DeloopSGNN: Revisiting Spectral GNNs Through the Lens of Spatial Aggregation

Duanyu Li*, Huijun Wu*†, Min Xie, Kai Lu, Wenzhe Zhang
Zhenwei Wu, Yong Dong, Ruibo Wang†

College of Computer Science and Technology, National University of Defense Technology
{liduanyu19,wuhuijun, xiemin, kailu, zhangwenzhe, zhenweiwu, yongdong, ruibo}@nudt.edu.cn

Abstract

Graph Neural Networks (GNNs) have been studied from two primary perspectives: spectral, which employs global graph signal filtering and is theoretically more expressive, and spatial, which builds on local neighborhood aggregation and generalizes well across diverse graph structures. While spectral GNNs are expected to perform better in theory, they often underperform in practice compared to spatial models. To better understand this gap, we introduce a novel theoretical framework for converting spectral GNNs into the spatial domain, allowing for more intuitive analysis. This transformation reveals that signal looping and repeated high-order aggregation are major causes of over-smoothing in spectral GNNs. By addressing these issues in the spatial domain and converting the model back to the spectral domain, we propose DeloopSGNN, a spectral GNN with improved expressive capacity. Experiments on benchmark datasets show that DeloopSGNN achieves consistently strong performance in terms of accuracy and adversarial robustness, demonstrating that spectral GNNs can benefit significantly from careful architectural design grounded in our proposed framework.

Code — https://github.com/duanyuli2000/DeloopSGNN

Introduction

Graph Neural Networks (GNNs) have become a fundamental architecture for learning on non-Euclidean data (2023; 2025a; 2025b). Within this family, Spectral Graph Neural Networks (Spectral GNNs) possess notable theoretical advantages rooted in graph signal processing, leveraging frequency-domain filtering and graph Fourier transforms (Wang and Zhang 2022). In heterophilic graph settings, spectral methods often surpass spatial approaches by more precisely modulating signal propagation across different frequency bands.

However, a long-standing core contradiction remains unresolved: despite their strong theoretical grounding, spectral methods have not consistently and significantly outperformed more intuitive Spatial GNNs in general tasks (2022). Current spectral research remains heavily confined to optimizations in filter design and basis construction (He et al. 2021; Xu et al. 2024; Huang et al. 2024), failing to address the fundamental performance bottleneck.

This contradiction is most evident in adaptive filter design: adding learnable parameters should, in principle, increase model flexibility, yet in practice it rarely improves performance. This is usually blamed on overfitting, which would normally still lower training loss as model size grows. In spectral models, however, training loss barely decreases with more parameters, indicating deeper theoretical limitations beyond overfitting. In contrast, spatial GNNs remain popular because their message-passing scheme directly mirrors real-world node-neighborhood interactions, making them intuitive and easy to use. This raises a key question: can we transfer the clarity and practicality of spatial methods to the spectral domain, thereby exposing the fundamental limits of spectral approaches and enabling more powerful models?

Based on these challenges, this paper addresses the following three core questions:

- **Q1:** How to rigorously map spectral filters to spatial message passing?
- **Q2:** What factors contribute to spectral models occasionally performing worse than spatial models in practice?
- **Q3:** How can these insights guide spectral GNNs beyond current performance and depth limits?

The key breakthrough of this work lies in constructing a closed loop from theoretical discovery to model innovation:

We first introduce S2SMT, a Spectral-to-Spatial Mapping Theorem that establishes an equivalence between spectral filters and spatial message passing. Within this framework, we explain the essentially similar behavior of GCN-SVD (Entezari et al. 2020) and SGC (Wu et al. 2019a), and clarify, from a spatial perspective, why GCN-SVD's performance varies across different attacks. Using S2SMT, we identify a core dilemma of spectral GNNs: deep aggregation still causes oversmoothing, leading to spectral signal loss and performance stagnation, contradicting the belief that spectral GNNs inherently alleviate oversmoothing. We further show that the corresponding deep spatial structures exhibit feature-space rank collapse, while loop-induced information backflow captures representation degradation in the spatial domain, together explaining why spectral models often fail to reach their full potential.

*These authors contributed equally.
†Corresponding author.

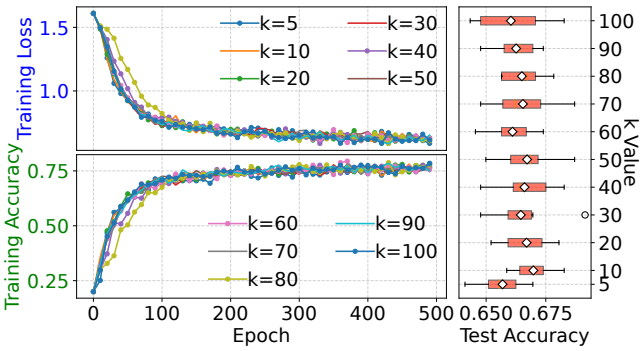


Figure 1: GPRGNN Performance on Chameleon Dataset with Different k Values

To address these issues, we propose DeloopSGNN, which imposes an acyclic constraint on the equivalent spatial model, converting deep propagation into deep acyclic propagation and mitigating oversmoothing. From this perspective, we find that adversarial attacks such as Metattack inject noise through short loops that construct deep cross-class connections; by removing these loops, DeloopSGNN blocks perturbation propagation, yielding an unintended yet significant gain in adversarial robustness.

We evaluate DeloopSGNN on 16 datasets of varying scales, covering both homophilic and heterophilic graphs, under two widely used adversarial attack methods. DeloopSGNN not only substantially improves the standard-task performance of spectral models but also achieves state-of-the-art robustness comparable to advanced defense methods.

Preliminaries & Related Work

This section outlines the mathematical foundations of graph neural networks, summarizes core spectral and spatial methods, and reviews adversarial attacks relevant to our study.

Preliminaries

Consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathbf{A}, \mathbf{X})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is the set of n nodes, $\mathbf{A} \in \{0,1\}^{n \times n}$ is the adjacency matrix with $\mathbf{A}_{ij} = 1$ if and only if nodes v_i and v_j are connected, and $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the node feature matrix with $\mathbf{x}_i \in \mathbb{R}^d$ representing the feature vector of node v_i. We also define key matrices frequently used in spectral graph theory: the normalized adjacency matrix $\tilde{\mathbf{A}}$ and the normalized Laplacian matrix $\tilde{\mathcal{L}} = \mathbf{I} - \tilde{\mathbf{A}}$, which is a real symmetric positive semi-definite matrix with eigenvalues $\tilde{\lambda}_i \in [0, 2)$. Its spectral decomposition $\tilde{\mathcal{L}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Lambda}}\tilde{\mathbf{U}}^T$ forms the basis of the spectral graph Fourier transform.

Spectral Graph Neural Networks

Spectral GNNs leverage graph signal processing by applying spectral filters to node features transformed via graph Fourier basis(Bo et al. 2023b). Prior work focuses on filter design: ChebNet (2019) uses Chebyshev polynomials, BernNet (2021) applies Bernstein-based band-pass filters, JacobiConv (2022) introduces learnable Jacobi filters, and Specformer (2023a) models eigenvalue dependencies using Transformer.

These models perform well on heterophilic graphs (e.g., Chameleon (2021)) by mitigating local homophily and promoting global information flow(Liao et al. 2024). However, in some cases, even with added learnable parameters (e.g., K-order coefficients in GPRGNN), training loss shows little improvement (see Fig. 1). This defies the conventional overfitting explanation, indicating deeper structural bottlenecks in spectral GNNs beyond parameter tuning. This study targets that core limitation.

Spatial Graph Neural Networks

Spatial methods aggregate neighborhood information via message passing, directly encoding graph topology. Representative models include GCN's weighted averaging (2016), GAT's attention (2017), and GraphSAGE's sampling (2017). For heterophilic graphs, H2GCN (2020) decouples node and neighbor features to capture higher-order neighborhoods, while MGNN (2023) combines spring-energy minimization with adaptive edge attention. Compared with spectral methods, spatial GNNs offer greater topological interpretability because their weights and aggregation explicitly correspond to real-world relations. Despite lacking formal spectral foundations, advanced spatial models perform on par with spectral ones on most benchmarks—including heterophilic graphs (2022)—thereby challenging the presumed superiority of spectral methods and revealing a gap between spectral theory and empirical performance. Their main limitation is oversmoothing: deep GNNs tend to produce indistinguishable node features (2019), largely due to vanishing Dirichlet energy (2020). Spectral models mitigate this via signal filtering (2023a), whereas spatial models require explicit mechanisms to prevent oversmoothing.

Due to their distinct formalisms and theoretical bases, spatial and spectral approaches have long been treated as separate GNN paradigms. Although recent work has begun to connect them (Chen et al. 2023; Guo et al. 2024), these efforts mostly target specific architectures or special cases. In this work, we rigorously prove that any spectral filtering operation admits an equivalent finite-step spatial aggregation, thereby establishing a general spectral-to-spatial transformation framework.

Graph Adversarial Attacks

Graph adversarial attacks reduce model robustness by perturbing graph structures or node features. These attacks are typically categorized as targeted (affecting specific nodes) or untargeted (causing overall performance degradation and posing higher security risks). To assess robustness, we employ two widely used untargeted attacks: PGD (2017), which uses projected gradient descent to introduce bounded perturbations that maximize prediction error; and Metattack (2019), a meta-learning approach that stealthily alters the graph structure to lower accuracy. Both are strong baselines and serve as key evaluation methods in this study.

Theory & Discovery

We first formulate the Spectral-to-Spatial Mapping Theorem and verifies it through two steps: (1) deconstructing GCN-SVD via spatial interpretation, and (2) revealing limitations of spectral methods and proposing targeted improvements.

Spectral-to-Spatial Mapping Theorem

We establish a rigorous mapping from spectral filters to spatial aggregation, providing the theoretical foundation for spectral-to-spatial transformation in GNNs. Specifically, we prove that any spectral filter $g(\Lambda)$ can be equivalently expressed as a spatial operator using at most n deterministic message-passing steps, where n is the number of nodes. We also derive the explicit coefficient mapping between spectral and spatial forms. To achieve this, we formalize the problem and define two classes of graph signal processing operators.

A graph processing operator can be defined in either the spectral or spatial domain. In the spectral domain, it uses a filter function $g : [0, 2] \to \mathbb{R}$; in the spatial domain, it relies on aggregation coefficients $\{\beta_k\}$. The respective outputs are:

$$\mathbf{Z}_{\text{spec}} = \tilde{\mathbf{U}}\, g(\tilde{\mathbf{\Lambda}})\, \tilde{\mathbf{U}}^T\mathbf{X}, \quad \mathbf{Z}_{\text{spat}} = \sum_{k=0}^{K} \beta_k\, \tilde{\mathbf{A}}^k\mathbf{X}.$$

Moreover, for any spectral filter $g(\lambda)$ there exists a unique set of spatial coefficients $\{\beta_k\}_{k=0}^{K-1}$ (with $K \le n$) such that

$$\tilde{\mathbf{U}}\, g(\tilde{\mathbf{\Lambda}})\, \tilde{\mathbf{U}}^T \;=\; \sum_{k=0}^{K} \beta_k\, \tilde{\mathbf{A}}^k$$

The proof is developed in the following three steps.

Step 1: Polynomial Exact Representation. Given that the eigenvalues $\{\lambda_i\}_{i=1}^{n}$ of $\tilde{\mathcal{L}}$ are finite and bounded by n, the polynomial interpolation theorem guarantees that for any function $g(\lambda)$, there exists a polynomial $P(\lambda)$ of degree at most $n-1$ such that the spectral operator admits the exact form $g(\tilde{\mathbf{\Lambda}}) = \sum_{k=0}^{K} \alpha_k \tilde{\mathcal{L}}^k$.

Step 2: Polynomial Basis Transformation. Using $\tilde{\mathcal{L}} = \mathbf{I} - \tilde{\mathbf{A}}$, expand the binomial form of $\tilde{\mathcal{L}}^k$:

$$\begin{aligned} g(\tilde{\mathbf{\Lambda}}) &= \sum_{k=0}^{K} \alpha_k \tilde{\mathcal{L}}^k = \sum_{k=0}^{K} \alpha_k \sum_{m=0}^{k} \binom{k}{m} (-1)^m \tilde{\mathbf{A}}^m \\ &= \sum_{m=0}^{K} [\sum_{k=m}^{K} \alpha_k \binom{k}{m} (-1)^m] \tilde{\mathbf{A}}^m \end{aligned}$$

The spatial kernel coefficients β_m are explicitly given by:

$$\beta_m = \sum_{k=m}^{K} \alpha_k \binom{k}{m} (-1)^m$$

Step 3: Invertibility Verification. Let's set the coefficient mapping matrix $\mathbf{H} \in \mathbb{R}^{K\times K}$ converts α into β:

$$[\beta_0, \ldots, \beta_K]^T = \mathbf{H}[\alpha_0, \ldots, \alpha_K]^T$$

The elements of $\mathbf{H}$ are defined as $\mathbf{H}_{mk} = \binom{k}{m}(-1)^m$ for $k \ge m$, and zero otherwise. Since $\mathbf{H}$ is upper triangular with non-zero diagonal entries, it is invertible.

This establishes a bijective correspondence between spectral and spatial coefficients, completing the proof of the theorem.

Spatial-to-Spectral Irreversibility

While S2SMT establishes an equivalence from spectral to spatial operations, most spatial GNNs do not fit this form. GAT is a typical example: its layer-specific attention weights $\alpha_{vu}^{(l)}$ and nonlinear activation σ in

$$h_v^{(l)} = \sigma\left(\sum_{u\in\mathcal{N}(v)} \alpha_{vu}^{(l)} \mathbf{W}^{(l)} h_u^{(l-1)}\right)$$

yield dynamic, nonlinear, and layer-dependent information flow. As a result, the spatial-to-spectral mapping is generally irreversible, due to (i) nonlinear aggregation, where core operations use learnable nonlinear rather than fixed linear combinations, and (ii) layer dependencies, since deep aggregations rely on shallow-layer outputs.

Deconstructing Existing Model GCN-SVD

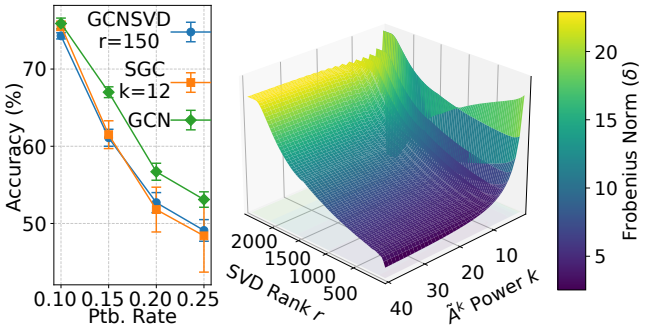
Building on the S2SMT, we revisit classical models from a spatial perspective, focusing on GCN-SVD. Although its low-rank approximation empirically improves robustness against adversarial attacks, its theoretical basis remains unclear. As stronger attacks emerge, GCN-SVD has occasionally failed (Zhang and Zitnik 2020), limiting its broader applicability. In essence, low-rank approximation acts as a band-stop filter: GCN-SVD suppresses mid-frequency signals via a low-rank adjacency matrix, while deep SGC achieves a similar effect through high-order propagation.

- **SGC (k-th power)**: The spectral response of $\tilde{\mathbf{A}}^k$ is $g(\lambda) = (1-\lambda)^k$, where $\lambda \in [0, 2]$. This function exponentially suppresses mid-frequency signals ($\lambda \to 1$) while retaining low and high frequencies.
- **GCN-SVD (Rank-r truncation)**: Given the singular value decomposition $\tilde{\mathbf{A}} = \mathbf{U}\Sigma\mathbf{V}^T$, where $\Sigma = \text{diag}(\sigma_i) = \text{diag}(|1-\lambda_i|)$, the rank-$r$ approximation is $\text{SVD}_r(\tilde{\mathbf{A}}) = \mathbf{U}_r\,\Sigma_r\,\mathbf{V}_r^T$. It retains the top-$r$ singular values, capturing dominant low and high-frequency components, while discarding smaller ones to suppress mid-frequency signals.

Given their spectral similarity, we conjecture that GCN-SVD and SGC exhibit similar robustness. As shown in Fig. 2a, with $r = 150$ and $k = 12$, both models show nearly identical performance—and failure—against Metattack on Cora. Operator analysis (Fig. 2b) further shows that their core operators converge toward isomorphic forms as parameters vary, as formalized below:

$$\lim_{r\to 1^+, k\to\infty} \left\| \text{SVD}_r(\tilde{\mathbf{A}}) - \tilde{\mathbf{A}}^k \right\|_F = 0$$

Results indicate that the spectral filters of both methods become equivalent after parameter tuning. Through S2SMT,



(a) (b)

Figure 2: Comparative analysis of GCN-SVD and SGC variants: (a) robustness under adversarial attacks, (b) Frobenius norm distance between $\text{SVD}_r(\tilde{\mathbf{A}})$ and $\tilde{\mathbf{A}}^k$ operators.

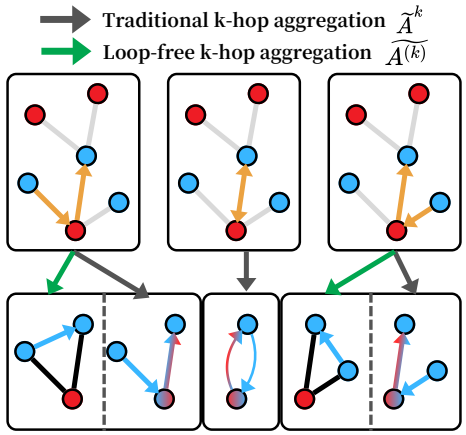


Figure 3: Comparison of $\tilde{\mathbf{A}}^k$ and $\widetilde{\mathbf{A}^{(k)}}$ Aggregation

GCN-SVD obtains a spatial interpretation: since SGC's $\tilde{\mathbf{A}}^k$ corresponds to k-hop aggregation, GCN-SVD's low-rank operation can be approximated similarly in the spatial domain. This view helps intuitively explain the core mechanisms behind GCN-SVD's robustness and vulnerability:

- **Robustness to low-budget/weak attacks:** The low-rank approximation—equivalent to deep aggregation—dissipates local perturbations via multi-hop propagation, gradually diluting adversarial signals across neighborhoods.
- **Vulnerability to high-budget/strong attacks:** The same deep aggregation mechanism establishes global signal propagation paths, which instead accelerate and amplify the spread of strong perturbations, expanding both the affected range and impact magnitude, ultimately leading to defense failure.

Defect Analysis of Spectral Methods

S2SMT shows that any spectral filter can be equivalently expressed as a spatial aggregation $\sum_{k=0}^{K} \beta_k \tilde{\mathbf{A}}^k$, revealing that spectral methods are intrinsically bounded by spatial operator structures. The next section analyzes their fundamental limitations from a spatial perspective.

Algebraic Nature Analysis: The equivalent spatial form comprises tunable coefficients β_k and aggregation operators $\tilde{\mathbf{A}}^k$. Since β_k directly correspond to spectral coefficients, they introduce no significant spectral distortion. We therefore focus on $\tilde{\mathbf{A}}^k$, starting from the eigen-decomposition $\tilde{\mathbf{A}} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$, where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$, with $|\lambda_1| = 1$ and $|\lambda_i| < 1$ for all $i > 1$. The k-th power satisfies $\tilde{\mathbf{A}}^k = \mathbf{V}\mathbf{\Lambda}^k\mathbf{V}^{-1}$, and in the limit, this operator projects any input signal onto the principal eigenspace:

$$\lim_{k\to\infty} \tilde{\mathbf{A}}^k\mathbf{X} = \mathbf{V} \cdot \text{diag}(1, 0, \ldots, 0) \cdot \mathbf{V}^{-1}\mathbf{X} = \mathbf{v}_1\mathbf{v}_1^T\mathbf{X}$$

Key Insight: The analysis shows spectral methods are fundamentally limited by the algebraic convergence of their equivalent spatial operators—i.e., spatial oversmoothing. In long-range aggregation, spectral GNNs rely on high-order terms $\tilde{\mathbf{A}}^k$, whose convergence remains unaffected by coefficient tuning. As a result, spectral models inevitably suffer signal degradation in deep propagation, limiting their capacity to capture long-range dependencies.

Spatial Deconstruction: From the spatial perspective, the root cause of the above defects lies in over-smoothing. Unlike the over-smoothing found in other complex spatial GNNs, this work focuses on avoiding the over-smoothing caused by the operator $\tilde{\mathbf{A}}^k$ to effectively alleviate spectral defects. As shown in Fig. 3, taking the central node of  as an example, when $k = 2$, $\tilde{\mathbf{A}}^2$ contains self-loop paths, mixing in low-order (1-hop) neighbor information; meanwhile, the 2-hop neighbors' information is diluted due to aggregation twice during propagation. In summary, the main reasons why $\tilde{\mathbf{A}}^k$ causes over-smoothing include:

1. **Signal Backflow:** $\tilde{\mathbf{A}}^k$ contains many computational loops (such as 3-hop paths $v \to u \to v \to w$), causing repeated traversal of low-order neighbors. Information accumulates, amplifies, and backflows through multiple loops, injecting a large amount of low-order noise.
2. **Repeated Aggregation:** The k-time averaging aggregation causes the original k-hop neighbor features to be repeatedly diluted along the propagation paths, weakening the distinguishability of distant information.

The corresponding solution will be proposed in the next section. From a spatial perspective, this paper proposes DeloopSGNN, which effectively mitigates oversmoothing by imposing acyclic constraints on the aggregation operator $\tilde{\mathbf{A}}^k$, achieving significant results.

Deloop Spectral GNN

The preceding analysis links spectral GNNs' limitations to signal loss from oversmoothing in $\tilde{\mathbf{A}}^k$, caused by signal reflux via loops and dilution from repeated aggregation. To address this, we propose **DeloopSGNN**, which reconstructs $\tilde{\mathbf{A}}^k$ by enforcing acyclic paths and integrating weighted high-order neighbor aggregation. This design blocks feed-

back loops, reduces low-order noise and feature confusion, and offers a spatial correction for spectral shortcomings.

We reorder the operation by applying the power before normalization, preventing information loss from repeated aggregation and preserving full k-hop neighborhoods. This also enables imposing an acyclic path constraint on $\mathbf{A}^k$, producing $\mathbf{A}^{(k)}$, and replacing $\tilde{\mathbf{A}}^k$ with the constrained $\widetilde{\mathbf{A}^{(k)}}$. Details of the acyclic constraint follow in the next section.

Acyclic Path Constraints

Constructing the k-th order acyclic adjacency matrix $\mathbf{A}^{(k)}$ requires solving the NP-hard acyclic path counting problem. Building on the k-order acyclic path framework from Wu et al., we extend it to matrix form to define $\mathbf{A}^{(k)}$. Exact computation is computationally feasible up to order five, which is sufficient for most practical applications; beyond that, approximation methods are employed.

1. **First-order:** Includes only direct adjacency relations and thus inherently satisfies the acyclic property: $\mathbf{A}^{(1)} = \mathbf{A}$
2. **Second-order:** To ensure acyclicity, self-loops in second-order paths are removed by zeroing the diagonal of $\mathbf{A}^2$, where $\mathcal{Z}(\cdot)$ denotes the operation that sets diagonal entries to zero: $\mathbf{A}^{(2)} = \mathcal{Z}(\mathbf{A}^2)$
3. **Third-order:** Eliminates 2-length cycles (e.g., $v \to u \to v$) by removing cyclic components from $\mathbf{A}^{(2)}\mathbf{A}$, where $\odot$ denotes the Hadamard (element-wise) product, $\mathbf{D_f}$ is the degree expansion matrix with $\mathbf{D_f}_{ij} = d_j$, $\mathbf{J}$ is the all-ones matrix:
$$\mathbf{A}^{(3)} = \mathcal{Z}\left(\mathbf{A}^{(2)}\mathbf{A} - \mathbf{A} \odot (\mathbf{D_f} - \mathbf{J})\right)$$
4. **Fourth-order:** Eliminates both 2-length and 3-length cycles through multi-order path correction. Here, $\mathcal{D}(\cdot)$ denotes the diagonal expansion operator, defined as $\mathcal{D}(x)_{ij} = x_{jj}$:
$$\mathbf{A}^{(4)} = \mathcal{Z}(\mathbf{A} \odot \mathbf{P}_1^{(4)} + (\mathbf{J} - \mathbf{A}) \odot \mathbf{P}_0^{(4)})$$
$$\mathbf{P}_1^{(4)} = \mathbf{A}^{(3)}\mathbf{A} - (\mathbf{D_f} - 4\mathbf{J}) \odot \mathbf{A}^{(2)} - \mathcal{D}(\mathbf{A}^{(2)}\mathbf{A})$$
$$\mathbf{P}_0^{(4)} = \mathbf{A}^{(3)}\mathbf{A} - (\mathbf{D_f} - \mathbf{J}) \odot \mathbf{A}^{(2)}$$
5. **Fifth-order:** Further eliminates complex cyclic structures through weighted combinations of multi-order paths:
$$\mathbf{A}^{(5)} = \mathcal{Z}(\mathbf{A} \odot \mathbf{P}_1^{(5)} + (\mathbf{J} - \mathbf{A}) \odot \mathbf{P}_0^{(5)})$$
$$\begin{aligned}\mathbf{P}_1^{(5)} = {} & \mathbf{A}^{(4)}\mathbf{A} - \mathbf{A}^{(3)} \odot (\mathbf{D_f} - 4\mathbf{J}) - \mathcal{D}(\mathbf{A}^{(3)}\mathbf{A}) \\ & - \mathbf{A}^2 \odot (\mathcal{D}(\mathbf{A}^{(2)}\mathbf{A}) - 2\mathbf{A}^{(2)} + 3\mathbf{J}) \\ & + 3\mathbf{A}(\mathbf{A}^{(2)} \odot \mathbf{A}) + \mathbf{A}^{(2)} \odot (\mathbf{A}^{(2)} - 1)\end{aligned}$$
$$\begin{aligned}\mathbf{P}_0^{(5)} = {} & \mathbf{A}^{(4)}\mathbf{A} - \mathbf{A}^{(3)} \odot (\mathbf{D_f} - \mathbf{J}) \\ & - \mathbf{A}^2 \odot \mathcal{D}(\mathbf{A}^{(2)}\mathbf{A}) + 3\mathbf{A}(\mathbf{A} \odot \mathbf{A}^{(2)})\end{aligned}$$
6. **Sixth-order and beyond:** Constructing exact acyclic paths of order six or higher is NP-hard. To address this, we use a recursive approximation algorithm that builds each acyclic path matrix from the previous one, reducing computational cost while approximating acyclic structure: $\mathbf{A}^{(k)} = \mathcal{Z}(\mathbf{A}^{(k-1)}\mathbf{A})$.

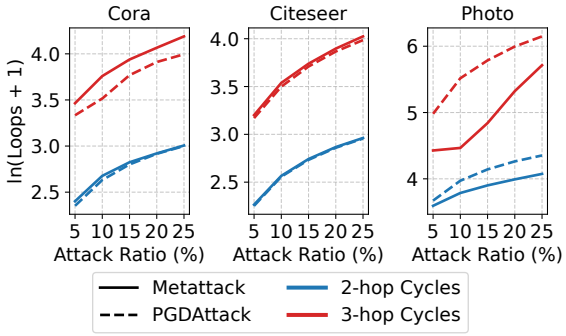


Figure 4: Short-cycle Injection by Adversarial Attacks

DeloopSGNN and Robust Analysis

The inter-layer propagation function of DeloopSGNN aggregates information using acyclic path matrices and learnable weights, and is formally expressed as:

$$\mathbf{Z} = \sum_{k=0}^{K} \beta_k \widetilde{A^{(k)}} \tag{1}$$

In addition, this extension facilitates the construction of corresponding spectral-domain operators via the S2SMT transformation matrix $\mathbf{H}$, thereby enhancing compatibility with arbitrary spectral GNN frameworks. In this work, we adopt the operator filter defined in Equation 1.

Robust Analysis Wu et al. highlight that adversarial attacks often compromise graph structure by injecting malicious edges. We further observe that such perturbations frequently create numerous short cycles (see Fig. 4), intensifying feature confusion. The acyclic pathway constraint effectively blocks this structural noise. Hence, we hypothesize that DeloopSGNN not only mitigates oversmoothing but also improves adversarial robustness—an assumption later confirmed by experiments.

Experiments

Building on the preceding theoretical analysis, this section experimentally validates DeloopSGNN's performance, focusing on two key objectives:

1. **Generalization:** Assess DeloopSGNN's performance on representative graph datasets compared to mainstream methods;
2. **Robustness:** Evaluate its defense effectiveness and stability under structural perturbations in common adversarial attack scenarios.

Experimental Setup

This section details the experimental setup for reproducibility and consistency. Experiments ran on a workstation with dual NVIDIA RTX A6000 GPUs (48GB each), a 16-core Intel Xeon W5-3435X CPU, and 128GB RAM. The system runs Ubuntu 22.04 with CUDA 12.7. Models were implemented in PyTorch 2.4, using a fixed random seed of 3.

Data	Nodes↑	Features	Edges	$\mathcal{H}_{\text{node}}$
Cornell	183	1703	277	0.3009
Texas	183	1703	279	0.0567
Reed98	962	745	18812	0.4539
Citeseer	2120	3703	3679	0.7108
Amherst41	2235	1193	90954	0.4666
Chameleon	2277	2325	31371	0.2471
Cora	2485	1433	5069	0.8145
Johnshopkins55	5157	2406	186572	0.4958
Squirrel	5201	2089	198353	0.2172
Photo	7487	745	119043	0.8488
Actor	7600	932	26659	0.2199
Film	7600	932	26659	0.2199
Computers	13381	767	245778	0.8017
Cornell5	18621	4735	790753	0.4803
Pubmed	19717	500	44324	0.7924
Penn94	22470	128	170823	0.8834

*$\mathcal{H}_{\text{node}} = \frac{1}{n}\sum_{v\in V}\frac{|\{u\in N(v)|y_u=y_v\}|}{|N(v)|} \in [0,1]$ denotes the node homophily indicator. A higher value indicates stronger homophily

Table 1: Dataset Information

Dataset Setup To thoroughly assess DeloopSGNN's generalization, we select 16 widely used benchmark datasets covering diverse scales, domains, and homophily levels. Dataset statistics are summarized in Table 1, providing comprehensive and representative evaluation. To assess adversarial robustness, we apply structural perturbations on four benchmark datasets (Cora(2024b), Citeseer(2024a), Photo, Computers) using Metattack and PGD. Perturbation rates $\epsilon \in \{0\%, 5\%, 10\%, 15\%\}$ simulate varying attack strengths.

Data Preprocessing For each dataset, only the largest connected component is retained to minimize peripheral node effects. Data splits vary by homophily: high-homophily datasets ($\mathcal{H}_{\text{node}} \geq 0.5$) use a 1:1:8 train/validation/test ratio for thorough generalization assessment, while low-homophily datasets adopt a 6:2:2 split to ensure sufficient training under complex structures.

Benchmark Models To evaluate DeloopSGNN, we compare it with representative spectral methods—GPRGNN, ChebNet, FAGCN (Bo et al. 2021), EvenNet (Lei et al. 2022), BernNet, and JacobiConv—as well as spatial methods including GCN, GAT, and H2GCN. For adversarial robustness, we consider strong defense baselines: GCN-SVD, GNNGuard (Zhang and Zitnik 2020), GCNJaccard (Wu et al. 2019b), NoisyGCN (Dai et al. 2022), and MidGCN (Huang et al. 2023).

Training Configuration: Models are trained with Adam optimizer (learning rate 0.01, weight decay 0.0005) for up to 1000 epochs with early stopping. Each experiment is run 10 times, reporting mean and standard deviation for significance. DeloopSGNN's path order K is tuned from 1 to 6 based on six degrees of separation, with the best value selected via grid search on the validation set. Baselines use default hyperparameters from their original implementations.

Performance Evaluation

Generalization Ability Evaluation Table 2 compares DeloopSGNN with mainstream GNNs across 16 datasets, showing its clear performance advantage. DeloopSGNN ranks first in 11 and second in 5 cases, consistently leading across diverse tasks and graph structures, demonstrating strong generalization across various settings.

DeloopSGNN excels across graphs with varying homophily. In heterophilic settings, it consistently outperforms advanced spectral models, surpassing BernNet by 8.1% on Squirrel and outperforming JacobiConv, the previous state-of-the-art. On medium- and high-homophily graphs (e.g., Penn94, Photo, Cora), it ranks among the top two, demonstrating strong adaptability. Low standard deviations across datasets further confirm its stable and robust training.

Beyond its performance, DeloopSGNN also shows strong computational efficiency and scalability. On large graphs like Cornell5, it runs without out-of-memory errors, demonstrating hardware adaptability. Its mathematically precise acyclic path construction enables highly parallel sparse operations with space complexity below $O(n^2)$, reducing overhead while preserving expressiveness—making it practical for large-scale deployment.

In summary, DeloopSGNN delivers an efficient, scalable, and generalizable graph-learning framework that achieves stable training and strong performance in both homophilic and heterophilic settings. These results confirm that the S2SMT theory effectively overcomes spectral-method limitations, laying a solid theoretical and practical foundation for future advances in spectral GNNs.

Robustness Evaluation Table 3 highlights DeloopSGNN's strong robustness under adversarial attacks. Across 32 test cases (2 attacks × 4 datasets × 4 perturbation levels), it ranks first in 28 and second in 3, significantly outperforming baselines and rivaling state-of-the-art robust GNNs. For example, on the Photo dataset under Metattack, most models exhibit sharp performance drops and high variance, while DeloopSGNN maintains stability, with only a 1.9% accuracy drop at 10% perturbation and consistent results at higher levels. Similar patterns are observed across other datasets and attacks, demonstrating DeloopSGNN's resilience and reliable training behavior.

DeloopSGNN's robustness stems from its unique acyclic path design, originally created to counter oversmoothing rather than adversarial attacks. This gives it inherent resistance to perturbations without extra defense modules. Under clean conditions ($\epsilon = 0\%$), it consistently leads in performance and avoids the accuracy drops common in over-defensive methods like GCN-SVD and GCN-Jaccard.

Overall, DeloopSGNN delivers strong robustness and stable training under adversarial attacks, while maintaining excellent accuracy in clean settings. This balance between defense and performance makes it well-suited for real-world applications where security is critical.

Conclusion

This paper presents a theoretical framework for the equivalence and transformation between spatial and spectral graph

Dataset	Spatial GNNs			Spectral GNNs						DeloopSGNN
($\mathcal{H}_{\text{node}} \downarrow$)	GCN	GAT	H2GCN	FAGCN	GPRGNN	ChebNet	EvenNet	BernNet	JacobiConv	
Penn94	90.4 ± 0.1	91.5 ± 0.1	**92.5 ± 0.2**	91.2 ± 0.7	91.5 ± 0.2	90.4 ± 0.2	91.9 ± 0.2	91.4 ± 0.1	91.3 ± 0.1	92.2 ± 0.2
Photo	93.5 ± 0.1	94.1 ± 0.1	94.4 ± 0.3	93.8 ± 1.0	93.7 ± 0.5	91.9 ± 1.0	94.1 ± 0.7	94.2 ± 0.2	93.5 ± 0.5	**94.8 ± 0.1**
Cora	83.3 ± 0.5	83.1 ± 0.8	83.6 ± 0.7	84.1 ± 0.6	84.0 ± 1.1	82.3 ± 0.8	82.8 ± 0.6	81.8 ± 1.8	82.8 ± 1.1	**84.4 ± 0.5**
Computers	89.5 ± 0.4	90.2 ± 0.2	90.0 ± 0.3	89.2 ± 0.5	87.6 ± 1.0	85.5 ± 2.3	88.6 ± 0.7	88.7 ± 0.4	81.6 ± 5.0	**90.7 ± 0.4**
Pubmed	86.5 ± 0.1	85.1 ± 0.2	87.0 ± 0.2	86.2 ± 0.2	87.2 ± 0.1	86.2 ± 0.2	87.3 ± 0.2	87.4 ± 0.1	87.0 ± 0.2	**87.7 ± 0.1**
Citeseer	71.9 ± 0.9	70.4 ± 2.0	72.6 ± 0.5	72.9 ± 1.2	69.8 ± 1.0	71.5 ± 0.9	72.6 ± 0.8	66.9 ± 1.0	71.2 ± 1.6	**73.9 ± 0.8**
Johnshopkins55	68.0 ± 0.1	66.6 ± 0.8	64.7 ± 1.6	65.3 ± 1.2	63.9 ± 0.3	59.6 ± 0.4	63.5 ± 0.2	66.4 ± 0.7	65.0 ± 0.6	**68.3 ± 0.6**
Cornell5	75.4 ± 0.3	73.2 ± 1.5	74.4 ± 0.6	74.0 ± 0.6	73.7 ± 0.3	OOM	77.6 ± 0.7	**77.9 ± 0.3**	73.7 ± 0.6	77.7 ± 0.8
Amherst41	68.1 ± 0.5	64.4 ± 1.8	68.9 ± 1.0	68.4 ± 3.0	66.2 ± 0.4	62.4 ± 1.1	68.4 ± 1.5	69.1 ± 1.0	69.3 ± 4.1	**71.4 ± 1.3**
Reed98	74.9 ± 2.7	**76.8 ± 0.5**	69.2 ± 2.3	69.9 ± 4.7	67.9 ± 6.9	70.9 ± 3.4	65.9 ± 3.5	64.3 ± 5.4	72.3 ± 3.4	75.1 ± 2.9
Cornell	59.3 ± 2.6	62.1 ± 4.5	72.3 ± 4.6	63.8 ± 3.1	88.5 ± 1.8	72.6 ± 2.8	93.6 ± 2.5	87.0 ± 1.9	82.1 ± 2.6	**94.1 ± 1.3**
Chameleon	61.9 ± 1.3	64.7 ± 1.5	65.4 ± 1.9	56.9 ± 2.0	52.9 ± 1.1	45.5 ± 2.2	63.4 ± 0.9	67.6 ± 1.1	49.6 ± 1.4	**69.3 ± 0.9**
Actor	31.2 ± 0.3	34.1 ± 0.6	38.3 ± 0.9	36.6 ± 0.8	34.6 ± 0.4	35.5 ± 0.5	37.4 ± 0.5	38.5 ± 0.6	35.1 ± 0.4	**41.5 ± 0.5**
Film	31.2 ± 0.3	33.8 ± 1.4	38.3 ± 0.9	36.6 ± 0.6	34.6 ± 0.4	35.5 ± 0.5	37.4 ± 0.2	**38.5 ± 0.4**	38.3 ± 0.8	38.4 ± 0.8
Squirrel	46.9 ± 0.6	45.9 ± 2.4	45.8 ± 1.3	41.9 ± 0.5	29.1 ± 1.3	27.0 ± 3.3	43.0 ± 1.2	48.3 ± 1.2	54.6 ± 1.6	**56.4 ± 1.4**
Texas	65.1 ± 3.4	75.1 ± 3.9	91.6 ± 2.4	63.9 ± 5.1	88.9 ± 1.2	78.0 ± 2.8	**93.8 ± 1.8**	92.3 ± 1.8	89.3 ± 2.3	93.0 ± 0.8

Table 2: Comparison of models on various datasets, with first and second scores in **bold** and underline, respectively.

	Dataset	ϵ	GCN	GAT	GCNSVD	GNNGuard	GCNJaccard	MidGCN	NoisyGCN	DeloopSGNN
Metattack	Cora	0%	83.6 ± 0.6	83.5 ± 0.7	76.5 ± 0.4	83.4 ± 0.3	81.8 ± 0.6	83.7 ± 0.4	83.6 ± 0.3	**84.4 ± 0.5**
		5%	80.6 ± 0.3	80.0 ± 0.8	75.4 ± 0.5	80.3 ± 0.4	81.3 ± 0.3	81.6 ± 0.4	80.6 ± 0.3	**82.6 ± 0.4**
		10%	75.9 ± 0.5	75.8 ± 1.6	72.8 ± 0.7	76.0 ± 0.5	78.0 ± 0.3	78.7 ± 0.5	76.4 ± 0.5	**79.5 ± 0.6**
		15%	66.6 ± 1.0	66.4 ± 3.0	68.6 ± 0.5	67.2 ± 0.4	72.9 ± 0.7	**72.9 ± 0.6**	67.6 ± 0.5	72.0 ± 2.7
	Citeseer	0%	68.6 ± 1.2	68.7 ± 1.8	68.1 ± 1.3	69.3 ± 0.4	69.4 ± 0.6	71.3 ± 1.2	69.3 ± 0.7	**73.9 ± 0.8**
		5%	64.8 ± 0.7	64.5 ± 2.8	68.8 ± 0.6	65.7 ± 0.7	66.9 ± 1.1	68.1 ± 0.9	65.5 ± 0.7	**69.6 ± 0.9**
		10%	60.9 ± 1.0	60.4 ± 3.2	**67.3 ± 0.7**	61.2 ± 1.0	64.2 ± 1.0	65.6 ± 1.2	61.5 ± 0.6	66.8 ± 1.1
		15%	55.8 ± 1.3	57.3 ± 2.6	**65.6 ± 1.4**	56.1 ± 0.9	61.5 ± 0.9	62.1 ± 0.9	57.0 ± 1.2	64.1 ± 1.1
	Photo	0%	93.9 ± 0.3	94.3 ± 0.2	88.6 ± 0.2	93.5 ± 0.4	93.9 ± 0.3	90.3 ± 0.3	93.5 ± 0.2	**94.8 ± 0.1**
		5%	90.7 ± 0.9	88.7 ± 1.1	87.0 ± 0.3	90.4 ± 1.3	91.1 ± 0.4	86.9 ± 0.3	90.9 ± 0.7	**94.2 ± 0.3**
		10%	86.0 ± 1.0	83.5 ± 2.0	83.5 ± 0.4	86.2 ± 1.6	86.3 ± 1.2	85.1 ± 0.4	84.9 ± 1.7	**92.9 ± 0.7**
		15%	76.3 ± 2.7	60.0 ± 6.5	65.9 ± 1.6	74.4 ± 3.8	73.5 ± 3.9	73.4 ± 1.8	71.9 ± 7.8	**89.1 ± 1.8**
	Computers	0%	89.2 ± 0.4	90.5 ± 0.3	77.0 ± 1.7	88.6 ± 0.8	89.4 ± 0.5	81.1 ± 1.1	88.8 ± 0.7	**90.7 ± 0.4**
		5%	85.1 ± 0.8	85.5 ± 0.9	74.0 ± 1.0	80.0 ± 7.1	84.8 ± 1.0	74.3 ± 1.1	81.8 ± 3.0	**87.2 ± 0.7**
		10%	75.7 ± 2.9	72.6 ± 6.2	68.7 ± 1.2	69.6 ± 8.1	75.8 ± 2.2	71.6 ± 1.2	72.6 ± 5.8	**85.1 ± 0.8**
		15%	76.5 ± 4.0	78.1 ± 2.3	66.5 ± 0.9	53.6 ± 10.0	73.9 ± 2.9	67.8 ± 2.2	61.5 ± 13.2	**84.4 ± 0.9**
PGD Attack	Cora	0%	83.4 ± 0.5	82.8 ± 1.2	76.3 ± 0.5	83.7 ± 0.3	81.6 ± 0.6	83.5 ± 0.3	83.7 ± 0.2	**84.4 ± 0.5**
		5%	79.0 ± 0.5	79.4 ± 1.2	75.5 ± 0.5	79.0 ± 0.2	79.4 ± 0.4	79.8 ± 0.3	79.2 ± 0.4	**80.7 ± 0.5**
		10%	77.1 ± 0.4	77.5 ± 0.5	73.7 ± 0.4	76.8 ± 0.4	77.3 ± 0.4	77.2 ± 0.4	77.3 ± 0.2	**78.6 ± 0.4**
		15%	75.2 ± 0.6	75.5 ± 0.9	73.0 ± 0.4	75.1 ± 0.3	75.2 ± 0.6	75.1 ± 0.5	75.5 ± 0.3	**76.6 ± 0.5**
	Citeseer	0%	68.7 ± 0.8	67.5 ± 1.2	68.2 ± 1.0	69.0 ± 0.6	69.7 ± 1.3	70.2 ± 1.1	69.6 ± 0.8	**73.9 ± 0.8**
		5%	67.8 ± 1.0	65.9 ± 1.9	68.0 ± 1.3	66.8 ± 0.6	67.2 ± 0.8	68.4 ± 1.1	67.2 ± 1.7	**69.5 ± 1.0**
		10%	65.2 ± 0.7	64.2 ± 1.8	**67.3 ± 0.8**	65.0 ± 1.3	66.2 ± 1.4	66.4 ± 0.7	65.4 ± 1.4	66.8 ± 1.2
		15%	62.8 ± 1.5	63.6 ± 2.2	65.9 ± 1.1	63.2 ± 1.7	65.0 ± 1.2	65.5 ± 1.2	63.0 ± 1.3	**66.7 ± 1.2**
	Photo	0%	93.9 ± 0.2	94.1 ± 0.3	88.6 ± 0.3	93.7 ± 0.3	94.0 ± 0.2	90.3 ± 0.3	93.7 ± 0.3	**94.8 ± 0.1**
		5%	85.4 ± 0.2	85.8 ± 0.7	85.6 ± 0.3	85.1 ± 0.3	85.5 ± 0.3	84.7 ± 0.5	85.2 ± 0.3	**89.3 ± 0.4**
		10%	81.7 ± 0.4	79.8 ± 2.8	82.1 ± 0.3	82.1 ± 0.4	81.9 ± 0.5	81.6 ± 0.6	81.7 ± 0.6	**86.2 ± 0.8**
		15%	79.4 ± 0.6	78.2 ± 3.1	78.5 ± 0.6	79.1 ± 0.8	79.3 ± 0.6	80.3 ± 0.7	79.5 ± 1.1	**83.5 ± 1.3**
	Computers	0%	89.4 ± 0.4	90.1 ± 0.6	77.5 ± 0.8	88.6 ± 0.5	89.5 ± 0.2	80.8 ± 0.8	88.6 ± 0.7	**90.7 ± 0.4**
		5%	80.8 ± 1.0	82.2 ± 0.2	74.8 ± 1.6	80.2 ± 1.2	80.6 ± 1.4	75.2 ± 0.6	79.5 ± 1.1	**84.3 ± 0.7**
		10%	75.5 ± 1.9	78.1 ± 1.2	69.1 ± 1.2	68.4 ± 1.8	73.2 ± 2.4	72.4 ± 1.4	72.7 ± 1.9	**81.2 ± 0.6**
		15%	69.3 ± 2.3	73.2 ± 5.4	65.4 ± 1.9	69.1 ± 1.0	68.9 ± 1.8	69.9 ± 1.3	68.4 ± 1.6	**79.0 ± 0.8**

Table 3: Robust performance under Metattack and PGD Attack, with first and second scores in **bold** and underline, respectively.

neural networks. Building on this foundation, we propose DeloopSGNN—a novel spectral GNN that integrates the intuitive structure of spatial models with the theoretical strengths of spectral methods. By mitigating signal loops and high-order aggregation effects, DeloopSGNN significantly improves expressive power and robustness. We believe this work offers valuable insights for advancing spectral GNN design.

Acknowledgments

This project is supported by the National Natural Science Foundation of China (Grant Nos. 62306328, U24A2033, 62302512, 62421002) and the Science and Technology Innovation Program of Hunan Province (Grant Nos. 2023RC3021, 2024RC1047), as well as the Youth Independent Innovation Science Foundation of National University of Defense Technology (Grant No. ZK24-02).

References

Bo, D.; Shi, C.; Wang, L.; and Liao, R. 2023a. Specformer: Spectral graph neural networks meet transformers. *arXiv preprint arXiv:2303.01028*.

Bo, D.; Wang, X.; Liu, Y.; Fang, Y.; Li, Y.; and Shi, C. 2023b. A survey on spectral graph neural networks. *arXiv preprint arXiv:2302.05631*.

Bo, D.; Wang, X.; Shi, C.; and Shen, H. 2021. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 3950–3957.

Cai, C.; and Wang, Y. 2020. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*.

Chen, Z.; Chen, F.; Zhang, L.; Ji, T.; Fu, K.; Zhao, L.; Chen, F.; Wu, L.; Aggarwal, C.; and Lu, C.-T. 2023. Bridging the gap between spatial and spectral domains: A unified framework for graph neural networks. *ACM Computing Surveys*, 56(5): 1–42.

Cui, G.; and Wei, Z. 2023. Mgnn: Graph neural networks inspired by distance geometry problem. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 335–347.

Dai, E.; Jin, W.; Liu, H.; and Wang, S. 2022. Towards robust graph neural networks for noisy graphs with sparse labels. In *Proceedings of the fifteenth ACM international conference on web search and data mining*, 181–191.

Entezari, N.; Al-Sayouri, S. A.; Darvishzadeh, A.; and Papalexakis, E. E. 2020. All you need is low (rank) defending against adversarial attacks on graphs. In *Proceedings of the 13th international conference on web search and data mining*, 169–177.

Feng, Y.; Liang, W.; Wan, X.; Liu, J.; Li, M.; and Liu, X. 2025a. Incremental Multi-View Clustering: Exploring Stream-View Correlations to Learn Consistency and Diversity. *IEEE Transactions on Knowledge and Data Engineering*.

Feng, Y.; Liang, W.; Wan, X.; Liu, J.; Liu, S.; Qu, Q.; Guan, R.; Xu, H.; and Liu, X. 2025b. Incremental Nyström-based Multiple Kernel Clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 16613–16621.

Guo, J.; Huang, K.; Yi, X.; Su, Z.; and Zhang, R. 2024. Rethinking spectral graph neural networks with spatially adaptive filtering. *arXiv preprint arXiv:2401.09071*.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

He, M.; Wei, Z.; Xu, H.; et al. 2021. Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. *Advances in neural information processing systems*, 34: 14239–14251.

Huang, J.; Du, L.; Chen, X.; Fu, Q.; Han, S.; and Zhang, D. 2023. Robust mid-pass filtering graph convolutional networks. In *Proceedings of the ACM Web Conference 2023*, 328–338.

Huang, K.; Wang, Y. G.; Li, M.; et al. 2024. How universal polynomial bases enhance spectral graph neural networks: Heterophily, over-smoothing, and over-squashing. *arXiv preprint arXiv:2405.12474*.

Kipf, T. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907*.

Lei, R.; Wang, Z.; Li, Y.; Ding, B.; and Wei, Z. 2022. Evennet: Ignoring odd-hop neighbors improves robustness of graph neural networks. *Advances in neural information processing systems*, 35: 4694–4706.

Liao, N.; Liu, H.; Zhu, Z.; Luo, S.; and Lakshmanan, L. V. 2024. Benchmarking spectral graph neural networks: A comprehensive study on effectiveness and efficiency. *arXiv preprint arXiv:2406.09675*.

Liu, X.; Chen, J.; and Wen, Q. 2023. A survey on graph classification and link prediction based on gnn. *arXiv preprint arXiv:2307.00865*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Oono, K.; and Suzuki, T. 2019. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*.

Rozemberczki, B.; Allen, C.; and Sarkar, R. 2021. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2): cnab014.

Sun, Y.; Deng, H.; Yang, Y.; Wang, C.; Xu, J.; Huang, R.; Cao, L.; Wang, Y.; and Chen, L. 2022. Beyond Homophily: Structure-aware Path Aggregation Graph Neural Network. In *IJCAI*, 2233–2240.

Tangb, S.; Lib, B.; and Yua, H. 2019. Chebnet: Efficient and stable constructions of deep neural networks with rectified power units using chebyshev approximation. *arXiv preprint arXiv:1911.05467*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, X.; and Zhang, M. 2022. How powerful are spectral graph neural networks. In *International conference on machine learning*, 23341–23362. PMLR.

Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. 2019a. Simplifying graph convolutional networks. In *International conference on machine learning*, 6861–6871. Pmlr.

Wu, H.; Wang, C.; Tyshetskiy, Y.; Docherty, A.; Lu, K.; and Zhu, L. 2019b. Adversarial examples on graph data: Deep insights into attack and defense. *arXiv preprint arXiv:1903.01610*.

Wu, J.; Chen, N.; Zhou, C.; Che, H.; Han, C.; and Liu, Q. 2020. Computing the Number of Loop-free k-hop Paths of Networks. *IEEE Transactions on Services Computing*, 15(4): 2114–2128.

Xu, L.; Chen, J.; Han, Z.; and Zhang, Y. 2024. TaylorNet: A novel approach for spectral filter learning on graph data. *Neurocomputing*, 605: 128358.

Yang, X.; Min, E.; Liang, K.; Liu, Y.; Wang, S.; Zhou, S.; Wu, H.; Liu, X.; and Zhu, E. 2024a. Graphlearner: Graph node clustering with fully learnable augmentation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 5517–5526.

Yang, X.; Wang, Y.; Liu, Y.; Wen, Y.; Meng, L.; Zhou, S.; Liu, X.; and Zhu, E. 2024b. Mixed graph contrastive network for semi-supervised node classification. *ACM Transactions on Knowledge Discovery from Data*.

Zhang, X.; and Zitnik, M. 2020. Gnnguard: Defending graph neural networks against adversarial attacks. *Advances in neural information processing systems*, 33: 9263–9275.

Zhu, J.; Yan, Y.; Zhao, L.; Heimann, M.; Akoglu, L.; and Koutra, D. 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, 33: 7793–7804.

Zügner, D.; and Günnemann, S. 2019. Adversarial Attacks on Graph Neural Networks via Meta Learning. In *International Conference on Learning Representations (ICLR)*.