Beyond Fixed Depth: Adaptive Graph Neural Networks for Node Classification Under Varying Homophily

Asela Hevapathige[1], **Asiri Wijesinghe**[2], **Ahad N. Zehmakan**[1]

[1]School of Computing, Australian National University, Canberra, Australia
[2]Data61, CSIRO, Canberra, Australia
asela.hevapathige@anu.edu.au, asiriwijesinghe.wijesinghe@data61.csiro.au, ahadn.zehmakan@anu.edu.au

Abstract

Graph Neural Networks (GNNs) have achieved significant success in addressing node classification tasks. However, the effectiveness of traditional GNNs degrades on heterophilic graphs, where connected nodes often belong to different labels or properties. While recent work has introduced mechanisms to improve GNN performance under heterophily, certain key limitations still exist. Most existing models apply a fixed aggregation depth across all nodes, overlooking the fact that nodes may require different propagation depths based on their local homophily levels and neighborhood structures. Moreover, many methods are tailored to either homophilic or heterophilic settings, lacking the flexibility to generalize across both regimes. To address these challenges, we develop a theoretical framework that links local structural and label characteristics to information propagation dynamics at the node level. Our analysis shows that optimal aggregation depth varies across nodes and is critical for preserving class-discriminative information. Guided by this insight, we propose a novel adaptive-depth GNN architecture that dynamically selects node-specific aggregation depths using theoretically grounded metrics. Our method seamlessly adapts to both homophilic and heterophilic patterns within a unified model. Extensive experiments demonstrate that our approach consistently enhances the performance of standard GNN backbones across diverse benchmarks.

Introduction

Graph Neural Networks (GNNs) have achieved remarkable success in various graph-based downstream tasks across diverse domains, including social network analysis (Li et al. 2023; Benslimane et al. 2023), bioinformatics (Zhang et al. 2021; Yi et al. 2022), and anomaly detection (Kim et al. 2022; Bei et al. 2025). Particularly, GNNs have demonstrated strong performance in node classification tasks due to their ability to integrate structural characteristics and node feature information into meaningful representations (Sun et al. 2024; Khoshraftar and An 2024).

However, many mainstream GNNs are built upon the homophily assumption, where connected nodes tend to share similar labels or characteristics (McPherson, Smith-Lovin, and Cook 2001; Zhu et al. 2020). While this assumption has proven effective for many real-world networks, such as citation networks and knowledge graphs (Yang, Cohen, and Salakhudinov 2016), it significantly limits GNNs' performance on heterophilic graphs, where connected nodes often have different labels or properties (Zheng et al. 2023a). Such heterophilic patterns are common in many real-world scenarios, including social networks and web graphs (Pei et al. 2020; Zhu et al. 2021; Platonov et al. 2023).

To address this limitation, researchers have begun exploring specialized heterophily-aware GNN architectures that explicitly account for the dissimilarity between connected nodes and develop alternative message-passing strategies (Zhu et al. 2021; Zheng et al. 2022; Luan et al. 2022; Zhu et al. 2023; Wang et al. 2024a). Nonetheless, certain limitations still exist. First, categorizing graphs as homophilic or heterophilic based on simple neighborhood label ratios oversimplifies the complex nature of information propagation in graphs. Real-world graphs are intricate structural entities with different regions or nodes exhibiting varying levels of homophily. Different nodes have varying sensitivity to information aggregation, depending on their structural context, features, and neighbor label composition. A more nuanced understanding of this phenomenon is required to design GNN architectures that accommodate these differences. Second, methods specifically designed for heterophilic graphs often lack the flexibility to effectively handle both homophilic and heterophilic contexts within the same architecture, limiting their applicability and generalizability across diverse graph types.

We aim to address these limitations through a fundamental observation: optimal information propagation strategies are inherently different across nodes within the same graph. The uniform nature of information aggregation in current GNNs (i.e., fixed layer depth applied to all nodes) limits its node classification performance, as different nodes have varying information aggregation needs based on their local neighbourhood properties, including degree distribution, label homophily ratio, and feature similarity patterns within their close neighbourhoods. This motivates us to theoretically connect structural and label characteristics to information propagation dynamics at the node level, gaining a more nuanced understanding of node-specific aggregation needs beyond global graph homophily characteristics.

Building on these theoretical insights, we derive node-

specific metrics that quantify individual node aggregation requirements without being constrained by global graph homophily measures. We then utilize these insights to develop dynamic GNN architectures that adaptively determine aggregation depth for each node, improving their ability to handle node-specific information propagation needs. The main contributions of our work are summarized as follows:

- **Theoretical Foundation**: We establish theoretical connections between structural and label characteristics and information propagation dynamics, deriving node-specific metrics that quantify optimal aggregation depth requirements in GNNs.
- **Novel Adaptive Depth Architecture**: We propose a novel GNN architecture that dynamically adjusts aggregation depth for individual nodes based on their specific information propagation needs, with two distinct variants to accommodate different computational requirements.
- **Unified Homophilic/Heterophilic Handling**: Our architecture provides a single framework that seamlessly accommodates both homophilic and heterophilic graph patterns without requiring separate architectures or preprocessing steps.
- **Empirical Validation**: We provide comprehensive experiments demonstrating that our adaptive depth architecture consistently improves node classification performance of mainstream GNN backbones across diverse graph structures and homophily levels.

Related Work

Heterophily-Aware GNNs While traditional GNNs assume homophily, such an inductive bias is believed to result in performance degradation in less homophilic graphs (Lim et al. 2021). Geom-GCN (Pei et al. 2020) is one of the early works that identified this performance bottleneck. Since then, there has been a plethora of works targeting enhancing GNN performance in heterophilic graphs. Some works propose filtering neighbours to improve aggregation performance on heterophilic graphs by excluding or deprioritising dissimilar nodes (Bo et al. 2021; Luan et al. 2022; Zheng et al. 2023b; Guo et al. 2024). Furthermore, some works have shown that incorporating higher-order neighbourhoods could alleviate this issue, as it enables the aggregation process to access more homophilic information (Zhu et al. 2020; Wang et al. 2021, 2022; Li et al. 2022; Yu et al. 2024; Li, Pan, and Kang 2024). There have been some works introducing structural constraints such as neighbour ordering (Song et al. 2023) and selective aggregation mechanisms (He et al. 2022; Haruta, Konishi, and Kurokawa 2023; Wang et al. 2024c) to enable more discriminative node representations by controlling how information flows between nodes of different classes. While the aforementioned works have focused on improving GNN architecture, some studies have taken an alternative path by rewiring the input graph to increase homophily (Guo et al. 2023; Li, Kim, and Wang 2023; Bi et al. 2024; Bose, Banerjee, and Das 2025).

Recent theoretical findings have shown that heterophily does not always negatively impact node classification. Ma et al. (2022) first demonstrated this, showing that moderate levels of heterophily are more detrimental to GNNs than conditions of extreme heterophily. Further, Wang et al. (2024a) elaborated that class separability in node classification using GNNs depends on both neighborhood distribution distances and node degrees, establishing that the impact of heterophily is nuanced rather than uniformly negative.

Our work is fundamentally different from the above. We present a granular node-level theoretical framework that decomposes GNN aggregation based on neighbourhood label composition, demonstrating how the classification quality of individual nodes scales across multiple layers. Our work shows that nodes with varying neighborhood compositions benefit from different propagation depths, resulting in an adaptive architecture.

Depth-Adaptive GNNs There have been few works exploring the impact of adaptive depth in GNNs. Wu et al. (2024) employed reinforcement learning to design a flexible GNN architecture that adaptively searches for optimal parameters of components, including GNN depth, aggregation functions, and pooling operations for graph classification. ADMP-GNN (Abbahaddou et al. 2025) introduced a depth allocation policy for GNNs using centrality heuristics to cluster structurally similar nodes and assign optimal layer depth based on their validation performance. Recently, Hevapathige, Zehmakan, and Wang (2025) integrated learnable Bakry-Émery curvature to determine node-specific aggregation depth, aiming to enhance feature distinctiveness.

In contrast to these works, our method is specifically designed to address heterophily challenges in node classification by analysing how neighbourhood label composition affects the propagation depth.

Theoretical Analysis

We develop a theoretical framework to understand how neighborhood profile impacts node classification performance in GNNs under different homophily conditions, establishing the basis for our adaptive depth allocation strategy. For node v, we define its profile as the tuple (d_v^+, d_v^-, d_v) describing the distribution of same-label neighbours, opposite-label neighbours, and total degree within its neighbourhood. All proofs for the theorems are in the Appendix.

Preliminaries We consider an undirected graph $G = (V, E, \mathbf{X}, \mathbf{y})$, where V represents the set of nodes, E is the set of edges, $\mathbf{X} \in \mathbb{R}^{|V| \times d}$ is the matrix of initial node feature representations with a feature dimension of d, and $\mathbf{y} \in \{0,1\}^{|V|}$ is the vector of node labels. Each node $v \in V$ is associated with a feature vector $\mathbf{x}_v \in \mathbb{R}^d$ (i.e., $\mathbf{x}_v = \mathbf{X}_{v,:}$), and has a label $y_v \in \{0,1\}$ (i.e., $y_v = \mathbf{y}_v$). For theoretical tractability, we focus on the binary classification setting. Let $\mathcal{N}(v)$ be the neighborhood of node v, and $d_v = |\mathcal{N}(v)|$ its degree. We define same-label neighbors as $\mathcal{N}_v^+ = \{u \in \mathcal{N}(v) : y_u = y_v\}$ and opposite-label neighbors as $\mathcal{N}_v^- = \{u \in \mathcal{N}(v) : y_u \neq y_v\}$. Their cardinalities are $d_v^+ = |\mathcal{N}_v^+|$ and $d_v^- = |\mathcal{N}_v^-|$, with $d_v = d_v^+ + d_v^-$.

Class Concepts To establish the assumptions for our analysis, we formally define the concepts of class prototypes, signal variance, and noise variance.

Definition 1 (Class Concepts)**.** *We define the following graph-wide parameters:*

- *Class prototypes:* $\boldsymbol{\mu}^0, \boldsymbol{\mu}^1 \in \mathbb{R}^d$ *where* $\boldsymbol{\mu}^c = \mathbb{E}[\mathbf{x}_u | y_u = c]$ *for* $c \in \{0, 1\}$
- *Signal variance:* $\Delta^2 = \|\boldsymbol{\mu}^0 - \boldsymbol{\mu}^1\|^2$
- *Noise variance:* $\sigma^2_{intra} = Var[\mathbf{x}_u | y_u = c]$ *for any* $c \in \{0, 1\}$

Signal variance assesses the separation between different classes, while noise variance evaluates the variation within a single class. Note that, under the homoscedasticity assumption (Yang, Tu, and Chen 2019), we assume equal noise variance across all classes. Higher signal variance leads to better class separation, while lower noise variance ensures tight clustering within classes, resulting in better classification quality (Fisher 1936).

GNN Architecture We utilize Graph Convolutional Network (GCN) (Kipf and Welling 2017) to examine homophily/heterophily effects, aligning with numerous theoretical analyses in this area (Ma et al. 2022; Wang et al. 2024b). The fundamental node update mechanism of GCN is:

$$\mathbf{X}^{(l+1)} = \sigma(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{X}^{(l)}\mathbf{W}^{(l)})$$

where $\mathbf{W}^{(l)}$ is the learnable parameter matrix at layer l, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix with added self-loops, $\tilde{\mathbf{D}}$ is the degree matrix corresponding to $\tilde{\mathbf{A}}$, and $\mathbf{X}^{(0)} = \mathbf{X}$ are the initial node features. For analytical tractability, we analyze a simplified aggregation scheme that captures the essential neighborhood averaging behavior. Following (Wu et al. 2019), which shows that most of the benefit in GCNs comes from local averaging rather than from nonlinear activation functions, we focus on the uniform averaging operation obtained through row normalization $\tilde{\mathbf{D}}^{-1}\tilde{\mathbf{A}}$.

$$\mathbf{h}_v = \frac{1}{d_v + 1}\left(\mathbf{x}_v + \sum_{u \in \mathcal{N}(v)} \mathbf{x}_u\right)$$

Assumptions We consider a contextual stochastic block model (CSBM) (Deshpande et al. 2018) where nodes are partitioned into two classes. Each node's feature follows a class-specific Gaussian distribution: $\mathbf{x}_v = \boldsymbol{\mu}^{y_v} + \boldsymbol{\epsilon}_v$ with $\boldsymbol{\epsilon}_v \sim \mathcal{N}(\mathbf{0}, \sigma^2_{\text{intra}}\mathbf{I}_d)$, where $\mathbf{I}_d$ is the $d \times d$ identity matrix. Under this model, for any node v, same-label neighbors $u \in \mathcal{N}_v^+$ have features $\mathbf{x}_u = \boldsymbol{\mu}^{y_v} + \boldsymbol{\epsilon}_u^+$, while opposite-label neighbors $u \in \mathcal{N}_v^-$ have features $\mathbf{x}_u = \boldsymbol{\mu}^{1-y_v} + \boldsymbol{\epsilon}_u^-$, where $1 - y_v$ denotes the opposite class label in the binary setting. All noise terms $\boldsymbol{\epsilon}_v, \{\boldsymbol{\epsilon}_u^+\}_{u \in \mathcal{N}_v^+}, \{\boldsymbol{\epsilon}_u^-\}_{u \in \mathcal{N}_v^-}$ are independent and follow $\mathcal{N}(\mathbf{0}, \sigma^2_{\text{intra}}\mathbf{I}_d)$. This CSBM setup is consistent with established theoretical frameworks for analyzing heterophily in GNNs (Ma et al. 2022).

An in-depth analysis of the real-world implications of these assumptions and justifications on why our theoretical insights would hold for a multi-class setting is provided in the Appendix.

Aggregation Effect Analysis

In this section, we analyse how the neighbourhood aggregation impacts the node classification quality under varying homophily/heterophily conditions. To analyze this effect, we decompose GNN aggregation based on neighbourhood labels.

Definition 2 (Label-Based Aggregation)**.** *We decompose the GNN aggregation operation based on neighbor labels as:*

$$\mathbf{h}_v = \frac{1}{d_v + 1}\left(\mathbf{x}_v + \sum_{u \in \mathcal{N}_v^+} \mathbf{x}_u + \sum_{u \in \mathcal{N}_v^-} \mathbf{x}_u\right)$$

To quantify how neighbourhood aggregation affects the original class signal, we introduce signal preservation factor.

Definition 3 (Signal Preservation Factor)**.** *For node* v*, the signal preservation factor is:*

$$\alpha_v = \frac{1 + d_v^+ - d_v^-}{d_v + 1}$$

We present the following theorem, which characterises how neighbourhood label composition affects representation quality after aggregation.

Theorem 1 (Label Aggregation Effect)**.** *For any node* v *in the graph, under the stated graph-wide assumptions, the aggregated representation has expected value:*

$$\mathbb{E}[\mathbf{h}_v | y_v] = \frac{1 + d_v^+}{d_v + 1}\boldsymbol{\mu}^{y_v} + \frac{d_v^-}{d_v + 1}\boldsymbol{\mu}^{1-y_v}$$

The signal variance after aggregation is:

$$\|\mathbb{E}[\mathbf{h}_v | y_v = 0] - \mathbb{E}[\mathbf{h}_v | y_v = 1]\|^2 = \alpha_v^2\Delta^2$$

The noise variance is: $Var[\mathbf{h}_v | y_v] = \frac{\sigma^2_{intra}}{d_v+1}$
The node-specific classification quality is: $Q_v = \frac{\alpha_v^2(d_v+1)\Delta^2}{\sigma^2_{intra}}$

Employing Theorem 1, we derive the following observations.

Corollary 1 (Strong Homophily)**.** *When* $d_v^+ \gg d_v^-$*, the signal preservation factor* $\alpha_v \approx 1$*, and classification quality scales linearly with degree:* $Q_v \propto d_v$*.*

In strongly homophilic neighborhoods, aggregation generally benefits nodes regardless of their degree, since $\alpha_v \approx 1$ guarantees that the expected aggregated representation preserves class signal. However, higher-degree nodes achieve better performance since increased sample (neighborhood) size decreases the probability of "error".

Corollary 2 (Strong Heterophily)**.** *When* $d_v^- \gg d_v^+$*,* α_v *transitions from 0 at low degrees to -1 at high degrees. Classification quality shows poor performance at low degrees due to signal cancellation (*$Q_v \approx 0$ *when* d_v *is small), but achieves performance comparable to homophilic cases at high degrees (*$Q_v \propto d_v$ *when* d_v *is large).*

Strong heterophily isn't inherently bad. It becomes problematic when there aren't enough neighbours to establish a reliable pattern. With sufficient degree, heterophilic nodes can perform as well as homophilic ones by learning from the opposite relationships.

Corollary 3 (Mixed Homophily/Heterophily)**.** *When $d_v^+ \approx d_v^-$ (balanced same/opposite-label neighbors), $\alpha_v \approx \frac{1}{d_v+1}$, causing signal cancellation that worsens with degree, leading to poor classification performance.*

Nodes with balanced neighbourhoods experience signal cancellation, as competing signals from same-class and opposite-class neighbours neutralize each other. This effect is particularly severe for high-degree nodes, where $\alpha_v \to 0$ as d_v increases, leaving such nodes with insufficient directional information for reliable classification.

Multi-Layer Analysis

We extend Theorem 1 to multi-layer setting to analyze the iterative aggregation effect in GNNs.

Theorem 2 (Iterative Aggregation Effect)**.** *Assume a n-layer GNN where: (1) label-conditioned features remain independent across layers, and (2) each layer performs the same neighborhood aggregation pattern with degree d_v. Then, for node v, after n layers:*

- *Signal variance:* $\alpha_v^{2n}\Delta^2$
- *Noise variance:* $\frac{\sigma_{intra}^2}{(d_v+1)^n}$
- *Classification quality:* $Q_v^n = \frac{\alpha_v^{2n}(d_v+1)^n\Delta^2}{\sigma_{intra}^2}$

Theorem 2 shows how single-layer effects compound over multiple layers. Signal preservation factor α_v is raised to the power $2n$, meaning if $|\alpha_v| < 1$ (signal degradation), the effect becomes exponentially worse with depth. Conversely, noise reduction compounds beneficially, but the net effect depends on whether signal preservation dominates.

Remark 1. *While our main analysis assumes oversimplified behaviour, real GNNs exhibit deviations due to oversmoothing and feature correlation. The extended theoretical analysis in the Appendix addresses these practical effects.*

Adaptive Depth Graph Neural Networks

We introduce AD-GNN, a novel architecture that leverages our theoretical findings to enhance message propagation in GNNs by allocating adaptive depth. The key insight is that different nodes benefit differently from deeper layers based on their local neighbourhood structure and multi-layer aggregation effects.

Let Q_v^n denote the classification quality for node v after n layers of message passing, and Q_v^0 represents the initial feature quality before any message passing. For node v and target depth n, we define the Depth Benefit Metric:

Definition 4 (Depth Benefit Metric)**.** *For node v and target depth n, we define the Depth Benefit Metric:*

$$\varepsilon_v^n = \frac{Q_v^n}{Q_v^0} = \left(\alpha_v^2 \cdot (d_v+1)\right)^n$$

The metric represents the multiplicative improvement in classification quality where $\varepsilon_v^n > 1$ indicates benefit, $\varepsilon_v^n = 1$ indicates no change, and $\varepsilon_v^n < 1$ indicates degradation.

Stopping Depth Assignment

Let $t_{\max}$ be the maximum allowable depth in the network. For each vertex $v \in V$, we define its stopping depth $T(v)$ as:

$$T(v) = \max\left\{t \in \{1,2,\ldots,t_{\max}\} \,\middle|\, \tilde{\varepsilon}_v^{t_{\max}} \geq \tau_{\boldsymbol{\theta}}(t)\right\}$$

where $\tilde{\varepsilon}_v^{t_{\max}}$ is the depth benefit score normalized to $[0,1]$ using min-max scaling across all nodes, and $\tau_{\boldsymbol{\theta}} : \mathbb{N} \to [0,1]$ is a monotonically increasing adaptive threshold function parameterized by $\boldsymbol{\theta}$. The monotonically increasing property ensures progressive filtering where nodes with lower depth benefit scores are gradually excluded as network depth increases, preventing nodes that cease to benefit from message passing at layer t from re-entering at deeper layers $t' > t$. The threshold function is defined as:

$$\tau_{\boldsymbol{\theta}}(t) = \lambda + (1-\lambda)\cdot\theta(t)$$

where $\lambda \in [0,1]$ is a hyperparameter, and $\theta(t) : \mathbb{N} \to [0,1]$ is a learnable monotonically increasing function. The parameter λ establishes a minimum threshold ensuring $\tau_{\boldsymbol{\theta}}(t) \in [\lambda, 1]$, while the learnable component $\theta(t)$ adaptively optimizes the threshold across layers.

Message Passing Framework

For each vertex v and step $t \leq T(v)$, the message passing consists of aggregation and update steps:

$$\mathbf{m}_v^{(t)} = \text{AGG}\left(\{\!\{\mathbf{h}_u^{(\min\{t-1,T(u)\})} \mid u \in \mathcal{N}(v)\}\!\}\right)$$

$$\mathbf{h}_v^{(t)} = \text{UPD}\left(\mathbf{h}_v^{(t-1)}, \mathbf{m}_v^{(t)}\right)$$

For any $t > T(v)$, we have $\mathbf{h}_v^{(t)} = \mathbf{h}_v^{(t-1)}$. The adaptive depth mechanism can enhance existing GNN architectures such as GCN (Kipf and Welling 2017), GAT (Veličković et al. 2018), and GraphSAGE (Hamilton, Ying, and Leskovec 2017) by utilizing their original aggregation and update functions along with node-specific depth allocation based on the depth benefit metric.

Depth Benefit Metric Calculation

Computing the depth benefit metric requires estimating α_v for each node, which is challenging due to incomplete label information during semi-supervised training. We use feature similarity to estimate same-label probabilities between adjacent nodes:

$$p_{uv} = f_{\boldsymbol{\delta}}(\mathbf{h}_u, \mathbf{h}_v)$$

where $f_{\boldsymbol{\delta}} : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$ is a learnable, permutation-invariant function that maps pairs of node embeddings to same-label probabilities. The expected counts for same-label and opposite-label neighbors are:

$$\hat{d}_v^+ = \sum_{u\in\mathcal{N}(v)} f_{\boldsymbol{\delta}}(\mathbf{h}_u, \mathbf{h}_v), \quad \hat{d}_v^- = \sum_{u\in\mathcal{N}(v)} (1 - f_{\boldsymbol{\delta}}(\mathbf{h}_u, \mathbf{h}_v))$$

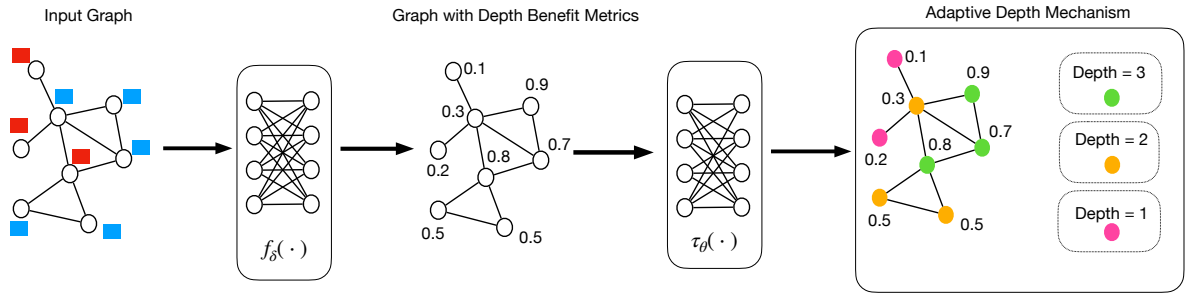


Figure 1: High-level workflow of AD-GNN: First, the depth benefit metric for each node is computed using their node profile and learned label probabilities. Then, the depth for each node is determined based on a learnable threshold mechanism. Nodes that have higher depth benefit metrics will receive more aggregation layers compared to others.

The estimated signal preservation factor and depth benefit metric are:

$$\hat{\alpha}_v = \frac{1 + \hat{d}_v^+ - \hat{d}_v^-}{d_v + 1}, \quad \hat{\varepsilon}_v^{t_{\max}} = \left(\hat{\alpha}_v^2 \cdot (d_v + 1)\right)^{t_{\max}}$$

This creates a differentiable mechanism that enables end-to-end optimization where depth allocation adapts based on learned similarity assessments and predicted depth benefits.

Training Objective

To ensure $f_{\boldsymbol{\delta}}$ learns meaningful similarity assessments, we incorporate regularization using available label information on training nodes:

$$\mathcal{L}_{\text{reg}} = -\frac{1}{|E_{\text{train}}|} \sum_{(u,v)\in E_{\text{train}}} [y_{uv} \log\left(f_{\boldsymbol{\delta}}(\mathbf{h}_u, \mathbf{h}_v)\right) + (1 - y_{uv}) \log\left(1 - f_{\boldsymbol{\delta}}(\mathbf{h}_u, \mathbf{h}_v)\right)]$$

where $E_{\text{train}} = \{(u,v) \in E : u, v \in V_{\text{train}}\}$ is the set of edges between training nodes, and $y_{uv} = \mathbb{I}[y_u = y_v]$ is the true same-label indicator. The total training objective becomes:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \mathcal{L}_{\text{reg}}$$

where $\mathcal{L}_{\text{task}}$ is the downstream task loss. The High-level overview of AD-GNN is depicted in Figure 1.

Fast Variant

To accelerate computation, we derive AD-GNN$_{\text{fast}}$ by replacing the learnable similarity function with a static degree-based approximation. Leveraging high-degree assortativity (Arcagni et al. 2017), the principle that high-degree nodes tend to connect to other high-degree nodes and share similar labels in the network, we compute same-label probability as:

$$p_{uv} = \frac{d_u \times d_v}{\max_{(i,j)\in E}(d_i \times d_j)}$$

The estimated depth benefit metric becomes:

$$\hat{\varepsilon}_v^{t_{\max}} = \left(\left(\hat{\alpha}_{v,\text{degree}}\right)^2 \cdot (d_v + 1)\right)^{t_{\max}}$$

where $\hat{\alpha}_{v,\text{degree}}$ is computed using the degree-based same-label probability estimates. This variant eliminates the computational overhead of learning $f_{\boldsymbol{\delta}}$ and requires no regularization term, limiting the training objective to the downstream task loss alone.

Complexity Analysis

We analyse the complexity of AD-GNN variants separately from the backbone GNN architecture. AD-GNN incurs $\mathcal{O}(|E| \times d)$ complexity for feature-based label probability computation, where d is the feature dimension, $\mathcal{O}(|E| + |V|)$ for depth benefit metric computation, $\mathcal{O}(|E| + |V|)$ per-layer complexity for threshold filtering mechanism, and $\mathcal{O}(|E_{\text{train}}|)$ for regularization term. This takes the total complexity of AD-GNN to $\mathcal{O}(|E| \times d + t_{\max} \times (|E| + |V|))$. AD-GNN$_{\text{fast}}$ achieves a reduced complexity of $\mathcal{O}(t_{\max} \times (|E| + |V|))$ by eliminating feature-based label probability computation using degree-based similarity approximation, which incurs only $\mathcal{O}(|E|)$ complexity. Additionally, the fast variant does not require computing the regularization term, further reducing training overhead. Note that, since AD-GNN progressively filters edges at each layer, it also reduces the backbone GNN complexity by operating on smaller edge sets.

Experiments

Datasets We evaluate our approach for the node classification task using 11 datasets that cover both homophilic and heterophilic settings. Homophilic datasets include Cora ML, Citeseer, Pubmed, and DBLP from the CitationFull benchmark (Bojchevski and Günnemann 2018), as well as the Photo dataset from the Amazon benchmark (Shchur et al. 2018). Heterophilic datasets include Texas, Cornell, Wisconsin, Squirrel, Chameleon, and Film datasets from WebKB benchmark (Pei et al. 2020). Additionally, we employ the ogbn-arxiv dataset from OGB benchmark (Hu et al. 2020) for our scalability analysis.

Additional details, including dataset statistics, model hyperparameters, computational resources, and implementation details, are provided in the Appendix.

Baselines We employ three classical GNNs: GCN (Kipf and Welling 2017), GAT (Veličković et al. 2018), and GraphSAGE (Hamilton, Ying, and Leskovec 2017), along with three modern GNNs: MixHop (Abu-El-Haija et al. 2019), GATv2 (Brody, Alon, and Yahav 2022), and DirGNN (Rossi et al. 2024), as our backbones.

Evaluation Setting We use a 60/20/20 random split strategy for the training/validation/test sets and report the mean

Methods	Cora-ML	Citeseer	Pubmed	Photo	DBLP	Film	Squirrel	Chameleon	Cornell	Wisconsin	Texas
GCN	87.07 ±1.21	76.68 ±1.64	86.74 ±0.47	89.30 ±0.82	83.93 ±0.34	30.26 ±0.79	39.47 ±1.47	40.89 ±4.12	55.14 ±8.46	61.60 ±7.00	60.00 ±6.45
AD-GCN	87.32 ±1.25	79.14 ±1.00	88.39 ±0.32	94.10 ±0.31	84.14 ±0.44	42.54 ±1.15	40.04 ±0.99	43.66 ±0.73	88.51 ±4.87	93.88 ±3.03	92.30 ±4.52
AD-GCN$_{\text{fast}}$	87.34 ±1.35	79.13 ±0.99	88.41 ±0.37	94.10 ±0.31	83.90 ±0.97	41.39 ±1.46	40.88 ±1.09	43.40 ±0.56	87.02 ±3.49	94.38 ±2.86	91.64 ±5.16
GAT	84.12 ±0.55	75.46 ±1.72	87.24 ±0.55	90.81 ±0.22	80.61 ±1.21	26.28 ±1.73	35.62 ±2.06	39.21 ±3.08	53.64 ±11.1	60.00 ±11.0	61.21 ±8.17
AD-GAT	85.02 ±1.64	79.92 ±0.76	87.38 ±0.33	94.03 ±0.34	83.94 ±0.40	41.25 ±0.77	36.73 ±0.83	40.52 ±1.55	86.17 ±4.69	91.50 ±2.42	90.49 ±5.72
AD-GAT$_{\text{fast}}$	84.37 ±1.80	79.95 ±0.78	87.43 ±0.34	93.84 ±0.33	83.45 ±0.43	41.03 ±0.96	36.92 ±0.99	40.36 ±2.54	85.96 ±3.18	92.62 ±2.82	90.82 ±4.65
GraphSAGE	86.52 ±1.32	76.04 ±1.30	88.45 ±0.50	94.23 ±0.62	86.16 ±0.50	34.23 ±0.99	36.09 ±1.99	37.77 ±4.14	75.95 ±5.01	81.18 ±5.56	82.43 ±6.14
AD-GraphSAGE	87.14 ±1.56	79.84 ±1.26	88.99 ±0.64	94.61 ±0.35	85.00 ±1.60	40.92 ±1.46	40.88 ±0.87	40.10 ±1.45	89.57 ±4.30	94.62 ±2.56	92.95 ±2.84
AD-GraphSAGE$_{\text{fast}}$	87.11 ±1.63	79.88 ±1.27	88.88 ±0.60	94.54 ±0.41	85.03 ±1.39	40.90 ±1.49	41.08 ±0.73	39.79 ±1.87	90.64 ±4.28	94.25 ±2.38	92.79 ±2.95
MixHop	87.29 ±1.19	70.75 ±2.95	80.75 ±2.29	94.83 ±0.41	84.27 ±0.31	32.22 ±2.34	38.85 ±0.89	42.94 ±1.01	73.51 ±6.34	75.88 ±4.90	77.84 ±7.73
AD-MixHop	87.66 ±1.24	81.01 ±1.36	90.09 ±0.58	95.09 ±0.57	84.26 ±0.81	43.37 ±1.17	39.76 ±0.90	46.29 ±1.19	90.21 ±3.59	94.75 ±2.22	94.43 ±2.56
AD-MixHop$_{\text{fast}}$	87.73 ±1.31	80.95 ±1.33	90.23 ±0.42	94.55 ±0.57	84.46 ±0.79	43.14 ±1.80	40.27 ±1.05	45.05 ±1.83	90.00 ±3.16	92.00 ±3.41	90.00 ±5.65
GATv2	85.10 ±1.84	76.31 ±1.62	88.77 ±0.39	94.03 ±0.44	84.85 ±0.61	34.90 ±0.79	35.24 ±0.63	42.53 ±1.18	61.35 ±3.21	64.12 ±4.81	67.84 ±4.75
AD-GATv2	85.17 ±1.50	80.10 ±0.99	89.26 ±0.58	94.33 ±0.70	84.26 ±0.52	40.21 ±2.11	37.96 ±0.94	42.99 ±1.25	86.38 ±4.48	91.25 ±1.94	90.16 ±4.69
AD-GATv2$_{\text{fast}}$	85.02 ±1.26	79.11 ±1.48	88.25 ±0.24	94.23 ±0.68	84.26 ±0.52	42.02 ±1.88	40.82 ±0.41	42.63 ±1.03	87.66 ±5.19	92.37 ±2.20	92.46 ±4.47
DirGNN	85.66 ±0.31	77.71 ±0.78	86.94 ±0.55	95.38 ±0.32	81.22 ±0.54	35.76 ±1.68	38.67 ±1.09	42.94 ±1.66	76.51 ±6.14	80.50 ±5.50	76.25 ±6.31
AD-DirGNN	87.25 ±1.42	78.36 ±1.82	89.35 ±0.30	95.55 ±0.51	83.52 ±0.39	41.92 ±1.82	40.58 ±0.94	42.06 ±0.77	91.70 ±2.60	95.13 ±1.89	92.95 ±2.21
AD-DirGNN$_{\text{fast}}$	85.85 ±1.90	78.35 ±1.80	89.07 ±0.29	94.99 ±0.59	82.80 ±0.65	41.23 ±1.75	39.36 ±1.12	41.70 ±1.07	89.57 ±3.86	93.38 ±2.50	92.79 ±3.04

Table 1: Node classification accuracy ± standard deviation (%). The best results are highlighted. Baseline results are sourced from Suresh et al. (2021); Platonov et al. (2023); Zheng et al. (2023b), and Chen et al. (2025).

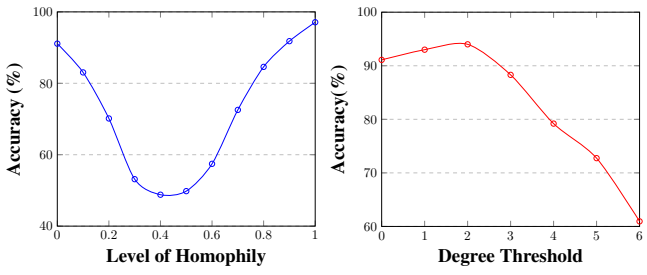
and standard deviation of accuracy over 10 random initialization, similar to the setup in Suresh et al. (2021); Zheng et al. (2023b). For Squirrel and Chameleon datasets, we employ the data splits provided by Platonov et al. (2023), which contain filtered datasets with duplicate nodes removed. We report baseline results from previous papers using the same experimental setup. If unavailable, we generate baseline results based on the hyperparameters from the original papers. When a baseline model is modified using the AD-GNN, the resulting version is named with the prefix "AD-", such as AD-GCN. Also, the AD-GNN fast variant is denoted by an additional subscript $_{\text{fast}}$, such as AD-GCN$_{\text{fast}}$.

Exp–1. Node Classification We present the node classification results in Table 1. Our approach consistently outperforms the baseline methods in both homophilic and heterophilic graph benchmarks. Notably, our adaptive layer mechanism demonstrates greater performance improvements on heterophilic graphs compared to homophilic ones. This is mainly due to heterophilic datasets being more vulnerable to signal degradation than homophilic ones, as demonstrated in the theoretical analysis. Additionally, we observe that our method enhances the performance of GNNs that are designed with inductive biases for heterophilic graph settings, such as MixHop and DirGNN. This shows that our approach complements these models by capturing additional structural and label-dependent information that might otherwise be overlooked. Furthermore, the fast variant of AD-GNN also performs well, achieving results comparable to and sometimes surpassing those of the original AD-GNN, offering a scalable yet powerful solution.

Exp–2. Case Study We conduct experiments to empirically validate the observations presented in Theorem 1. In these experiments, we generate a synthetic graph using a stochastic block model with controllable homophily by explicitly forming a specified proportion of edges between same-class versus different-class nodes. We then measure the GCN performance under two cases.

In Figure 2(a), we evaluate performance across varying homophily degrees, from ideal heterophily to ideal homophily. Performance remains stable under both extremes, confirming Corollary 1 and Corollary 2. However, we observe performance decline in mixed homophily scenarios due to signal cancellation between neighbors with balanced same or opposite-label configurations, consistent with Corollary 3.

Figure 2(b) validates the low-degree scenario highlighted in Corollary 2. In this scenario, we examine strong heterophily while avoiding aggregation of nodes that fall below a certain degree threshold. The results demonstrate that excluding low-degree nodes (specifically, those with a degree of 1-2) from aggregation can improve performance. This improvement is attributed to the complete signal cancellation that occurs in low-degree nodes under strong heterophily conditions. By avoiding these nodes, we can achieve better performance. Conversely, preventing aggregation for high-degree nodes results in a decline in performance.

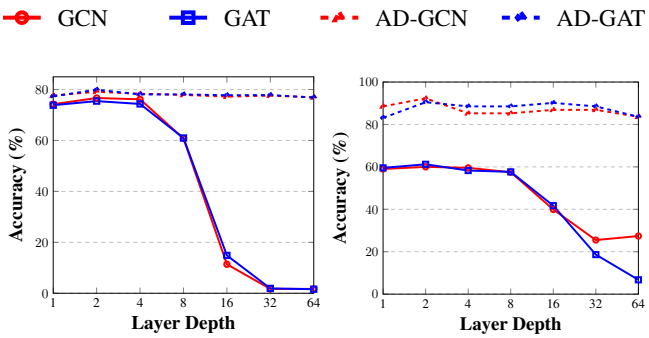


(a) (b)

Figure 2: Case studies to empirically validate observations from Theorem 1.

Exp–3. Oversmoothing Analysis Figure 3 shows AD-GNN performance under varying layer depth. While traditional GNNs exhibit rapid degradation with increased depth,

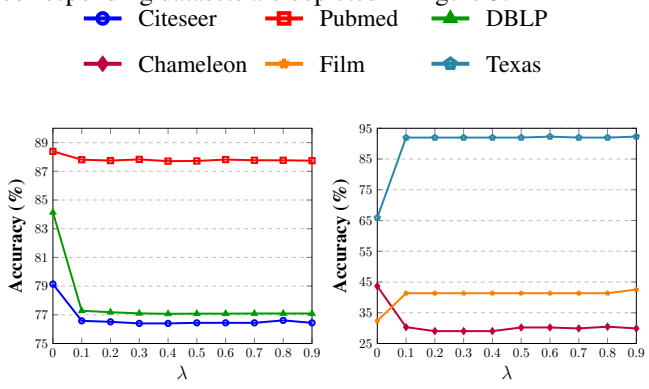
AD-GNN variants maintain consistent performance across deeper layers, effectively mitigating oversmoothing. This robustness can be attributed to our adaptive layer mechanism, which regulates information propagation to prevent excessive signal degradation.



(a) Citeseer (b) Texas

Figure 3: Oversmoothing comparison.

Exp–4. Hyperparameter Sensitivity Analysis We analyse the impact of hyperparameter λ for both homophilic and heterophilic datasets in Figure 4. Degree distributions for corresponding datasets are depicted in Figure 5.

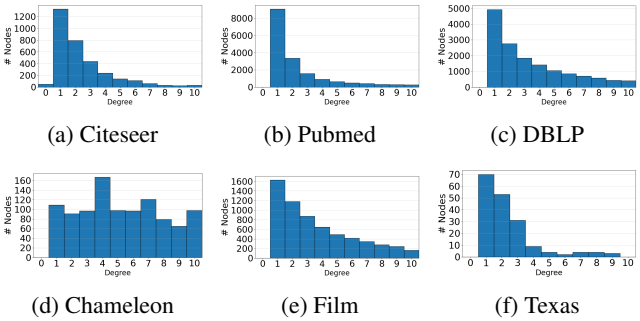


(a) Homophilic Graphs (b) Heterophilic Graphs

Figure 4: Sensitivity analysis of parameter λ on AD-GCN.

For all homophilic graphs (Citeseer, Pubmed, and DBLP), the best results are achieved with $\lambda = 0$. This is primarily because setting $\lambda = 0$ guarantees that every node is aggregated at least once. According to Corollary 1, aggregation under strong homophily is always beneficial, regardless of the degree.

Heterophilic graphs (Chameleon, Film, and Texas) exhibit some interesting deviations. Chameleon performs better when nodes have at least one aggregation layer (i.e., when $\lambda = 0$). This is mainly because most nodes in the graph have moderate to high degrees, which makes aggregation beneficial (as noted in Corollary 2). In contrast, the majority of nodes in Texas and Film have lower degrees (1-2). According to Corollary 2, under strong heterophily, low-degree nodes experience signal cancellation (i.e. $\alpha_v = 0$) during the aggregation process. Therefore, setting $\lambda > 0$ (meaning some nodes do not aggregate at all) would improve performance.



(a) Citeseer (b) Pubmed (c) DBLP

(d) Chameleon (e) Film (f) Texas

Figure 5: Degree distribution of datasets.

Exp-5. Scalability Analysis We assess the scalability of AD-GNN using a large dataset (ogbn-arxiv) with varying depths. The hidden layer size is fixed at 128 for all methods. Each model is evaluated based on the number of learnable parameters, average runtime per epoch, and accuracy. The results are depicted in Table 2. Both AD-GCN variants exhibit a reasonable increase in computational complexity compared to the base model, while providing performance enhancements. Specifically, AD-GCN$_{\text{fast}}$ incurs minimal computational overhead, with a parameter count similar to that of GCN and a runtime increase of only around 5%.

	Depth = 4			Depth = 8		
	# Param (K)	Time (ms)	Acc. (%)	# Param (K)	Time (ms)	Acc. (%)
GCN	55.5	277.13	69.53	122.5	564.73	68.39
AD-GCN	88.5	405.85	70.32	155.6	655.52	70.63
AD-GCN$_{\text{fast}}$	55.5	286.40	70.18	122.5	580.15	70.42

Table 2: Performance comparison of AD-GCN variants.

Additional Experiments The Appendix includes four experiments: (1) comparing AD-GNN with additional state-of-the-art GNNs on node classification, (2) evaluating AD-GNN with a modified depth benefit metric from our extended analysis, (3) visualizing how the depth benefit metric varies with node degrees and AD-GNN's cluster embedding quality, and (4) providing empirical justifications for employing degree assortativity in AD-GNN$_{\text{fast}}$.

Conclusion, and Future Work

In this work, we provide novel theoretical insights into the impact of layer depth on the classification quality of nodes under varying homophily for node classification. Our findings indicate that strong heterophily is not necessarily negative, and the aggregation requirements of a node depend on the distribution of same-label neighbours, opposite-label neighbours, and total degree within its neighbourhood. By applying our theoretical insights, we develop a novel GNN plugin that adaptively determines the layer depth for each node, thereby improving their classification quality. Experiments on diverse graph structures and multiple GNN back-

bones demonstrate that our solution can consistently uplift performance in the node classification task.

Limitations of our theoretical analysis are provided in the Appendix. We plan to offer a more detailed theoretical analysis with relaxed assumptions in our future work. Furthermore, we plan to learn the hyperparameter λ in a data-driven manner.

References

Abbahaddou, Y.; Malliaros, F. D.; Lutzeyer, J. F.; and Vazirgiannis, M. 2025. ADMP-GNN: Adaptive Depth Message Passing GNN. *arXiv preprint arXiv:2509.01170*.

Abu-El-Haija, S.; Perozzi, B.; Kapoor, A.; Alipourfard, N.; Lerman, K.; Harutyunyan, H.; Ver Steeg, G.; and Galstyan, A. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, 21–29. PMLR.

Arcagni, A.; Grassi, R.; Stefani, S.; and Torriero, A. 2017. Higher order assortativity in complex networks. *European Journal of Operational Research*, 262(2): 708–719.

Bei, Y.; Zhou, S.; Shi, J.; Ma, Y.; Wang, H.; and Bu, J. 2025. Guarding Graph Neural Networks for Unsupervised Graph Anomaly Detection. *IEEE Transactions on Neural Networks and Learning Systems*.

Benslimane, S.; Azé, J.; Bringay, S.; Servajean, M.; and Mollevi, C. 2023. A text and GNN based controversy detection method on social media. *World Wide Web*, 26(2): 799–825.

Bi, W.; Du, L.; Fu, Q.; Wang, Y.; Han, S.; and Zhang, D. 2024. Make heterophilic graphs better fit gnn: A graph rewiring approach. *IEEE Transactions on Knowledge and Data Engineering*.

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.

Bojchevski, A.; and Günnemann, S. 2018. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *International Conference on Learning Representations*.

Bose, K.; Banerjee, S.; and Das, S. 2025. Can Graph Neural Networks Tackle Heterophily? Yes, With a Label-Guided Graph Rewiring Approach! *IEEE Transactions on Neural Networks and Learning Systems*.

Brody, S.; Alon, U.; and Yahav, E. 2022. How Attentive are Graph Attention Networks? In *International Conference on Learning Representations*.

Chen, J.; Deng, B.; Chen, C.; Zheng, Z.; et al. 2025. Graph neural ricci flow: Evolving feature from a curvature perspective. In *The Thirteenth International Conference on Learning Representations*.

Chien, E.; Peng, J.; Li, P.; and Milenkovic, O. 2021. Adaptive Universal Generalized PageRank Graph Neural Network. In *International Conference on Learning Representations*.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Deshpande, Y.; Sen, S.; Montanari, A.; and Mossel, E. 2018. Contextual stochastic block models. *Advances in Neural Information Processing Systems*, 31.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2): 179–188.

Ge, J.; Wang, X.; and Shi, W. 2021. Link prediction of the world container shipping network: A network structure perspective. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(11).

Guo, J.; Du, L.; Bi, W.; Fu, Q.; Ma, X.; Chen, X.; Han, S.; Zhang, D.; and Zhang, Y. 2023. Homophily-oriented heterogeneous graph rewiring. In *Proceedings of the ACM web conference 2023*, 511–522.

Guo, J.; Huang, K.; Zhang, R.; and Yi, X. 2024. ES-GNN: Generalizing graph neural networks beyond homophily with edge splitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

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

Haruta, S.; Konishi, T.; and Kurokawa, M. 2023. A Novel Graph Aggregation Method Based on Feature Distribution Around Each Ego-node for Heterophily. In *Asian Conference on Machine Learning*, 452–466. PMLR.

He, D.; Liang, C.; Liu, H.; Wen, M.; Jiao, P.; and Feng, Z. 2022. Block modeling-guided graph convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 4022–4029.

Hevapathige, A.; Zehmakan, A. N.; and Wang, Q. 2025. Depth-Adaptive Graph Neural Networks via Learnable Bakry-'Emery Curvature. *31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33: 22118–22133.

Khoshraftar, S.; and An, A. 2024. A survey on graph representation learning methods. *ACM Transactions on Intelligent Systems and Technology*, 15(1): 1–55.

Kim, H.; Lee, B. S.; Shin, W.-Y.; and Lim, S. 2022. Graph anomaly detection with graph neural networks: Current status and challenges. *IEEE Access*, 10: 111820–111829.

Kingma, D. 2015. Adam: a method for stochastic optimization. In *International Conference on Learning Representations*.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

Li, B.; Pan, E.; and Kang, Z. 2024. Pc-conv: Unifying homophily and heterophily with two-fold filtering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, 13437–13445.

Li, S.; Kim, D.; and Wang, Q. 2023. Restructuring graph for higher homophily via adaptive spectral clustering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 8622–8630.

Li, X.; Sun, L.; Ling, M.; and Peng, Y. 2023. A survey of graph neural network based recommendation in social networks. *Neurocomputing*, 549: 126441.

Li, X.; Zhu, R.; Cheng, Y.; Shan, C.; Luo, S.; Li, D.; and Qian, W. 2022. Finding global homophily in graph neural networks when meeting heterophily. In *International conference on machine learning*, 13242–13256. PMLR.

Lim, D.; Hohne, F.; Li, X.; Huang, S. L.; Gupta, V.; Bhalerao, O.; and Lim, S. N. 2021. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. *Advances in neural information processing systems*, 34: 20887–20902.

Luan, S.; Hua, C.; Lu, Q.; Zhu, J.; Zhao, M.; Zhang, S.; Chang, X.-W.; and Precup, D. 2021. Is heterophily a real nightmare for graph neural networks to do node classification? *arXiv preprint arXiv:2109.05641*.

Luan, S.; Hua, C.; Lu, Q.; Zhu, J.; Zhao, M.; Zhang, S.; Chang, X.-W.; and Precup, D. 2022. Revisiting heterophily for graph neural networks. *Advances in neural information processing systems*, 35: 1362–1375.

Ma, Y.; Liu, X.; Shah, N.; and Tang, J. 2022. Is Homophily a Necessity for Graph Neural Networks? In *International Conference on Learning Representations*.

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1): 415–444.

Pei, H.; Wei, B.; Chang, K. C. C.; Lei, Y.; and Yang, B. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *8th International Conference on Learning Representations, ICLR 2020*.

Platonov, O.; Kuznedelev, D.; Diskin, M.; Babenko, A.; and Prokhorenkova, L. 2023. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*.

Rossi, E.; Charpentier, B.; Di Giovanni, F.; Frasca, F.; Günnemann, S.; and Bronstein, M. M. 2024. Edge directionality improves learning on heterophilic graphs. In *Learning on graphs conference*, 25–1. PMLR.

Shchur, O.; Mumme, M.; Bojchevski, A.; and Günnemann, S. 2018. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*.

Singh, H.; and Singh, H. 2023. Analysis of Different Measures of Centrality to Identify Vital Nodes in Social Networks. In *International Conference on Advanced Network Technologies and Intelligent Computing*, 101–115. Springer.

Song, Y.; Zhou, C.; Wang, X.; and Lin, Z. 2023. Ordered GNN: Ordering Message Passing to Deal with Heterophily and Over-smoothing. In *The Eleventh International Conference on Learning Representations*.

Sun, H.; Li, X.; Wu, Z.; Su, D.; Li, R.-H.; and Wang, G. 2024. Breaking the entanglement of homophily and heterophily in semi-supervised node classification. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, 2379–2392. IEEE.

Suresh, S.; Budde, V.; Neville, J.; Li, P.; and Ma, J. 2021. 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*, 1541–1551.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Wang, J.; Guo, Y.; Yang, L.; and Wang, Y. 2024a. Understanding Heterophily for Graph Neural Networks. In *Forty-first International Conference on Machine Learning*.

Wang, J.; Guo, Y.; Yang, L.; and Wang, Y. 2024b. Understanding Heterophily for Graph Neural Networks. In *International Conference on Machine Learning*, 50489–50529. PMLR.

Wang, K.; Zhang, G.; Zhang, X.; Fang, J.; Wu, X.; Li, G.; Pan, S.; Huang, W.; and Liang, Y. 2024c. The heterophilic snowflake hypothesis: Training and empowering gnns for heterophilic graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3164–3175.

Wang, R.; Mou, S.; Wang, X.; Xiao, W.; Ju, Q.; Shi, C.; and Xie, X. 2021. Graph structure estimation neural networks. In *Proceedings of the web conference 2021*, 342–353.

Wang, T.; Jin, D.; Wang, R.; He, D.; and Huang, Y. 2022. Powerful graph convolutional networks with adaptive propagation mechanism for homophily and heterophily. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 4210–4218.

Wang, X.; and Xu, Y. 2019. An improved index for clustering validation based on Silhouette index and Calinski-Harabasz index. In *IOP Conference Series: Materials Science and Engineering*, volume 569, 052024. IOP Publishing.

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

Wu, Z.; Chen, J.; Al-Sabri, R.; Oloulade, B. M.; and Gao, J. 2024. Depth-adaptive graph neural architecture search for graph classification. *Knowledge-Based Systems*, 301: 112321.

Yan, Y.; Hashemi, M.; Swersky, K.; Yang, Y.; and Koutra, D. 2022. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *2022 IEEE International Conference on Data Mining (ICDM)*, 1287–1292. IEEE.

Yang, K.; Tu, J.; and Chen, T. 2019. Homoscedasticity: An overlooked critical assumption for linear regression. *General psychiatry*, 32(5): e100148.

Yang, Z.; Cohen, W.; and Salakhudinov, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, 40–48. PMLR.

Yi, H.-C.; You, Z.-H.; Huang, D.-S.; and Kwoh, C. K. 2022. Graph representation learning in bioinformatics: trends, methods and applications. *Briefings in Bioinformatics*, 23(1): bbab340.

Yu, Z.; Feng, B.; He, D.; Wang, Z.; Huang, Y.; and Feng, Z. 2024. LG-GNN: local-global adaptive graph neural network for modeling both homophily and heterophily. In *Proceedings of the thirty-third international joint conference on artificial intelligence*, 2515–2523.

Zhang, X.-M.; Liang, L.; Liu, L.; and Tang, M.-J. 2021. Graph neural networks and their current applications in bioinformatics. *Frontiers in genetics*, 12: 690049.

Zheng, X.; Wang, Y.; Liu, Y.; Li, M.; Zhang, M.; Jin, D.; Yu, P. S.; and Pan, S. 2022. Graph neural networks for graphs with heterophily: A survey. *arXiv preprint arXiv:2202.07082*.

Zheng, X.; Zhang, M.; Chen, C.; Zhang, Q.; Zhou, C.; and Pan, S. 2023a. Auto-heg: Automated graph neural network on heterophilic graphs. In *Proceedings of the ACM Web Conference 2023*, 611–620.

Zheng, Y.; Xu, J.; and Chen, L. 2024. Learn from heterophily: Heterophilous information-enhanced graph neural network. *arXiv preprint arXiv:2403.17351*.

Zheng, Y.; Zhang, H.; Lee, V.; Zheng, Y.; Wang, X.; and Pan, S. 2023b. Finding the missing-half: Graph complementary learning for homophily-prone and heterophily-prone graphs. In *International Conference on Machine Learning*, 42492–42505. PMLR.

Zhu, J.; Rossi, R. A.; Rao, A.; Mai, T.; Lipka, N.; Ahmed, N. K.; and Koutra, D. 2021. Graph neural networks with heterophily. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 11168–11176.

Zhu, J.; Yan, Y.; Heimann, M.; Zhao, L.; Akoglu, L.; and Koutra, D. 2023. Heterophily and graph neural networks: Past, present and future. *IEEE Data Engineering Bulletin*.

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.

Appendix

Analysis of our Theoretical Framework

While our theoretical framework offers a new perspective on GNN performance under varying homophily conditions, our analysis has several limitations. In this section, we discuss these limitations.

Simplified Aggregation Analysis We primarily focus on the uniform averaging operation, omitting several key components of real GCN architectures, including a symmetric normalisation scheme, weight transformation and non-linear activation functions. This is done to make our analysis more tractable. Our experimental results demonstrate that the insights derived under these relaxations are applicable to real-world settings. Nonetheless, incorporating these key components would strengthen the generalizability of our findings.

Layer Independence Assumption Theorem 2 assumes independence between features across layers, which might not hold due to feature correlations and oversmoothing. We have provided an extended theoretical analysis that incorporates these real-world factors and derives a more realistic depth benefit metric. Nonetheless, this assumption becomes more reasonable for shallow networks (i.e., 2-3 layers), which are generally optimal for node classification tasks. In such cases, the independence assumption serves as a useful first-order approximation.

Contextual Stochastic Block Model Structure Our analysis assumes a CSBM structure, which allows us to maintain Gaussian feature distributions and clear class separations. This aligns with existing theoretical frameworks built for GNNs that prioritise analytical tractability. However, it would be worthwhile to incorporate more realistic and complex graph structures to further strengthen the generalizability of our theoretical insights.

Binary Classification Restriction Our current theoretical analysis focuses on binary classification, which provides analytical feasibility. Our experimental results strongly validate the fact that our findings hold in multi-class settings. This transferability can be justified by several reasons.

- Regardless of distinguishing between two or k classes, nodes still benefit from same-label neighbours and are negatively impacted by opposite-label neighbours, making the high-level concept of the homophily-heterophily tradeoff identical.
- The presented tradeoff between signal preservation and noise reduction scales naturally to multi-class scenarios. In signal preservation, same-label neighbours reinforce the correct class prototype, while opposite-label neighbours (regardless of their specific labels) hinder the signal. The principle of reducing noise through neighbourhood averaging remains consistent, irrespective of the number of classes.
- Our key insight about balanced neighbourhoods causing signal cancellation extends naturally to a multi-class setting. Nodes with equal proportions of different label neighbours experience diluted class-specific signals, whether those different classes represent a single alternative (binary) or multiple alternatives (multi-class).

In our future work, we plan to formally extend our theoretical analysis to support a multi-class setting.

Architecture Generalizability Our current theoretical analysis is based on GCN with uniform neighbour aggregation. However, our experimental analysis incorporates multiple GNN backbones with diverse architectures, demonstrating that our insights also hold for these. Extending our theoretical analysis to more complex GNN architectures would further strengthen the generalizability of our insights.

Proofs

In this section, we provide proofs for the theorems in the main content.

Theorem 1 (Label Aggregation Effect)**.** *For any node v in the graph, under the stated graph-wide assumptions, the aggregated representation has expected value:*

$$\mathbb{E}[\mathbf{h}_v|y_v] = \frac{1+d_v^+}{d_v+1}\boldsymbol{\mu}^{y_v} + \frac{d_v^-}{d_v+1}\boldsymbol{\mu}^{1-y_v}$$

The signal variance after aggregation is:

$$\|\mathbb{E}[\mathbf{h}_v|y_v=0] - \mathbb{E}[\mathbf{h}_v|y_v=1]\|^2 = \alpha_v^2\Delta^2$$

The noise variance is: $Var[\mathbf{h}_v|y_v] = \frac{\sigma_{intra}^2}{d_v+1}$
The node-specific classification quality is: $Q_v = \frac{\alpha_v^2(d_v+1)\Delta^2}{\sigma_{intra}^2}$

Proof. We start by proving the Expected Representation. Starting from the aggregation formula:

$$\mathbf{h}_v = \frac{1}{d_v+1}\left(\mathbf{x}_v + \sum_{u\in\mathcal{N}_v^+}\mathbf{x}_u + \sum_{u\in\mathcal{N}_v^-}\mathbf{x}_u\right)$$

Taking expectations conditioned on the node's label y_v:

$$\mathbb{E}[\mathbf{h}_v|y_v] = \frac{1}{d_v+1}\Big(\mathbb{E}[\mathbf{x}_v|y_v] + \sum_{u\in\mathcal{N}_v^+}\mathbb{E}[\mathbf{x}_u|y_v] + \sum_{u\in\mathcal{N}_v^-}\mathbb{E}[\mathbf{x}_u|y_v]\Big)$$

By the feature assumptions, $\mathbb{E}[\mathbf{x}_v|y_v] = \boldsymbol{\mu}^{y_v}$, $\mathbb{E}[\mathbf{x}_u|y_v] = \boldsymbol{\mu}^{y_v}$ for $u \in \mathcal{N}_v^+$, and $\mathbb{E}[\mathbf{x}_u|y_v] = \boldsymbol{\mu}^{1-y_v}$ for $u \in \mathcal{N}_v^-$. Substituting:

$$\begin{aligned}\mathbb{E}[\mathbf{h}_v|y_v] &= \frac{1}{d_v+1}\Big(\boldsymbol{\mu}^{y_v} + d_v^+\cdot\boldsymbol{\mu}^{y_v} + d_v^-\cdot\boldsymbol{\mu}^{1-y_v}\Big)\\ &= \frac{1+d_v^+}{d_v+1}\boldsymbol{\mu}^{y_v} + \frac{d_v^-}{d_v+1}\boldsymbol{\mu}^{1-y_v}\end{aligned}$$

Then, we prove the Signal Variance. For class 0 nodes ($y_v = 0$):

$$\mathbb{E}[\mathbf{h}_v|y_v=0] = \frac{1+d_v^+}{d_v+1}\boldsymbol{\mu}^0 + \frac{d_v^-}{d_v+1}\boldsymbol{\mu}^1$$

For class 1 nodes ($y_v = 1$):

$$\mathbb{E}[\mathbf{h}_v|y_v = 1] = \frac{1 + d_v^+}{d_v + 1}\boldsymbol{\mu}^1 + \frac{d_v^-}{d_v + 1}\boldsymbol{\mu}^0$$

Computing the difference:

$$\begin{aligned}
&\mathbb{E}[\mathbf{h}_v|y_v = 0] - \mathbb{E}[\mathbf{h}_v|y_v = 1] \\
&= \frac{1 + d_v^+}{d_v + 1}\boldsymbol{\mu}^0 + \frac{d_v^-}{d_v + 1}\boldsymbol{\mu}^1 - \frac{1 + d_v^+}{d_v + 1}\boldsymbol{\mu}^1 - \frac{d_v^-}{d_v + 1}\boldsymbol{\mu}^0 \\
&= \frac{1 + d_v^+ - d_v^-}{d_v + 1}(\boldsymbol{\mu}^0 - \boldsymbol{\mu}^1)
\end{aligned}$$

Taking the squared norm:

$$\begin{aligned}
&\|\mathbb{E}[\mathbf{h}_v|y_v = 0] - \mathbb{E}[\mathbf{h}_v|y_v = 1]\|^2 \\
&= \left(\frac{1 + d_v^+ - d_v^-}{d_v + 1}\right)^2 \|\boldsymbol{\mu}^0 - \boldsymbol{\mu}^1\|^2 \\
&= \alpha_v^2 \Delta^2
\end{aligned}$$

Then, we prove the Noise Variance. The noise variance is:

$$\text{Var}[\mathbf{h}_v|y_v] = \text{Var}\left[\frac{1}{d_v + 1}\Big(\boldsymbol{\epsilon}_v + \sum_{u \in \mathcal{N}_v^+} \boldsymbol{\epsilon}_u^+ + \sum_{u \in \mathcal{N}_v^-} \boldsymbol{\epsilon}_u^-\Big)\right]$$

Since there are $1 + d_v^+ + d_v^- = d_v + 1$ independent noise terms, each with variance σ^2_{intra}:

$$\begin{aligned}
\text{Var}[\mathbf{h}_v|y_v] &= \frac{1}{(d_v + 1)^2} \cdot (d_v + 1)\sigma^2_{\text{intra}} \\
&= \frac{\sigma^2_{\text{intra}}}{d_v + 1}
\end{aligned}$$

Finally, we derive the Classification Quality. By definition, classification quality is the ratio of signal variance to noise variance:

$$\begin{aligned}
Q_v &= \frac{\alpha_v^2 \Delta^2}{\sigma^2_{\text{intra}}/(d_v + 1)} \\
&= \frac{\alpha_v^2 (d_v + 1)\Delta^2}{\sigma^2_{\text{intra}}}
\end{aligned}$$

This completes the proof. □

Theorem 2 (Iterative Aggregation Effect)**.** *Assume a n-layer GNN where: (1) label-conditioned features remain independent across layers, and (2) each layer performs the same neighborhood aggregation pattern with degree d_v. Then, for node v, after n layers:*

- *Signal variance:* $\alpha_v^{2n}\Delta^2$
- *Noise variance:* $\frac{\sigma^2_{intra}}{(d_v+1)^n}$
- *Classification quality:* $Q_v^n = \frac{\alpha_v^{2n}(d_v+1)^n\Delta^2}{\sigma^2_{intra}}$

Proof. We proceed by induction on the number of layers n.

First, consider the base case ($n = 1$). From Theorem 1, for a single aggregation layer:

$$\begin{aligned}
\mathbb{E}[\mathbf{h}_v^{(1)}|y_v] &= \frac{1 + d_v^+}{d_v + 1}\boldsymbol{\mu}^{y_v} + \frac{d_v^-}{d_v + 1}\boldsymbol{\mu}^{1-y_v} \\
\text{Var}[\mathbf{h}_v^{(1)}|y_v] &= \frac{\sigma^2_{\text{intra}}}{d_v + 1}
\end{aligned}$$

The signal variance is:

$$\begin{aligned}
&\|\mathbb{E}[\mathbf{h}_v^{(1)}|y_v = 0] - \mathbb{E}[\mathbf{h}_v^{(1)}|y_v = 1]\|^2 \\
&= \left(\frac{1 + d_v^+ - d_v^-}{d_v + 1}\right)^2 \Delta^2 = \alpha_v^2 \Delta^2
\end{aligned}$$

And classification quality:

$$Q_v^{(1)} = \frac{\alpha_v^2 \Delta^2}{\sigma^2_{\text{intra}}/(d_v + 1)} = \frac{\alpha_v^2 (d_v + 1)\Delta^2}{\sigma^2_{\text{intra}}}$$

Assume that after k layers:

$$\begin{aligned}
\|\mathbb{E}[\mathbf{h}_v^{(k)}|y_v = 0] - \mathbb{E}[\mathbf{h}_v^{(k)}|y_v = 1]\|^2 &= \alpha_v^{2k}\Delta^2 \\
\text{Var}[\mathbf{h}_v^{(k)}|y_v] &= \frac{\sigma^2_{\text{intra}}}{(d_v + 1)^k} \\
Q_v^{(k)} &= \frac{\alpha_v^{2k}(d_v + 1)^k \Delta^2}{\sigma^2_{\text{intra}}}
\end{aligned}$$

Consider the step ($k \to k + 1$): At layer $k + 1$, we apply the same aggregation operation to the outputs of layer k. Let $\mathbf{r}_0^{(k)} = \mathbb{E}[\mathbf{h}_v^{(k)}|y_v = 0]$ and $\mathbf{r}_1^{(k)} = \mathbb{E}[\mathbf{h}_v^{(k)}|y_v = 1]$. By hypothesis:

$$\|\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)}\|^2 = \alpha_v^{2k}\Delta^2$$

Applying one more aggregation layer:

$$\begin{aligned}
\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v = 0] &= \frac{1}{d_v + 1}\left(\mathbf{r}_0^{(k)} + \sum_{u \in \mathcal{N}_v^+} \mathbf{r}_0^{(k)} + \sum_{u \in \mathcal{N}_v^-} \mathbf{r}_1^{(k)}\right) \\
&= \frac{1 + d_v^+}{d_v + 1}\mathbf{r}_0^{(k)} + \frac{d_v^-}{d_v + 1}\mathbf{r}_1^{(k)}
\end{aligned}$$

Similarly:

$$\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v = 1] = \frac{1 + d_v^+}{d_v + 1}\mathbf{r}_1^{(k)} + \frac{d_v^-}{d_v + 1}\mathbf{r}_0^{(k)}$$

The signal difference becomes:

$$\begin{aligned}
&\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v = 0] - \mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v = 1] \\
&= \frac{1 + d_v^+ - d_v^-}{d_v + 1}(\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)}) \\
&= \alpha_v(\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)})
\end{aligned}$$

Taking the squared norm:

$$\begin{aligned}
&\|\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v = 0] - \mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v = 1]\|^2 \\
&= \alpha_v^2 \|\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)}\|^2 \\
&= \alpha_v^2 \cdot \alpha_v^{2k}\Delta^2 = \alpha_v^{2(k+1)}\Delta^2
\end{aligned}$$

For the noise variance, each aggregation layer independently reduces variance by factor $(d_v + 1)$:

$$\begin{aligned}
\text{Var}[\mathbf{h}_v^{(k+1)}|y_v] &= \frac{1}{d_v + 1}\text{Var}[\mathbf{h}_v^{(k)}|y_v] \\
&= \frac{1}{d_v + 1} \cdot \frac{\sigma^2_{\text{intra}}}{(d_v + 1)^k} \\
&= \frac{\sigma^2_{\text{intra}}}{(d_v + 1)^{k+1}}
\end{aligned}$$

Finally, the classification quality becomes:

$$Q_v^{(k+1)} = \frac{\alpha_v^{2(k+1)}\Delta^2}{\sigma_{\text{intra}}^2/(d_v+1)^{k+1}} = \frac{\alpha_v^{2(k+1)}(d_v+1)^{k+1}\Delta^2}{\sigma_{\text{intra}}^2}$$

This completes the induction and proves all three statements. □

Extended Theoretical Analysis

Multi-Layer Analysis with Feature Correlation and Oversmoothing

While our analysis in Theorem 2 assumes feature independence across layers, real GNN behaviour exhibits deviations due to feature correlation and over-smoothing effects. These deviations can be characterised by node-specific constants that we define below.

Definition 5 (Signal Calibration Factor)**.** *For node v at layer n, the signal calibration factor $\beta_{v,n}$ represents the multiplicative deviation from idealized signal preservation at that layer:*

$$\beta_{v,n} = \frac{\|\mathbb{E}[\mathbf{h}_v^{(n)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(n)}|y_v=1]\|^2}{\alpha_v^2\|\mathbb{E}[\mathbf{h}_v^{(n-1)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(n-1)}|y_v=1]\|^2}$$

For simplicity, when $\beta_{v,n}$ is approximately constant across layers, we denote it as β_v.

Definition 6 (Noise Evolution Factor)**.** *For node v at layer n, the noise evolution factor $\gamma_{v,n}$ captures the deviation from degree-based noise reduction at that layer:*

$$\gamma_{v,n} = \frac{(d_v+1)\mathit{Var}[\mathbf{h}_v^{(n)}|y_v]}{\mathit{Var}[\mathbf{h}_v^{(n-1)}|y_v]}$$

For simplicity, when $\gamma_{v,n}$ is approximately constant across layers, we denote it as γ_v.

From the above, we derive the modified theorem for multi-layer analysis below:

Theorem 3 (Multi-Layer Analysis with Calibration Factors)**.** *Consider an n-layer GNN with signal preservation factor $\alpha_v = \frac{1+d_v^+-d_v^-}{d_v+1}$. Let β_v and γ_v be the signal calibration and noise evolution factors respectively, representing systematic deviations from idealized aggregation behavior. Then after n layers:*

- *Signal variance:* $\|\mathbb{E}[\mathbf{h}_v^{(n)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(n)}|y_v=1]\|^2 = \beta_v^n\alpha_v^{2n}\Delta^2$
- *Noise variance:* $\mathit{Var}[\mathbf{h}_v^{(n)}|y_v] = \frac{\gamma_v^n\sigma_{\mathit{intra}}^2}{(d_v+1)^n}$
- *Classification quality:* $Q_v^{(n)} = \frac{\beta_v^n\alpha_v^{2n}(d_v+1)^n\Delta^2}{\gamma_v^n\sigma_{\mathit{intra}}^2} = \left(\frac{\beta_v\alpha_v^2(d_v+1)}{\gamma_v}\right)^n\frac{\Delta^2}{\sigma_{\mathit{intra}}^2}$

where $\frac{\Delta^2}{\sigma_{\mathit{intra}}^2}$ is the initial signal-to-noise ratio before aggregation.

Proof. We proceed by induction on the number of layers n, incorporating the theoretical calibration factors.

First, we consider the base case ($n=1$). From Theorem 1, the single-layer behavior gives:

$$\|\mathbb{E}[\mathbf{h}_v^{(1)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(1)}|y_v=1]\|^2 = \alpha_v^2\Delta^2$$
$$\text{Var}[\mathbf{h}_v^{(1)}|y_v] = \frac{\sigma_{\text{intra}}^2}{d_v+1}$$

However, real GNN behavior exhibits systematic deviations captured by the calibration factors:

$$\|\mathbb{E}[\mathbf{h}_v^{(1)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(1)}|y_v=1]\|^2 = \beta_v\alpha_v^2\Delta^2$$
$$\text{Var}[\mathbf{h}_v^{(1)}|y_v] = \frac{\gamma_v\sigma_{\text{intra}}^2}{d_v+1}$$

The classification quality becomes:

$$Q_v^{(1)} = \frac{\beta_v\alpha_v^2\Delta^2}{\gamma_v\sigma_{\text{intra}}^2/(d_v+1)} = \frac{\beta_v\alpha_v^2(d_v+1)\Delta^2}{\gamma_v\sigma_{\text{intra}}^2}$$

This establishes the base case. Assume that after k layers:

$$\|\mathbb{E}[\mathbf{h}_v^{(k)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(k)}|y_v=1]\|^2 = \beta_v^k\alpha_v^{2k}\Delta^2$$

$$\text{Var}[\mathbf{h}_v^{(k)}|y_v] = \frac{\gamma_v^k\sigma_{\text{intra}}^2}{(d_v+1)^k}$$

$$Q_v^{(k)} = \left(\frac{\beta_v\alpha_v^2(d_v+1)}{\gamma_v}\right)^k\frac{\Delta^2}{\sigma_{\text{intra}}^2}$$

Consider the step ($k \to k+1$). At layer $k+1$, we apply aggregation to the outputs of layer k. Let $\mathbf{r}_0^{(k)} = \mathbb{E}[\mathbf{h}_v^{(k)}|y_v=0]$ and $\mathbf{r}_1^{(k)} = \mathbb{E}[\mathbf{h}_v^{(k)}|y_v=1]$. By the inductive hypothesis:

$$\|\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)}\|^2 = \beta_v^k\alpha_v^{2k}\Delta^2$$

Applying one more aggregation layer with calibration factor:

$$\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v=0] = \frac{1+d_v^+}{d_v+1}\mathbf{r}_0^{(k)} + \frac{d_v^-}{d_v+1}\mathbf{r}_1^{(k)}$$
$$\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v=1] = \frac{1+d_v^+}{d_v+1}\mathbf{r}_1^{(k)} + \frac{d_v^-}{d_v+1}\mathbf{r}_0^{(k)}$$

The signal difference becomes:

$$\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v=1]$$
$$= \frac{1+d_v^+-d_v^-}{d_v+1}(\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)})$$
$$= \alpha_v(\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)})$$

Incorporating the signal calibration factor for this layer:

$$\|\mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v=0] - \mathbb{E}[\mathbf{h}_v^{(k+1)}|y_v=1]\|^2$$
$$= \beta_v\alpha_v^2\|\mathbf{r}_0^{(k)} - \mathbf{r}_1^{(k)}\|^2$$
$$= \beta_v\alpha_v^2 \cdot \beta_v^k\alpha_v^{2k}\Delta^2 = \beta_v^{k+1}\alpha_v^{2(k+1)}\Delta^2$$

Dataset	Homophily Ratio	#Nodes	#Edges	#Classes	#Features
Cora ML	0.79	2,995	16,316	7	2,879
Citeseer	0.71	3,327	4,732	6	3,703
Pubmed	0.79	19,717	44,338	3	500
Photo	0.83	7,650	119,081	8	745
DBLP	0.83	17,716	105,734	4	1,639
Film	0.24	7,600	33,544	5	931
Squirrel	0.22	5,201	217,073	5	2,089
Chameleon	0.25	2,277	36,101	5	2,325
Cornell	0.11	183	295	5	1,703
Wisconsin	0.16	251	499	5	1,703
Texas	0.06	183	309	5	1,703
ogbn-arxiv	0.65	169,343	1,166,243	40	128

Table 3: Dataset statistics.

For the noise variance, each layer introduces calibration factor γ_v:

$$\begin{aligned}\text{Var}[\mathbf{h}_v^{(k+1)}|y_v] &= \frac{\gamma_v}{d_v+1}\text{Var}[\mathbf{h}_v^{(k)}|y_v] \\ &= \frac{\gamma_v}{d_v+1}\cdot\frac{\gamma_v^k\sigma^2_{\text{intra}}}{(d_v+1)^k} \\ &= \frac{\gamma_v^{k+1}\sigma^2_{\text{intra}}}{(d_v+1)^{k+1}}\end{aligned}$$

The classification quality becomes:

$$\begin{aligned}Q_v^{(k+1)} &= \frac{\beta_v^{k+1}\alpha_v^{2(k+1)}\Delta^2}{\gamma_v^{k+1}\sigma^2_{\text{intra}}/(d_v+1)^{k+1}} \\ &= \frac{\beta_v^{k+1}\alpha_v^{2(k+1)}(d_v+1)^{k+1}\Delta^2}{\gamma_v^{k+1}\sigma^2_{\text{intra}}} \\ &= \left(\frac{\beta_v\alpha_v^2(d_v+1)}{\gamma_v}\right)^{k+1}\frac{\Delta^2}{\sigma^2_{\text{intra}}}\end{aligned}$$

This completes the induction and proves all three statements. □

Next, we extend the proposed depth benefit metric to incorporate the calibration phenomena.

Definition 7 (Modified Depth Benefit Metric)**.** *For node v and target depth n, incorporating calibration factors, the Depth Benefit Metric is:*

$$\varepsilon_v^n = \left(\frac{\beta_v\alpha_v^2(d_v+1)}{\gamma_v}\right)^n$$

where β_v is the signal calibration factor and γ_v is the noise evolution factor for node v.

Supplementary Experimental Details

Benchmark Dataset Statistics

Dataset statistics, including homophily ratios, graph sizes, feature dimensions, and number of labels, are depicted in Table 3.

Model Hyper-Parameters

We search model hyper-parameters in following ranges: number of layers $\in \{2,3,4\}$, dropout $\in \{0.1,0.5,0.6\}$, learning rate $\in \{0.001,0.005,0.01\}$, hidden dimension $\in \{128,256\}$, weight decay $\in \{1\times10^{-4},5\times10^{-4},1\times10^{-1}\}$, epochs $\in \{200,500,1000\}$, and $\lambda \in \{0.0,0.1,0.8,0.9\}$. We employ the Adam algorithm (Kingma 2015) for optimisation.

Computational Resources, and Implementation Details

All experiments were conducted on a Linux server equipped with an Intel Xeon W-2175 2.50GHz processor featuring 28 cores, an NVIDIA RTX A6000 GPU, and 512GB of RAM. Our implementation employs pytorch version 2.3.1, torchvision 0.18.1, torchaudio 2.3.1, torch-geometric 2.7.0, torch-cluster 1.6.3, torch-scatter 2.0.9, and torch_sparse 0.6.18.

Additional Experiments

Comparison with Existing GNNs We compare AD-GNN with existing state-of-the-art GNNs. Our baselines include general-purpose GNNs, as well as GNNs designed for heterophily, namely, GCN (Kipf and Welling 2017), GAT (Veličković et al. 2018), GraphSage (Hamilton, Ying, and Leskovec 2017), GCN-Cheby (Defferrard, Bresson, and Vandergheynst 2016), MixHop (Abu-El-Haija et al. 2019), SGC (Wu et al. 2019), WRGAT (Suresh et al. 2021), ACM-GCN (Luan et al. 2021), H2GCN (Zhu et al. 2020), GPR-GNN (Chien et al. 2021), GGCN (Yan et al. 2022), LINKX (Lim et al. 2021), GloGNN (Li et al. 2022), Dir-GNN (Rossi et al. 2024), and Hi-GNN (Zheng, Xu, and Chen 2024). The results are depicted in Table 4. As shown in the results, AD-GNN variants significantly outperform existing state-of-the-art GNNs in both homophilic and heterophilic graphs.

Node Classification Performance with Modified Depth Benefit Metric We evaluate AD-GNN with the modified depth benefit metric introduced in the extended theoretical analysis. The modified AD-GNN variant is named "AD-GNN$^{\#}$". The results are depicted in Table 5. The performance of AD-GNN$^{\#}$ is closely aligned with that of AD-GNN. We believe this is due to the fact that signal calibra-

Methods	Citeseer	Pubmed	Film	Cornell	Wisconsin	Texas
GCN	76.68 ± 1.64	87.38 ± 0.66	30.26 ± 0.79	57.03 ± 4.67	59.80 ± 6.99	59.46 ± 5.25
GAT	75.46 ± 1.72	87.62 ± 0.42	26.28 ± 1.73	58.92 ± 3.32	55.29 ± 8.71	58.38 ± 4.45
GraphSage	76.04 ± 1.30	88.45 ± 0.50	34.23 ± 0.99	75.95 ± 5.01	81.18 ± 5.56	82.43 ± 6.14
SGC	75.66 ± 1.37	87.15 ± 0.47	25.83 ± 1.09	55.41 ± 5.29	57.84 ± 4.83	58.11 ± 6.27
GCN-Cheby	76.25 ± 1.76	88.08 ± 0.52	36.11 ± 1.09	74.32 ± 7.46	79.41 ± 4.46	77.30 ± 4.07
MixHop	70.75 ± 2.95	80.75 ± 2.29	32.22 ± 2.34	73.51 ± 6.34	75.88 ± 4.90	77.84 ± 7.73
H_2GCN	76.72 ± 1.50	88.50 ± 0.64	35.86 ± 1.03	82.16 ± 4.80	86.67 ± 4.69	84.86 ± 6.77
ACM-GCN	75.56 ± 1.32	89.48 ± 0.58	35.09 ± 1.18	77.57 ± 5.26	83.53 ± 3.83	82.70 ± 6.27
GGCN	76.65 ± 1.91	88.25 ± 0.43	34.86 ± 0.87	71.35 ± 7.34	74.12 ± 5.37	65.14 ± 8.30
LINKX	72.00 ± 1.90	78.39 ± 1.09	27.06 ± 1.22	39.46 ± 17.89	55.88 ± 6.29	52.43 ± 9.21
GLOGNN	77.41 ± 1.65	89.62 ± 0.35	37.36 ± 1.34	82.16 ± 5.82	82.35 ± 5.11	69.19 ± 11.16
WRGAT	76.81 ± 1.89	88.52 ± 0.92	36.53 ± 0.77	81.62 ± 3.90	86.98 ± 3.78	83.62 ± 5.50
Dir-GNN	77.71 ± 0.78	86.94 ± 0.55	35.76 ± 6.31	76.51 ± 6.14	80.50 ± 5.50	76.25 ± 4.68
Hi-GNN	79.30 ± 2.13	89.43 ± 0.53	37.21 ± 1.35	80.00 ± 4.26	85.88 ± 3.18	86.22 ± 4.67
AD-GCN	79.14 ± 1.00	88.39 ± 0.32	42.54 ± 1.15	88.51 ± 4.87	93.88 ± 3.03	92.30 ± 4.52
AD-MixHop	81.01 ± 1.36	90.09 ± 0.58	43.37 ± 1.17	90.21 ± 3.59	94.75 ± 2.22	94.43 ± 2.56

Table 4: Node classification accuracy ± standard deviation (%). The best results are highlighted. Baseline results are sourced from Suresh et al. (2021), and Zheng, Xu, and Chen (2024).

Methods	Cora-ML	Citeseer	Pubmed	Photo	DBLP	Film	Squirrel	Chameleon	Cornell	Wisconsin	Texas
GCN	87.07 ± 1.21	76.68 ± 1.64	86.74 ± 0.47	89.30 ± 0.82	83.93 ± 0.34	30.26 ± 0.79	39.47 ± 1.47	40.89 ± 4.12	55.14 ± 8.46	61.60 ± 7.00	60.00 ± 6.45
AD-GCN	87.32 ± 1.25	79.14 ± 1.00	88.39 ± 0.32	94.10 ± 0.31	84.14 ± 0.44	42.54 ± 1.15	40.04 ± 0.99	43.66 ± 0.73	88.51 ± 4.87	93.88 ± 3.03	92.30 ± 4.52
AD-GCN#	87.00 ± 1.95	79.58 ± 1.38	89.22 ± 0.39	93.87 ± 0.65	84.10 ± 0.44	42.20 ± 1.44	40.64 ± 1.01	44.54 ± 1.13	89.79 ± 4.54	94.00 ± 2.89	91.97 ± 3.40
GAT	84.12 ± 0.55	75.46 ± 1.72	87.24 ± 0.55	90.81 ± 0.22	80.61 ± 1.21	26.28 ± 1.73	35.62 ± 2.06	39.21 ± 3.08	53.64 ± 11.1	60.00 ± 11.0	61.21 ± 8.17
AD-GAT	85.02 ± 1.64	79.92 ± 0.76	87.38 ± 0.33	94.03 ± 0.34	83.94 ± 0.40	41.25 ± 0.77	36.73 ± 0.83	40.52 ± 1.55	86.17 ± 4.69	91.50 ± 2.42	90.49 ± 5.72
AD-GAT#	85.25 ± 1.57	78.83 ± 1.69	88.61 ± 0.51	92.43 ± 1.16	84.56 ± 1.25	42.23 ± 1.81	39.18 ± 1.19	38.92 ± 2.23	90.85 ± 2.70	94.75 ± 2.67	92.95 ± 3.52
GraphSAGE	86.52 ± 1.32	76.04 ± 1.30	88.45 ± 0.50	94.23 ± 0.62	86.16 ± 0.50	34.23 ± 0.99	36.09 ± 1.99	37.77 ± 4.14	75.95 ± 5.01	81.18 ± 5.56	82.43 ± 6.14
AD-GraphSAGE	87.14 ± 1.56	79.84 ± 1.26	88.99 ± 0.64	94.61 ± 0.35	85.00 ± 1.60	40.92 ± 1.46	40.88 ± 0.87	40.10 ± 1.45	89.57 ± 4.30	94.62 ± 2.56	92.95 ± 2.84
AD-GraphSAGE#	86.73 ± 1.90	78.31 ± 1.53	89.62 ± 0.55	94.74 ± 0.66	84.62 ± 1.33	41.47 ± 1.33	39.80 ± 0.65	39.28 ± 1.17	89.57 ± 3.74	94.75 ± 2.08	93.44 ± 2.20

Table 5: Node classification performance of the AD-GNN variant with a modified depth benefit metric. The best results are highlighted.

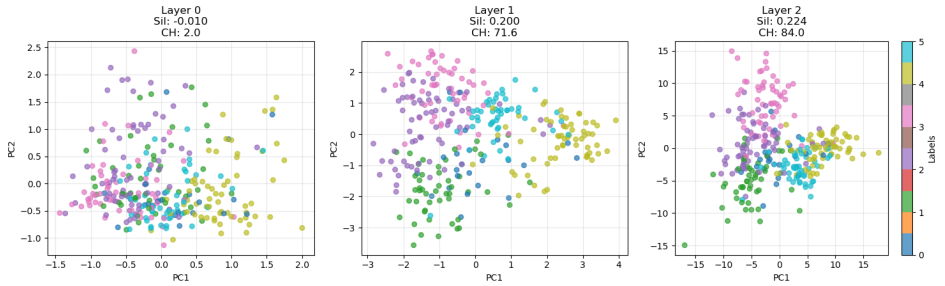
tion and noise evolution factors have less impact on shallow layers, where label-conditioned features remain nearly independent across layers, consistent with Theorem 2. Shallow layers are known to be more beneficial for node classification compared to deeper layers.

Visualization Analysis We provide two visualisation experiments related to AD-GNN. First, we analyse the embedding quality evolution of AD-GNN variants compared to the baselines in Figure 6. We also present the Silhouette Score and Calinski-Harabasz Score (Wang and Xu 2019) as metrics for cluster quality. Higher values of these metrics indicate better embedding clustering, resulting in improved classification. Based on the embedding visualisations, AD-GNN variants demonstrate superior performance compared to the baselines. AD-GNN achieves significantly better final layer separation, with a higher Silhouette score and a substantially higher Calinski-Harabasz score, compared to baselines. This indicates that AD-GNN's adaptive depth mechanism enables more effective feature learning and information propagation, resulting in better class boundaries in the final embedding space.

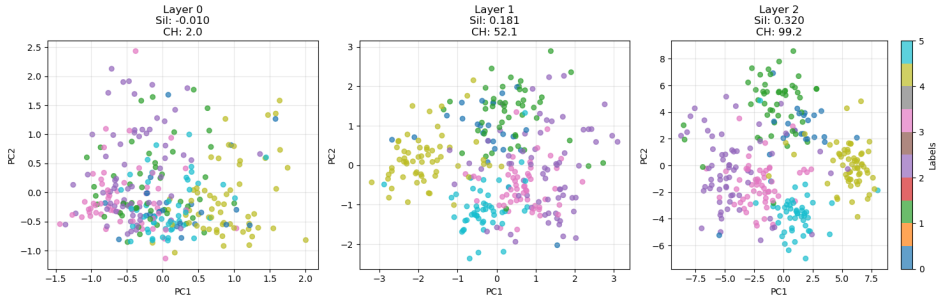
In the second experiment, we analyse the average depth benefit metric computed by AD-GCN across different node degrees in various datasets. The corresponding visualisation is shown in Figure 8. As observed, the depth benefit metric tends to increase with node degree across all datasets. This trend is primarily due to the decrease in noise variance as node degree increases. Also, heterophilic graphs exhibit lower depth benefit metrics, as nodes in these datasets tend to have mixed label neighbourhoods, compared to homophilic ones. Additionally, in heterophilic datasets, where most nodes have low degrees (such as Cornell and Texas), the depth benefit metric values are typically much lower. This is because avoiding aggregation for nodes with low degrees is beneficial under strong heterophily, as demonstrated in Corollary 2.

Comparison with Different Heuristics for AD-GNN$_{\text{fast}}$ We empirically justify the design choice of degree assortativity for AD-GNN$_{\text{fast}}$ by comparing it with alternative heuristics. The similarity baselines we employ include common neighbors ratio, Jaccard similarity, Adamic–Adar index, betweenness centrality, k-shell, and clustering coefficient (Ge, Wang, and Shi 2021; Singh and Singh 2023). Higher similarity metric values indicate more substantial label similarity. The performance and runtime comparisons are depicted in Table 6, and Figure 7, respectively.

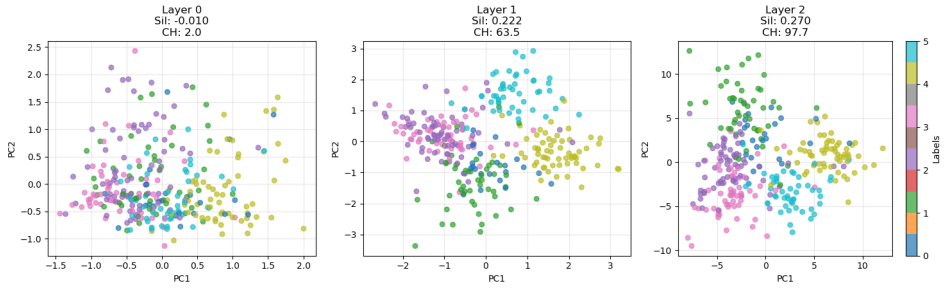
The results demonstrate that degree assortativity yields better classification performance and lower runtime compared to other heuristics. Given that degree assortativity requires only global degree statistics rather than expensive



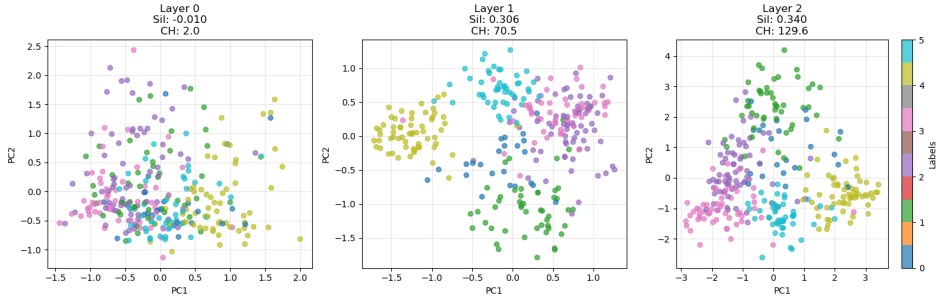
(a) GCN



(b) GAT



(c) AD-GCN



(d) AD-GAT

Figure 6: Layer-wise Embedding Visualization for Citeseer Dataset. Sil and CH acronyms refer to the silhouette score and the Calinski-Harabasz score, respectively.

neighbourhood computations, it is computationally efficient compared to pairwise similarity methods. Moreover, unlike local heuristics such as common neighbours or the Adamic-Adar index, which are sensitive to variations in local structure, degree assortativity captures higher-order global patterns. This makes it more robust against network sparsity, heterophily, and irregularities in local structures.

Methods	Citeseer	Pubmed	Chameleon	Wisconsin
Common Neighbors Ratio	78.44 $\pm$1.00	87.93 $\pm$0.36	43.20 $\pm$1.10	88.75 $\pm$2.18
Jaccard Similarity	78.49 $\pm$1.47	88.34 $\pm$0.36	43.33 $\pm$1.53	91.60 $\pm$2.18
Adamic-Adar Index	78.85 $\pm$1.01	88.34 $\pm$0.36	43.30 $\pm$1.14	90.00 $\pm$2.36
Centrality Similarity (Betweenness)	78.72 $\pm$1.49	88.09 $\pm$0.37	43.51 $\pm$1.29	88.62 $\pm$4.52
K-shell Similarity	78.61 $\pm$1.57	88.38 $\pm$0.35	41.55 $\pm$1.58	89.50 $\pm$5.89
Clustering Coefficient Similarity	78.62 $\pm$1.52	88.39 $\pm$0.51	42.78 $\pm$1.63	89.88 $\pm$3.60
Degree Assortativity (Ours)	79.13 $\pm$0.99	88.41 $\pm$0.37	43.40 $\pm$0.56	94.38 $\pm$2.86

Table 6: Comparison with different heuristics. The best results are highlighted.

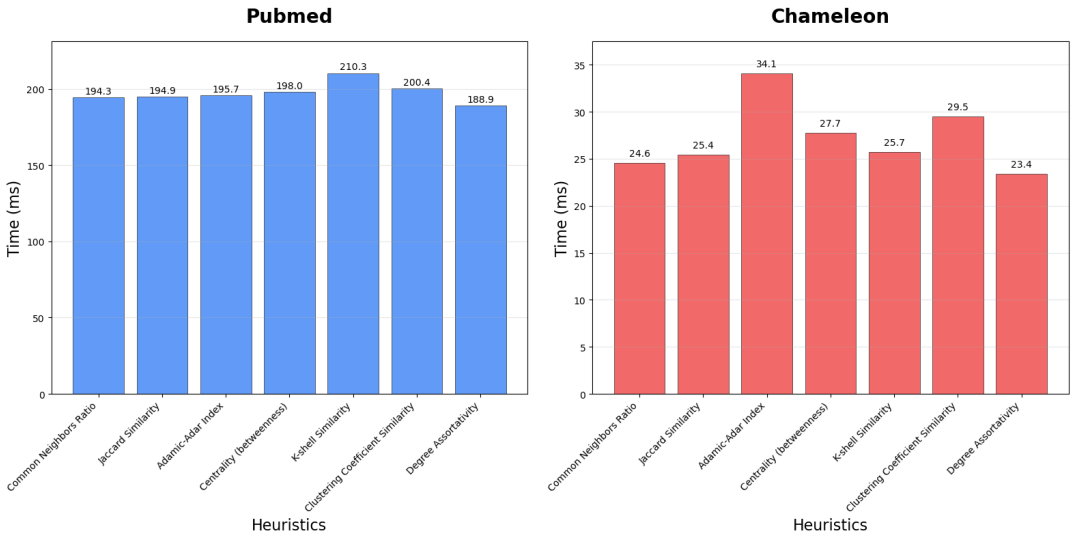
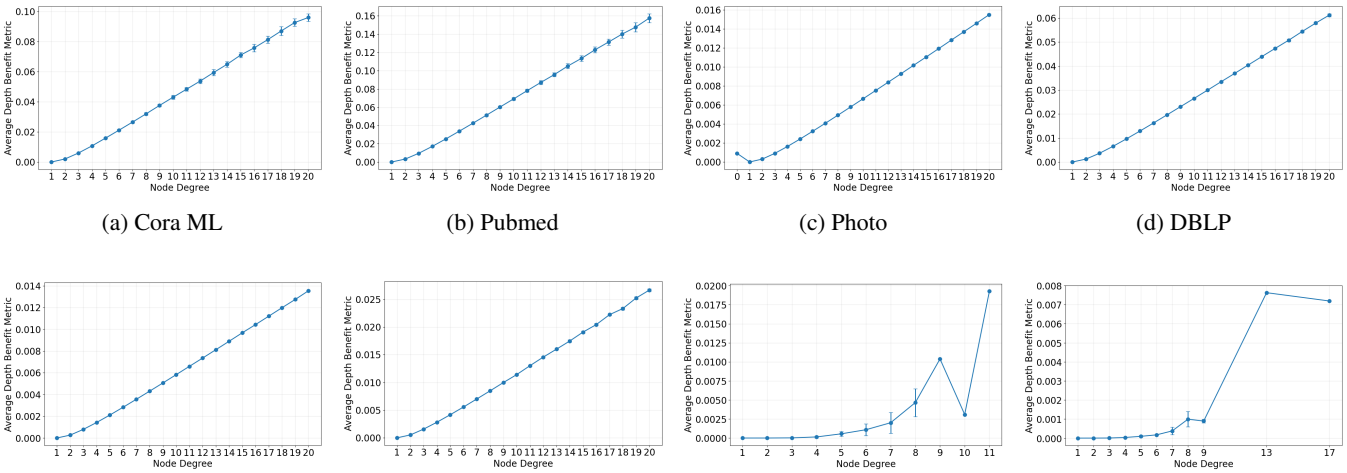


Figure 7: Runtime (per epoch) comparison between different heuristics.



(a) Cora ML (b) Pubmed (c) Photo (d) DBLP

(e) Squirrel (f) Chameleon (g) Cornell (h) Texas

Figure 8: Average depth benefit metric computed per degree in homophilic and heterophilic datasets.