one can write:

$$\begin{aligned} \mathbf{L}^{(\rho)} &= \underbrace{(\mathbf{L} + \epsilon \mathbf{I}) \circ \dots \circ (\mathbf{L} + \epsilon \mathbf{I})}_{\rho \text{ times}} \\ &= \underbrace{(\mathbf{D} - \mathbf{A} + \epsilon \mathbf{I}) \circ \dots \circ (\mathbf{D} - \mathbf{A} + \epsilon \mathbf{I})}_{\rho \text{ times}} \\ &= (\mathbf{D} + \epsilon \mathbf{I})^\rho + (-1)^\rho \underbrace{\mathbf{A} \circ \dots \circ \mathbf{A}}_{\rho \text{ times}} \\ &\rightarrow (\mathbf{L}^{(\rho)})_{ii} = (d_i + \epsilon)^\rho; (\mathbf{L}^{(\rho)})_{ij} = (-1)^\rho a_{ij}^\rho. \end{aligned} \quad (31)$$

Similarly for $\hat{\mathbf{L}}^{(\rho)}$ and by considering the zero-diagonality of \mathbf{A} and \mathbf{E} , one can write:

$$\begin{aligned} \hat{\mathbf{L}}^{(\rho)} &= \underbrace{(\hat{\mathbf{L}} + \epsilon \mathbf{I}) \circ \dots \circ (\hat{\mathbf{L}} + \epsilon \mathbf{I})}_{\rho \text{ times}} \\ &= \underbrace{(\mathbf{D} + \mathbf{D}_E + \epsilon \mathbf{I} - (\mathbf{A} + \mathbf{E})) \circ \dots \circ (\mathbf{D} + \mathbf{D}_E + \epsilon \mathbf{I} - (\mathbf{A} + \mathbf{E}))}_{\rho \text{ times}} \\ &= (\mathbf{D} + \mathbf{D}_E + \epsilon \mathbf{I})^\rho + (-1)^\rho \underbrace{(\mathbf{A} + \mathbf{E}) \circ \dots \circ (\mathbf{A} + \mathbf{E})}_{\rho \text{ times}} \\ &\rightarrow (\hat{\mathbf{L}}^{(\rho)})_{ii} = (d_i + d_{e_i} + \epsilon)^\rho; (\hat{\mathbf{L}}^{(\rho)})_{ij} = (-1)^\rho (a_{ij} + e_{ij})^\rho. \end{aligned} \quad (32)$$

□

Besides, the next lemma gives upper bounds for $\|\hat{\mathbf{L}}\|$ and Laplacian perturbation distance $\|\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)}\|$. Note that $r_1(M)$ and $c_1(M)$ denote the maximum Euclidean row and column norms of the matrix M .

Lemma 2. *The Laplacian perturbation distance $\|\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)}\|$ and $\|\hat{\mathbf{L}}\|$ are upper bounded by:*

$$\|\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)}\| \leq \sum_{m=1}^{\rho} \binom{\rho}{m} \left(\eta_m \|\mathbf{E}^{(m)}\| + (d^{max} + \epsilon)^{\rho-m} (d_e^{max})^m \right), \quad (33)$$

and

$$\|\hat{\mathbf{L}}^{(\rho)}\| \leq (\hat{d}^{max} + \epsilon)^{\rho} + \|\hat{\mathbf{A}}^{(\rho)}\|, \quad (34)$$

where $\hat{d}^{max} = d^{max} + d_e^{max}$, $d^{max} = \max_i d_i$, and $d_e^{max} = \max_i d_{e_i}$.

Proof. For the sake of simplicity, we start with $\|\hat{\mathbf{L}}^{(\rho)}\|$. Using (32) and by exploiting the triangular inequality principle of the norms, one can write:

$$\begin{aligned} \|\hat{\mathbf{L}}^{(\rho)}\| &\leq \|(\mathbf{D} + \mathbf{D}_E + \epsilon\mathbf{I})^{\rho} + (-1)^{\rho} \underbrace{(\mathbf{A} + \mathbf{E}) \circ \dots \circ (\mathbf{A} + \mathbf{E})}_{\rho \text{ times}}\| \\ &\leq \|(\mathbf{D} + \mathbf{D}_E + \epsilon\mathbf{I})^{\rho}\| + \|\underbrace{(\mathbf{A} + \mathbf{E}) \circ \dots \circ (\mathbf{A} + \mathbf{E})}_{\rho \text{ times: } \hat{\mathbf{A}}^{(\rho)}}\| \\ &\leq \underbrace{(d^{max} + d_e^{max} + \epsilon)^{\rho}}_{\hat{d}^{max}} + \|\hat{\mathbf{A}}^{(\rho)}\|. \end{aligned} \quad (35)$$

Regarding $\|\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)}\|$, it is easier to work on the Laplacian elements. We have:

$$\begin{aligned} \forall i \neq j : (\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)})_{ij} &= (-1)^{\rho+1} ((a_{ij} + e_{ij})^{\rho} - a_{ij}^{\rho}) \\ &= (-1)^{\rho+1} \left(\sum_{m=1}^{\rho} \binom{\rho}{m} a_{ij}^{\rho-m} e_{ij}^m \right), \end{aligned} \quad (36)$$

$$\begin{aligned} (\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)})_{ii} &= (d_i + \epsilon)^{\rho} - (d_i + \epsilon + d_{e_i})^{\rho} \\ &= - \sum_{m=1}^{\rho} \binom{\rho}{m} (d_i + \epsilon)^{\rho-m} d_{e_i}^m. \end{aligned} \quad (37)$$

Therefore, using (36) and (37), we can obtain a matrix-form to express $\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)}$ as:

$$\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)} = \sum_{m=1}^{\rho} \binom{\rho}{m} \left((-1)^{\rho+1} \mathbf{A}^{(\rho-m)} \circ \mathbf{E}^{(m)} + (\mathbf{D} + \epsilon\mathbf{I})^{(\rho-m)} \circ \mathbf{D}_E^{(m)} \right) \quad (38)$$

Next, by applying the triangular principle of the norm, one can write:

$$\|\mathbf{L}^{(\rho)} - \hat{\mathbf{L}}^{(\rho)}\| \leq \sum_{m=1}^{\rho} \binom{\rho}{m} \left(\|\mathbf{A}^{(\rho-m)} \circ \mathbf{E}^{(m)}\| + \|(\mathbf{D} + \epsilon\mathbf{I})^{(\rho-m)} \circ \mathbf{D}_E^{(m)}\| \right). \quad (39)$$

The next Lemma makes it possible to take further steps for simplifying the current bound in (39).

Lemma 3. (Lemma 2c, [47]). *For any symmetric matrices \mathbf{A}_1 and \mathbf{A}_2 , there is an upper bound on $\|\mathbf{A}_1 \circ \mathbf{A}_2\|$ as:*

$$\|\mathbf{A}_1 \circ \mathbf{A}_2\| \leq \min\{c_1(\mathbf{A}_1), r_1(\mathbf{A}_1)\} \|\mathbf{A}_2\| \quad (40)$$

Using Lemma 3, we can better simplify the bound in (39). We first apply Lemma 3 on the first term of the bound in (39):

$$\|(\mathbf{A}^{(\rho-m)} \circ \mathbf{E}^{(m)})\| \leq \underbrace{\min\{r_1(\mathbf{A}^{(\rho-m)}), c_1(\mathbf{A}^{(\rho-m)})\}}_{\eta_m} \|\mathbf{E}^{(m)}\|. \quad (41)$$

Similarly, the second term can be written as:

$$\begin{aligned} \|((\mathbf{D} + \epsilon\mathbf{I})^{(\rho-m)} \circ \mathbf{D}_E^{(m)})\| &= \underbrace{\|(\mathbf{D} + \epsilon\mathbf{I})^{(\rho-m)}\|}_{(d^{max} + \epsilon)^{\rho-m}} \underbrace{\|\mathbf{D}_E^{(m)}\|}_{(d_e^{max})^m} \\ &\leq \min\{r_1((\mathbf{D} + \epsilon\mathbf{I})^{(\rho-m)}), c_1((\mathbf{D} + \epsilon\mathbf{I})^{(\rho-m)})\} \|\mathbf{D}_E^{(m)}\|. \end{aligned} \quad (42)$$

Finally, by plugging (41) and (42) into (39), the desired result in (33) is obtained, and it completes the proof of Lemma 2. □

Now, using the triangular principle of the norms, the property of being Lipschitz for the nonlinearity function $\sigma(\cdot)$, and the results in Lemma 2, we come back to find an upper bound for $d(\mathbf{Y}, \hat{\mathbf{Y}})$ in (18) as follows:

$$\begin{aligned} \|\sigma(\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \hat{\mathbf{W}}) - \sigma(\mathbf{L}^{(\rho)} \mathbf{X} \mathbf{W})\| &\leq \|\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \hat{\mathbf{W}} - \mathbf{L}^{(\rho)} \mathbf{X} \mathbf{W}\| \\ &= \|\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \hat{\mathbf{W}} - \hat{\mathbf{L}}^{(\rho)} \mathbf{X} \mathbf{W} + \hat{\mathbf{L}}^{(\rho)} \mathbf{X} \mathbf{W} - \mathbf{L}^{(\rho)} \mathbf{X} \mathbf{W}\| \\ &\leq \|\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \hat{\mathbf{W}} - \hat{\mathbf{L}}^{(\rho)} \mathbf{X} \mathbf{W}\| + \|\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \mathbf{W} - \mathbf{L}^{(\rho)} \mathbf{X} \mathbf{W}\| \\ &\leq \|\hat{\mathbf{L}}^{(\rho)}\| \|\mathbf{X}\| \|\mathbf{W} - \hat{\mathbf{W}}\| + \|\hat{\mathbf{L}}^{(\rho)} - \mathbf{L}^{(\rho)}\| \|\mathbf{X}\| \|\mathbf{W}\| \\ &\leq \sqrt{v} \left(\left[(\hat{d}^{max} + \epsilon)^{\rho} + \|\hat{\mathbf{A}}^{(\rho)}\| \right] \delta_{\mathbf{W}} + \right. \\ &\quad \left. \sum_{m=1}^{\rho} \binom{\rho}{m} \left(\eta_m \|\mathbf{E}^{(m)}\| + (d^{max} + \epsilon)^{\rho-m} (d_e^{max})^m \right) \|\mathbf{W}\| \right). \end{aligned} \quad (43)$$

Then, using $\|\mathbf{E}\| \leq \xi$ and $\|\mathbf{D}_E\| \leq \xi_D$, where ξ and ξ_D are sufficiently small, one can use the first-order Taylor approximation (by considering only $m = 1$) to further simplify the upper bound as:

$$\begin{aligned} \|\sigma(\hat{\mathbf{L}}^{(\rho)} \mathbf{X} \hat{\mathbf{W}}) - \sigma(\mathbf{L}^{(\rho)} \mathbf{X} \mathbf{W})\| &\leq \sqrt{v} \left(\left[(\hat{d}^{max} + \epsilon)^{\rho} + \|\hat{\mathbf{A}}^{(\rho)}\| \right] \delta_{\mathbf{W}} \right. \\ &\quad \left. + \rho [\eta \|\mathbf{E}\| + (d^{max} + \epsilon)^{\rho-1} d_e^{max}] \|\mathbf{W}\| \right) + \mathcal{O}(\xi^2) + \mathcal{O}(\xi_D^2). \end{aligned} \quad (44)$$

and this completes the proof of Theorem 3. Note that, in the last line, we used $\|\mathbf{X}\| \leq \|\mathbf{X}\|_F = \sqrt{v}$. □

APPENDIX D EXPERIMENTAL STABILITY ANALYSIS

We validate Theorem 3 through experiments by analyzing how the theoretical bounds on the Right Hand Side (RHS) and Left Hand Side (LHS) depend on: 1) the Signal-to-Noise-Ratio (SNR) in both adjacency and weight matrices, 2) the Hadamard order ρ , 3) the graph sparsity (relevant to p_{ER}),

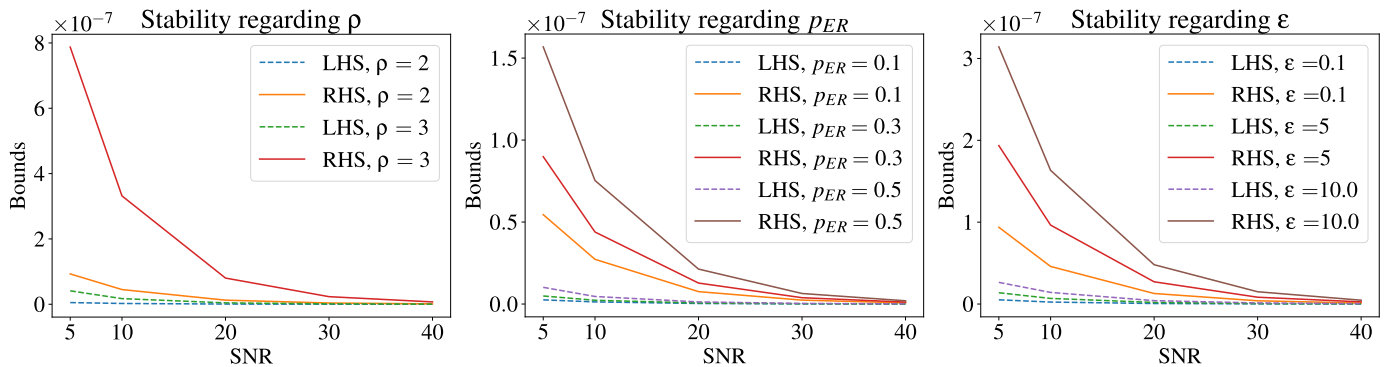


Fig. 7: Evaluation of Theorem 3 across different configurations for the Sobolev norm when certain perturbations are made to the adjacency matrix.

and 4) the Sobolev coefficient ϵ . More precisely, we build the underlying graph using the ER model with N nodes and edge probability p_{ER} . We then generate true and perturbed elements using (16) with varying SNRs $\in \{5, 10, 20, 30, 40\}$ to perturb the underlying adjacency matrix and weight matrix \mathbf{W} . Both the initial feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F_0}$ and weight matrix $\mathbf{W} \in \mathbb{R}^{F_0 \times F_1}$ are drawn from normal distributions. In all scenarios, we set $F_0 = 16$ and $F_1 = 2$, and the results are averaged over 100 random seeds. We analyze the following scenarios:

- Varying ρ : we test ρ in $\{2, 3\}$ by letting $N = 10$, $p_{ER} = 0.3$, and $\epsilon = 0.5$.
- Varying p_{ER} : we test p_{ER} in $\{0.1, 0.3, 0.5\}$ by letting $N = 10$, $\rho = 2$, and $\epsilon = 0.5$.
- Varying ϵ : we test ϵ in $\{0.5, 5, 10\}$ by letting $N = 10$, $\rho = 2$, and $p_{ER} = 0.3$.

Figure 7 shows the averaged values of RHS and LHS across different SNRs and varying ρ , p_{ER} , and ϵ . We find that higher values of ρ make the network more sensitive to perturbations, especially with low SNRs. Additionally, the network benefits from graph sparsity, as it is more robust to perturbations with lower values of p_{ER} . We also observe that high values of ϵ can negatively affect the robustness of S2-GNN. Finally, as the SNR increases, the upper bound becomes tighter, which is related to smaller values of $\delta_{\mathbf{W}}$, $\|\mathbf{E}\|$, d_e^{max} , and also ζ and ζ_D .

APPENDIX E HOMOPHILY INDEX

The homophily index for the datasets used to construct the graphs is presented in Table VII. $H(G) \rightarrow 1$ when graphs exhibit strong homophily, while $H(G) \rightarrow 0$ for graphs with strong heterophily [48]. Thus, low homophily corresponds to high heterophily, and vice versa. The learned graphs consistently exhibit stronger homophily, which directly correlates with the results in Table II, where methods applied to the learned graphs generally perform better.

APPENDIX F GRAPH CLASSIFICATION EXPERIMENTS

We perform additional comparisons between the state-of-the-art methods and S2-GNN for graph classification tasks.

TABLE VII: Homophily index $H(G)$ for the constructed graphs with different values of k .

Dataset	k -NN Graphs				Learned Graphs			
	$k = 10$	$k = 20$	$k = 30$	$k = 40$	$k = 10$	$k = 20$	$k = 30$	$k = 40$
Cancer-B	0.8625	0.8447	0.8334	0.8230	0.8838	0.8670	0.8568	0.8463
Cancer-M	0.6267	0.5954	0.5765	0.5629	0.6534	0.6242	0.6054	0.5921
20News	0.5557	0.4968	0.4615	0.4354	0.6103	0.5348	0.4919	0.4628
HAR	0.8849	0.8478	0.8211	0.8007	0.9087	0.8851	0.8660	0.8500
Isolet	0.7346	0.6851	0.6454	0.6105	0.7619	0.7255	0.6943	0.6664

TABLE VIII: Accuracy (in %) comparison for graph classification.

Model	ENZYMES	MUTAG	PROTEINS
Cheby [2]	29.39 ± 1.21	76.11 ± 2.54	67.24 ± 0.59
GCN [3]	22.69 ± 1.18	73.14 ± 3.86	68.15 ± 0.80
GAT [4]	22.12 ± 1.04	74.98 ± 5.53	64.89 ± 1.23
SGC [29]	19.71 ± 0.84	66.73 ± 3.86	61.29 ± 0.73
ClusterGCN [30]	22.56 ± 0.81	81.14 ± 1.11	67.54 ± 0.57
SuperGAT [28]	21.97 ± 0.97	71.70 ± 6.24	65.61 ± 1.11
Transformer [21]	24.74 ± 1.07	80.78 ± 2.33	64.71 ± 1.18
GATv2 [20]	20.78 ± 1.24	71.43 ± 6.00	64.47 ± 1.13
S2-GNN (ours)	23.66 ± 0.90	73.60 ± 1.46	67.20 ± 0.78

Table VIII shows the results in three datasets for graph classification: ENZYMES, MUTAG, and PROTEINS [49]. We observe that S2-GNN presents competitive results regarding previous state-of-the-art models. However, we believe that further theoretical and experimental studies should be considered in future work for graph classification.