On Performance Discrepancies Across Local Homophily Levels in Graph Neural Networks

Donald Loveland
University of Michigan, Ann Arbor
dlovelan@umich.edu

Jiong Zhu
University of Michigan, Ann Arbor
jiongzhu@umich.edu

Mark Heimann
Lawrence Livermore National Lab
heimann2@llnl.gov

Benjamin Fish
University of Michigan, Ann Arbor
benfish@umich.edu

Michael T. Schaub
RWTH Aachen University
schaub@cs.rwth-aachen.de

Danai Koutra
University of Michigan, Ann Arbor
dkoutra@umich.edu

Abstract

Graph Neural Network (GNN) research has highlighted a relationship between high homophily (i.e., the tendency of nodes of the same class to connect) and strong predictive performance in node classification. However, recent work has found the relationship to be more nuanced, demonstrating that simple GNNs can learn in certain heterophilous settings. To resolve these conflicting findings and align closer to real-world datasets, we go beyond the assumption of a global graph homophily level and study the performance of GNNs when the local homophily level of a node deviates from the global homophily level. Through theoretical and empirical analysis, we systematically demonstrate how shifts in local homophily can introduce performance degradation, leading to performance discrepancies across local homophily levels. We ground the practical implications of this work through granular analysis on five real-world datasets with varying global homophily levels, demonstrating that (a) GNNs can fail to generalize to test nodes that deviate from the global homophily of a graph, and (b) high local homophily does not necessarily confer high performance for a node. We further show that GNNs designed for globally heterophilous graphs can alleviate performance discrepancy by improving performance across local homophily levels, offering a new perspective on how these GNNs achieve stronger global performance.

1 Introduction

Deep learning with Graph Neural Networks (GNNs) has become common in many learning tasks over collaboration networks [1], social networks [2], financial networks [3], and more [4–6]. Yet, retrospectives on GNN performance are limited, leading to a poor understanding of the conditions that will cause a GNN's performance to degrade. Understanding when GNNs will fail promotes a proactive approach to GNN development, rectifying any issues that may arise once deployed to the public. One previously studied degradation mechanism of GNN performance is the presence of heterophilous connections [7–9]. Heterophily, or the tendency for nodes of different classes to connect, has been found in a variety of graph applications where sensitive factors influence the connective patterns, necessitating its study [10–13]. However, some recent works have argued that performance does not necessarily degrade with heterophily, and in fact simple GNN architectures, such as GCN, can perform well in certain settings [14–16]. These seemingly conflicting results indicate a gap in understanding, demanding further research on the influence of heterophily on GNNs.

To better understand the limitations of GNNs, we explore existing assumptions in the literature. Surprisingly, many assume that constituent nodes of a graph possess local homophily levels similar to the global homophily of the entire graph, disregarding the impact of a node's local homophily level on performance [14, 15, 17]. Among works that considers a local perspective, it is assumed that homophilous nodes perform better, biasing the interpretation of the results without consideration for other performance patterns [18, 19]. Assuming a constant local homophily hinders GNN development by obscuring whether new models improve performance across all nodes or certain node subsets. Furthermore, by myopically assuming higher local homophily indicates higher performance, potential discrepancies for nodes within graphs of varying global homophily levels has yet to be studied.

This work. We investigate how shifts in local homophily can impact GNNs, extending beyond current assumptions and aligning closer to real-world settings. Our analysis considers a GNN trained on nodes biased towards a graph's global homophily, and then applied to test nodes of varying local homophily levels. We *theoretically analyze* the scenario by obtaining a closed-form solution for a GNN's weight matrix and demonstrate, through perturbation analysis, that a GNN's performance can degrade when a node's local homophily level is shifted relative to the global homophily of the graph. We also demonstrate that when adopting aggregation mechanisms tailored for heterophilous graphs, predictions are less likely to degrade. We show that our theory generalizes to a variety of settings through a broad empirical analysis facilitated by our proposed *synthetic graph generator* that enables control over the local homophily levels of nodes. We also show the practical repercussions of our theoretical and empirical analyses on a representative set of five *real-world* datasets with varying global homophily levels. Across our experiments, we study nine different GNN architectures, demonstrating that those tailored to handle heterophily often maintain more uniform performance, minimizing discrepancies. Together, our analyses describe a new failure point of GNNs – an expected label distribution over a node's neighbors, stemming from an over-reliance on the global homophily of a graph – presenting a challenge for nodes with underrepresented homophily levels to be correctly predicted. While previous works have noted fairness issues in GNNs based on sensitive attributes, e.g. race or gender, determined exogenously [20, 21], our results point to a novel inequality rooted in a network's structure that could lead to unfairness in human-centric settings. Our contributions are:

- **Theoretical Analysis:** We show how a GNN's predictions change under a shift in local homophily level, providing intuition on how GNN performance can degrade for nodes with local homophily levels which differ from the global homophily level. We analyze this premise for specific homophilous and heterophilous designs, showing that heterophilous GNNs will generally be less susceptible to performance degradation as a node's local homophily level varies.

- **Synthetic Experiments and Model Comparison:** We perform empirical analysis by modifying the preferential attachment model to allow for more granular control over the distribution of local homophily levels. This capability facilitates empirical verification of our theory under more general graph structures and GNN architectures. Additionally, we perform the first node-level analysis that directly compares GNNs that assume homophily and GNNs that are adjusted for heterophily, demonstrating different levels of performance discrepancy across GNN designs.

- **Real-world Experiments:** We provide the first granular analysis of GNN performance as local homophily levels are varied across a set of five real-world datasets. We find that our theoretical performance degradation trends hold more generally, confirming GNNs designed for heterophily can aid in minimizing performance discrepancy across nodes with varying local homophily patterns.

2 Related Work

In this section, we begin by discussing GNN architectures designed to improve learning under heterophily. We then detail previous approaches towards local property and discrepancy analysis.

Learning GNNs in Diverse Neighborhoods. GNNs adopt an aggregation function to combine the ego-node's (the node being updated) features and the neighboring nodes' features. Depending on the neighborhood of a node, a particular aggregation mechanism may be insufficient to learn representations [8]. For example, GCN [22], GAT [23], and SGC [24] were built to learn over homophilous neighborhoods through their weighted average of the ego-node and neighboring nodes' features. To remedy this issue, models such as GraphSAGE [25], GPR-GNN [26], FA-GCN [27], GCNII [28], and LINKX [29] separate the ego and neighbor embeddings, either through a residual connection or concatenation. We note that LINKX is MLP-based, directly embedding the adjacency matrix through an MLP rather than traditional message passing. GPR-GNN and FA-GCN additionally

follow a predict-before-propagate paradigm to help alleviate the harm that can come from mixing representation learning and aggregation [30, 31], while GCNII utilizes identity mapping to mitigate oversmoothing, a known problem for homophily [8]. H2GCN adopts further decoupling across higher order neighborhoods, aggregating each k-hop neighborhood separately [17]. *In previous works, there is limited analysis demonstrating the impact of GNN architectures on the performance of nodes with varying local homophily. In this work, we provide the first granular analysis of these models, showing a new perspective on how they perform and their (in)ability to mitigate performance discrepancies.*

Local Property Analysis. Studies on GNN performance relative to an input graph's structural properties have gained traction; however, the adjustment to considering a per-node local perspective is still under-explored. For instance, many studies have argued the conditions in which a node is able to benefit from message passing with respect to homophily, but only consider a constant local homophily level for all nodes [8, 14, 15]. Du et al. offers the first local analysis, however the results are contradictory across datasets and are only performed for a single model [18]. More recent work has developed other homophily-inspired metrics to contextualize local performance, however the proposed metric can still fail to explain performance depending on the dataset [19]. Both works assume that higher local homophily should always improve performance, ultimately guiding their development of new architectures and metrics that reinforce this assumption. However, the conflicting results across datasets seen in both works indicate that this assumption may oversimplify the behavior of a GNN and fail to consider other drivers for performance degradation. Closely related to our work, Ma et al. analyzes the disparate treatment of individual nodes defined by their shortest path distance to the training dataset, showing a degradation in performance as distance increases [32]. We build upon the idea of structural property subgroup analysis, but instead consider variations in local homophily rather than distance to the training set, creating a shift in how performance is analyzed in the context of homophily. *Thus, we analyze the performance of GNNs, breaking the assumption that the local homophily levels are constant, and demonstrate how node predictions systematically degrade as the local homophily levels deviate from the global homophily of the training graph.*

3 Preliminaries

In this section, we provide key notations and definitions, the notation is summarized in App. A.1.

3.1 Graphs

Let $G = (V, E, \mathbf{X}, \mathbf{Y})$ denote a simple graph with node set V and edge set E, where $\mathbf{X} \in \mathbb{R}^{|V| \times f}$ represents the node feature matrix with f features per node and $\mathbf{Y} \in \{0, 1\}^{|V| \times c}$ represents the one-hot encoded node label matrix with c classes. A specific node $i \in G$ has feature vector $\mathbf{x_i}$, class $y_i \in \{1, ..., c\}$, and one-hot encoded class label vector $\mathbf{y_i}$. The edge set can also be represented as an adjacency matrix, $\mathbf{A} \in \{0, 1\}^{|V| \times |V|}$, where a value of 1 at index (i, j) denotes an edge between nodes i and j in G, otherwise the index is set to 0. We use both E and $\mathbf{A}$ throughout the paper, opting for E when explicitly discussing the edges of G and $\mathbf{A}$ when describing matrix computations on the edge set. A *k-hop neighborhood* of node $i \in V$, $N_k(i)$, denotes the subgraph induced by the nodes that are reachable within k-steps of i.

3.2 Node Classification with GNNs

We focus on node classification through a GNN, where the goal is to learn a mapping between $\mathbf{X}$ and $\mathbf{Y}$. This mapping is estimated through a subset of V, referred to as the set of training nodes n_{train}. For a k-layer GNN, learning is facilitated through message passing over k-hop neighborhoods of a graph. The steps, at a high level, include (1) embedding $\mathbf{X}$ through a non-linear transformation parameterized by a weight matrix $\mathbf{W}$ and (2) aggregating the embedded features across neighborhoods of each node. Message passing over all nodes in the graph can be computed through matrix multiplication, where the most basic formulation updates node representations through $\mathbf{R}_l = \sigma((\mathbf{A} + \mathbf{I})\mathbf{R}_{l-1}\mathbf{W}_l)$ for a layer $l \in \{1, 2, ..., k\}$ of the GNN, where $\mathbf{R}_0 = \mathbf{X}$ and σ is an activation function. The update is applied k times, resulting in final representations for each node that can be used for classification.

3.3 Homophily and Heterophily

We focus on edge homophily and present the following definitions to describe our analysis. We begin with the global homophily ratio of a graph, h, describing the overall homophily level in graphs; $h = 0$

indicates a fully heterophilous graph and $h = 1$ indicates a fully homophilous graph [11].

Definition 1 - Global Homophily Ratio. *The global homophily ratio h over a graph's edge set E is the fraction of edges in E that connect two nodes, u and v, with the same label, y_u and y_v:*

$$h = \frac{|\{(u,v) : (u,v) \in E \wedge y_u = y_v\}|}{|E|}. \tag{1}$$

Other global homophily metrics are discussed in Appendix A.3. Additionally, the empirical class compatibility matrix of a graph, $[\mathbf{H}_L]$ describes the probability of two nodes with certain labels connecting, where the (u,v)-th entry is the fraction of edges between a node in class u and a node in class v: $[\mathbf{H}_L]_{u,v} = \frac{|\{(i,j) : (i,j) \in E \wedge y_i = u \wedge y_j = v\}|}{|\{(i,j) : (i,j) \in E \wedge y_i = u\}|}$. However, both the global homophily ratio and compatibility matrix oversimplify the mixing patterns in a graph when there are varying neighborhood compositions. To perform more granular analysis on a per-node basis, we also define the local homophily ratio of a node t, h_t.

Definition 2 - Local Homophily Ratio. *The local homophily ratio of a node t, h_t, is the fraction of edges in the neighborhood of t that connect t to a neighbor u with the same class:*

$$h_t = \frac{|\{(u,t) : u \in N_1(t) \wedge y_u = y_t\}|}{|N_1(t)|}. \tag{2}$$

Given GNNs are often shallow and only depend on a small k-hop neighborhood for a single node prediction, it is natural to analyze GNNs through the local, rather than global, homophily ratio. Moreover, many real-world graphs display a wide range of local homophily ratios across the constituent nodes, as seen in App. A.5.2, necessitating local analysis.

4 Relationship between a Node's Local Homophily Level and Performance

In this section, we aim to characterize the impact of local homophily on the accuracy of node-level predictions by considering shifts in local homophily levels, at test time, relative to the global homophily level the GNN was trained on. We begin by revealing the drivers for performance discrepancies through theoretical analysis and discuss their implications on node-level performance. Leveraging these insights, we relax our assumptions in Section A.4.4 and show that our theory holds in more general settings via extensive empirical analysis on synthetic data. Additionally, we consider even more general real-world graphs (without any constraints) in Section 6.

Setup. Following previous theoretical GNN work [15, 17, 18, 33] and popular models such as SGC and LightGCN [24, 34], we study two different GNNs with different aggregation mechanisms. Our homophilous GNN, F, is formulated as $(\mathbf{A} + \mathbf{I})\mathbf{X}\mathbf{W}$, where $\mathbf{A} + \mathbf{I}$ is G's adjacency matrix with self-loops, directly mixing the features of the ego-node and neighbor nodes [22, 24, 34]. Our heterophilous GNN, F' is formulated as $(\mathbf{X} \parallel \mathbf{A}\mathbf{X})\mathbf{W}$, where $(\mathbf{X} \parallel \mathbf{A}\mathbf{X})$ is the concatenation of the ego-node features and aggregated neighboring features [17, 25, 29]. Both designs have yet to be analyzed through a localized perspective to characterize discrepancy across predictions.

Similar to the setup in [35], we consider a graph G with a subset of training nodes n_{train}, each of which has an associated node feature vector $\mathbf{x_i}$, one-hot encoded class label vector $\mathbf{y_i}$, 1-hop homophily ratio h, and degree d. For brevity, we focus on binary classification (though we consider multi-class settings in our experiments) and represent $\mathbf{y_i} = onehot(y_i) = \begin{bmatrix} 1 & 0 \end{bmatrix}$ when node i's class is $y_i = 0$ and $\begin{bmatrix} 0 & 1 \end{bmatrix}$ when $y_i = 1$. We consider node feature vectors from a uniform distribution and biased towards a particular class: when $y_i = 0, \mathbf{x_i} = \begin{bmatrix} (0.5 + p) & (0.5 - p) \end{bmatrix}$ and when $y_i = 1, \mathbf{x_i} = \begin{bmatrix} (0.5 - p) & (0.5 + p) \end{bmatrix}$, where parameter $p \in [0, 0.5]$ controls the 'agreement' between the node features and its class label, i.e. as p approach 0.5 the features become more similar to the class labels. The final prediction for node i is $\mathbf{argmax}\, \mathbf{z_i}$ where $\mathbf{z_i}$ is the output logit vector of the GNNs. We begin by solving for both GNN's optimal weight matrix $\mathbf{W}$, and then apply F and F' to a test node t. We specify the local homophily ratio for t as $h + \alpha_t = h_t$, where $\alpha_t \in [-h, 1 - h]$ is t's shift in local homophily level compared to the global homophily level. Under this setup, we analyze how t's prediction is impacted as its local homophily ratio h_t shifts relative to h, the global homophily ratio used to train F. In Theorem 1, without loss of generality, we consider the impacts of h_t when t's class label is $y_t = 0$ for GNN F. In Theorem 2, we consider the same setup for GNN F'.

Analysis & Implications. Our first theorem provides a *direct relationship between t's performance under a homophilous GNN and its local homophily ratio* as it deviates from the global homophily.

Theorem 1 *Consider a test node t with local homophily ratio $h + \alpha_t$, label $\mathbf{y_t} = \begin{bmatrix} 1 & 0 \end{bmatrix}$, and node features $\mathbf{x_t} = \begin{bmatrix} (0.5 + p) & (0.5 - p) \end{bmatrix}$. The class prediction from F for node t is a function of the global homophily level and the shift of the local homophily level, given by $\mathbf{argmax}\, \mathbf{z_t}$, where $\mathbf{z_t} = \mathbf{y_t} + b_{hom} \begin{bmatrix} \alpha_t & -\alpha_t \end{bmatrix}$ and $b_{hom} = d/(1 + d(2h - 1))$.*

Proof. The proof can be found in App. A.2.1. We provide additional analysis on a 2-layer variant of F, formulated as $(\mathbf{A} + \mathbf{I})^2 \mathbf{XW}$ in App. A.2.4.

Intuitively, this theorem implies that ***homophilous* GNNs are susceptible to performance discrepancies.** Specifically, we can expect the *performance to degrade* when a test node either becomes *more homophilous relative to a heterophilous graph* or *more heterophilous relative to a homophilous graph*. To further understand the implications of α_t, we analyze three settings that naturally arise for the global homophily: (1) $0 \leq h < 0.5$, (2) $h = 0.5$, and (3) $0.5 < h \leq 1$. We note that while the node degrees can influence the conditions that cause α_t to impact $\mathbf{z_t}$ (through b_{hom}), we show in App. A.2.1 that this mostly occurs for extremely low-degree nodes. Previous work corroborates our findings regarding the difficulty with low-degree nodes under heterophily [8, 17, 36]; however, our analysis extends this observation to demonstrate a significantly more complex interplay between node degree, global homophily, and shift in local homophily.

Setting 1: Heterophilous ($\mathbf{0 \leq h < 0.5}$): In this scenario, when $d(2h - 1) < -1$, $b_{hom} < 0$, leading to $\mathbf{z_t} = \mathbf{y_t} + |b_{hom}| \begin{bmatrix} -\alpha_t & \alpha_t \end{bmatrix}$, where b_{hom}'s sign has been distributed into the vector. Thus,

$$\mathbf{z_t} = \begin{cases} \mathbf{y_t} + |b_{hom}| \begin{bmatrix} |\alpha_t| & -|\alpha_t| \end{bmatrix}, & \text{if } h_t \leq h \\ \mathbf{y_t} + |b_{hom}| \begin{bmatrix} -\alpha_t & \alpha_t \end{bmatrix}, & \text{if } h_t > h, \end{cases} \tag{3}$$

where the sign of α_t has been integrated into the vectors. We can then deduce that (globally) heterophilous graphs, when $b_{hom} < 0$ is satisfied, **will cause F to degrade in performance as the test node's local homophily increases**, denoted by the score increase of the wrong class in the second case of Equation (3). Additionally, when local homophily decreases, the predictions will improve given the increase in score for the correct class of the first case in Equation (3), however as $h < 0.5$, α_t has a smaller possible range of values, minimizing the impact on F's predictions.

Setting 2: Mixed Homophily ($\mathbf{h = 0.5}$): When the graph is not strongly homophilous nor strongly heterophilous (i.e., $h = 0.5$), $b_{hom} = d$, leading to:

$$\mathbf{z_t} = \begin{cases} \mathbf{y_t} + d \begin{bmatrix} -|\alpha_t| & |\alpha_t| \end{bmatrix}, & \text{if } h_t \leq 0.5 \\ \mathbf{y_t} + d \begin{bmatrix} \alpha_t & -\alpha_t \end{bmatrix}, & \text{if } h_t > 0.5. \end{cases} \tag{4}$$

In this case, we find that F will have improved performance when the local homophily of a test node is increased. Conversely, decreased local homophily for a test node will decrease performance. This is the only case that agrees with previous work regarding high homophily as a direct indicator of performance. Notably, the prediction directly depends on d, potentially leading to performance variations that are dominated by degree, rather than local homophily.

Setting 3: Homophilous ($\mathbf{0.5 < h \leq 1}$): In this scenario, $b_{hom} > 0$, leading to:

$$\mathbf{z_t} = \begin{cases} \mathbf{y_t} + |b_{hom}| \begin{bmatrix} -|\alpha_t| & |\alpha_t| \end{bmatrix}, & \text{if } h_t \leq h \\ \mathbf{y_t} + |b_{hom}| \begin{bmatrix} \alpha_t & -\alpha_t \end{bmatrix}, & \text{if } h_t > h. \end{cases} \tag{5}$$

We can then deduce that (globally) homophilous graphs **will cause F to degrade in performance as a test node's local homophily decreases**, denoted by the score increase of the first case of Equation (5). When local homophily increases, the predictions will improve, however as $h > 0.5$, α_t has a smaller range of values, minimizing impact on the predictions.

Going beyond homophilous GNNs, Theorem 2 provides a *direct relationship between t's performance under a heterophilous GNN and its local homophily ratio* as it deviates from the global homophily ratio. At a high level, it implies that ***heterophilous* GNNs help alleviate performance discrepancies.**

Theorem 2 *Following the same setup from Theorem 1, the class prediction from F' for node t is a function of the global homophily level and the shift of the local homophily level, given by $\mathbf{argmax}\, \mathbf{z_t}$, where $\mathbf{z_t} = \mathbf{y_t} + b_{het} \begin{bmatrix} \alpha_t & -\alpha_t \end{bmatrix}$ and $b_{het} = d^2(2h - 1)/(1 + d^2(2h - 1)^2)$.*

Proof. The proof can be found in App. A.2.2.

Comparing to Theorem 1, we can *still expect performance degradation*, although at a different rate due to b_{het}. Our goal is to determine which of the two models produces the largest perturbations to the output logit vector, $\mathbf{z_t}$, introducing more significant discrepancies. In Figure 3 in App. A.2.2, we compare b_{het} and b_{hom}, the coefficients on the vector $[\alpha_t \quad -\alpha_t]$ for the heterophilous and homophilous GNNs, respectively. We find that the magnitude of b_{het} is often significantly smaller than the magnitude of b_{hom} when varying values of d and h. Thus, *we can expect* $[\alpha_t \quad -\alpha_t]$ *to have less impact on* $\mathbf{z_t}$ *for heterophilous models*, creating less discrepancy as compared to homophilous models. The full analysis comparing b_{hom} and b_{het} can be found in App. A.2.2.

5 Generalization of Theoretical Results via Synthetic Data Analysis

To display how the theoretical relationship between local homophily and classification performance generalizes, we introduce a graph generator that enables control over the local homophily ratios across a graph and conduct an extensive empirical analysis to study the following research questions: **(Q1)** What performance discrepancies arise across the range of local homophily values as the global homophily is varied? and **(Q2)** Do GNNs built specifically for heterophily display different discrepancy patterns across local homophily ranges as compared to homophilous architectures?

5.1 Synthetic Data Generation

Building on the preferential attachment model where a compatibility matrix governs edge likelihood [13, 17], we modify the generator to allow a node's homophily level to be either randomly assigned or defined by the compatibility matrix. We explain the generation process below, and provide the explicit steps in Algorithm 1, found in App. A.4.2. At a high level, the steps to add a node to a synthetic graph are: (1) Sample a class label, (2) Generate node features, and (3) Add connections based on assigned homophily level. We provide related work for graph generation in App. A.4.1, and property analysis in App. A.4.3.

Class and Feature Generation. For a node i, label y_i is sampled from probability distribution $P(\{0, ..., c\})$ with c possible classes. Features $\mathbf{x_i}$ are generated from a 2D Gaussian, where each dimension has a mean of ϵy_i and standard deviation of 1, where $\epsilon \in [0, 1]$ introduces noise into the features. When $\epsilon = 1$, the label y_i is explicitly encoded, when $\epsilon = 0$, y_i is unrecoverable.

Structure Generation. We define a class compatibility matrix $\mathbf{H}_L$ with diagonal elements h, denoting the probability of connecting nodes with similar classes (homophilous), and off-diagonals elements $(1-h)/c$, denoting the probability of connecting nodes with different classes (heterophilous) [13, 17]. During generation, a new node u is attached to an existing node v as $P((u, v) \in E) \propto [\mathbf{H}_L]_{y_u, y_v}$. To control the local homophily ratios, we introduce a uniformity parameter ρ such that with probability ρ, a node i's local homophily ratio h_i is sampled at random from a uniform distribution, $U(0, 1)$. As ρ increases, more local homophily ratios follow the uniform distribution, rather than the compatibility matrix. Since the preferential attachment model adds nodes sequentially, it is possible that the local homophily of nodes early in the generation process drift from their original values. For example, if a node i is initially generated with two homophilous connections and three heterophilous connections ($h_i = 0.4$), later nodes attached to i can result in a final $h_i \neq 0.4$. To correct this, we keep track of how far a node i has drifted from its original h_i through $drift_i$, and prioritize connections to high-drift nodes (i.e., a node $drift_i > \delta$, where δ is a hyperparameter that defines the drift threshold) that would return the node's local homophily ratio back to its original value.

5.2 Synthetic Evaluation Setup

We begin by detailing the GNNs and the synthetic graph generation process used in our experiments.

GNN Models. For our experiments, we consider a diverse set of GNN models including SGC [24], GCN [22], GAT [23], GraphSAGE [27], GCNII [28], H2GCN [17], GPR-GNN [26], FA-GCN [27], and LINKX [29]. SGC, GCN and GAT act as homophilous baselines, while GCNII, H2GCN, GPR-GNN, FA-GCN, and LINKX adopt mechanisms to improve learning in heterophilous settings. While GraphSAGE was not built for heterophily, later works [17] have noted its design can improve learning over heterophily; we include it to represent a midpoint between the models. We also include results for a graph-agnostic MLP. Experiments across SGC [24], GPR-GNN [26], and FA-GCN [27] can be
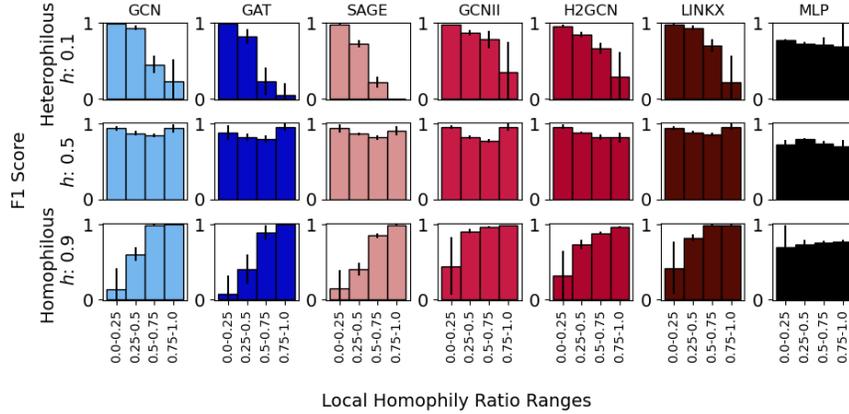
Figure 1: Performance for different GNN models (columns) on synthetic data generated with global homophily ratios $h \in \{0.1, 0.5, 0.9\}$ (rows) and uniformity $\rho = 0.5$, reported for different local homophily ranges. As the local homophily deviates from the global homophily, homophilous models (blue bars) fail to generalize, resulting in performance discrepancy, while heterophilous models (red bars) retain higher relative performance by up to 0.5 F1-score, alleviating discrepancy. Exact differences can be found in Figure 6. MLP achieves constant performance.

found in Appendix A.4.4. All models are hyperparameter tuned (parameters detailed in App A.6) and the model which maximizes performance on the validation set is applied to the test set.

Data and Evaluation. Using the proposed generator, we generate graphs by varying the global homophily ratio $h \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ with $\rho = 0.5$, $\epsilon = 0.5$, and $\delta = 5$, allowing us to study GNNs while jointly varying global and local homophily levels. An additional study for $\rho = 0.75$ is provided in Figure 9 of App. A.4.4, demonstrating how discrepancy can be alleviated when the entire range of local homophily ratios has a sufficient number of nodes. For each combination of h and ρ, we generate 10 graph with 5000 nodes and 100k edges (i.e. n = 5000, m = 20), and split the nodes into a 50-25-25% split for train, validation, and test. Additional discussion and experiments on varying the training set size are provided in Appendix A.4.4. To match our theoretical analysis, we focus on binary classification. For evaluation, we compare the performance across models and global homophily ratios as the local homophily ratio is varied. To compute localized performance, we split the test nodes into four groups based on their local homophily ratios and calculate an F1 score per group. We choose F1 score as the node subsets can become imbalanced with respect to their class after splitting. Per bin, we report the average and standard deviation for F1 over the 10 generated graphs. Results are presented in Figures 1 of the main text, and Figures 7, 8 in App. A.4.4

5.3 Synthetic Results

(Q1) Performance discrepancies across local and global homophily ratios. In Figure 1, we visualize the performance of each model by binning nodes of similar homophily for each of the global homophily ratios (analysis on $h \in \{0.3, 0.7\}$ and additional GNNs in App A.4.4). We first find that the MLP, as expected, performs significantly worse with an F1 score of 0.75 as compared to the GNN's which achieve an F1 score of 0.9 and higher, highlighting the positive contribution of the graph structure on predictions. Next, we highlight the clear trend between performance and local homophily depending on the global homophily ratio: when $h < 0.5$ (lower global homophily), performance often degrades as local homophily increases, while when $h > 0.5$ (higher global homophily), performance often degrades as local homophily decreases. The results align well with our theoretical analysis, showing strong generalization of our findings to more complex graph structures and GNN architectures. Together, **our theoretical and empirical results indicate that assuming high homophily always indicates high performance may oversimplify the GNN's behavior, leading retrospective analyses astray when diagnosing performance degradation.**

(Q2) Performance discrepancies for homophilous vs. heterophilous GNNs. To understand how different GNN designs amplify or reduce discrepancy, we analyze the trend of local performances across models. While all models achieve similar global performance, we observe that the models perform differently depending on the local homophily range. As shown in Figure 1 (and Figure 6 in App. A.4.4), homophilous models often have higher performance for test nodes with local homophily

levels that are close to the global homophily, as expected from Theorem 1, whereas heterophily-based models perform better than homophilous models for nodes with local homophily levels far from the global homophily, agreeing with Theorem 2. These insights highlight the different behaviors of the two designs, indicating that **models designed for heterophily are able to alleviate performance discrepancies across nodes, while homophilous models exacerbate discrepancy in strongly homophilous/heterophilous graphs**, Interestingly, LINKX, despite being MLP based, is competitive as compared to the other heterophilous GNNs in minimizing discrepancy. We attribute this to the separation of ego-node and neighbor-node embeddings, a key design in the other heterophilous models [17, 37]. These results verify that heterophilous models offer a better performance trade-off between nodes with over- and under-represented local homophily levels, displaying minor degradation on the over-represented nodes and significant improvement on the under-represented nodes.

6 Real-world Empirical Evaluation

We now demonstrate how our results extend to real-world datasets.

Data and Setup. We choose five real-world graphs: two homophilous (Cora [38], Coauthor-CS [39]) and three heterophilous (Wisconsin [7], Squirrel [40], and Arxiv-Year [29]). Each heterophilous dataset varies greatly in size, providing an opportunity to measure the impact of the number of nodes and density on discrepancy. Additionally, we choose Wisconsin and Squirrel due to their historically inconclusive performance when comparing GNNs to non-graph-based deep learning baselines [14, 17]. Our analysis aims to provide insight into why prior results have been inconclusive and explain their poor performance from a local perspective. For Wisconsin and Squirrel, we perform 30 random 50-25-25% splits of the nodes to obtain the train, validation, and test sets. For Arxiv-Year, Cora, and Coauthor-CS we perform 5 random splits over the same ratio as the graph are much larger. We train the same architectures as in the synthetic experiments, seen in Section 5.2.

We report results in Figure 2 of the main text, and Figure 11 in App. A.5.1. While we group all test nodes into four local homophily ranges, we limit the ranges of Wisconsin in Figure 2 due to having less than three test nodes in local homophily ranges above 0.6. Additionally, features may be more informative in certain local homophily ranges, obscuring when performance degrades due to homophily or uninformative features. Thus, we report the difference in F1 score between the different GNNs and the graph-agnostic MLP, ΔF1 $\equiv$ (F1-score GNN - F1-score MLP), to disentangle how performance relies on the node features as compared to the graph structure.

6.1 Performance Discrepancies in Homophilous vs. Heterophilous Real-World Datasets

Homophilous Graphs. The homophilous graphs are shown in the two right-most plots of Figure 2. First, we highlight the nearly **0.6 drop in ΔF1 across both datasets for test nodes with local homophily ratios far from the global homophily ratio**, performing worse than an MLP as denoted by the negative F1 score difference. Furthermore, this degradation is consistent across all of the GNN architectures, with H2GCN, which leverages heterophilous designs, maintaining generally higher performance as the nodes become more heterophilous. This result demonstrates the practical implications of our theoretical analysis, allowing us to additionally demonstrate that degradation can occur under rich feature sets. Notably, the MLP outperforms all of the GNN architectures in the heterophilous parts of the graph, implying the neighborhood information is actively corrupting performance in these regions due to the GNN's reliance on the global homophily.

Heterophilous Graphs. The heterophilous datasets are shown on the three left-most plots of Figure 2. Compared to the homophilous datasets, Wisconsin and Squirrel have poor performance relative to the MLP for nearly all local homophily ranges. Our local analysis identifies a mechanism for *how* this arises: exacerbated performance degradation for nodes with local homophily ratios far from the global homophily ratio. Despite the MLP model outperforming the GNN models across Wisconsin and Squirrel, there is still a notable degradation in performance as the local homophily ranges increase, **causing a 0.2-0.4 drop in ΔF1 between the bins furthest from the global homophily ratio**. We hypothesize that degree can influence whether GNNs degrade in performance on heterophilous graphs, as seen in our theoretical analysis in App. A.2.1, causing this drop to be lower as compared to the homophilous graphs. Aligning closer to the homophilous datasets, each model's best performance on Arxiv-Year is within the bin closest to the global homophily ratio, demonstrating a scenario where the heterophilous neighborhood is beneficial to learning. Following the trends seen in the synthetic
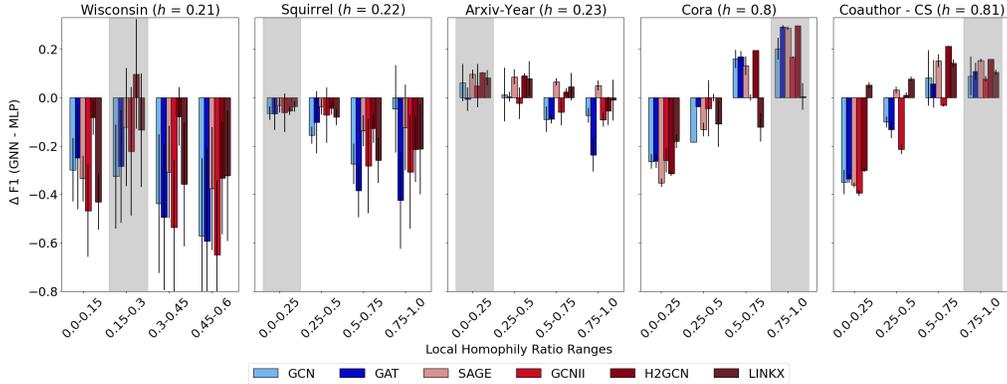
Figure 2: Real-words graphs: Difference in performance, ΔF1=F1(GNN) - F1(MLP), for GNN models across ranges of local homophily ratios (more GNNs in App. A.5.1), averaged over multiple splits. Our results elucidate *how* models achieve different global performance, where heterophilous models (bars in reds) better tend to combat the systematic performance discrepancy seen in homophilous models (bars in blues). Gray indicates the range that the global homophily ratio falls in; negative bars indicate worse performance than MLP.

data, the heterophilous models also display higher uniformity across the homophily levels for each datasets. Surprisingly, for Squirrel, we observe a significant difference in MLP performance across the various local homophily ranges, indicating that the features are intrinsically more informative in certain regions. When accounting for this through ΔF1, we see similar degradation to that in the synthetic and homophilous datasets, again with the heterophilous GNNs tending to perform best.

6.2 Discussion

Two key insights emerge that have yet to be established in other analyses: (a) nodes with higher local homophily are not inherently easier to classify, as seen by the relative drops in performance on heterophilous datasets, and (b) heterophilous designs improve learning across nearly all local homophily ranges, not just one particular range, alleviating performance discrepancies across nodes. We note that there might be additional factors which give rise to performance variations across local homophily ratios. For instance, previous works have identified that, under certain settings, nodes with extreme homophily levels can be easier to classify [14, 18, 19]. Moreover, the ease of classification has been tied to the interaction of degree and homophily, pointing towards nodes with high degree and homophily as the easiest [8]. Performance variations due to raw structural properties (e.g. degree or local homophily ratio) and local homophily shift are not necessarily mutually exclusive, nor do they conflict with our results. Instead, we hypothesize both factors can interact, amplifying performance discrepancies across homophily ranges, further necessitating work which studies their interplay.

7 Conclusion

In this work, we take a local perspective focused on discerning how nodes of varying local homophily levels can experience performance discrepancies. We first theoretically demonstrated that classification performance degrades as the local homophily ratio of a node deviates from the global homophily ratio. To demonstrate the generalizability of our findings, we proposed a new parsimonious synthetic graph generator that enabled the generation of graphs with varying global and local homophily. We demonstrated that our theoretical insights still hold in more general settings, finding that performance degradation can occur in either highly homophilous or heterophilous settings. Furthermore, we showed that this discrepancy in performance can be reduced by using GNN models which adopt explicit mechanisms to support heterophily. Our experiments on real-world datasets of varying global homophily ratios confirm the practical implications of our insights, exhibiting similar disparity patterns. The discovery and characterization of GNN degradation through shifts in local homophily relative to a graph's global homophily necessitates the development of new GNNs that are able to explicitly handle such data shifts. Additionally, our findings highlight how, in human-facing applications of GNNs, individuals might experience disparate treatment under a GNN due to structural properties of the underlying graph, opening new research directions in algorithmic fairness.

8 Acknowledgments

References

[1] Andrew Mccallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2), 11 2000. 1

[2] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The World Wide Web Conference*, page 417–426, New York, NY, USA, 2019. Association for Computing Machinery. 1

[3] Daixin Wang, Zhiqiang Zhang, Jun Zhou, Peng Cui, Jingli Fang, Quanhui Jia, Yanming Fang, and Yuan Qi. Temporal-aware graph neural network for credit risk prediction. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 702–710. 1

[4] Valerio Ciotti, Moreno Bonaventura, Vincenzo Nicosia, Pietro Panzarasa, and Vito Latora. Homophily and missing links in citation networks. *EPJ Data Science*, 5, 11 2015. 1

[5] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. *In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

[6] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BylA_C4tPr. 1

[7] 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*, 2020. 1, 8, 26

[8] 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 *2022 IEEE International Conference on Data Mining (ICDM)*, pages 1287–1292, 2022. 2, 3, 5, 9

[9] Jiong Zhu, Ryan A. Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K. Ahmed, and Danai Koutra. Graph neural networks with heterophily. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021. 1

[10] Bas Hofstra, Rense Corten, Frank Van Tubergen, and Nicole B Ellison. Sources of segregation in social networks: A novel approach using facebook. *American Sociological Review*, 82(3): 625–656, 2017. 1

[11] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*. 4

[12] Jorge Brea, Javier Burroni, and Carlos Sarraute. Inference of users demographic attributes based on homophily in communication networks. *NetMob*, 2018.

[13] Fariba Karimi, Mathieu Génois, Claudia Wagner, Philipp Singer, and Markus Strohmaier. Homophily influences ranking of minorities in social networks. *Scientific reports*, 2018. 1, 6, 20, 21

[14] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks? In *International Conference on Learning Representations*, 2022. 1, 2, 3, 8, 9

[15] Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Is heterophily a real nightmare for graph neural networks to do node classification? *arXiv*, 2021. 2, 3, 4

[16] Jiong Zhu, Yujun Yan, Mark Heimann, Lingxiao Zhao, Leman Akoglu, and Danai Koutra. Heterophily and graph neural networks: Past, present and future. *IEEE Data Engineering Bulletin*. URL https://par.nsf.gov/biblio/10435541. 1

[17] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 3, 4, 5, 6, 8, 15, 21, 26

[18] Lun Du, Xiaozhou Shi, Qiang Fu, Xiaojun Ma, Hengyu Liu, Shi Han, and Dongmei Zhang. Gbk-gnn: Gated bi-kernel graph neural networks for modeling both homophily and heterophily. In *Proceedings of the ACM Web Conference 2022*, pages 1550–1558, 2022. 2, 3, 4, 9

[19] Andrea Cavallo, Claas Grohnfeldt, Michele Russo, Giulio Lovisotto, and Luca Vassio. 2-hop neighbor class similarity (2ncs): A graph structural metric indicative of graph neural network performance. *3rd Workshop on Graphs and more Complex structures for Learning and Reasoning (GCLR) at AAAI*, 2023. 2, 3, 9, 20

[20] Yushun Dong, Jing Ma, Chen Chen, and Jundong Li. Fairness in graph mining: A survey. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2022. 2

[21] Wenbin Zhang, Jeremy C Weiss, Shuigeng Zhou, and Toby Walsh. Fairness amidst non-iid graph data: A literature review. *arXiv preprint arXiv:2202.07170*, 2022. 2

[22] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. 2, 4, 6, 26

[23] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. 2, 6

[24] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019. 2, 4, 6

[25] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017. 2, 4, 15

[26] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*, 2021. 2, 6

[27] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *In AAAI*, 2021. 2, 6

[28] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, 2020. 2, 6

[29] Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. *Advances in Neural Information Processing Systems*, 34:20887–20902, 2021. 2, 4, 6, 8, 20

[30] Ganqu Cui, Jie Zhou, Cheng Yang, and Zhiyuan Liu. Adaptive graph encoder for attributed graph embedding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2020. 3

[31] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations*, 2019. 3

[32] Jiaqi Ma, Junwei Deng, and Qiaozhu Mei. Subgroup generalization and fairness of graph neural networks. In *Advances in Neural Information Processing Systems*, 2021. 3

[33] Yu Wang, Yuying Zhao, Yushun Dong, Huiyuan Chen, Jundong Li, and Tyler Derr. Improving fairness in graph neural networks via mitigating sensitive attribute leakage. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 1938–1948. Association for Computing Machinery, 2022. 4

[34] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 639–648. Association for Computing Machinery, 2020. 4

[35] Jiong Zhu, Junchen Jin, Donald Loveland, Michael T Schaub, and Danai Koutra. On the relationship between heterophily and robustness of graph neural networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022. 4

[36] Bei Lin, You Li, Ning Gui, Zhuopeng Xu, and Zhiwu Yu. Multi-view graph representation learning beyond homophily. *ACM Trans. Knowl. Discov. Data*, 17(8), jun 2023. ISSN 1556-4681. URL https://doi.org/10.1145/3592858. 5

[37] 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? *International Conference on Learning Representations*, 2023. 8, 26

[38] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93, Sep. 2008. doi: 10.1609/aimag.v29i3.2157. URL https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2157. 8

[39] Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *International Conference on Learning Representations*, 2017. 8

[40] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2), 2021. 8

[41] John Palowitch, Anton Tsitsulin, Brandon Mayer, and Bryan Perozzi. Graphworld: Fake graphs bring real insights for gnns. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 3691–3701, 2022. 20

[42] Chaokun Wang, Binbin Wang, Bingyang Huang, Shaoxu Song, and Zai Li. Fastsgg: Efficient social graph generation using a degree distribution generation model. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 564–575, 2021. 20

[43] Seiji Maekawa, Yuya Sasaki, George Fletcher, and Makoto Onizuka. Gencat: Generating attributed graphs with controlled relationships between classes, attributes, and topology. *Information Systems*, 115:102195, 2023. 20

[44] Albert-Laszlo Barabasi and Reka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, Oct. 1999. 21

[45] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in Neural Information Processing Systems*, 33:22118–22133, 2020. 26

A   Appendix

A.1   Notation Table

In this section, we give a summary of the notation used throughout the main text.

Table 1: Graph- and Node-level Notations and Definitions

Notation	Definitions
G	Graph G
V	Node set V for G
E	Edge set E for G
$\mathbf{I}$	Identity matrix
$\mathbf{A}$	Adjacency matrix for G. Shape $|V| \times |V|$
$\mathbf{X}$	Feature matrix for G. Shape $|V| \times f$ where f is the number of features.
$\mathbf{Y}$	One-hot encoded class matrix for G. Shape $|V| \times c$ where c is the number of classes.
$\mathbf{H}_L$	Class compatibility matrix..
h	Global homophily ratio.
i	A node $i \in G$
$\mathbf{x_i}$	Feature vector for node i
y_i	Class label for node i
$\mathbf{y_i}$	One-hot encoded class label vector for node i
$N_k(i)$	k-hop neighborhood for a node i
h_i	Local homophily ratio for a node i
α_i	Shift in local homophily ratio for a node i, relative to the global homophily ratio of its graph, $h + \alpha_i = h_i$.

Table 2: Graph Neural Network Notations and Definitions

Notation	Definitions
n_{train}	Training nodes of a GNN
$\mathbf{W_l}$	The weight matrix for a layer l of a GNN
$\mathbf{R_l}$	The hidden representations for nodes at layer l of a GNN. R_0 denotes the input features.
$\mathbf{r_i^l}$	The hidden representation vector for a node i at layer l
σ	Activation function of a GNN.
$\mathbf{z_i}$	Output logit vector of a GNN for node i.

Table 3: Synthetic Graph/Evaluation Notations and Definitions

Notation	Definitions
n	Number of nodes
m	Number of edges added per generation step
ϵ	Noise strength for generated features.
ρ	Probability to sample a new homophily ratio.
δ	Hyperparameter to determine if a node has drifted in homophily level.
$\Delta F1$	F1 Score of GNN - F1 Score of MLP (graph agnostic).

A.2   Theoretical Relationship between a Node's Local Homophily Level and Performance

In this section, we provide the proof for Theorem 1 in the main text. We additionally perform theoretical analysis on the influence of degree on Theorem 1, as well as the impact of higher order aggregation. We provide a high-level takeaway for each additional analysis.

A.2.1 Proof for Theorem 1: Homophilous Model $(\mathbf{A} + \mathbf{I})\mathbf{XW}$

As described in the main text, we consider a graph G with a subset of nodes n_{train}. Each node $i \in n_{train}$ has an associated node feature vector $\mathbf{x_i}$, one-hot encoded class label vector $\mathbf{y_i}$, 1-hop homophily ratio h, and degree d. For brevity, we focus on binary classification, and represent $\mathbf{y_i}$ as $[1 \quad 0]$ when $y = 0$ and $[0 \quad 1]$ when $y = 1$. The node feature vectors are initialized as $\mathbf{x_i} = [(0.5 + p) \quad (0.5 - p)]$ when $y_i = 0$ and $\mathbf{x_i} = [(0.5 - p) \quad (0.5 + p)]$ when $y_i = 1$. The GNN, F, is formulated as $(\mathbf{A} + \mathbf{I})\mathbf{XW}$, where $\mathbf{A} + \mathbf{I}$ is G's adjacency matrix with self-loops and $\mathbf{W}$ is F's weight matrix, and trained through n_{train}. The final prediction for a node i is $\mathbf{argmax}\, \mathbf{z_i}$ where $\mathbf{z_i}$ is the output logit vector of F. Focusing first on the $(\mathbf{A} + \mathbf{I})\mathbf{X}$ term, a row (node) i in this matrix with a class 0 will be equal to:

$$y_i = 0 : \mathbf{r_i^1} = \mathbf{x_i} + \frac{hd}{2}[1+p \quad 1-p] + \frac{(1-h)d}{2}[1-p \quad 1+p] \tag{6}$$

similarly, a row (node) i in this matrix of class 1 will be equal to:

$$y_i = 1 : \mathbf{r_i^1} = \mathbf{x_i} + \frac{hd}{2}[1-p \quad 1+p] + \frac{(1-h)d}{2}[1+p \quad 1-p] \tag{7}$$

The computations for each $\mathbf{r_i^1}$ represent the sum of the node vector for node i, the feature vectors from the homophilous connections of node i, and the feature vectors of the heterophilous connections for node i. Condensing these terms, the simplified expressions for each $\mathbf{r_i^1}$ is shown in Equations 8 and 9

$$y_i = 0 : \mathbf{r_i^1} = \frac{1}{2}[(1+d+p+dp(2h-1)) \quad (1-p+d(1+p-2hp))] \tag{8}$$

$$y_i = 1 : \mathbf{r_i^1} = \frac{1}{2}[(1-p+d(1+p-2hp)) \quad (1+d+p+dp(2h-1))] \tag{9}$$

We can then represent $(\mathbf{A+I})\mathbf{XW}$ as $\mathbf{RW}$ as seen in Equation 10 and set it equal to the label matrix $\mathbf{Y}$ after a transformation by the weight matrix $\mathbf{W}$.

$$\mathbf{R} = \frac{1}{2}\begin{bmatrix} \vdots & \vdots \\ (1+d+p+dp(2h-1)) & (1-p+d(1+p-2hp)) \\ \vdots & \vdots \\ (1-p+d(1+p-2hp)) & (1+d+p+dp(2h-1)) \\ \vdots & \vdots \end{bmatrix}_{|T| \times 2} \tag{10}$$

$$\mathbf{R}_{|T| \times 2}\mathbf{W}_{2 \times 2} = \mathbf{Y}_{|T| \times 2} = \begin{bmatrix} \vdots & \vdots \\ 1 & 0 \\ \vdots & \vdots \\ 0 & 1 \\ \vdots & \vdots \end{bmatrix} \tag{11}$$

Our goal is to now solve for the optimal $\mathbf{W}$ in this system of equations. While the system of equations is overdetermined, each row of the same class share the same solution. Thus, we can sample the unique data points, leaving us with the optimal $\mathbf{W}$ in Equation 12 from solving the reduced system of equations, where $c_1 = \dfrac{1}{2(p - d^2p + 2dhp + 2d^2hp)}$.

$$\mathbf{W} = c_1 \begin{bmatrix} 1+d+p+dp(2h-1) & -1+p-d(1+p-2hp) \\ -1+p-d(1+p-2hp) & 1+d+p+dp(2h-1) \end{bmatrix} \tag{12}$$

We now consider a new test point, t with label $y_t = 0$ and subsequently features $\mathbf{x_t} = \frac{1}{2}[(1+p) \quad (1-p)]$. In addition, we assume that t has a homophily ratio $h + \alpha_t = h_t \in [0,1]$ where α_t represents a shift away from h. When α_t is positive, we can interpret this as a test node with a higher homophily ratio, and when α_t is negative, a lower homophily ratio. Under this new homophily ratio, $\mathbf{r_t^1}$ can be calculated as:

$$\mathbf{r_t^1} = \frac{h_t d + 1}{2}[1 + p \quad 1 - p] + \frac{(1 - h_t)d}{2}[1 - p \quad 1 + p] \tag{13}$$

Similar to equation 8, the simplified expressions for $\mathbf{r_t^1}$ is given as:

$$\mathbf{r_t^1} = \frac{1}{2}[(1 + d + p + dp(2h_t - 1)) \quad (1 - p + d(1 + p - 2h_t p))] \tag{14}$$

We can now compute $\mathbf{r_t^1 W}$ and analyze the associated predictions. The output of $\mathbf{r_t^1 W}$ is:

$$\mathbf{z_t} = \frac{1}{1 + d(2h - 1)}[(1 + d(h + h_t - 1)) \quad d(h - h_t)] \tag{15}$$

It is easy to check that when $h_t = h$, we recover the correct prediction of $[1, 0]$ as expected. We now consider what happens when $h_t = h + \alpha_t$ where $\alpha_t \in [-h, 1 - h]$. Since we are interested in understanding how the predictions would change as a byproduct of α_t, we look at the difference between predictions when $h_t = h$. Below we compute $\Delta \mathbf{z_t}$, the change in the logit vector output for a test point with local homophily h_t, where $b_{hom} = \frac{d}{1 + d(2h - 1)}$.

$$\Delta \mathbf{z_t} = \frac{b_{hom}}{d}[(1 + d(h + (h + \alpha_t) - 1)) \quad d(h - (h + \alpha_t))]$$
$$- \frac{b_{hom}}{d}[(1 + d(h + h - 1)) \quad d(h - h)] \tag{16}$$
$$\Delta \mathbf{z_t} = b_{hom}[\alpha_t \quad -\alpha_t] \quad \square \tag{17}$$

A.2.2 Proof for Theorem 2: Heterophilous Model $(\mathbf{X} \parallel \mathbf{AX})\mathbf{W}$

Following the same set up previously described for the GNN parameterized as $(\mathbf{A} + \mathbf{I})\mathbf{XW}$, we also consider a GNN, F', parameterized as $(\mathbf{X} \parallel \mathbf{AX})\mathbf{W}$, concatenating the features from the ego-node and the aggregated neighboring features. This separation through concatenation is found in a variety of GNNs, and is often denoted as a core design to promote heterophilous learning [17, 25].

Focusing first on the aggregation step of F', $\mathbf{AX}$, a node i of class 0 will have aggregated features from its neighbors, $\mathbf{r_i^1}$, defined as:

$$y_i = 0 : \mathbf{r_i^1} = \frac{hd}{2}[1 + p \quad 1 - p] + \frac{(1 - h)d}{2}[1 - p \quad 1 + p] \tag{18}$$

similarly, a node i of class 1 will have aggregated features from its neighbors:

$$y_i = 1 : \mathbf{r_i^1} = \frac{hd}{2}[1 - p \quad 1 + p] + \frac{(1 - h)d}{2}[1 + p \quad 1 - p] \tag{19}$$

Then, the concatenated vectors $(\mathbf{X} \parallel \mathbf{AX})\mathbf{W}$ for nodes of class 0 and 1 are expressed as:

$$y_i = 0 : \mathbf{x_i} \parallel \mathbf{r_i^1} = \frac{1}{2}[1 + p \quad 1 - p \quad d + dp(2h - 1) \quad d(1 + p - 2hp)] \tag{20}$$

$$y_i = 1 : \mathbf{x_i} \parallel \mathbf{r_i^1} = \frac{1}{2}[1 - p \quad 1 + p \quad d(1 + p - 2hp) \quad d + dp(2h - 1)] \tag{21}$$

We refer to the collection of concatenated vectors as $\mathbf{R} \in \mathcal{R}^{|V| \times 4}$. Then, the goal, similar to the case of the GNN F, is to solve for W such that:

$$\mathbf{R}_{|T| \times 4} \mathbf{W}_{4 \times 2} = \mathbf{Y}_{|T| \times 2} = \begin{bmatrix} \vdots & \vdots \\ 1 & 0 \\ \vdots & \vdots \\ 0 & 1 \\ \vdots & \vdots \end{bmatrix} \tag{22}$$

Following a similar procedure to the analysis of F, we reduce the system of equations by sampling the unique points and solve for the optimal $\mathbf{W}$. However, unlike in F, $\mathbf{R}$ is not a square matrix after sampling, and thus we use a right pseudo-inverse through $\mathbf{R}^{-1} = \mathbf{R}^{\mathbf{T}}(\mathbf{R}\mathbf{R}^{\mathbf{T}})^{-1}$ to attain a solution to the system of equations.

We now consider a new test point, t with label $y_t = 0$, similar to the set up for F. Then:

$$\mathbf{x_t} \parallel \mathbf{r_t^1} = \frac{1}{2} \begin{bmatrix} 1+p & 1-p & d+dp(2h_t-1) & d(1+p-2h_tp) \end{bmatrix} \tag{23}$$

We can now compute $\mathbf{r_t^1}\mathbf{W} = \mathbf{z_t}$:

$$\mathbf{z_t} = \frac{1}{1+d^2(2h-1)^2} \begin{bmatrix} 1+d^2(2h-1)(h+h_t-1)) & (d^2(2h-1)(h-h_t)) \end{bmatrix} \tag{24}$$

As before, when $h_t = h$, it is easy to see that the label is recovered. We now consider when $h_t = h + \alpha_t$ and look at the difference in predictions relative to when $h_t = h$. $\Delta\mathbf{z_t}$ for a test point with local homophily h_t predicted by the heterophilous model, where $b_{het} = \dfrac{d^2(2h-1)}{1+d^2(2h-1)^2}$ is:

$$\Delta\mathbf{z_t} = b_{het} \begin{bmatrix} \alpha_t & -\alpha_t \end{bmatrix} \quad \square \tag{25}$$

Thus, we can still expect heterophilous models which adopt the ego-neighbor separation design to degrade in performance as the local homophily of a node changes, just as studied in the Section 4 of the main text. However, the change in coefficient, b_{hom} vs. b_{het}, demonstrates a significantly different degradation pattern.

The impact of b_{hom} and b_{het}. We analyze how the different coefficients influence the degradation of performance. To do this, we compare the difference in magnitude between b_{het} and b_{hom}, where a positive difference indicates a smaller magnitude for b_{hom} (indicating that b_{het} has a stronger influence on $\Delta\mathbf{z_t}$ as compared to b_{hom}), while a negative difference indicates a smaller magnitude for b_{het} (indicating that b_{hom} has a stronger influence on $\Delta\mathbf{z_t}$ as compared to b_{het}). As the goal is to identify which GNN formulation is likely to produce discrepancies, we prefer models with smaller magnitudes given this will minimize the influence of α_t.
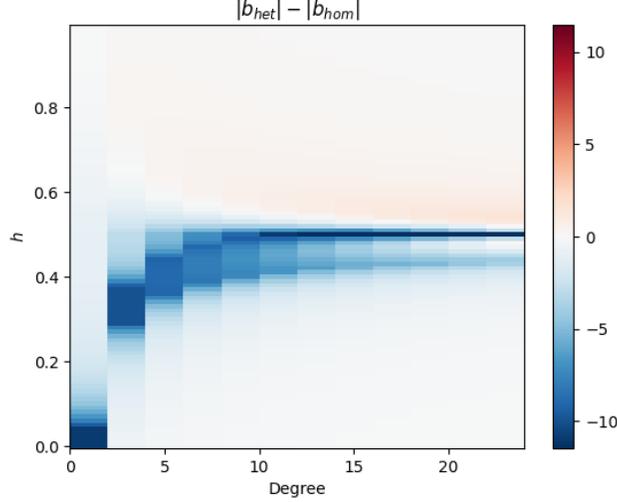
The comparison of b_{het} and b_{hom} is performed in Figure 3, where we sample points for the degree from $\{1, 5, 10, 15, 20, 25\}$ and h from 0 to 1 in intervals of 0.001. We can see that the difference in the coefficients is predominantly negative, indicating that the GNN F', formulated as $(\mathbf{X} \parallel \mathbf{AX})\mathbf{W}$, is less impacted by shifts in local homophily. Thus, although both the homophilous and heterophilous formulations are impacted by discrepancies induced by α_t, we can expect the predictions of F' to experience less discrepancy as the magnitude of α_t becomes larger.

A.2.3 The Impact of Degree on $\Delta\mathbf{z_t}$ on Homophilous Models (AXW)

As in the main text, we provide an outline of the cases below, with commentary on the impact of degree d on the cases.

Setting 1: Heterophilous ($0 \leq h < 0.5$): In this scenario, when $d(2h-1) < -1$, $b_{hom} < 0$, leading to $\mathbf{z_t} = \mathbf{y_{t,0}} + |b_{hom}| \begin{bmatrix} -\alpha_t & \alpha_t \end{bmatrix}$ where the sign of b_{hom} has been distributed into the vector. There are then two cases:

$$\mathbf{z_t} = \begin{cases} \mathbf{y_t} + |b_{hom}| \begin{bmatrix} |\alpha_t| & -|\alpha_t| \end{bmatrix}, & \text{if } h_t \leq h \\ \mathbf{y_t} + |b_{hom}| \begin{bmatrix} -\alpha_t & \alpha_t \end{bmatrix}, & \text{if } h_t > h. \end{cases} \tag{26}$$

Figure 3: The difference in magnitudes for b_{het} and b_{hom} for parameter combinations of degree and h. Red indicates parameter combinations where b_{hom}, coming from the homophilous model, is smaller, and blue indicates parameter combinations where b_{het}, coming from the heterophilous model, is smaller. Smaller coefficients are preferred to minimize the discrepancies on predictions induced by α_t. Predominantly negative values indicates that the heterophilous model formulated as $(\mathbf{X} \parallel \mathbf{AX})\mathbf{W}$ is less sensitive to α_t and thus produces smaller discrepancies.

where the sign of α_t has been integrated into the vectors. To better understand the conditions that lead to this scenario, we note that d must be greater than $\dfrac{1}{1 - 2h}$ to lead to $b_{hom} < 0$. However, even for moderate values of h, the required degree to elicit a change is small, thus the heterophily shift is often what will dominate the prediction, not the degree. Furthermore, as $d(2h - 1)$ becomes more negative, b_{hom} will tend towards $\dfrac{1}{1 + (2h - 1)}$, demonstrating that the influence of d on predictions decreases as d gets larger. As such, we can expect changes in the local homophily to be the dominant factor in prediction changes.

Setting 2: Mixed Homophily ($h = 0.5$): When the graph is not strongly homophilous nor strongly heterophilous, $h = 0.5 \rightarrow b_{hom} = d$, leading to the following changes for $\mathbf{z_t}$:

$$\mathbf{z_t} = \begin{cases} \mathbf{y_t} + d \begin{bmatrix} -|\alpha_t| & |\alpha_t| \end{bmatrix}, & \text{if } h_t \leq h \\ \mathbf{y_t} + d \begin{bmatrix} \alpha_t & -\alpha_t \end{bmatrix}, & \text{if } h_t > h. \end{cases} \tag{27}$$

As the exception from other cases, the change in performance directly depends on the degree d and thus performance variations may be dominated by degree, rather than local homophily, as d becomes large. Intuitively, if the global homophily level is 0.5, we can expect the model to not make use of this signal.

Setting 3: Homophilous ($0.5 < h \leq 1$): In this scenario, $b_{hom} > 0$, thus:

$$\mathbf{z_t} = \begin{cases} \mathbf{y_t} + |b_{hom}| \begin{bmatrix} -|\alpha_t| & |\alpha_t| \end{bmatrix}, & \text{if } h_t \leq h \\ \mathbf{y_t} + |b_{hom}| \begin{bmatrix} \alpha_t & -\alpha_t \end{bmatrix}, & \text{if } h_t > h. \end{cases} \tag{28}$$

Similar to Case 1, as $d(2h - 1)$ becomes more positive, b_{hom} will approach $\dfrac{1}{1 + (2h - 1)}$, leading to d becoming less influential on the prediction. As such, we can expect changes in the local homophily to be the dominant factor in prediction changes.

High-level Takeaway: Notably, the heterophilous and homophilous settings are not necessarily symmetric – the heterophilous setting requires the degree criteria to be satisfied in order to attain the predictive pattern, while the homophilous setting is always true. We suspect this plays a role in our (as well as previous) analysis as our theoretical analysis suggests homophilous datasets will

always degrade in performance as test nodes become heterophilous, while heterophilous datasets will only degrade when the degree criteria is met. This phenomenon is seen in Figure 2 of our real-world analysis, where the performance disparities are generally more drastic in the homophilous datasets.

A.2.4 Additional Theoretical Analysis on 2-layer GNN

We follow a similar set up as in the 1-layer case for the graph G, but now consider a 2-layer GNN, F, formulated as $(\mathbf{A} + \mathbf{I})^2 \mathbf{X} \mathbf{W}$. During the first aggregation step of F, a node i of class 0 will have updated features as specified in Equation 6, and a node i of class 1 will have updated features as specified in Equation 7.

During the second aggregation step of F, a node i of class 0 will have updated features $\mathbf{r_i^2}$ defined as:

$$y_i = 0 : \mathbf{r_i^2} = \frac{hd+1}{2} \left[(1 + d + p + dp(2h-1)) \quad (1 - p + d(1 + p - 2hp)) \right] + \frac{(1-h)d}{2} \left[(1 - p + d(1 + p - 2hp)) \quad (1 + d + p + dp(2h-1)) \right] \tag{29}$$

similarly, a node i of class 1 will have updated features:

$$y_i = 1 : \mathbf{r_i^2} = \frac{hd+1}{2} \left[(1 - p + d(1 + p - 2hp)) \quad (1 + d + p + dp(2h-1)) \right] + \frac{(1-h)d}{2} \left[(1 + d + p + dp(2h-1)) \quad (1 - p + d(1 + p - 2hp)) \right] \tag{30}$$

Each $\mathbf{r_i^2}$ is the sum of the updated node vector for node i after one aggregation step ($\mathbf{r_i^1}$), the updated feature vectors from the homophilous connections of node i, and the updated feature vectors of the heterophilous connections for node i. Condensing these terms, the simplified expressions for each $\mathbf{r_i^2}$ is shown in Equations 31 and 32

$$y_i = 0 : \mathbf{r_i^2} = \frac{1}{2} \left[1 + p + d^2(1 + (1-2h)^2 p) + d(2 + (4h-2)p), \right. \\ \left. 1 - p + d(2 + (2-4h)p) - d^2(-1 + (1-2h)^2 p) \right] \tag{31}$$

$$y_i = 1 : \mathbf{r_i^2} = \frac{1}{2} \left[1 - p + d(2 + (2-4h)p) - d^2(-1 + (1-2h)^2 p), \right. \\ \left. 1 + p + d^2(1 + (1-2h)^2 p) + d(2 + (4h-2)p) \right] \tag{32}$$

Similar to the 1-layer case, $(\mathbf{A} + \mathbf{I})^2 \mathbf{X} \mathbf{W}$ can be represented as $\mathbf{R} \mathbf{W}$, where $\mathbf{R}$ is the matrix of updated features after two layers of feature aggregation. $\mathbf{R} \mathbf{W}$ is then set equal to the label matrix $\mathbf{Y}$ after a transformation by the weight matrix $\mathbf{W}$. Again, we sample the unique data points, leaving us with the optimal $\mathbf{W}$ after solving the reduced system of equations.

We now consider a new test point, t with label $y_t = 0$ and features $\mathbf{x_t} = \frac{1}{2} \left[(1+p) \quad (1-p) \right]$. In addition, we assume that t has a homophily ratio $h + \alpha_t = h_t \in [0, 1]$ where α_t represents a shift away from h. When α_t is positive, we can interpret this as a test node with a higher homophily ratio, and when α_t is negative, a lower homophily ratio. Under this new homophily ratio, $\mathbf{r_t^2}$ can be calculated as:

$$\mathbf{r_t^2} = \frac{1}{2} \left[1 + p + d^2(1 + (1-2h_t)^2 p) + d(2 + (4h_t - 2)p), \right. \\ \left. 1 - p + d(2 + (2-4h_t)p) - d^2(-1 + (1-2h_t)^2 p) \right] \tag{33}$$

We can now compute $\mathbf{r_t^2} \mathbf{W} = \mathbf{z_t^2}$, the output of the 2-layer GNN, and analyze the associated predictions. The output of $\mathbf{r_t^2} \mathbf{W}$ is shown in Equation 34, where $b_{2hom} = \dfrac{d}{(1 + d(2h-1))^2}$

$$\mathbf{z_t^2} = \frac{b_{2hom}}{d} \left[1 + 2d(h + h_t - 1) + d^2(1 - 2h + 2h^2 - 2h_t + 2h_t^2), \right.$$
$$\left. 2d(h - h_t)(1 + d(-1 + h + h_t)) \right] \tag{34}$$

We now consider what happens when $h_t = h + \alpha_t$ where $\alpha_t \in [-h, 1-h]$. Since we are interested in understanding how the predictions would change as a byproduct of α_t, we look at the difference between predictions when $h_t = h$. Below we compute $\Delta \mathbf{z_t^2}$, the change in the logit vector output for the 2-layer GNN on the test point with local homophily h_t.

$$\mathbf{z_t^2} = \frac{b_{2hom}}{d} \left[1 + 2d(h + (h + \alpha_t) - 1) + d^2(1 - 2h + 2h^2 - 2(h + \alpha_t) + 2(h + \alpha_t)^2), \right.$$
$$\left. 2d(h - (h + \alpha_t))(1 + d(-1 + h + (h + \alpha_t))) \right] \tag{35}$$

$$\Delta \mathbf{z_t^2} = 2b_{2hom} \left[\alpha_t(1 + d(2h - 1 + \alpha_t)) \quad -\alpha_t(1 + d(2h - 1 + \alpha_t)) \right]$$
$$\Delta \mathbf{z_t^2} = 2(1 + d(2h - 1 + \alpha_t))b_{2hom} \left[\alpha_t \quad -\alpha_t \right] \quad \square \tag{36}$$

The change in the logit vector, $\Delta \mathbf{z_t^2}$, for the 2-layer GNN, is similar to the change in the logit vector for the 1-layer GNN, except with an additional multiplicative factor. Specifically, if we refer to the change in the logit vector for the 1-layer GNN as $\Delta \mathbf{z_t^1}$, $\Delta \mathbf{z_t^2} = (2(1 + d(2h - 1 + \alpha_t))\frac{b_{hom}}{d} \Delta \mathbf{z_t^1}$. We focus on analyzing this multiplicative factor, given we already understand the behavior of $\Delta \mathbf{z_t^1}$ as studied in Section 4. Our analysis is facilitated by letting the element-wise division of $\Delta \mathbf{z_t^2}$ and $\Delta \mathbf{z_t^1}$ be equal to $\mathbf{q} = \frac{\Delta \mathbf{z_t^2}}{\Delta \mathbf{z_t^1}} = \frac{2b_{hom}(1 + d(2h - 1 + \alpha_t))}{d} \begin{bmatrix} 1 & 1 \end{bmatrix} = \frac{2(1 + d(2h - 1 + \alpha_t))}{1 + d(2h - 1)} \begin{bmatrix} 1 & 1 \end{bmatrix} = \frac{2(1 + d(2h - 1) + d\alpha_t)}{1 + d(2h - 1)} \begin{bmatrix} 1 & 1 \end{bmatrix}$ and studying the coefficient of $\mathbf{q}$. Specifically, we analyze two extreme cases where t is either part of a strongly heterophilous or strongly homophilous graph to see what patterns emerge.

Setting 1: Strongly Heterophilous ($h \approx 0$): We study this scenario by analyzing the limit of $\mathbf{q}$ as h approaches 0, with $d > 1$.

$$\lim_{h \to 0} \mathbf{q} = \frac{2(1 + d(\alpha_t - 1))}{1 - d} \tag{37}$$

In this scenario $0 \le a_t < 1$. When $d(\alpha_t - 1) < -1$, we see that the coefficient is positive, thus following the same degradation trend as in the 1-layer GNN. Conversely, when $d(\alpha_t - 1) > -1$ the coefficient is negative, indicating that the performance degradation does not occur. However, this requires that $\alpha_t > \frac{d-1}{d}$ which quickly becomes unlikely to occur as d grows, as even low values of d require extremely large α_t values to satisfy the inequality, e.g. $d = 4$ requires $\alpha_t > \frac{3}{4}$. We additionally analyze the magnitude of this coefficient, finding that $\frac{2(1 + d(\alpha_t - 1))}{1 - d} > 1$ when $\alpha_t < \frac{d-1}{2d}$, suggesting that the 2-layer GNN more rapidly degrades in performance as compared to the 1-layer GNN when for moderate values of α_t, and behaves similarly to the 1-layer when α_t is large and d is small.

Setting 2: Strongly Heterophilous ($h \approx 1$): We study this scenario by analyzing the limit of $\mathbf{q}$ as h approaches 1, with $d > 1$.

$$\lim_{h \to 1} \mathbf{q} = \frac{2(1 + d(\alpha_t + 1))}{1 + d} \tag{38}$$

In this scenario $-1 < a_t \le 0$. As in the 1-layer case, the coefficient is strictly positive, thus following the same degradation trend as in the 1-layer GNN. Similar to the heterophilous case, we find that

$\frac{2(1 + d(\alpha_t + 1))}{1 + d} > 1$ when $\alpha_t > -\frac{d - 1}{2d}$, mirroring the behavior of the 2-layer GNN applied to heterophilous graphs with increased degradation.

High-level Takeaway: When moving from a 1-layer to 2-layer GNN, we find that the trend of performance degradation still holds, with performance degrading as the local homophily ratio of a node deviates from the global homophily used to train the GNN. Additionally, we show that the rate of performance degradation for local homophily ratios far from the global homophily ratio is generally increased when a 2-layer GNN is applied, relative to a 1-layer GNN.

A.3 Evaluation through Edge Homophily

In this section we discuss other edge-based homophily metrics proposed in the literature. Additionally, we discuss how our results generalize to higher-order neighborhoods.

Other Edge-based Homophily Metrics: In this work, we focus on edge homophily as a means of characterizing performance. Recent work has pointed out that the global edge homophily metric, provided in Equation 1, can become susceptible to imbalance in multi-class settings [29]. To remedy this issue, the authors propose a class homophily metric which accounts for the size of each class within the global homophily calculation. As our analysis focuses on local homophily analysis, the issues in regards to the global homophily metric do not directly apply when stratifying across local homophily ranges. Additionally, the class homophily metric has no direct application to local analysis, given it depends on graph-level properties. While it is possible that the new class homophily metric can change the global homophily characterization, we find that the heterophilous datasets in this work have similar metrics as computed by both the class homophily and traditional global edge homophily [29], suggesting either metric would lead to similar conclusions. On heterophilous datasets with larger class imbalances, it may be prudent to use the class homophily metric to better contextualize discrepancy.

Higher-Order Behavior: Two natural question arise as to whether the results hold when the GNN operates on higher-order information, as well as if the results can be contextualized by higher-order information. We first note that through the theoretical analysis in Appendix A.2.4, we can see that a GNN operating on higher order information (2-hop neighborhoods) will experience similar degradation patterns. Given the relationship under the 2-hop setting is more complex, we prioritize the 1-hop theorem in the main text to provide a more intuitive understanding of how performance degrades. Moreover, all of the GNNs trained are cross validated on depths between 2 and 4, indicating that empirically the higher order information again clearly trends with our homophily measure. From the perspective of characterization, operating on higher-order neighborhoods often follows similar patterns as the one-hop case [19, 29], up to a certain point, as these metrics will approximate the global homophily when the size of the neighborhoods approach the entire graph. Moreover, there are practical challenges such as extremely large computation times, and ambiguity in metric design, that make it difficult to apply higher-order metrics. For instance, it is unclear if the metric should compare (a) the ego-node label with the labels for nodes in the k-hop neighborhood, (b) the labels between neighbors within the k-hop neighborhood, or (c) the ego-node label as compared with the labels of nodes exactly k-hops from the ego-node. In this work, we focus on the 1-hop case given its ubiquity across research on homophily/heterophily, as well as its intuitive definition.

A.4 Synthetic Datasets

A.4.1 Graph Generation Related Work

Karimi et al. proposed the homophily-based preferential attachment model that is used as the foundation for our synthetic generation method [13]. More recently, GraphWorld has been proposed to allow for the study of GNN performance across diverse graphs with varying global homophily, size, and class imbalance [41]. Additionally, FastSGG has been proposed to generate graphs for social network settings with an emphasis on control of the graph's degree distribution [42]. Despite these varying approaches, GraphWorld, FastSGG, and Karimi et al's preferential attachment model only consider homophily through a fixed global parameter, providing no insight into, or control of, the local homophily distribution in each graph. GenCAT remedies this issue by introducing a node-class matrix to control the homophily level for each node [43]. However, while this matrix provides the flexibility to generate a graph with an arbitrary local homophily distribution, in practice there are no

---

Algorithm 1: Synthetic Graph Generation

Input: Total nodes n, m edges to add per step, label probability distribution $P(\{0, ..., c\})$, uniformity probability ρ, class compatibility matrix $\mathbf{H}_L$, drift change cutoff δ

1   Initialize G with m nodes and m edges according to $\mathbf{H}_L$;   `// Details in App. A.4.2`
2   Initialize vector $drift$ to hold change in homophily per node $drift_0, ..., drift_m = 0$
3   **for** $u = m$ *to* n **do**
4   Add node u to G
5   Sample label $y_u \sim P(\{0, ..., c\})$
6      $N_1(u) = \text{GetNeighbors}(G, m, u, drift, \rho, \mathbf{H}_L, \delta)$
7   **for** v *in* $N_1(u)$ **do**
8   Add edge (u, v) to G
9   **if** $y_u == y_v$ **then**
10            $drift[v] \mathrel{+}= 1$;   `// Drifted more homophilous`
11   **else**
12            $drift[v] \mathrel{-}= 1$;   `// Drifted more heterophilous`
13      $drift[u] = 0$

Result: Graph G

Algorithm 2: GetNeighbors

Input: G, m edges to add, node u, $drift$ vector, uniformity probability ρ, class compatibility matrix $\mathbf{H}_L$, drift change cutoff δ

1   **if** $q \sim U(0, 1) \leq \rho$ **then**   `// Connect based on random local homophily` h'
2   Sample new local homophily ratio $h' \sim U(0, 1)$
3      $hom\_drift = \{q : drift_q > \delta\}$;   `// Get nodes with large homophilous drift`
4      $het\_drift = \{s : drift_s < -\delta\}$;   `// Get nodes with large heterophilous drift`
5      $N_1(u) = \{\text{Sample up to } round(mh') \text{ nodes with label } y_u \text{ from } het\_drift\}$
6   **if** $|N_1(u)| < round(mh')$ **then**
7          $N_1(u) \cup \{\text{Sample } round(mh') - |(N_1(u))| \text{ nodes with label } y_u \text{ from } G \setminus het\_drift\}$
8      $N_1(u) \cup \{\text{Sample up to } round(m(1 - h')) \text{ nodes with label} \neq y_u \text{ from } hom\_drift\}$
9   **if** $|N_1(u)| < m$ **then**
10         $N_1(u) \cup \{\text{Sample } m - |N_1(u)| \text{ nodes with label} \neq y_u \text{ from } G \setminus hom\_drift\}$
11   **else**   `// Connect based on original compatibility matrix` $\mathbf{H}_L$
12 $N_1(u) = \{\text{Sample } m \text{ nodes with probability based on } \mathbf{H_L}\}$

Result: $N_1(u)$

---

mechanisms proposed to enforce a particular distribution over it. Instead, each node's homophily is assumed to be completely independent, which is both unrealistic and cumbersome to instantiate. As such, systematically controlling this distribution would require the introduction of a parameterization over the node-class matrix, which is not proposed in the work. Our approach solves this problem by including a new *single* parameter into the preferential attachment model that enforces nodes to deviate from the global homophily ratio, introducing nodes with a local homophily that has been shifted relative to the global homophily ratio. This capability allows us to easily learn over graphs with a wide range of local homophily ratios in a highly controlled manner, demonstrating the practical implications of our theory across different GNNs.

A.4.2   Synthetic Generation Algorithm - Additional Details

Initializing the Generated Graph: We follow the same strategy to initialize the graph as is done in previous homophily-based preferential attachment models, as well as in the classic Barabasi-Albert (BA) algorithm [13, 17, 44]. We begin by generating a node with label assigned from $P(0, ..., c)$ and add it to the empty graph G. We then sequentially add $m - 1$ nodes to G, where each node is first assigned a class label and then attached to one other node in the graph. The attachment process for these nodes follows the GetNeighbors function as outlined in the main text, with only one edge added, rather than m edges. The result of this process is a connected graph of m nodes and m edges, allowing any subsequent nodes to have m connections without needing to create multiple edges between the same nodes.
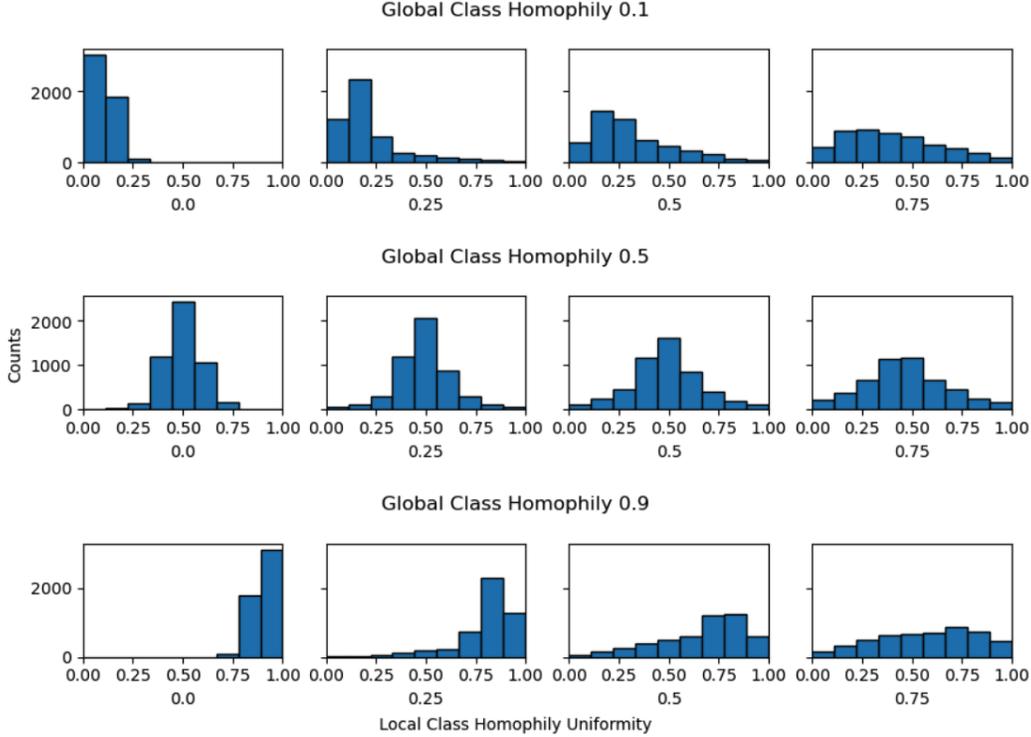
Figure 4: Distribution of local homophily ratios generated as the global class homophily is varied (rows) and the uniformity ρ is varied (columns). We see the nodes spread out to occupy the full range of homophily values around $\rho = 0.5$ while still maintaining a clear peak.

A.4.3 Synthetic Dataset Properties

The synthetic dataset generation process is described in Section 5 of the main text. Here, we detail some properties of the synthetic data, including the distribution of local homophily ratios as the uniformity parameter is varied and degree information.

In Figure 4, we show how the distribution of local homophily changes as a function of the uniformity parameter. On the left-most plot for each row, we show the distribution of local homophily ratios on the unmodified preferential attachment model. Despite the minor deviation from the set global homophily ratio, the values are highly concentrated and do not provide samples across the entire range of local homophily values. This is similarly seen for the plot second to the left, which denotes $\rho = 0.25$, where the extreme homophily and heterophily values do not have enough nodes for analysis. The plot second from the right, $\rho = 0.5$, demonstrates the first scenario where there are ample data points across the the full range of local homophily and thus what we focus on in our analysis. Finally, the plot farthest to the right, $\rho = 0.75$, demonstrates a scenario where the distribution approaches uniformity. As we are interested in the problem of distribution shift at test time, this scenario is of less interest.

In Figure 5, we show the degree distribution for each of the graphs of varying uniformity level. As compared to the unmodified preferential attachment model, the uniformity parameter introduces a small amount of high degree nodes seen to the right of the distribution in each plot. This is due to the correction phase of the generation process, as described in the main text. However, as the uniformity parameter increases, the amount of high degree nodes becomes less severe. Moreover, even in the case of $h = 0.5, \rho = 0.25$, the high degree nodes only account for around 1% of the nodes with the other graphs only containing even less.

### A.4.4 Additional Synthetic Results

In this section, we include a series of additional experiments to supplement the findings in the main text. For our synthetic data, we include results for $h \in \{0.3, 0.7\}$, as well as additional models across all synthetic graph setups.
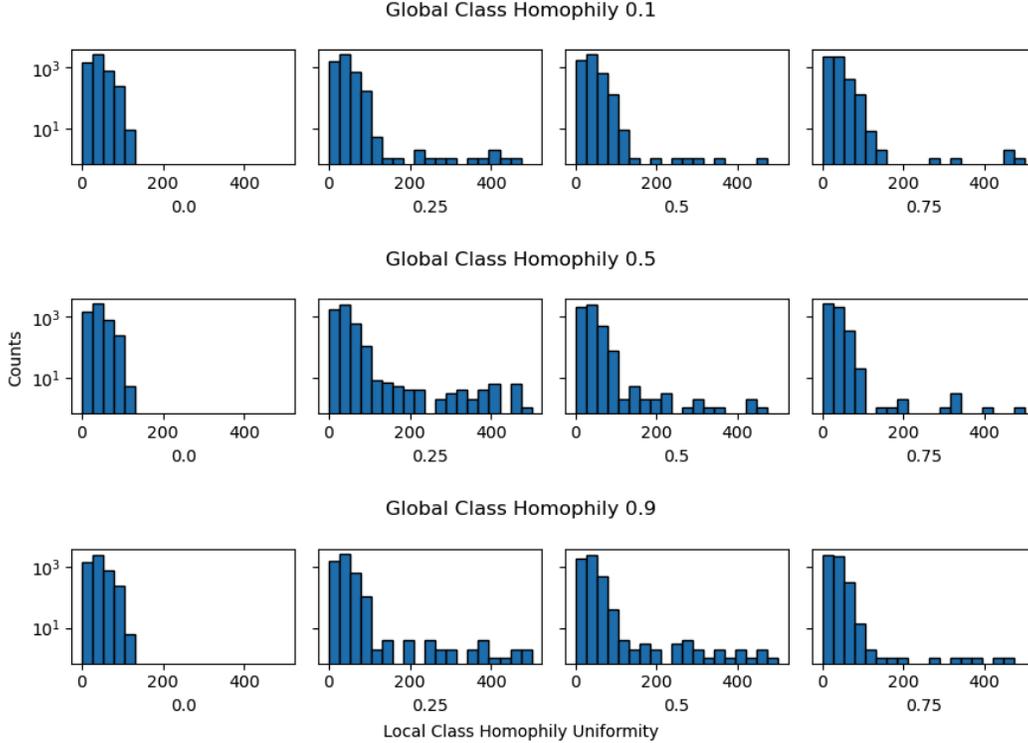
Figure 5: Distribution of degrees generated as the global class homophily is varied (rows) and the uniformity ρ is varied (columns). We see that ρ does not significantly impact the degree distribution other than for a small percentage of nodes.

Pair-wise Differences between Homophilous and Heterophilous Models. In this section we re-plot Figure 1, directly subtracting the performance of the homophilous models from the heterophilous models. We show the difference in performance for all comparisons in Figure 6, where the homophilous model is subtracted from the heterophilous model. Thus, positive values are indicative of the homophilous model performing better, while negative values are indicative of the heterophilous model performing better. We show this for our two extreme settings, $h = 0.1, 0.9$. These plots visually reinforce our findings as the local homophily ranges near the global homophily ratio are always positive (homophilous models performing better), while the local homophily ranges far from the global homophily ratio are always negative (heterophilous models performing better). Additionally, the relative performance differences between the homophilous and heterophilous models in the positive and negative sections of Figure 6 are significant – heterophilous models achieve upwards of 0.45 F1 improvement in under-represented local homophily ranges while often losing less than 0.05 F1 on the over-represented local homophily ranges.

Additional Experiments for $h \in \{0.3, 0.7\}$. In this section we highlight the performance of GCN, GAT, SAGE, GCNII, H2GCN, LINKX, and MLP models on synthetic settings where $h \in \{0.3, 0.7\}$. The results are shown in 7. While the discrepancy is not as significant as before, this is not unexpected as the theory indicates models will experience the most degradation at extreme local homophily shifts. However, almost every model, except for GCNII, degrades with a drop in F1 around 0.2 as the local homophily level shifts relative to the global homophily. Regarding model design, GCNII maintains a strong performance across all ranges in both settings, as does H2GCN in the $h = 0.7$ setting, showing the importance of heterophilous design to minimize disparity.
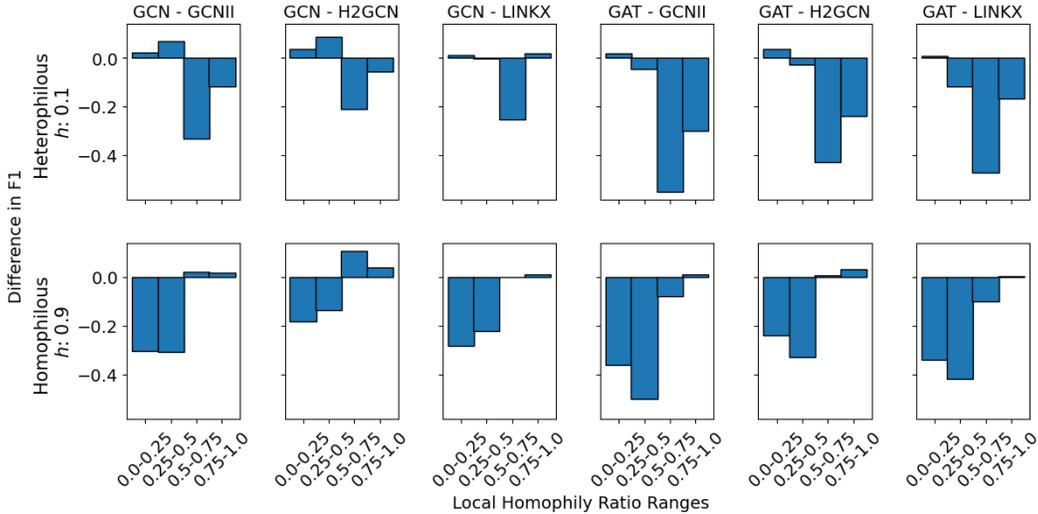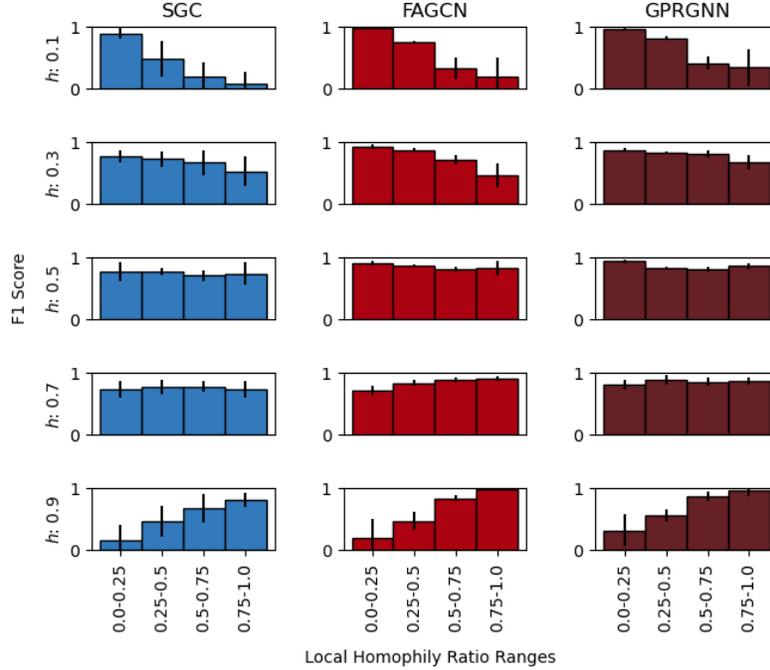
Figure 6: Performance differences between homophilous and heterophilous models (columns) on synthetic data generated with global homophily ratios (columns) $h \in \{0.1, 0.9\}$ and uniformity $\rho = 0.5$. Each bar represents the average difference in F1 score between the two models specified in the column header for nodes with a local homophily ratio between the values specified on the x-axis.
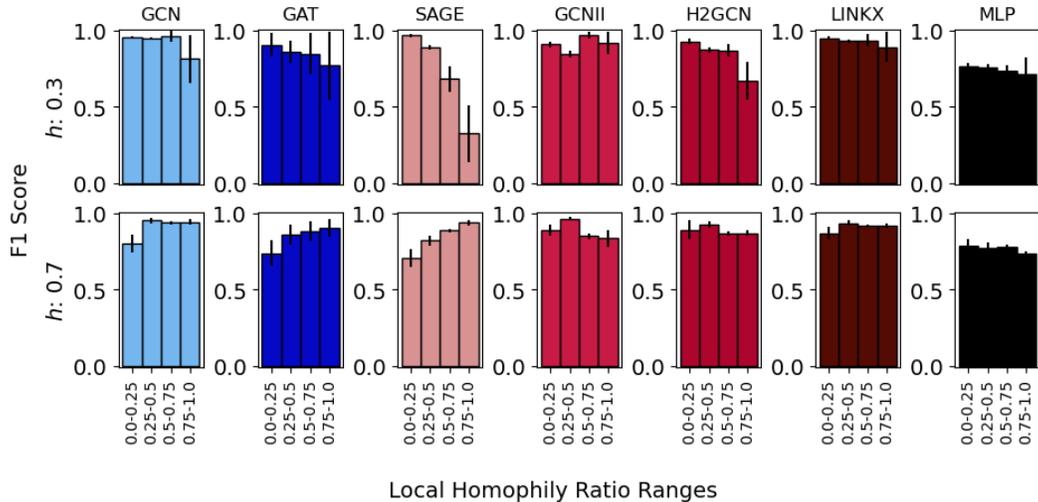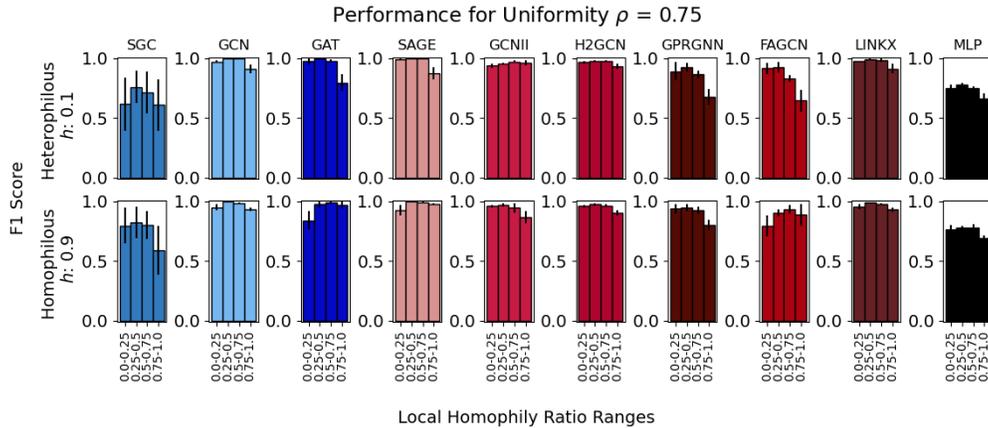

Figure 7: Performance across models (columns) on synthetic data generated with global homophily ratios (rows) $h \in \{0.3, 0.7\}$ and uniformity $\rho = 0.5$. Each bar represents the F1 score for nodes with a local homophily ratio between the values specified on the x-axis. Error bars denote the standard deviation in performance. Blue bars indicate models with homophilous designs, red bar indicates models with heterophilous designs.

Additional Models for All h. In addition to the models used in the main text, i.e. GCN, GAT, SAGE, GCNII, and H2GCN, we include results on SGC, FAGCN and GPRGNN. SGC is a linearized GNN architecture with a simple weighted aggregation tailored towards homophily settings. FAGCN and GPRGNN are both architectures designed to improve learning under heterophilous graphs, with the ability to differentiate homophilic and heterophilic nodes through gating mechanisms and adaptable weights, respectively. In Figure 8, we show the results for these three models on each set of synthetic graph parameters. Similar to the main text, we find that SGC significantly degrades in performance as the local homophily ratio deviates from the global homophily ratio, even more so than GCN or GAT. Additionally, we see that FAGCN and GPRGNN both retain relatively strong performance, albeit slightly less than H2GCN and GCNII. We find that FAGCN and GPRGNN are much more sensitive to hyperparameters, as compared to H2GCN and GCNII, where the wrong choice can cause

their performance to devolve close to that of GCN. While the general trend of heterophilous GNNs performing better on the full range of local homophily values remains true, H2GCN's concatenation based mechanisms provides significant practical value due to not relying on hyperparameters for strong performance.

Figure 8: Performance across models (rows) on synthetic data generated with global homophily ratios (columns) $h \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and uniformity $\rho = 0.5$. Each bar represents the F1 score for nodes with a local homophily ratio between the values specified on the x-axis. Error bars denote the standard deviation in performance. Blue bars indicate models with homophilous designs, red bar indicates models with heterophilous designs.

Figure 9: Performance across models (columns) on synthetic data generated with global homophily ratios (rows) $h \in \{0.1, 0.9\}$ and uniformity $\rho = 0.75$. Each bar represents the F1 score for nodes with a local homophily ratio between the values specified on the x-axis. Error bars denote the standard deviation in performance. Blue bars indicate models with homophilous designs, red bar indicates models with heterophilous designs. By introducing more support across the range of local homophily ratios, each GNN is able to achieve much lower discrepancy across groups.

Additional Performance for $\rho = 0.75$. To further explain our choice for $\rho = 0.5$ in Section A.4.4, we train all of our models (GNNs and MLP) on additional synthetic datasets where $\rho = 0.75$. The distribution of local homophily ratios for $\rho = 0.75$ can be seen in Figure 4. We do not consider $\rho = 0.25$ as there are not enough data points across the full local homophily range to achieve a reasonable signal, as seen in Figure 4. For our experiments, we follow the same setup as in Section A.4.4 by training each model on 10 different graphs of a particular parameter combination and analyze the performance trends across groups of nodes with varying local homophily level. We focus on $h = 0.1$ and $h = 0.9$ to maximize the possible shifts in local homophily, relative to h. The results are displayed in Figure 9 for all models. As expected, the performance discrepancy is significantly less than in Figure 1, across all model. Moreover, the difference in performance between the homophilous and heterophilous GNNs is less significant. As each GNN now has ample support across the full range local homophily range during training, each model is able to generalize, irrespective of the homophily level. While $\rho = 0.5$ is a useful tool to demonstrate how discrepancy can arise in GNN models, Figure 9 is valuable in showing a set of conditions that can allow GNNs to perform well, irrespective of design.

Additional Discussion and Experiments on Training Set Size. For our synthetic analysis, we use a training/validation/test split ratio of 50/25/25%. Based on previous works for GNNs applied to *heterophilous* settings, most use similar (or even higher) ratio to study their models [7, 17, 45]. While studies have shown that GNNs applied to *homophilous* settings can attain strong performance with low training node ratios [22], it is common for heterophilous graphs to require more training data as the local homophily patterns are more complex. Despite this known behavior, the impact of training size on disparity is not well studied. To address this gap, we include a set of experiments on our synthetic data where the amount of training data is varied. All of the parameters, aside from the amount of training data, is kept the same as described in Section of the main text. As the main text uses 50% of the nodes for training, we study the scenarios of 30%, and 10%.

Through the experiments depicted in Figure 10, we find that the amount of training data does not significantly alter the discrepancy patterns. This further verifies the findings of the main text that discrepancy is a fundamental issue embedded within the models and training distribution, rather than an artifact of training set size. We do note that many of the models have amplified degradation as the homophily level deviates from the global homophily ratio, as compared to the original results with a 50% training set ratio. For example, GCN, GAT, and H2GCN nearly drop to an F1 score of 0 for the furthest bins from the global homophily ratio as the training set ratio drops. This is not unexpected, as the smaller training sets can cause less nodes to fall within these bins during training. Thus, training set size can *influence* the distribution of local homophily levels, which impacts the discrepancy patterns, but it is not a direct issue. Some models retain comparable, or even slightly better, performance irrespective of training set size, however this is likely due to variations in training and are often accompanied by larger variance bars.

A.5 Real-world Datasets

### A.5.1 Additional Models for Real-world Datasets

Similar to our expanded analysis on the synthetic data, we include SGC, FAGCN, and GPRGNN results for the real-world datasets from the main text. As seen in Figure 11, the performance trends across the varying local homophily ranges generally stay consistent with the findings in the main text. In particular, we see that for Wisconsion, Arxiv-Year, Cora, and Coauthor-CS, the highest performing subgroups for each model is in, or near, the gray region, indicating that the global homophily ratio is influencing the performance. Squirrel demonstrates a slightly different behavior by not degrading at the furthest group, however does follow the trend across the first three groupings. In the main text, we note that there is noise in this regime due to the lack of nodes. Additionally, previous work has pointed out that the squirrel dataset can become easier to predict on due to duplicates across the structure [37]. As such, it may be possible that the data leakage caused by these duplicates can influence performance, making certain homophily ranges easier.

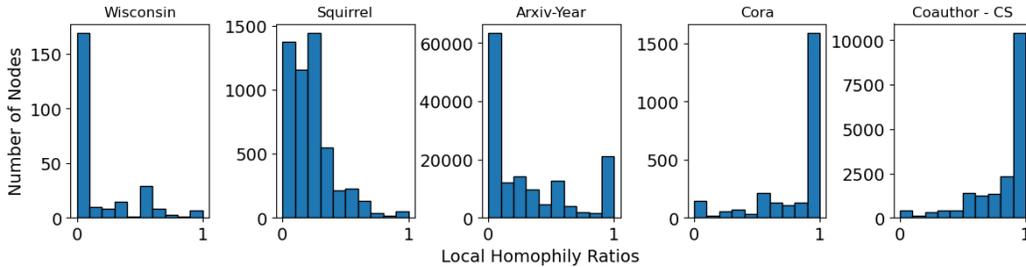### A.5.2 Local Homophily Distribution for Real-world Datasets

For each the dataset analyzed in this work, we include a histogram of their local homophily distribution. While Cornell and Wisconsin are very similar and predominately heterophilous, the other datasets provide a wide spread of nodes across the varying local homophily ratios.
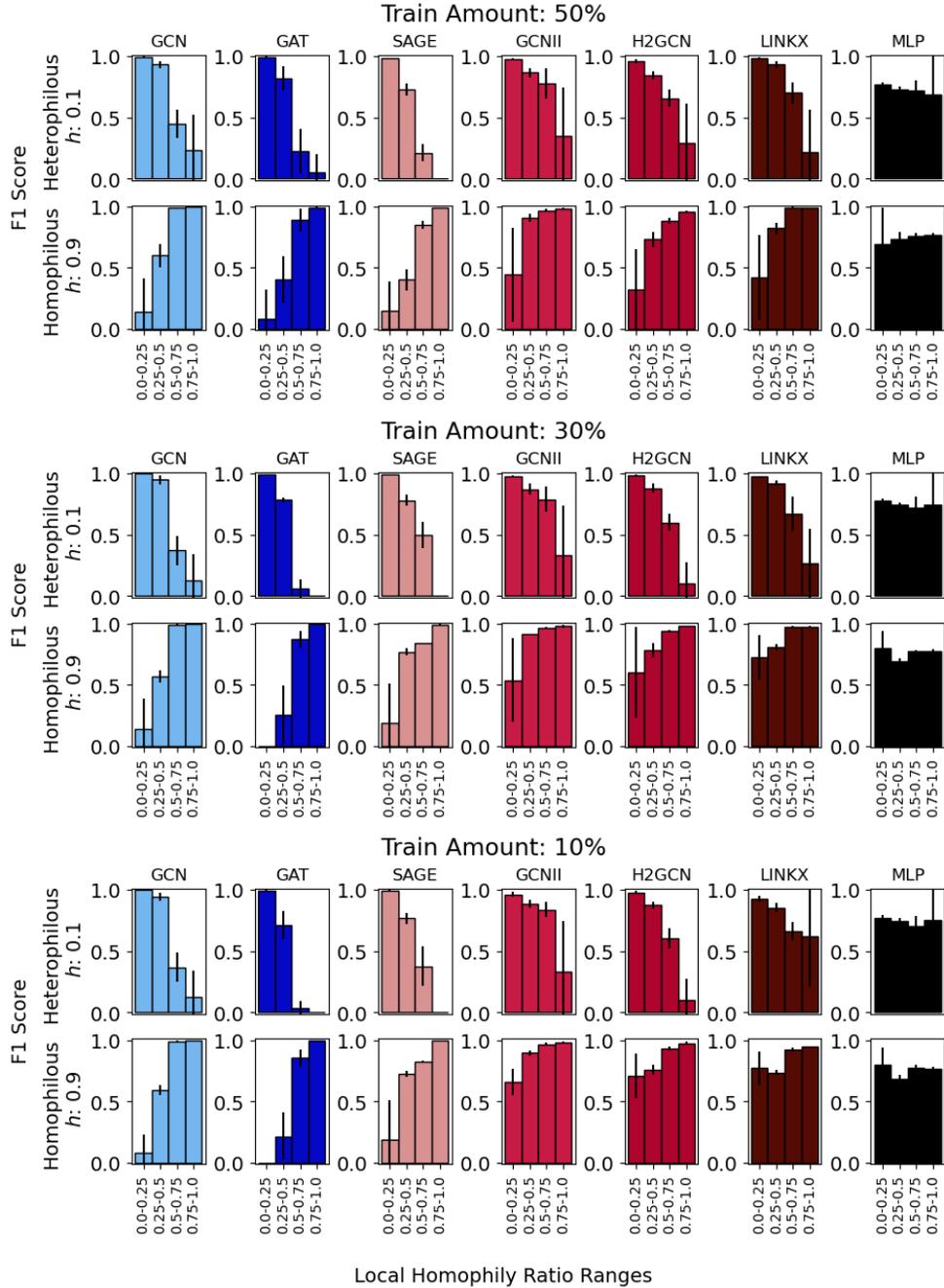
Figure 10: Performance across models (columns) on synthetic data generated with global homophily ratios (rows) $h \in \{0.1, 0.9\}$ and uniformity $\rho = 0.5$ with training set ratios of 50% (top, the original results), 30% (middle) and 10% (bottom). Performance degradation patterns generally remain the same, highlighting discrepancy as a fundamental model and data distribution problem, rather than a direct artifact of training set size.
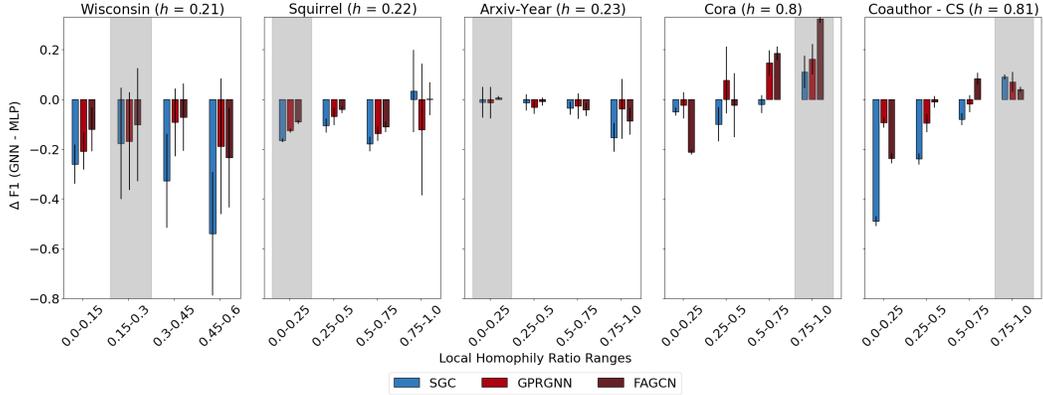
Figure 11: F1 performance difference between GNNs and MLP (ΔF1) for varying local homophily ratios in real-world graphs. For a homophily range specified on the x-axis, the 5 bars denote ΔF1 for each GNN across test nodes that fall within the range (ranges adjusted per dataset such that each range has at least three nodes). Gray regions indicate the range that the global homophily ratio falls in. More negative bars indicate worse performance, while more positive bars indicate better performance. Error bars are the standard deviation in performance across experiment runs. Blue bars denote GNNs with homophilous designs, while red bars denote GNNs with heterophilous designs. All results align with previous findings where best performance is generally within the gray regions, and systematic performance disparity emerges as the ranges move further from the global homophily.

Figure 12: Histograms depicting the distribution of local homophily ratios for the real-world datasets analyzed in the main text. Notably, all datasets, despite their clear peak, possess nodes across the entire range of possible local homophily ratios

A.6 Models and Hyperparameter Tuning

In this section we detail the models and hyperparameters tuned for each experiment.

For synthetic experiments:

1. **MLP**: Implementation directly from PyTorch
   - Hidden Dim: 16, 32
   - Depth: 2, 3
   - Dropout: 0.3, 0.5

2. **GCN**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.gcn_conv)
   - Hidden Dim: 16, 32
   - Depth: 2, 3
   - Dropout: 0.3, 0.5

3. **GAT**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.gat_conv)
   - Hidden Dim: 16, 32
   - Depth: 2, 3
   - Heads: 1, 2
   - Dropout: 0.3, 0.5

28

4. **SAGE**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.sage_conv)
   - Hidden Dim: 16, 32
   - Depth: 2, 3
   - Dropout: 0.3, 0.5

5. **GCN-II**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.gcn2_conv)
   - Hidden Dim: 16, 32
   - Depth: 4, 8
   - Dropout: 0.3, 0.5

6. **H2GCN**: Open source PyTorch implementation from https://github.com/CUAI/Non-Homophily-Large-Scale
   - Hidden Dim: 16, 32
   - Depth: 2, 3
   - Dropout: 0.3, 0.5

7. **SGC**: Implementation directly from PyTorch Geometric (torch_geometric/nn/conv/sg_conv)
   - Depth: 2, 3

8. **GPR-GNN**: Open source PyTorch implementation from https://github.com/jianhao2016/GPRGNN
   - Hidden Dim: 16, 32
   - Depth: 2, 3
   - Alpha: 0.1, 0.5, 0.9
   - K: 10
   - Dropout: 0.3, 0.5

9. **FA-GNN**: Open source PyTorch implementation from https://github.com/bdy9527/FAGCN
   - Hidden Dim: 16, 32
   - Depth: 2, 3
   - Epsilon: 0.1, 0.5, 0.9
   - Dropout: 0.3, 0.5

10. **LINKX**: Implementation directly from PyTorch Geometric (torch_geometric/nn/models/LINKX)
    - Hidden Dim: 16, 32
    - Depth: 2, 3
    - Dropout: 0.3, 0.5

For real-world experiments:

1. **MLP**: Implementation directly from PyTorch
   - Hidden Dim: 32, 64
   - Depth: 2, 3, 4
   - Dropout: 0.3, 0.5

2. **GCN**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.gcn_conv)
   - Hidden Dim: 32, 64
   - Depth: 2, 3, 4
   - Dropout: 0.3, 0.5

3. **GAT**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.gat_conv)
   - Hidden Dim: 32, 64
   - Depth: 2, 3, 4
   - Heads: 1, 2

- Dropout: 0.3, 0.5

4. **SAGE**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.sage_conv)

   - Hidden Dim: 32, 64
   - Depth: 2, 3, 4
   - Dropout: 0.3, 0.5

5. **GCN-II**: Implementation directly from PyTorch Geometric (torch_geometric.nn.conv.gcn2_conv)

   - Hidden Dim: 32, 64
   - Depth: 4, 8, 16
   - Dropout: 0.3, 0.5

6. **H2GCN**: Open source PyTorch implementation from https://github.com/CUAI/Non-Homophily-Large-Scale

   - Hidden Dim: 32, 64
   - Depth: 2, 3, 4
   - Dropout: 0.3, 0.5

7. **SGC**: Implementation directly from PyTorch Geometric (torch_geometric/nn/conv/sg_conv)

   - Depth: 2, 3, 4

8. **GPR-GNN**: Open source PyTorch implementation from https://github.com/jianhao2016/GPRGNN

   - Hidden Dim: 32, 64
   - Depth: 2, 3, 4
   - Alpha: 0.1, 0.5, 0.9
   - K: 10
   - Dropout: 0.3, 0.5

9. **FA-GNN**: Open source PyTorch implementation from https://github.com/bdy9527/FAGCN

   - Hidden Dim: 32, 64
   - Depth: 2, 3, 4
   - Epsilon: 0.1, 0.5, 0.9
   - Dropout: 0.3, 0.5

10. **LINKX**: Implementation directly from PyTorch Geometric (torch_geometric/nn/models/LINKX)

    - Hidden Dim: 32, 64
    - Depth: 2, 3, 4
    - Dropout: 0.3, 0.5

All models have a final linear layer at the end of the convolution section to produce the final predictions. Any unspecified parameters, such as the additional parameters introduced by GCN-II, H2GCN, and LINKX are left as the defaults in their respective code bases.