GRIPEDGE: HETEROPHILY-AWARE GRAPH LEARNING VIA ATTENTIONAL FEATURE-SPECTRAL NEIGHBOUR PROPAGATION IN DENSE GRAPHS*

Anonymous authors
Paper under double-blind review

ABSTRACT

Node classification in heterophilic graphs remains a challenging task, as connected nodes often belong to different classes and exhibit heterogeneous features. The assumption of homophily, which is typical in GNNs, encounters problems such as oversmoothing and reduced separability and leads to low performance on dense heterophilic benchmarks such as *Squirrel* and *Chameleon*. We therefore propose a unified framework that improves feature representation, structural learning and spectral aggregation. Our approach combines attention-based mechanisms to integrate local and global neighborhood information, spectral modulation to capture oscillatory node–edge patterns and edge augmentation inspired by structure learning to refine graph connectivity. Extensive experiments demonstrate that our model consistently offers robust and discriminative node embeddings and outperforms state-of-the-art methods on the task of node classification in dense graphs.

1 INTRODUCTION

Node classification in graphs is a fundamental task in machine learning, with applications in social networks, biological networks and recommendation systems (Kipf & Welling, 2017; Veličković et al., 2018). Traditional GNNs rely on the homophily assumption, which posits that neighboring nodes share similar features or classes (Hamilton et al., 2017). While valid for many benchmarks, this assumption fails in heterophilic graphs, where adjacent nodes often differ in class and features (Yan et al., 2021), causing oversmoothing and reduced discriminative power (Li et al., 2018; Li & et al., 2023).

Recent research has highlighted that effective graph learning in heterophilic settings requires addressing challenges across multiple dimensions: structural design, feature aggregation, representation learning, and spectral filtering. We group related approaches into four categories, each reflecting a core strategy for graph representation learning.

Feature-driven models rely on explicit node attributes to reduce dependence on noisy graph structures, with architectures such as MixHop (Abu-El-Haija et al., 2019), LINKX (Lim et al., 2021), Graphformer (Ying et al., 2021) and Gophormer (Yang et al., 2021) exploring multi-hop feature mixing, layer-wise feature transformation or attention-based fusion of local and global contexts. Self-supervised contrastive learning reduces reliance on labels by maximizing agreement between positive pairs and pushing apart negatives (Veličković et al., 2019; Hassani & Khasahmadi, 2020; Qiu et al., 2020), yielding discriminative embeddings. In heterophilic graphs, methods like HLCL (Yang & Mirzasoleiman, 2024), RGSL (Xie et al., 2024), HGIF (Ren et al., 2024), Co-GSL (Liu et al., 2021), SLAPS (Zheng et al., 2021), and GCC (Qiu et al., 2020) face hard negatives, where nodes from different classes may appear similar, leading to misleading supervision. Graph Structure Learning (GSL) infers task-specific adjacency matrices with node representations when the graph is incomplete or noisy (Wu & et al., 2023). Methods such as FAGCN (Bo et al., 2021), Geom-GCN (Pei et al., 2020), LDS (Franceschi et al., 2019), PTDNet (Zhang et al., 2020), and KDGA (Wu et al., 2022) refine or augment edges. In heterophilic graphs, GSL must balance preserving informative

*We utilized ChatGPT to assist in text creation and refinement, support literature research and facilitate idea generation and development.

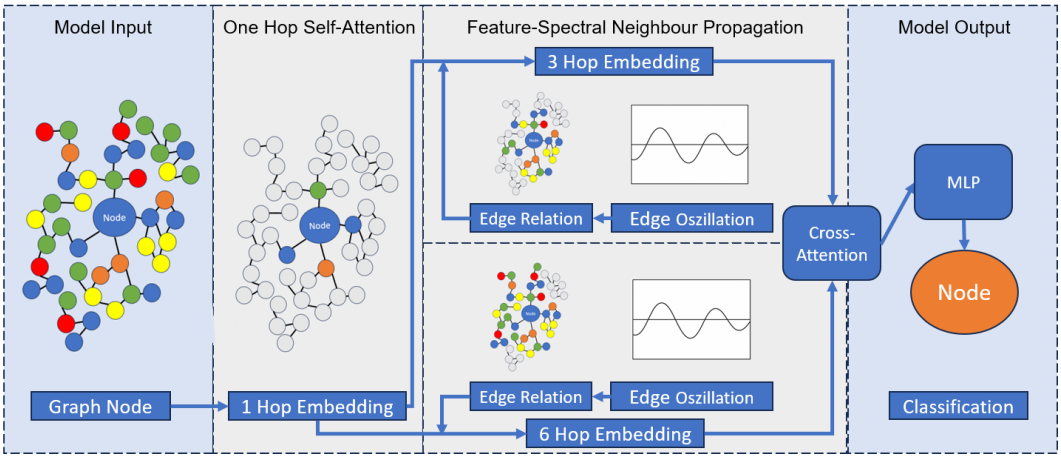

Figure 1: Our model GRIP first generates one-hop self-attention embeddings, which are subsequently processed by feature–spectral GCN layers to obtain multi-hop representations. These representations are then fused through cross-attention to yield the final node embedding.

edges with denoising spurious links. Spectral approaches analyze graph signals in the frequency domain to enhance node feature aggregation and filtering, with models such as GPR-GNN (Chien et al., 2020), att-Node-level NLSFs (Lin, 2024), and APPNP (Klicpera et al., 2019) exploiting low- and high-frequency components to improve expressivity. In heterophilic graphs, these methods must carefully modulate oscillatory signals, as class information can be entangled with high-frequency noise.

Contributions. We introduce a unified framework for node classification in heterophilic dense graphs that integrates advances in feature learning, spectral aggregation and structure refinement see fig. 1. First, similar to models as Graphformer (Ying et al., 2021) and Gophormer (Yang et al., 2021), our model enhances feature-based learning with attention: a hop-1 self-attention module precedes the GCN layers to capture fine-grained local signals, while cross-attention after the GCN layers fuses local and global representations.

Second, to provide embeddings with as much relevant information as possible from both nearby and distant neighbors, the framework leverages local and global aggregation of feature- and spectral-based neighborhood relationships. In particular, we incorporate a novel spectral approach to capture oscillatory node–edge patterns. This spectral modulation extends beyond feature-only methods like MixHop (Abu-El-Haija et al., 2019) or LINKX (Lim et al., 2021), and it is related to frequency-based approaches such as att-Node-level NLSFs (Lin, 2024).

Finally, inspired by structure learning, we adopt KDGA-style edge augmentation (Wu et al., 2022) to refine graph connectivity, preserving heterophilic links and strengthening meaningful structures. The integration of these components yields strong performance on dense heterophilic benchmarks like Squirrel and Chameleon, better exploiting heterophilic signals. Our model will be available on GitHub upon acceptance.

2 Related Work

In this section, we review related work most relevant to our model, focusing on three key areas already outlined in the introduction: feature learning, structure learning, and spectral methods. Note that contrastive learning approaches are not discussed here as they are not applied in our framework.

2.1 Feature Learning in Graphs

Feature-based approaches in graphs can be grouped into two categories (Yuan et al., 2023): the first exploits distant neighbors to capture semantic similarity beyond immediate adjacency, while the second adapts GNN architectures to handle heterogeneous neighbors more effectively.

MixHop (Abu-El-Haija et al., 2019) aggregates multi-hop neighborhoods, and WRGAT (Suresh et al., 2021) models long-range dependencies with adaptive edge weights. Transformer-inspired models such as Graphormer (Ying et al., 2021) and Gophormer (Yang et al., 2021) encode structural priors and global context. These methods leverage node attributes and multi-hop feature mixing, often via attention, to fuse local and global information. While reducing reliance on noisy structures, capturing long-range dependencies remains difficult. The second category modifies GNNs for heterogeneous neighbors. H2GCN (Zhu, 2020) separates ego features from neighbors to mitigate oversmoothing, LINKX (Lim et al., 2021) disentangles feature and structural processing, and GGCN (Yan et al., 2021) applies signed, weighted combinations of previous-layer representations.

Our approach aligns with the first category, integrating multi-hop information. Similar to Transformer-based methods, we use attention to capture fine-grained neighborhood differences while retaining GCN-style message passing. This hybrid combines attention expressivity with GCN efficiency, enabling deeper inspection of node features and explicit fusion of local and global neighbor information to capture short-, mid-, and long-range dependencies.

2.2 Structural Graph Learning

Structural Graph Learning (GSL) addresses incomplete, noisy, or missing graph structures by jointly learning task-specific adjacency matrices and node representations (Wu & et al., 2023). GSL typically combines structure optimization with feature or label propagation, refining or augmenting edges to capture relevant information (Bo et al., 2021; Pei et al., 2020; Franceschi et al., 2019; Zhang et al., 2020; Wu et al., 2022). In heterophilic graphs, preserving informative edges while denoising spurious links remains challenging.

GSL approaches fall into three groups. The first, classical adaptive structure modification, adds or removes edges to improve information flow, e.g., FAGCN (Bo et al., 2021) adjusts edge weights via feature similarity, GEOM-GCN (Pei et al., 2020) selects neighbors in both original and latent spaces, and IDGL (Chen et al., 2020) iteratively optimizes the adjacency matrix. The second group, distillation-based approaches, includes KDGA (Wu et al., 2022), using a student–teacher strategy to stabilize edge learning, and LDS (Franceschi et al., 2019), which iteratively refines edges and node representations probabilistically. The third group integrates structural refinement into features or architecture. JK-Net (Xu et al., 2018) and ResNet+adj use aggregated representations or ResNet-style skip connections to weight neighbor contributions and stabilize the graph.

Our method excels on dense graphs, so we enhance performance by adding edges to the learned graph. Direct addition fails if the graph is unlearned or fully learned, making embeddings hard to modify. We thus adopt KDGA, adding and removing edges during training and using teacher–student distillation to improve connectivity and capture meaningful relationships.

2.3 Spectral Methods in Graphs

Spectral graph theory underlies many graph learning methods by analyzing Laplacian eigenvalues and eigenvectors, with eigenvectors forming a Fourier basis and eigenvalues representing frequencies. Spectral GNNs use this to design filters controlling information flow and improve feature aggregation (Klicpera et al., 2019). In heterophilic graphs, oscillatory components must be carefully modulated to separate class signals from high-frequency noise. We categorize spectral models into three classes to clarify their designs.

The first category uses polynomial approximations of Laplacian eigenvalues for multi-hop aggregation, e.g., ChebNet (Defferrard et al., 2016) via Chebyshev polynomials. The second learns task-specific spectral responses, e.g., APPNP (Klicpera et al., 2019) combining predictions with personalized PageRank diffusion, and GPR-GNN (Chien et al., 2020) learning polynomial coefficients to emphasize selected frequency bands. The third category incorporates attention, nonlinear operations, or hybrid designs, e.g., Specformer (Bo et al., 2023) and att-Node-level NLSFs (Lin, 2024), exploiting both low- and high-frequency components to reduce oversmoothing and improve expressivity.

Our model explicitly uses edge oscillation to modulate edge–feature relations, emphasizing meaningful high-frequency differences while downweighting noisy connections. This spectral perspec-

tive complements feature-based similarity measures, placing the model in the third category with oscillation-aware, nonlinear modulation beyond standard polynomial filters.

3 Feature-Spectral Neighbour Propagation Framework

3.1 Problem Formulation

Let $G = (V, E)$ be a graph with n nodes, where each node $v_i \in V$ is associated with an input feature vector $x_i \in \mathbb{R}^d$, collected in the feature matrix $X \in \mathbb{R}^{n \times d}$. In the transductive setting, the graph structure $A \in \mathbb{R}^{n \times n}$ is fixed and used to learn improved node representations $Z \in \mathbb{R}^{n \times d'}$. These embeddings are then used to perform a downstream node-level task, such as (semi-)supervised node classification. Graph convolution networks (GCNs) update node representations through iterative message passing. At each layer t, a node v_i aggregates messages from its neighbors $\mathcal{N}(v_i)$ and updates its representation based on the aggregated message and its previous representation according to the update equations

$$\begin{aligned} m_i^{(t)} &= \text{AGGREGATE}^{(t)}\{h_j^{(t-1)} \mid v_j \in \mathcal{N}(v_i)\} \\ \text{and } h_i^{(t)} &= \text{COMBINE}^{(t)}\big(h_i^{(t-1)}, m_i^{(t)}\big), \end{aligned} \tag{1}$$

where $h_i^{(t)}$ denotes the representation of node v_i at layer t. In general, a graph neural network layer updates each node's representation in two steps. First, the node aggregates messages from its neighbors to form $m_i^{(t)}$, which encodes information about the local neighborhood. Second, the node combines this aggregated message with its previous representation to produce the updated embedding $h_i^{(t)}$. The exact choice of aggregation and combination functions can vary between GCN architectures. Some methods may incorporate attention mechanisms, normalization, or spectral information, but the core principle remains the same: information flows from neighbors to the target node through iterative message passing (Chen et al., 2020; Yuan et al., 2023).

3.2 Initial one-hop self-attention node embeddings

Before performing standard GCN message passing, we first enhance each node's initial embedding by attending over its immediate neighbors. While standard GCNs aggregate neighbor features using uniform or degree-normalized weights, self-attention computes a context-dependent weighting for each neighbor in parallel. This enables the model to capture subtle differences between neighbors and fine-grained relational patterns, which is especially important in heterophilic graphs (Veličković et al., 2018; Yang et al., 2021).

Each node uses query Q, key K, and value V, which are created from the raw node features via simple linear projections, to compute attention over its neighbors. We include self-loops, meaning that each node attends not only to its neighbors but also to itself; in other words, we use the extended neighborhood $\mathcal{N}^+(i) = \mathcal{N}(i) \cup \{i\}$, see Lampert & Scholtes (2023). Attention is then computated by

$$\begin{aligned} e_{ij} &= \frac{Q_i^\top K_j}{\sqrt{F_{\text{hidden}}}}, \\ \alpha_{ij} &= \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}^+(i)} \exp(e_{ik})} \\ \text{and } h_i' &= \sum_{j \in \mathcal{N}^+(i)} \alpha_{ij} V_j. \end{aligned} \tag{2}$$

The raw attention score e_{ij} measures the compatibility between node v_i and its neighbor v_j, with F_{hidden} denoting the dimensionality of the hidden node embeddings. The normalized attention weight α_{ij} is obtained by applying a softmax over all neighbors in the extended neighborhood. This allows the model to assign more importance to the most relevant neighbors while reducing the influence of less informative ones.

Before passing node embeddings into the actual GCN layer, the original input h_i from eq. (2) is replaced by these optimized embeddings h_i'. In this way, one-hop self-attention refines the initial

node representations, providing the GCN with richer and more informative features for subsequent message passing. This preprocessing step can improve the ability of the GCN to capture subtle differences between neighbors and model complex relational patterns in the graph.

3.3 FEATURE-SPECTRAL NEIGHBOUR PROPAGATION

After the one-hop self-attention step from section 3.2, the refined embeddings h'_i serve as input to the subsequent GCN-style layer. Each h'_j denotes the optimized representation of neighbor node v_j, obtained from the self-attention mechanism, which already captures context-dependent neighbor relations. The goal of this layer is to provide wide neighborhood information, which can then be selectively used through attention. To achieve this, we first compute feature-driven edge relations, then measure spectral oscillations in the Laplacian space, and finally fuse both signals into a compact and expressive representation. For each edge $(i, j) \in E$, we compute a feature-driven attention score (Chen & Chen, 2021)

$$e_{ij}^{\text{feat}} = \text{MLP}\big([Wh'_i \,\|\, Wh'_j]\big), \tag{3}$$

where W is a linear projection and $\|$ denotes concatenation. In parallel, we compute a spectral term

$$e_{ij}^{\text{spec}} = f_\theta\left(\sum_{k=1}^{K} \lambda_k \cdot (u_{k,i} - u_{k,j})^2\right) \tag{4}$$

to measure how nodes differ in the Laplacian eigenbasis. Here, λ_k are the eigenvalues and $u_{k,i}$ are components of the Laplacian eigenvectors. The eigenpairs (λ_k, u_k) are obtained from the graph Laplacian L. Eigenvalues correspond to frequency scales of the graph, while eigenvectors encode oscillation modes. Only the K leading eigenvectors of the Laplacian are used for the spectral term, where K is manually set as a hyperparameter. Thus, e_{ij}^{spec} quantifies how strongly two nodes oscillate relative to each other in the spectral domain. This provides a structural signal complementary to the feature-based similarity. The two signals e_{ij}^{feat} and e_{ij}^{spec} are then fused via modulation (Liu et al., 2023a)

$$e_{ij} = e_{ij}^{\text{feat}} \cdot \big(1 + \tanh(e_{ij}^{\text{spec}})\big), \tag{5}$$

where the spectral signal acts only in a moderating way—it slightly amplifies e_{ij}^{feat} depending on spectral differences.

The final attention weights α_{ij} are obtained by applying a softmax normalization over e_{ij} across the neighbors of i, and the node update is then perfomed by

$$h_i^{(t)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij}\, Wh'_j \;+\; (Lh')_i. \tag{6}$$

Here, the weighted sum aggregates neighbor embeddings according to their learned importance, while the Laplacian diffusion term $(Lh')_i$ (Sahbi, 2021) corresponds to the *COMBINE* step from Equation (2) in GCN layers. The graph Laplacian L is defined as $L = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$, where A is the adjacency matrix and D the degree matrix. The aim is to refine h'_i into $h_i^{(t)}$, producing richer node representations that integrate both structural and spectral information.

3.4 MULTI-HOP CROSS-ATTENTION

The objective of our approach is to provide node embeddings with as much relevant information as possible. To this end, the GCN layer is applied in two parallel streams: one executes the layer p times to aggregate information up to p hops, capturing local structures, while the other applies it $q > p$ times to integrate more distant neighbors, see fig. 1 for an illustration with $p = 3, q = 6$. This design avoids excessive dilution of p-hop features while still incorporating global context through q-hop embeddings. The two representations are then fused via cross-attention, allowing local embeddings to selectively integrate broader information. Together, the p- and q-hop embeddings yield balanced representations that capture both local details and global context.

The integration of p-hop and q-hop embeddings is performed via a cross-attention mechanism. Let $\mathbf{H}^p, \mathbf{H}^q \in \mathbb{R}^{N \times D}$ denote the node embeddings computed by the p-hop and q-hop GCN layers as presented in Section 3.3. Linear projections are applied to obtain the query, key, and value vectors:

the query vectors $\mathbf{Q}^p$ are derived from the p-hop embeddings $\mathbf{H}^p$, while the key $\mathbf{K}^q$ and value vectors $\mathbf{V}^q$ are derived from the q-hop embeddings $\mathbf{H}^q$.

The attention mechanism is designed to allow each node's p-hop embedding to selectively incorporate information from the q-hop embeddings. First, the attention weight

$$\alpha_i = \sigma\left(\frac{\mathbf{Q}_i^p \cdot \left(\mathbf{K}_i^q\right)^\top}{\sqrt{D}}\right) \tag{7}$$

of each node i is computed via the sigmoid function $\sigma(\cdot)$. It balances the contribution of global q-hop and local p-hop information, enabling each node to adaptively integrate both contexts. The fused embeddings

$$\mathbf{H}_i^{\text{out}} = \alpha_i \mathbf{V}_i^q + (1 - \alpha_i)\mathbf{H}_i^p \tag{8}$$

are then passed to the classification head for the downstream task of node classification (Liu et al., 2025).

4 Evaluation

4.1 Experimental Settings

We conduct experiments on six real-world datasets. Three are homophilic (Cora, CiteSeer, PubMed) and three are heterophilic (Chameleon, Squirrel, Actor), summarized in Table 1. The homophilic datasets are citation networks, while Chameleon and Squirrel are Wikipedia networks and Actor is an actor co-occurrence network. We use widely adopted datasets for comparability with prior work, avoiding very small or sparse graphs, since our model is designed for dense graphs emphasizing neighborhood aggregation. We do not apply the Squirrel and Chameleon cleaning procedure suggested in Platonov et al. (2023), since this procedure is less commonly used and in our view and nodes with identical neighbors can legitimately have different features (Yuan et al., 2023; Sen et al., 2008; Pei et al., 2020).

Table 1: Dataset statistics

Dataset	Nodes	Edges	Features	Classes
Chameleon	2,277	36,051	2,325	5
Squirrel	5,201	216,933	2,089	5
Actor	7,600	29,926	932	5
Cora	2,708	10,556	1,433	7
CiteSeer	3,327	9,104	3,703	6
PubMed	19,717	88,648	500	3

For the experiments presented in Section 4.2, we adopt commonly used variants of data splits. Specifically, for homophilic datasets, we follow the standard protocol of selecting 20 nodes per class for training, and randomly assigning 500 nodes for validation and 1000 nodes for testing (Chien et al., 2021b; Lin, 2024; Kipf & Welling, 2016). This setup aligns with the typical splits employed in the literature. For heterophilic graphs, we use a split of 48% training, 32% validation, and 20% test nodes (Liu et al., 2023b; Yuan et al., 2023), although some works report alternative splits such as 60%/20%/20% (Chien et al., 2021a).

To ensure robust evaluation, we generate 10 random splits per dataset and report performance using the 95% confidence interval (Lin, 2024), noting where other studies report standard deviation instead (Yuan et al., 2023). To assess generalization, we perform additional experiments on similarly dense but smaller graphs (2.5% train, 2.5% validation, 95% test nodes) for heterophilic datasets (Chien et al., 2021a; Lin, 2024); this is not done for homophilic datasets due to the limited training nodes per class.

For all splits, a new model, trainer, and optimizer are initialized. We employ GRIP with input dimension equal to the number of node features, output dimension equal to the number of classes, hidden dimension 64, three p-hop layers, and six q-hop layers (Section 3.4). A dropout rate of 0.1 is applied, with Adam optimizer (learning rate 0.01, weight decay 5×10^{-4}), and the loss function is cross-entropy loss for classification. For spectral information, we select at most $k = 32$ Laplacian eigenvectors to provide a compact spectral representation.

4.2 Edge Augmentation via Graph Structure Learning

Since GRIP shows strong performance on edge-centric tasks, we extend it to GRIPedge. GRIP is first trained until convergence, after which we adopt the teacher–student framework of Wu et al. (2022). Here, the student continues training on an augmented graph, learning jointly from the teacher's predictions and its own objective to improve generalization. Initialized with GRIP weights, the student produces embeddings H^{out} from Eq. 11. We denote $Z = H^{\text{out}}$, which are then passed to a multi-view edge scorer. Given embeddings $Z \in \mathbb{R}^{N\times d}$, edge scores combine similarity and difference terms:

$$S = \sigma\big(ZW_1Z^\top \; + \; \langle (Z_i - Z_j)W_2, (Z_i - Z_j)\rangle\big)\,, \tag{9}$$

with learnable $W_1, W_2 \in \mathbb{R}^{d\times d}$ and sigmoid $\sigma(\cdot)$. Edges are dropped from the original Graph G with probability p_{drop}, while new ones are sampled from the top-k candidates of S with probability p_{add}. The final adjacency is a convex combination:

$$A^{(t)} = \alpha_t\, G + (1-\alpha_t)\,\tilde{S}, \tag{10}$$

where α_t decreases linearly from α_{start} to α_{end}. Early epochs thus rely primarily on G, while later epochs progressively incorporate more augmented edges. The student minimizes a cross-entropy classification loss with additional distillation and regularization:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda_t T^2\,\mathrm{KL}\big(\mathrm{softmax}\big(\tfrac{z_s}{T}\big)\;\|\;\mathrm{softmax}\big(\tfrac{z_t}{T}\big)\big) + \lambda_{\text{reg}}\,\|A^{(t)}\|_F^2, \tag{11}$$

where z_s, z_t are student and teacher logits, T is the distillation temperature, and λ_t balances the distillation contribution. Here, $\|A^{(t)}\|_F^2$ denotes the Frobenius norm of the adjacency, encouraging sparsity, and λ_t is linearly annealed from $\lambda_{\text{kd,start}}$ (strong teacher guidance at early epochs) to $\lambda_{\text{kd,end}}$ (greater reliance on the student at later epochs).

We adopt hidden dimension 64, with edge dropout $p_{\text{drop}} = 0.1$, learning rate 0.002 and allow up to $k = 8$ candidate edges to be added with probability $p_{\text{add}} = 0.5$. Graph mixing gradually shifts from $\alpha_{\text{start}} = 0.9$ to $\alpha_{\text{end}} = 0.5$, while the distillation weight decreases from $\lambda_{\text{kd,start}} = 1.0$ to $\lambda_{\text{kd,end}} = 0.2$ with $T = 2.0$. Finally, the regularization term $\lambda_{\text{reg}} = 0.1$ promotes sparsity in the learned adjacency. GRIPedge follows the experimental setup described in Section 4.1, but for each split we report the maximum accuracy obtained from either the teacher (GRIP) or the student, as the teacher occasionally achieves better performance.

4.3 Experimental Results

Our model is evaluated and compared with a range of baseline methods, whose results are taken from five studies examining semi-supervised node classification accuracy (%) on heterophilic and homophilic datasets (Chien et al., 2021a; Zhu, 2020; Yuan et al., 2023; Lin, 2024; Liu et al., 2025). From these five papers, we select a subset of models, focusing primarily on those that perform well on heterophilic graphs and ensuring that there are methods in each of the following groups. Feature-based methods (Group 1), including GAT (Veličković et al., 2017), MLP, WRGAT (Suresh et al., 2021), H2GCN (Zhu, 2020), GraphSAGE (Hamilton et al., 2017), and MixHop (Abu-El-Haija et al., 2019), rely on explicit node features without directly leveraging the graph structure. Contrastive learning approaches (Group 2) such as MVGRL (Hassani & Khasahmadi, 2020), NWR-GAE (Tang et al., 2022), GREET (Liu et al., 2023b), and MUSE (Yuan et al., 2023) employ self-supervised learning to obtain robust node embeddings by maximizing similarities and differences between node pairs. Graph structure learning methods (Group 3), for example ResNet+adj, FAGCN (Bo et al., 2021), GEOM-GCN (Pei et al., 2020), HDP (Zheng et al., 2025), and AFMF (Liu et al., 2025), aim to learn or refine the graph structure itself to improve node representations and classification performance. Finally, spectral methods (Group 4), including Cheby+JK (Zhu, 2020), GCN+JK (Zhu, 2020), GCN (Kipf & Welling, 2016), FSGNN (Maurya et al., 2022), GloGNN (Li et al., 2022), and att-Node-level NLSFs (Lin, 2024), analyze the graph in the frequency domain to efficiently aggregate and transform node features. This grouping highlights the primary strategy employed by each method within the context of graph representation learning.

From table 2, feature-based methods generally perform best on homophilic graphs, while spectral approaches achieve higher accuracy on heterophilic graphs. Many methods show dataset-specific strengths, though exceptions exist, such as NLSF (Lin, 2024), which performs consistently well across all datasets.

Table 2: Node classification mean accuracy (%) on heterophilic and homophilic datasets; * = standard deviation, otherwise 95% confidence interval. See text for the grouping of the methods. Our method outperforms all other methods on two of the three heterohilic datasets.

		heterophilic			homophilic		
Method	**Group**	**Chameleon**	**Squirrel**	**Actor**	**Cora**	**Citeseer**	**PubMed**
GAT	1	56.38±2.2*	32.09±3.3*	28.06±1.5*	81.9±0.9*	70.7±1.1*	80.1±0.6*
MLP	1	46.91±2.2*	29.28±1.3*	35.66±0.9*	56.1±0.3*	56.9±0.4*	71.4±0.1*
WRGAT	1	65.24±0.9*	48.85±0.8*	36.53±0.8*	**88.2±2.3***	76.8±1.9*	88.5±0.9*
H2GCN	1	59.39±2.0*	37.90±2.0*	35.86±1.0*	87.8±1.4*	**77.1±1.6***	**89.6±0.3***
GraphSAGE	1	58.73±1.7*	41.61±0.7*	34.23±1.0*	86.9±1.0*	76.0±1.3*	88.5±0.5*
MixHop	1	60.50±2.5*	43.80±1.5*	32.22±2.3*	No data / split	No data / split	No data / split
MVGRL	2	51.07±2.7*	35.47±1.3*	30.02±0.7*	83.0±0.3*	72.8±0.5*	79.6±0.4*
NWR-GAE	2	72.04±2.6*	64.81±1.8*	30.17±0.2*	83.6±1.6*	71.5±2.4*	83.4±0.9*
GREET	2	63.64±1.3*	42.29±1.4*	36.55±1.0*	83.8±0.9*	73.1±0.8*	80.3±1.0*
MUSE	2	72.37±2.2*	54.19±3.0*	**38.55±1.3***	82.2±0.4*	71.1±0.4*	82.9±0.6*
ResNet+adj	3	71.1±2.2	65.5±1.6	No data / split	No data / split	No data / split	No data / split
FAGCN	3	64.2±2.0	47.6±1.9	No data / split	No data / split	No data / split	No data / split
GEOM-GCN	3	60.9	38.1	31.6	20.4±1.1	20.3±0.9	58.2±1.2
HDP	3	71.6±2.5*	62.1±1.6*	37.3±0.7*	No data / split	No data / split	No data / split
AFMF	3	78.2±1.2*	72.06±1.6*	35.4±0.7*	No data / split	No data / split	No data / split
Cheby+JK	4	63.8±2.3*	45.0±1.7*	35.1±1.4*	No data / split	No data / split	No data / split
GCN+JK	4	63.4±2.0*	40.5±1.6*	34.2±0.9*	No data / split	No data / split	No data / split
GCN	4	59.63±2.3*	36.28±1.5*	30.83±0.8*	81.9±0.9	70.7±1.1	80.1±0.6
FSGNN	4	77.9±0.5	68.9±1.7	No data / split	No data / split	No data / split	No data / split
GloGNN	4	70.0±2.1	61.2±2.0	No data / split	No data / split	No data / split	No data / split
NLSFs	4	79.8±1.2	68.2±1.9	No data / split	85.4±1.8	75.4±0.8	82.2±1.2
GRIP	1	79.6±1.1	73.2±0.6	29.1±0.6	75.2±1.4	63.1±1.4	68.4±3.4
GRIPedge	1	**79.9±1.7**	**73.7±1.5**	29.2±0.5	75.6±1.2	63.4±1.1	68.8±2.4

The GRIPedge model achieves higher performance than state-of-the-art methods on dense heterophilic datasets. Its performance, however, decreases rapidly on graphs with less neighborhood information. Notably, GRIPedge performs worse on the on more sparse datasets. This behavior can be attributed to the GRIP architecture: its attention mechanisms are most effective on datasets with rich neighborhood information. In contrast, other approaches, such as NLSF, which focus more on mathematically optimized usage of the GCN layers themselves, generally perform better on smaller datasets and sparser graphs.

Table 3: Node classification accuracy (%); values show mean ± 95% confidence interval (CI).

Model	**Group**	**Chameleon**	**Squirrel**	**Actor**
GAT	1	42.19±1.3	28.21±0.9	29.46±0.9
GraphSAGE	1	41.92±0.7	27.64±2.1	30.85±1.8
ChebNetII	4	46.37±3.1	34.40±1.1	33.48±1.2
JacobiConv	4	49.66±1.9	33.65±0.8	34.61±0.7
Specformer	4	49.79±1.2	38.24±0.9	34.12±0.6
OptBasisGNN	4	47.12±2.4	37.66±1.1	34.84±1.3
NLSFs	4	**50.58±1.3**	**38.39±0.9**	**35.13±1.0**
GRIP	1	45.18±1.9	34.45±0.8	24.66±0.5
GRIPedge	1	45.49±2.1	35.39±0.7	24.76±0.5

4.4 Generalization and Robustness under Graph Sparsification

In many real-world scenarios, labeled data is limited. We evaluate GRIPedge on a strongly reduced dataset using the 2.5/2.5/95 split from Section 4.1, with baseline results taken from Lin (2024) and Table 2 shows that spectral methods generally perform best on heterophilic datasets. While GRIPedge does not top the rankings, it remains relatively stable on denser heterophilic graphs like Squirrel and Chameleon, demonstrating robustness even with very few labels, see table 3.

This is also illustrated in fig. 2, where, based on the 48/32/20 split, we halve the number of edges in 6 random splits and reduce the number of nodes by 25% in 10 random splits. In both scenarios, $\text{GRIP}_{\text{edge}}$ shows robustness on the Squirrel and Chameleon datasets. For the Actor dataset, there are few edges, providing neighborhood information and the existing connections do not adequately reflect the graph structure, which impairs classification performance. Notably, Actor even

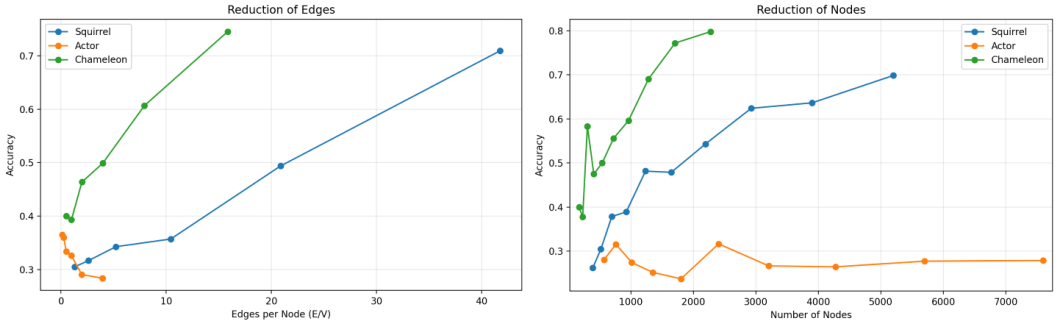

Figure 2: Robustness under Node and Edge Sparsification

benefits from having fewer edges, as removing uninformative connections can improve node classification (Ye & Ji, 2019; Kohn et al., 2024). This highlights that GRIPedge is most effective on dense, informative graphs, while performance drops when the graph lacks sufficient structural information.

4.5 Ablation Studies

In the following, we aim to evaluate the individual mechanisms of the GRIP model and investigate their impact on its overall performance. As discussed in the previous sections, the two main mechanisms are the use of attention mechanisms and the extension of neighborhood information through spectral modulation combined with the p-/q-hop layer architecture. Within the framework of the ablation studies, we remove three components individually from the model under otherwise identical conditions: **GRIP_linear** replaces the 1-hop neighborhood attention before applying the GCN layers with a simple linear projection, **GRIP_p_hop** removes the use of cross-attention with the q-hop layer output so that only the p-hop neighbor information is used, and **GRIP_feature** omits the spectral modulation of the feature-based edge relations attention.

Results show that all three mechanisms improve performance by 1-5%, with the strongest gains from self-attention; for dense graphs like Squirrel and Chameleon, attention enhances neighborhood aggregation, whereas for Actor, edges contribute little to informative neighborhood features, and even without attention slightly better results are obtained (cf. Section 4.4). This analysis is conducted on the three heterophilic datasets using the 48/32/20 splits described in Section 4.1, see table 4.

Table 4: Accuracy on GRIP ablation (mean ± 95% CI)

Model	Chameleon	Squirrel	Actor
GRIP_linear	76.7±1.6	67.3±1.3	**31.6±0.6**
GRIP_p_hop	77.2±0.9	69.9±0.8	28.1±0.5
GRIP_feature	78.5±0.7	72.4±0.7	29.0±0.5
GRIP	**79.6±1.1**	**73.2±0.6**	29.1±0.6

5 Conclusion

We propose a unified framework for node classification in heterophilic graphs that integrates multiple complementary strategies. Combining attention mechanisms with classical GCN layers, our model captures both local interactions and global neighborhood dependencies. Neighborhood aggregation is enhanced via spectral oscillation modulation, retaining informative high-frequency signals while filtering noise. To further strengthen graph representations, we adopt an edge augmentation technique inspired by KDGA, preserving critical heterophilic links and improving structural robustness. Experiments on dense heterophilic benchmarks such as *Squirrel* and *Chameleon* show that our approach consistently outperforms state-of-the-art methods, producing more discriminative and robust node embeddings. As a future direction, mathematically optimized spectral methods like att-Node-level NLSFs, which leverage symmetries to reduce complexity and enhance generalization, could serve as a core component. Extending these with task-specific mechanisms, such as feature-based attention, may further improve performance on challenging dense heterophilic graphs and advance broadly effective graph learning models.

REFERENCES

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International Conference on Machine Learning (ICML)*, 2019.

Deyu Bo, Xiao Wang, Chuan Shi, and Emiao Zhu. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Deyu Bo, Chuan Shi, Lele Wang, and Renjie Liao. Specformer: Spectral graph neural networks meet transformers. *arXiv preprint arXiv:2303.01028*, 2023. URL https://arxiv.org/abs/2303.01028.

Jun Chen and Haopeng Chen. Edge-featured graph attention network. *arXiv preprint arXiv:2101.07671*, 2021. URL https://arxiv.org/abs/2101.07671.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://arxiv.org/abs/2006.13009.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. *arXiv preprint arXiv:2006.07988*, 2020. URL https://arxiv.org/abs/2006.07988.

Eli Chien, Hongwei Pei, Yujun Xu, Jie Zhuang, Hanqing Wang, Lenka Cowen, Shuiwang Ji, and Pan Li. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations (ICLR)*, 2021a.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations (ICLR)*, 2021b. URL https://openreview.net/forum?id=3fd51494885a4f0252dd144ae51025065fef2186.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning (ICML)*, 2020.

Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. URL https://arxiv.org/abs/1609.02907.

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*, 2019.

Matthias Kohn, Marcel Hoffmann, and Ansgar Scherp. Edge-splitting mlp: Node classification on homophilic and heterophilic graphs without message passing. In *Proceedings of the Third Learning on Graphs Conference*, pp. 11:1–11:21, 2024. URL https://proceedings.mlr.press/v269/kohn25a.html.

Moritz Lampert and Ingo Scholtes. The self-loop paradox: Investigating the impact of self-loops on graph neural networks. *arXiv preprint arXiv:2312.01721*, 2023. URL https://arxiv.org/abs/2312.01721.

Jintang Li and et al. Oversmoothing: A nightmare for graph contrastive learning? In *International Conference on Learning Representations (ICLR)*, 2023.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

X. Li, R. Zhu, Y. Cheng, C. Shan, S. Luo, D. Li, and W. Qian. Finding global homophily in graph neural networks when meeting heterophily. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 13242–13256. PMLR, 2022. URL https://proceedings.mlr.press/v162/li22ad.html.

Derek Lim, Felix Hohne, Xiang Xu, Rose Yu Lim, Christopher De Sa, and Jure Leskovec. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

YWE Lin. Equivariant machine learning on graphs with nonlinear spectral filters. *arXiv preprint arXiv:2406.01249*, 2024. URL https://arxiv.org/abs/2406.01249.

Jie Liu, Renxiang Guan, Zihao Li, Jiaxuan Zhang, Yaowen Hu, and Xueyong Wang. Adaptive multi-feature fusion graph convolutional network for hyperspectral image classification. *Remote Sensing*, 15(23):5483, 2023a. doi: 10.3390/rs15235483.

Kang Liu, Xinyu Li, Lian Liu, Zhihao Xv, Aohang Pei, and Runshi Ji. Afmf: Adaptive fusion of multi-hop neighborhood features in graph convolutional network. *Journal of King Saud University - Computer and Information Sciences*, 37(7):191–200, 2025. doi: 10.1007/s44443-025-00222-z. URL https://link.springer.com/article/10.1007/s44443-025-00222-z.

Nian Liu, Xiao Wang, Hui Han, and Chuan Shi. Self-supervised heterogeneous graph neural network with co-contrastive learning. *arXiv preprint arXiv:2105.09111*, 2021.

Yixin Liu, Yizhen Zheng, Daokun Zhang, Vincent CS Lee, and Shirui Pan. Beyond smoothing: Unsupervised graph representation learning with edge heterophily discriminating. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37(4), pp. 25573–25581, 2023b. URL https://doi.org/10.1609/aaai.v37i4.25573.

S. K. Maurya, X. Liu, and T. Murata. Simplifying approach to node classification in graph neural networks. *Journal of Computational Science*, 62:101695, 2022. doi: 10.1016/j.jocs.2022.101695. URL https://doi.org/10.1016/j.jocs.2022.101695.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2020.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tJbbQfw-5wv.

Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *Knowledge Discovery and Data Mining (KDD)*, 2020.

Lingfei Ren, Ruimin Hu, Zheng Wang, Yilin Xiao, Dengshi Li, Junhang Wu, Jinzhang Hu, Yilong Zang, and Zijun Huang. Heterophilic graph invariant learning for out-of-distribution of fraud detection. In *Proceedings of the 32nd ACM International Conference on Multimedia (MM '24)*, pp. —, 2024. doi: 10.1145/3664647.3681312. URL https://dl.acm.org/doi/10.1145/3664647.3681312.

Hichem Sahbi. Learning laplacians in chebyshev graph convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 2064–2075. IEEE, 2021. doi: 10.1109/ICCVW54120.2021.00247.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–93, 2008.

Susheel Suresh, Vinith Budde, Jennifer Neville, Pan Li, and Jianzhu Ma. Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1541–1551, 2021. doi: 10.1145/3447548.3467373. URL https://arxiv.org/abs/2106.06586.

Mingyue Tang, Carl Yang, and Pan Li. Graph auto-encoder via neighborhood wasserstein reconstruction. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=ATUh28lnSuW.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations (ICLR)*, 2019.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Lirong Wu, Haitao Lin, Yufei Huang, and Stan Z. Li. Knowledge distillation improves graph structure augmentation for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. URL https://papers.neurips.cc/paper_files/paper/2022/file/4d4a3b6a34332d80349137bcc98164a5-Paper-Conference.pdf.

Yuxia Wu and et al. Exploring adaptive structure learning for heterophilic graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Xuanting Xie, Zhao Kang, and Wenyu Chen. Robust graph structure learning under heterophily. *arXiv preprint arXiv:2403.03659*, 2024.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80, pp. 5453–5462. PMLR, 2018. URL https://proceedings.mlr.press/v80/xu18c/xu18c.pdf.

Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Ling-bin Yang, Yifan Zhang, Ke Chen, Chen Gong, Ya Zhang, and Zhi-Hua Zhou. Gophormer: Ego-graph transformer for node classification. arXiv preprint arXiv:2110.13094, 2021.

Wenhan Yang and Baharan Mirzasoleiman. Graph contrastive learning under heterophily via graph filters (hlcl). In *Uncertainty in Artificial Intelligence (UAI)*, 2024. URL https://openreview.net/forum?id=khvJM3uFk8. Accepted as UAI 2024 poster / conference paper.

Yang Ye and Shihao Ji. Sparse graph attention networks. *arXiv preprint arXiv:1912.00552*, 2019. URL https://arxiv.org/abs/1912.00552.

Chenyu Ying, Jiaxuan You, Christopher Morris, Xieyuanli Ouyang, Ferran Alet, Petra Prna, and Jure Leskovec. Do transformers really perform badly on graphs? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Mengyi Yuan, Minjie Chen, and Xiang Li. Muse: Multi-view contrastive learning for heterophilic graphs. *arXiv preprint arXiv:2307.16026*, 2023. URL https://arxiv.org/abs/2307.16026. Zugriff am 17. September 2025.

Wentao Zhang, Zhen Zhang, Shiyu Chang, and Jiawei Han. Ptdnet: A parameterized topological denoising network. *arXiv preprint arXiv:2011.07057*, 2020. URL https://arxiv.org/abs/2011.07057.

Zhe Zheng, Soumya Pal, Mark Coates, and Michael Rabbat. Self-supervision improves structure learning for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. URL https://arxiv.org/abs/2102.05034.

Zhuonan Zheng, Sheng Zhou, Hongjia Xu, Ming Gu, Yilun Xu, Ao Li, Yuhong Li, Jingjun Gu, and Jiajun Bu. Heterophilous distribution propagation for graph neural networks. *Neural Networks*, 184:107014, 2025. doi: 10.1016/j.neunet.2024.107014. URL https://doi.org/10.1016/j.neunet.2024.107014.

Author Zhu. Beyond homophily in graph neural networks: Current limitations and effective designs. *arXiv preprint arXiv:2006.07988*, 2020.