Understanding GNNs and Homophily in Dynamic Node Classification

Michael Ito
University of Michigan

Danai Koutra
University of Michigan

Jenna Wiens
University of Michigan

Abstract

Homophily, as a measure, has been critical to increasing our understanding of graph neural networks (GNNs). However, to date this measure has only been analyzed in the context of static graphs. In our work, we explore homophily in dynamic settings. Focusing on graph convolutional networks (GCNs), we demonstrate theoretically that in dynamic settings, current GCN discriminative performance is characterized by the probability that a node's *future* label is the same as its neighbors' *current* labels. Based on this insight, we propose *dynamic homophily*, a new measure of homophily that applies in the dynamic setting. This new measure correlates with GNN discriminative performance and sheds light on how to potentially design more powerful GNNs for dynamic graphs. Leveraging a variety of dynamic node classification datasets, we demonstrate that popular GNNs are not robust to low dynamic homophily. Going forward, our work represents an important step towards understanding homophily and GNN performance in dynamic node classification.

1 INTRODUCTION

In node classification tasks, graph neural networks (GNNs) obtain strong performance on highly homophilous graphs, where most edges connect nodes of similar labels. In contrast, on many heterophilous graphs where most edges connect nodes of opposing labels, GNN performance degrades (Pei et al., 2020). Measuring homophily of a graph-based task has since been used in attempts to characterize the discriminative power of GNNs and the types of graphs for which GNN performance is limited (Zhu et al., 2020). Several works have focused on understanding the relationship between GNNs and homophily, both at the graph and node level (Pei et al., 2020; Loveland and Koutra, 2025; Ma et al., 2022; Zhu et al., 2024a; Luan et al., 2024a). These analyses have inspired key GNN designs that provide good generalization performance on homophilous and heterophilous graphs in static node classification (Chien et al., 2021; Bo et al., 2021; Zhu et al., 2021; Yan et al., 2022; Ito et al., 2025). Homophily has thus been crucial in advancing the study of GNNs since it has increased our understanding of GNN limitations and in turn led to the development of new methods to overcome these limitations.

However, to date, the vast majority of works analyzing GNNs and homophily in node classification have assumed a static graph with unchanging features, labels, and structure (Platonov et al., 2024). In many real-world node classification tasks, the graph changes over time. For example, consider a node classification problem from epidemiology where the goal is to predict the spread of a contagion on a temporal contact network of individuals. Here, node features change across time due to time-varying individual characteristics, node labels change across time due to the spread of the contagion, and the graph structure changes across time due to changes in contact patterns of the individuals.

Inspired by this problem, we explore notions of homophily and GNNs in the context of node classification on dynamic graphs. We theoretically analyze graph convolutional networks (GCNs), a widely used GNN variant, and demonstrate that in dynamic settings, GCN discriminative performance is characterized by the probability that node future labels are the same as their neighbors' current labels. Based on this finding, we propose *dynamic homophily*, a new homophily measure that is highly correlated with the discriminative performance of GCNs in dynamic settings. Our theory further suggests potential designs for more powerful dynamic GNNs that can overcome low dynamic homophily settings. Empirically, we apply dynamic homophily to dynamic node classification datasets from epidemiology, social network analysis, and molecular bi-

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

ology, demonstrating that our theoretical analyses hold in real-world dynamic graphs for a variety of GNNs. In summary, we make the following contributions.

- **Theoretical Analysis.** We present a theoretical analysis showing that in dynamic settings both the separation and variances of node representations produced by a GCN can be represented as a function of the probability that a node's future label is the same as their neighbors' current labels. As a result, GCN discriminative performance can be represented as a function of this probability. Our analysis sheds light on how to potentially design more powerful dynamic GNNs robust to low dynamic homophily settings.
- **New Homophily Definition for Dynamic Graphs.** Based on our theoretical findings, we propose a novel definition of homophily for the dynamic setting called dynamic homophily.
- **Real-world & Synthetic Experiments.** Applied to dynamic node classification tasks, we show that dynamic homophily accurately correlates with the discriminative performance of many GNNs.

2 PRELIMINARIES

We first introduce notation and define node classification on a static graph. We then provide an overview of message passing, a common framework for node classification, focusing on linear GCNs, a key tool in our theoretical analysis. We next provide a background on the homophily measures on which we build and formalize our problem setting of node classification on dynamic graphs.

2.1 Notation

We define the static graph $G = (V, \mathbf{A}, \mathbf{X}, \mathbf{y})$, where V is set of nodes, $\mathbf{A} \in \{0,1\}^{|V|\times|V|}$ is the adjacency matrix, $\mathbf{X} \in \mathbb{R}^{|V|\times d}$ is the node feature matrix, and $\mathbf{y} \in \mathbb{R}^{|V|}$ is the vector of node labels. Let C be the set of nodes classes. Let $\mathbf{d} \in \mathbb{N}^{|V|}$ be the vector of node degrees. For node $i \in V$, we denote the node feature vector and node label as $\mathbf{x}(i)$ and $y(i)$, respectively. We denote its degree and its set of one-hop neighbors as $d(i)$ and $\mathcal{N}(i)$, respectively. Let $\hat{\mathcal{N}}(i)$ be the set of one-hop neighbors after the addition of a self-loop.

2.2 Node Classification on a Static Graph

Let G be a static graph with adjacency matrix $\mathbf{A}$ from the space of adjacency matrices $\mathcal{A}$. Given a random sample of node representations $\mathbf{X}_{\text{train}} = \{\mathbf{x}(0), \ldots, \mathbf{x}(n_{\text{train}})\}$ from input space $\mathcal{X}$, and their labels $\mathbf{y}_{\text{train}} = \{y(0), \ldots, y(n_{\text{train}})\}$ from output space $\mathcal{Y}$, the node classification task on a static graph is to learn a classifier $f : \mathcal{X} \times \mathcal{A} \to \mathcal{Y}$ such that the expected loss $\mathbb{E}[\mathcal{L}(f(\mathbf{x}, \mathbf{A}), y)]$ over the training set is minimized, where $\mathcal{L}$ is some loss function. Once learned, f can be used to label the remaining nodes in the graph.

2.3 Message-Passing GNNs and Homophily

GNNs leverage the graph structure $\mathbf{A}$ and node feature matrix $\mathbf{X}$ to learn new representations $\mathbf{H} \in \mathbb{R}^{|V|\times h}$. Most GNNs follow a message-passing scheme, where each GNN layer updates each node's representation using the representations of its immediate neighbors (Gilmer et al., 2017). Formally, the l-th layer of a GNN can be summarized at the node level as the following propagation rule for all $i \in V$,

$$\mathbf{h}^{(l+1)}(i) = \textsc{Aggregate}^{(l)}(\{\mathbf{h}^{(l)}(j) : j \in \hat{\mathcal{N}}(i)\}), \quad (1)$$

where $\mathbf{h}^{(l)}(j)$ is the representation for node j at layer l of a GNN and Aggregate is a function that treats $\hat{\mathcal{N}}(i)$ as a set of nodes. In our work, we focus on linear GCNs which are GNNs that leverage mean aggregation determined by the propagation rule:

$$\mathbf{h}^{(l+1)}(i) = \frac{1}{d(i)+1} \sum_{j \in \hat{\mathcal{N}}(i)} \mathbf{h}^{(l)}(j). \quad (2)$$

GCNs are widely used due to their high performance in many tasks (Kipf and Welling, 2017; Wu et al., 2019). Linear GCNs have also been rigorously analyzed due to their simplicity, and various limitations of message passing GNNs have been discovered, such as oversmoothing and the heterophily problem (Li et al., 2018; Zhu et al., 2020). Similarly, in our analysis, we leverage linear GCNs as a key tool in demonstrating the limitations of GNNs in dynamic node classification.

Homophily is the probability that a node forms an edge with another node with the same label Zhu et al. (2020). Intuitively, when homophily is high, message passing is beneficial since different classes will be well separated in feature space after message passing. Formally, given a static graph G, static homophily measured with respect to all nodes in V is defined as:

$$h^S = \mathbb{P}(y(i) = y(j) \mid j \in \mathcal{N}(i)). \quad (3)$$

2.4 Our Problem Setting: Node Classification on a Dynamic Graph

A dynamic graph $\boldsymbol{G}_{0:T}$ is defined as a sequence of static graphs $\boldsymbol{G}_{0:T} = \{G_0, \ldots, G_T\}$, where $T+1$ is the number of total timesteps. The static graph at time t

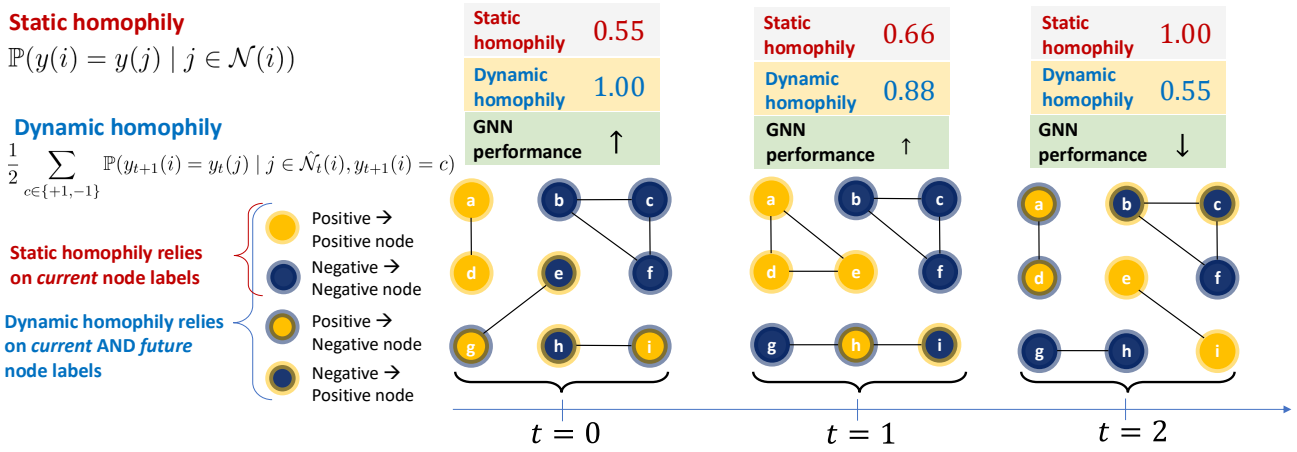


Figure 1: Toy dynamic graph where the task is to predict future node labels. When $t = 0$, GNNs obtain good performance in predicting future node labels. Dynamic homophily is high since all nodes have the same *future label* as their neighbors' current label while, static homophily is low since many nodes (e, g, h, i) and their neighbors have different labels at the *current timestep*. When $t = 2$, GNNs obtain poor performance. Dynamic homophily is low since many nodes (a, b, c, d) have a different *future label* than their neighbors' current label, while static homophily is high since all nodes and their neighbors have the same current label at $t = 2$.

is defined as $G_t = (V_t, \mathbf{A}_t, \mathbf{X}_t, \mathbf{y}_t)$ where V_t is the set of nodes at time t, $\mathbf{A}_t \in \{0,1\}^{|V|\times|V|}$ is the adjacency matrix at time t, $\mathbf{X}_t \in \mathbb{R}^{|V_t|\times d}$ is the feature matrix at time t, and $\mathbf{y}_t \in \mathbb{R}^{|V|}$ is the label vector at time t. Given training data $\{G^{(k)}_{0:T}\}_{k=1}^{n_{\text{train}}}$, the goal of node classification on dynamic graphs is to learn a classifier $f : \mathcal{X}_t \times \mathcal{A}_t \times \mathcal{Y}_t \to \mathcal{Y}_{t+1}$ such that the expected loss $\mathbb{E}[\mathcal{L}(f(x_t, \mathbf{A}_t, y_t), y_{t+1})]$ is minimized for all $t \in [0, T]$. f can then be applied to a new dynamic graph, with the goal of predicting labels at the next time step.

3 THEORETICAL ANALYSIS

We first introduce our setup, outlining our framework for characterizing GCN discriminative performance in the dynamic setting. We then present our main results, characterizing GCN discriminative performance in the dynamic setting and providing potential insights on how to design more powerful dynamic GNNs. Based on our theoretical results, we introduce a new measure of homophily, dynamic homophily, that correlates with GNN performance in the dynamic setting. We lastly provide an extensive comparison between our results in the dynamic setting and existing results in the static setting. In Figure 1, we provide an overview of our definitions, building intuition and highlighting the differences between static and dynamic homophily.

3.1 Setup

In this section, we show that the expected distance between nodes of different classes and the variance of nodes at time t characterize GCN discriminative performance at time t. This intermediate result provides a framework for our theoretical analysis. Specifically, by measuring the expected distance and the variance of the node representations after GCN layers at time t, we can quantify GCN discriminative performance at time t. We utilize the following assumptions.

Assumptions. Let $\boldsymbol{G}_{0:T}$ be a dynamic graph. For all $t \in [0, T]$ and for all $i \in V_t$, $y_t(i) \in \{-1, +1\}$, and $x_t(i) \sim N(y_t(i) \cdot \mu_t, \sigma_t^2) \in \mathbb{R}$ where N is the normal distribution. We do not assume any specific temporal process on the labels of $\boldsymbol{G}_{0:T}$, allowing our results to generalize across many dynamic settings. We also do not make any explicit assumptions on the node homophily distribution. We assume $f^{(l)}$ is a linear GCN. While our analysis relies on these typical assumptions (Wu et al., 2019; Zhu et al., 2020) for tractability, we empirically verify our claims when these assumptions do not hold.

Since the task at each time step is binary, we utilize the AUROC as a metric for discriminative power in favor of other metrics such as the misclassification rate which is affected by class imbalances. The AUROC of a GCN is the probability that it ranks a random positive node higher than a random negative node. Thus, it accurately measures a GCN's ability to discriminate between the two classes. The following lemma states that we can explicitly characterize the AUROC of a linear GCN applied at time t in terms of the expected distance between node representations of opposing classes and their variances at time t.

Lemma 3.1. *The expected AUROC of $f^{(l)}$ at timestep*

t can be written as follows,

$$\mathbb{E}[A_t(f^{(l)})] = 1 - \Phi\left(-\frac{\mathbb{E}_{i|i\in V_{t+1}^+}[\mathbf{h}_t^{(l)}(i)] - \mathbb{E}_{j|j\in V_{t+1}^-}[\mathbf{h}_t^{(l)}(j)]}{\mathbb{V}_{i,j|i\in V_{t+1}^+, j\in V_{t+1}^-}[\mathbf{h}_t^{(l)}(i) + \mathbf{h}_t^{(l)}(j)]}\right) \tag{4}$$

where Φ is the cumulative distribution function of the Gaussian distribution, and V_{t+1}^+ and V_{t+1}^- are positive and negative nodes at time $t+1$, respectively.

We prove Lemma 3.1 in Appendix B.1. Lemma 3.1 tells us that the expected AUROC at time t is monotonically increasing in the ratio of expected distance over the variances of the nodes at time t. Intuitively, an increase in the expected distance increases the distance *between* the two classes, while a decrease in the variances decreases the distance *within* the two classes.

3.2 Main Results

To characterize GCN discriminative performance, we first measure the expected distance in node representations between the positive and negative classes after l GCN layers, showing that the distance is characterized by the probability that node future labels are the same as their neighbors' current labels. We next show that node representation variances of the positive and negative classes after l GCN layers are also characterized by this probability. Based on these findings, we propose dynamic homophily, a new measure of homophily for dynamic node classification that better reflects GNN discriminative power compared to static homophily. Our results further provide potential insights in how to design more powerful GNNs in the dynamic setting. In the following theorem, we present our first result measuring the expected difference in node representations after l GCN layers.

Theorem 3.2. *At time t, the difference in expected node representations between a future positive and negative node after l layers of a GCN can be expressed as:*

$$\mathbb{E}_{i|i\in V_{t+1}^+}[\mathbf{h}_t^{(l)}(i)] - \mathbb{E}_{j|j\in V_{t+1}^-}[\mathbf{h}_t^{(l)}(j)] = 2\cdot\mu_t\cdot(h_t^+ + h_t^- - 1)^l. \tag{5}$$

where h_t^+ is the probability node i's label at time $t+1$ is the same as its neighbor's label at time t given node i's label is positive at $t+1$ such that:

$$h_t^+ = \mathbb{P}(y_{t+1}(i) = y_t(j) \mid j\in\hat{\mathcal{N}}_t(i), y_{t+1}(i) = +1), \tag{6}$$

and h_t^- is the probability node i's label at time $t+1$ is the same as its neighbor's label at time t given node i's label is negative at $t+1$ such that:

$$h_t^- = \mathbb{P}(y_{t+1}(i) = y_t(j) \mid j\in\hat{\mathcal{N}}_t(i), y_{t+1}(i) = -1). \tag{7}$$

We denote h_t^+ and h_t^- as the positive and negative class dynamic homophily levels, respectively.

We prove Theorem 3.2 in Appendix B.2. Before discussing the implications of Theorem 3.2, we first discuss the intuition behind our new homophily measures. The main idea is to measure node neighbors' contributions at time t towards node future labels at time $t+1$. In this manner, the positive and negative class dynamic homophily levels accurately capture relationships across time. Theorem 3.2 tells us that the expected distance in node representations at time t is a function of these homophily levels at time t. Specifically, the distance in node representations is *polynomial* in the sum of the positive and negative class dynamic homophily levels. Thus, increases in the sum of these homophily levels increases the separation of node representations, and we expect the discriminative performance of GCNs to increase.

While it is well understood that in the limit as $l\to\infty$ GCNs *oversmooth*, and the difference in node representations becomes 0, rendering nodes indistinguishable (Li et al., 2018), Theorem 3.2 tells us that the rate of oversmoothing is a function of the dynamic homophily levels, thus bridging the gap between homophily and oversmoothing in the dynamic setting similar to analyses in the static setting (Bodnar et al., 2022; Yan et al., 2022). We now demonstrate that the empirical distance concentrates around its expected distance.

Theorem 3.3. *For $\epsilon > 0$, the probability that at time t the distance between the empirical and expected distance after l GCN layers is larger than ϵ is bounded as follows:*

$$\mathbb{P}(|(\boldsymbol{\mu}_{\boldsymbol{V_{t+1}^+}}^{(\boldsymbol{l})} - \boldsymbol{\mu}_{\boldsymbol{V_{t+1}^-}}^{(\boldsymbol{l})}) - (\mathbb{E}_{i|i\in V_{t+1}^+}[\mathbf{h}_t^{(l)}(i)] - \mathbb{E}_{i|i\in V_{t+1}^-}[\mathbf{h}_t^{(l)}(i)])| \geq \epsilon) \leq \mathcal{O}(e^{-\epsilon^2 L_{t,+}^{(l)}} + e^{-\epsilon^2 L_{t,-}^{(l)}}) \tag{8}$$

,

where $\boldsymbol{\mu}_{\boldsymbol{V_{t+1}^+}}^{(\boldsymbol{l})}$ and $\boldsymbol{\mu}_{\boldsymbol{V_{t+1}^-}}^{(\boldsymbol{l})}$ are the empirical mean representations after l GCN layers over future positive and negative nodes, respectively, and

$$L_{t,+}^{(l)} = \frac{\left|V_{t+1}^+\right|^2}{\sigma^4\cdot\sum_{i\in V_{t+1}^+}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2} \tag{9}$$

$$L_{t,-}^{(l)} = \frac{\left|V_{t+1}^-\right|^2}{\sigma^4\cdot\sum_{i\in V_{t+1}^-}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2}. \tag{10}$$

We prove Theorem 3.3 in Appendix B.3. Theorem 3.3 tells us that the distance at time t between the empirical means and the expected representations after l GCN layers is small with high probability. In particular, this probability depends exponentially on $\left|V_{t+1}^+\right|$, the number of positive nodes, $\left|V_{t+1}^-\right|$, the number of

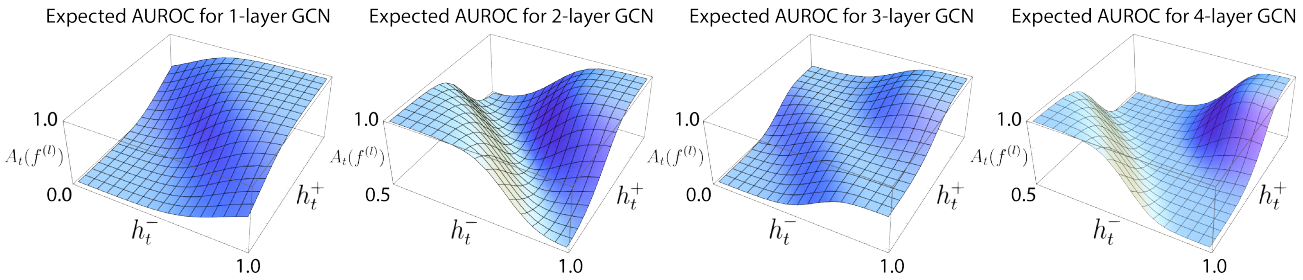


Figure 2: Expected AUROC across GCN layers as a function of dynamic homophily levels. The AUROC of odd layer GCNs is monotonically increasing in dynamic homophily levels, while the AUROC of even layer GCNs increase as both dynamic homophily levels approach 0 and 1, providing potential insights into how to design more powerful dynamic GNNs that can adapt to dynamic homophily levels across time.

negative nodes, and $d_t(i)$, the node degrees. If these quantities are large, the empirical means are close to their expectations with high probability, and the sum of the positive and negative dynamic homophily levels correlate with the empirical distance at time t. Our next theorem upper bounds GCN discriminative performance in terms of positive and negative class dynamic homophily levels.

Theorem 3.4. *The expected AUROC of $f^{(l)}$ at timestep t can be upper bounded as follows,*

$$\mathbb{E}[A_t(f^{(l)})] \leq 1 - \Phi\left(-\frac{2 \cdot \mu_t \cdot (h_t^+ + h_t^- - 1)^l}{v_{t+1}^+(l) + v_{t+1}^-(l)}\right). \tag{11}$$

where $v_{t+1}^+(l)$ and $v_{t+1}^-(l)$ are the lower bounds of the variances of the future positive and negative nodes after l GCN layers, respectively, and are defined recursively in terms of the dynamic homophily levels as follows,

$$\begin{aligned} v_{t+1}^+(l) &= {h_t^+}^2 \cdot v_{t+1}^+(l-1) + (1-h_t^+)^2 \cdot v_{t+1}^-(l-1) \\ v_{t+1}^-(l) &= {h_t^-}^2 \cdot v_{t+1}^-(l-1) + (1-h_t^-)^2 \cdot v_{t+1}^+(l-1) \\ v_{t+1}^+(0) &= v_{t+1}^-(0) = \sigma_t^2. \end{aligned} \tag{12}$$

Moreover the probability that at time t the distance between the empirical AUROC and the expected AUROC of a l layer GCN is larger than ϵ is bounded as follows,

$$\mathbb{P}\left(\left|A_t(f^{(l)}) - \mathbb{E}[A_t(f^{(l)})]\right| \geq \epsilon\right) \leq e^{\frac{-2 \cdot |V_{t+1}^+| \cdot |V_{t+1}^-| \cdot \epsilon^2}{|V_{t+1}^+| + |V_{t+1}^-|}} \tag{13}$$

We prove Theorem 3.4 in Appendix B.5. Theorem 3.4 tells us that for GCNs with an odd number of layers, the AUROC upper bound is monotonically increasing in the dynamic homophily levels for the positive and negative class, while for GCNs with an even number of layers the upper bound is monotonically increasing as both homophily levels approach 0 and 1. To demonstrate this intuition, consider 1 and 2-layer GCNs along binary stochastic block models (SBMs). When the positive and negative class homophily levels increase, more edges within communities are present, and a 1-layer GCN's AUROC can only increase. When both homophily levels are low, the SBM resembles a bipartite graph, and 2-layer GCNs are able to recover the correct representations.

We illustrate Theorem 3.4 by visualizing the AUROC upper bound as a function of positive and negative dynamic class homophily levels for different GCN layers in Figure 2. Notice that we recover known mid-homophily pitfalls in the static setting (Luan et al., 2024b) since the worst AUROC is obtained when both positive and negative dynamic class homophily levels are 0.5. More generally, the best AUROC across different positive and negative class dynamic homophily levels is achieved by different layers of a GCN. This result leads to potential insights in how to design more powerful dynamic GNNs that can adapt to both high and low dynamic homophily settings across time. Specifically, one way to overcome low dynamic homophily while maintaining high performance in high dynamic homophily settings is to leverage the intermediate representations of GCNs similar to designs in the static setting (Zhu et al., 2020) since even layers can recover from low dynamic homophily, while odd layers may obtain the best AUROC in high dynamic homophily settings. This allows a GCN to obtain the best performance across different positive and negative class dynamic homophily levels as the dynamic graph changes across time.

New Homophily Measure for Dynamic Graphs. Our investigation of GCN discriminative performance in the dynamic setting suggests a new homophily measure for the dynamic setting. In the binary case, we define the overall dynamic homophily as the average of the positive and negative dynamic homophily levels. Defining dynamic homophily as an average correctly accounts for a GNNs ability to discriminate between each of the classes since if the average decreases the distance between classes also decreases. More generally, in the multiclass case, we use the following definition:

Definition 3.5 (Dynamic homophily)**.** The dynamic homophily at timestep t is defined as,

$$h_t^D = \frac{1}{|C|} \sum_{c \in C} h_t^c, \text{ where} \tag{14}$$

$$h_t^c = \mathbb{P}(y_{t+1}(i) = y_t(j) \mid j \in \hat{\mathcal{N}}_t(i), y_{t+1}(i) = c). \tag{15}$$

In the dynamic multiclass setting, there is no single metric that fully captures the performance of a GNN similar to the static multiclass setting. In order to better capture performance in multiclass settings, in Appendix C.1 we propose and analyze the dynamic compatibility matrix, an extension of the class compatibility in static settings Zhu et al. (2021, 2023). Moreover, while our definition for dynamic homophily assumes a discrete dynamic graph, dynamic homophily can be easily extended to continuous dynamic graphs, and in Appendix C.2 we propose a straightforward extension.

4 EXPERIMENTAL SETUP

To demonstrate that our theoretical results hold in real-world dynamic graphs and for a variety of GNNs, we test if dynamic homophily correlates well with GNN performance across dynamic node classification tasks. We first consider tasks where the goal is to predict the spread of a signal since these tasks are fundamental problems arising in a variety of domains (Centola and Macy, 2007; Guilbeault et al., 2018). For example in public health, epidemiologists are interested in how a disease spreads, or in social network analysis, moderators aim to prevent the spread of misinformation (Leskovec et al., 2007; Gomez-Rodriguez et al., 2013). We next consider a biological task where the goal is to determine active proteins at each timestep, extending our experiments beyond signal spreading. Determining active proteins is an important task since it elucidates signaling pathways, information flow, and various protein functions (Przytycka et al., 2010; Holme and Saramäki, 2012). We explore the relationship between dynamic homophily and GNN performance across time, considering both standard GNN designs and designs that aim to address low static homophily.

4.1 Pseudo-synthetic Graph Datasets

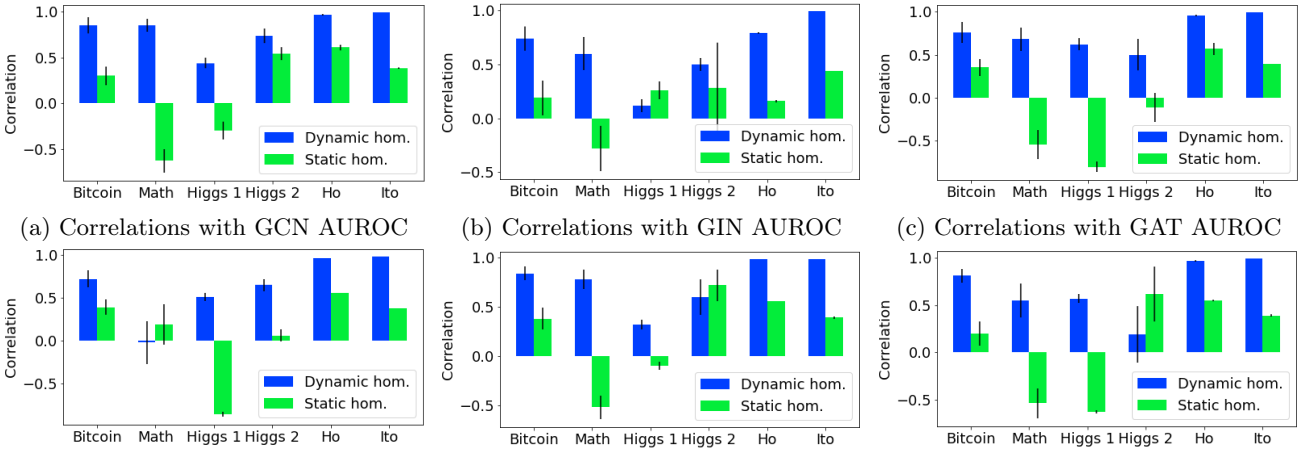
To model the spread of infectious disease, we start by generating the dynamic graph structure. Given the structure, we generate features and labels based on an epidemiological model. In generating our structures, we utilize real-world dynamic graphs representing various types of social interactions: **UCI**, a social network at the University of California Irvine (Panzarasa et al., 2009), **Bitcoin**, a transaction network from a Bitcoin platform (Kumar et al., 2016), and **Math**, a forum network on the website Math Overflow (Paranjape et al., 2017). In each network, structure changes over time such that $\mathbf{A}_t \neq \mathbf{A}_{t+1}$ for all t.

Given these real network structures, we synthetically generate labels and features using the susceptible-infected (SI) epidemiological model (Kermack and McKendrick, 1927; Newman, 2002). Using the SI model, we independently generate 60 dynamic graphs for each structure. Each graph is generated by first sampling node infection parameters and statuses at $t = 0$, then simulating the SI process. Here, node features include a node's infectivity and susceptibility parameters used in determining its future label and infection status at time t. With these dynamic graphs, we randomly split these data into equally sized sets of 20 graphs for training, validation, and testing. Thus, all nodes in the test set are unseen since the dynamic graphs are new.

4.2 Real-world Dynamic Graph Datasets

Higgs Networks. We utilize dynamic graphs from the Higgs dataset (De Domenico et al., 2013), a collection of dynamic social networks. The dataset monitors the spread of information about the discovery of the Higgs boson on Twitter before, during, and after its announcement. Here, a node's feature vector is a learnable node embedding concatenated with whether or not the information has yet reached the node. The task is to predict the times at which the information will reach each user. In our experiments, we consider four separate dynamic graphs each spanning 24 hours: **Higgs 1**, **Higgs 2**, **Higgs 3**, and **Higgs 4**, each representing different days of the Twitter network. For each dynamic graph we split the graph chronologically into 7 train graphs, 7 validation graphs, and 10 test graphs. In the Higgs networks, all nodes appear in the train, validation, and test sets. Each dynamic graph exhibits vastly different spreading behaviors since each day corresponds to a different period of the spreading process (De Domenico et al., 2013).

Protein-Protein Interaction Networks. We also utilize dynamic graphs from a biological repository of dynamic protein-protein interaction networks (DPPIN) (Fu and He, 2022). The datasets consist of dynamic protein-protein interactions of yeast cells where the task is to predict the times at which proteins are active. Here, node features include information describing proteins such as protein class and whether proteins are unknown or verified. Node features also include active status at time t. In our experiments, we consider four separate dynamic graphs: **Gavin**, **Ho**, **Ito**, and **Uetz** each spanning 36 timestamps of different protein interaction networks. We split the graph chronologically into 10 train graphs, 10 validation graphs, and 16 test



(a) Correlations with GCN AUROC (b) Correlations with GIN AUROC (c) Correlations with GAT AUROC

(d) Correlations with SAGE AUROC (e) Correlations with GCNII AUROC (f) Correlations with FA-GCN AUROC

Figure 3: Mean ± standard deviations of Spearman's rank correlation coefficient between GNN AUROC and homophily measures across graphs in the test set for a subset of datasets. For most GNN and dynamic graph combinations, dynamic homophily has a higher correlation with GNN performance compared to static homophily.

graphs. Similar to the Higgs networks, all nodes appear in the train, validation, and test sets. The full details of all dynamic graphs are summarized in Appendix D[1].

4.3 Training and Evaluation Details

Models. To explore the differences between homophilous and heterophilous GNNs, we train and evaluate many different GNNs: **SGC** (Wu et al., 2019), **GCN** (Kipf and Welling, 2017), **GIN** (Xu et al., 2019), and **GAT** (Veličković et al., 2018) are homophilous baselines since they do not explicitly include designs to address heterophily, while **SAGE** (Hamilton et al., 2017), **GCNII** (Li et al., 2018), and **FA-GCN** (Bo et al., 2021) are heterophilous since they adopt additional designs to improve performance in heterophilous settings (Loveland et al., 2023; Zhu et al., 2023). Given the construction of the dynamic graphs and existing results in Fu and He (2022), we do not expect dynamic GNNs that leverage the full temporal signal to perform better than static ones. Yet for completeness, in the Appendix we include results on the following dynamic GNNs: **DGNN** (Manessi et al., 2020; Narayan and Roe, 2018; Chen et al., 2022), **GCRN** (Seo et al., 2018), and **EvolveGCN** (Pareja et al., 2020), where we find consistent results for the dynamic GNNs.

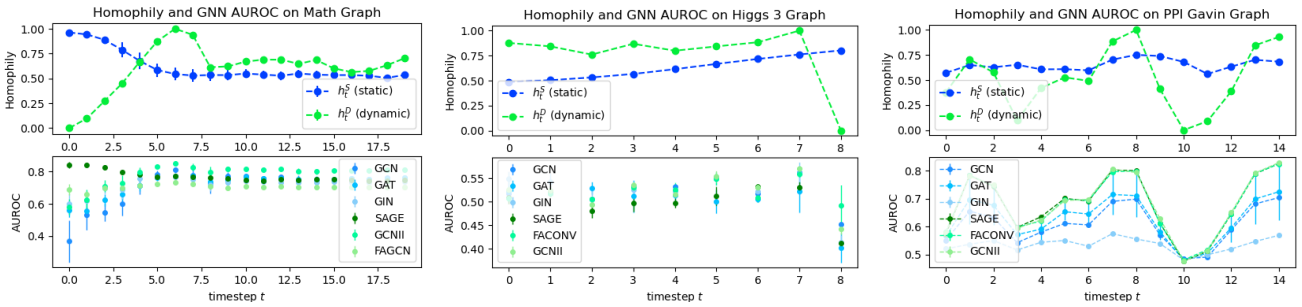
Training and Evaluation. During training, we apply a GNN to each static graph at time t making predictions about the labels $\mathbf{y}_{t+1}$ at the next timestep, minimizing the binary cross entropy loss. In the infectious disease and Higgs networks we only consider predictions for nodes that the signal has not reached. We evaluate each model on the same held out test set of graphs, measuring the AUROC. We then report the mean AUROC across time for each dynamic graph, and the mean and standard deviation of the AUROCs across all graphs in the test set. To measure to what extent dynamic homophily captures GNN discriminative power, we measure Spearman's rank correlation coefficient between dynamic homophily and GNN performance across time for each graph in the test set. We report the mean and standard deviation of the correlations across the graphs in the test set. For comparison, we also measure the correlation between static homophily with respect to snapshots of the dynamic graph and GNN performance. We provide training and evaluation details including computation of dynamic and static homophily, hyperparameter procedures, and reproducibility guidelines in Appendix D.

[1]Code can be found at: https://github.com/MLD3/UnderstandingDynamicGraphs

5 EXPERIMENTAL RESULTS

We empirically compare a variety of GNNs based on their performance as dynamic homophily changes. We aim to answer the following research questions:

- **RQ1**: To what extent do dynamic homophily and static homophily, based on snapshots in time, correlate with GNN discriminative performance in general dynamic node classification tasks where our assumptions made in our analyses are violated?
- **RQ2**: Do the observed trends change when considering GNNs specifically designed to perform well in settings with low static homophily? And how do such GNNs perform in settings with low dynamic homophily?



(a) Math dynamic graph (b) Higgs 3 dynamic graph (c) Gavin dynamic graph

Figure 4: Mean and standard deviations of dynamic homophily, static homophily, and AUROC across all graphs in the test set for the Math, Higgs 3, and Gavin dynamic graph. For all three dynamic graphs, static homophily stays high across time, while dynamic homophily and GNN performances exhibit the same trends. We do not connect AUROCs across signal spreading tasks since we only predict for nodes which the signal has not reached.

Table 1: Mean ± standard deviation of GNN AUROC across time for each dynamic graph in the test set. In general, GNNs with heterophilous designs (SAGE, GCNII, FA-GCN) outperform GNNs with homophilous designs (SGC, GCN, GIN, GAT), suggesting that heterophilous design choices can help alleviate low dynamic homophily.

	Epidemiological Dynamic Graphs			Higgs Dynamic Graphs			Protein-protein Dynamic Graphs		
GNN	**UCI**	**Bitcoin**	**Math**	**Higgs 1**	**Higgs 2**	**Higgs 3**	**Gavin**	**Ho**	**Ito**
SGC (*hom.*)	0.84 ± 0.01	0.85 ± 0.01	0.82 ± 0.01	0.54 ± 0.02	0.53 ± 0.01	**0.53 ± 0.00**	0.60 ± 0.00	0.60 ± 0.00	0.67 ± 0.00
GCN (*hom.*)	0.84 ± 0.01	0.85 ± 0.01	0.82 ± 0.01	0.56 ± 0.01	0.54 ± 0.01	0.52 ± 0.01	0.61 ± 0.00	0.60 ± 0.00	0.67 ± 0.00
GIN (*hom.*)	0.85 ± 0.01	**0.88 ± 0.01**	0.86 ± 0.01	0.49 ± 0.00	0.52 ± 0.01	0.52 ± 0.00	0.54 ± 0.00	0.53 ± 0.00	0.63 ± 0.00
GAT (*hom.*)	0.79 ± 0.01	0.84 ± 0.01	0.81 ± 0.01	0.60 ± 0.02	0.51 ± 0.01	0.50 ± 0.01	0.63 ± 0.03	0.60 ± 0.01	**0.68 ± 0.00**
SAGE (*het.*)	**0.86 ± 0.01**	0.87 ± 0.01	**0.88 ± 0.01**	**0.61 ± 0.00**	**0.55 ± 0.00**	0.50 ± 0.01	**0.68 ± 0.00**	0.68 ± 0.00	**0.68 ± 0.00**
GCNII (*het.*)	0.84 ± 0.01	**0.88 ± 0.01**	**0.88 ± 0.01**	0.56 ± 0.01	0.53 ± 0.00	0.52 ± 0.01	**0.68 ± 0.00**	**0.69 ± 0.00**	**0.68 ± 0.00**
FA-GCN (*het.*)	0.79 ± 0.01	0.83 ± 0.01	0.80 ± 0.01	0.57 ± 0.01	0.54 ± 0.01	0.52 ± 0.00	**0.68 ± 0.00**	**0.69 ± 0.00**	**0.68 ± 0.00**

5.1 (RQ1) How does static and dynamic homophily correlate with GNN AUROC?

In Figure 3, we compare average correlations obtained by dynamic and static homophily with GNN performance for a representative subset of GNN and dynamic graph combinations. We observe similar trends on the set of full results and present them in the Appendix A. Across 43 out of 44 homophilous GNN and dynamic graph combinations, the average correlation between dynamic homophily and GNN performance exceeds the correlation between static homophily and GNN performance. Across the 44 combinations, dynamic homophily achieves a median correlation of 0.75 interquartile range (IQR): (0.59, 0.96) which is significantly greater than the median correlation achieved by static homophily of 0.26 IQR: (-0.15, 0.47).

Data in Figure 3 are averaged across graphs in the test set, but also across time. To fully capture changes to GNN performance, we measure AUROC, dynamic homophily, and static homophily at each timestep for the Math, Higgs 3, and Gavin graphs (Figure 4). While dynamic homophily exhibits the same trends as homophilous GNN performance, static homophily does not since it stays high across all three graphs for all timesteps. We observe consistent results on the remaining graphs and include them in Appendix E.

5.2 (RQ2) How do heterophilous GNNs perform in low dynamic homophily?

Similar to trends across homophilous GNNs, we find that dynamic homophily also correlates well with heterophilous GNN performance, while static homophily does not. Across the 33 combinations of heterophilous GNNs and dynamic graphs, dynamic homophily achieves a median correlation of 0.78 IQR: (0.55, 0.97) which is significantly greater than the median correlation achieved by static homophily of 0.38 IQR: (0.04, 0.54). GNNs with heterophilous designs also tend to perform higher on average compared to GNNs with homophilous designs (Table 1). In Figure 4, we compare the performance across time between the two design types. When dynamic homophily is high, the task is generally easy, and we observe strong discriminative performance from both homophilous and heterophilous GNNs. However, when dynamic homophily is low, the task becomes much more difficult, and performance gaps emerge between the two design types in favor of the heterophilous GNNs. The results suggest that heterophilous designs in the static setting could also alleviate low dynamic homophily in the dynamic setting.

Although dynamic homophily generally correlates well with the discriminative performance for GNNs of dif-

ferent designs and dynamic graph datasets, there are specific combinations of GNNs and dynamic graphs in which dynamic homophily is only weakly correlated with GNN performance. Specifically, the performance of **SAGE** does not correlate with dynamic homophily for the Math graph. This may be due to several limitations. First, our theoretical analysis relies on the GCN aggregation. Thus, dynamic homophily is not guaranteed to correlate well with GNNs that leverage more complex aggregations. Second, as shown in Theorems 3.3 and 3.4, dynamic homophily may not reflect the distance in node representations at a particular timestep if there are too few nodes in both classes at that timestep.

6 DISCUSSION AND CONCLUSION

In the context of node classification on dynamic graphs, we present the first theoretical and empirical analysis of GNNs and homophily. Specifically, we analyze GCNs and characterize their discriminative performance in dynamic settings. Based on our analysis, we propose dynamic homophily, a new homophily measure that characterizes GCN discriminative power in a dynamic setting.

Our work builds on previous work that has focused on GNN performance and homophily in static settings. More specifically, Ma et al. (2022), Yan et al. (2022), Li et al. (2022) and Zhu et al. (2024b) all measured GCN node representations and their relationship to node-level homophily in static graphs. Their analyses focus on distance between classes in order to improve GNN performance in static settings. We also look at the node representations, but in addition to studying the distance between classes, we analyze the distance within classes which provides additional insights into designing better GNNs for dynamic settings.

Our analysis suggests that leveraging intermediate GNN representations can mitigate low dynamic homophily and allow a GNN to adapt to the dynamic graph as it changes across time. In our empirical analyses, we demonstrate that dynamic homophily is highly correlated with the performance of many variations of GNNs across a variety of dynamic node classification tasks from epidemiology, social network analysis, and molecular biology. More broadly, our results indicate that popular homophilous and heterophilous GNNs are not robust to low dynamic homophily, and as a result new approaches may be warranted. Going forward, our results have the potential to inform future GNN designs. Moreover, our work is an important building block for future work that can analyze more complex dynamic settings such as homophily in the context of local performance discrepancies extending the global view of dynamic homophily and general dynamic graph tasks beyond node classification. Overall, our work represents a step toward understanding the discriminative power of GNNs in the dynamic setting.

Acknowledgements

This material is based on work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Department of Energy Computational Science Graduate Fellowship under Award Number DE-SC0023112. It was also partially supported by National Science Foundation under Grant No. IIS 2212143. We thank the anonymous reviewers and members of the MLD3 lab for their valuable feedback.

References

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.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. volume 33, pages 7793–7804, 2020.

Donald Loveland and Danai Koutra. Unveiling the impact of local homophily on gnn fairness: In-depth analysis and new benchmarks. In *SIAM International Conference on Data Mining*, 2025.

Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks? In *International Conference on Learning Representations*, 2022.

Jiong Zhu, Gaotang Li, Yao-An Yang, Jing Zhu, Xuehao Cui, and Danai Koutra. On the impact of feature heterophily on link prediction with graph neural networks. In *Advances in Neural Information Processing Systems*, 2024a.

Sitao Luan, Chenqing Hua, Qincheng Lu, Liheng Ma, Lirong Wu, Xinyu Wang, Minkai Xu, Xiao-Wen Chang, Doina Precup, Rex Ying, Stan Z. Li, Jian Tang, Guy Wolf, and Stefanie Jegelka. The heterophilic graph learning handbook: Benchmarks, models, theoretical analysis, applications and challenges, 2024a. URL `https://arxiv.org/abs/2407.09618`.

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

Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3950–3957, 2021.

Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. Graph neural networks with heterophily. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11168–11176, 2021.

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. IEEE, 2022.

Michael Ito, Jiong Zhu, Dexiong Chen, Danai Koutra, and Jenna Wiens. Learning laplacian positional encodings for heterophilous graphs. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. PMLR, 2025.

Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. Characterizing graph datasets for node classification: Homophily-heterophily dichotomy and beyond. *Advances in Neural Information Processing Systems*, 36, 2024.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

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

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.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Liò, and Michael M. Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in GNNs. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

Sitao Luan, Chenqing Hua, Minkai Xu, Qincheng Lu, Jiaqi Zhu, Xiao-Wen Chang, Jie Fu, Jure Leskovec, and Doina Precup. When do graph neural networks help with node classification? investigating the homophily principle on node distinguishability. *Advances in Neural Information Processing Systems*, 36, 2024b.

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*, 2023.

Damon Centola and Michael Macy. Complex contagions and the weakness of long ties. *American journal of Sociology*, 113(3):702–734, 2007.

Douglas Guilbeault, Joshua Becker, and Damon Centola. Complex contagions: A decade in review. *Complex spreading phenomena in social systems: Influence and contagion in real-world social networks*, pages 3–25, 2018.

Jure Leskovec, Lada A Adamic, and Bernardo A Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5–es, 2007.

Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Modeling information propagation with survival theory. In *International conference on machine learning*, pages 666–674. PMLR, 2013.

Teresa M Przytycka, Mona Singh, and Donna K Slonim. Toward the dynamic interactome: it's about time. *Briefings in bioinformatics*, 11(1):15–29, 2010.

Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the American Society for Information Science and Technology*, 60(5):911–932, 2009.

Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 221–230. IEEE, 2016.

Ashwin Paranjape, Austin R Benson, and Jure Leskovec. Motifs in temporal networks. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 601–610, 2017.

William Ogilvy Kermack and Anderson G McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

Mark EJ Newman. Spread of epidemic disease on networks. *Physical review E*, 66(1):016128, 2002.

Manlio De Domenico, Antonio Lima, Paul Mougel, and Mirco Musolesi. The anatomy of a scientific rumor. *Scientific reports*, 3(1):2980, 2013.

Dongqi Fu and Jingrui He. Dppin: A biological repository of dynamic protein-protein interaction network data. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 5269–5277. IEEE, 2022.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. accepted as poster.

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

Donald Loveland, Jiong Zhu, Mark Heimann, Benjamin Fish, Michael T. Schaub, and Danai Koutra. On performance discrepancies across local homophily levels in graph neural networks. In *Learning on Graphs Conference (LOG)*, volume 231 of *Proceedings of Machine Learning Research*, page 6. PMLR, 2023.

Franco Manessi, Alessandro Rozza, and Mario Manzo. Dynamic graph convolutional networks. *Pattern Recognition*, 97:107000, 2020.

Apurva Narayan and Peter HO'N Roe. Learning graph dynamics using deep neural networks. *Ifac-Papersonline*, 51(2):433–438, 2018.

Jinyin Chen, Xueke Wang, and Xuanheng Xu. Gc-lstm: Graph convolution embedded lstm for dynamic network link prediction. *Applied Intelligence*, pages 1–16, 2022.

Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25*, pages 362–373. Springer, 2018.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5363–5370, 2020.

Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. Finding global homophily in graph neural networks when meeting heterophily. In *International Conference on Machine Learning*, pages 13242–13256. PMLR, 2022.

Jing Zhu, Xiang Song, Vassilis N. Ioannidis, Danai Koutra, and Christos Faloutsos. Touchup-g: Improving feature representation through graph-centric finetuning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2662–2666. ACM, 2024b.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Shivani Agarwal, Thore Graepel, Ralf Herbrich, Sariel Har-Peled, Dan Roth, and Michael I Jordan. Generalization bounds for the area under the roc curve. *Journal of Machine Learning Research*, 6(4), 2005.

Derek Lim, Felix Matthew Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Prasad Bhalerao, and Ser-Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278, 2019.

Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal graph benchmark for machine learning on temporal graphs. *Advances in neural information processing systems*, 2023.

Aric Hagberg and Drew Conway. Networkx: Network analysis with python. *URL: https://networkx. github. io*, 2020.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

Checklist

1. For all models and algorithms presented, check if you include:

(a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. **Yes**

(b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. **Yes**

(c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. **Yes**.

2. For any theoretical claim, check if you include:

(a) Statements of the full set of assumptions of all theoretical results. **Yes**

(b) Complete proofs of all theoretical results. **Yes**

(c) Clear explanations of any assumptions. **Yes**

3. For all figures and tables that present empirical results, check if you include:

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). **No**

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). **Yes**

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). **Yes**

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). **Yes**

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets. **Yes**

(b) The license information of the assets, if applicable. **Not Applicable**

(c) New assets either in the supplemental material or as a URL, if applicable. **Not Applicable**

(d) Information about consent from data providers/curators. **Not Applicable**

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. **Not Applicable**

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. **Not Applicable**

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. **Not Applicable**

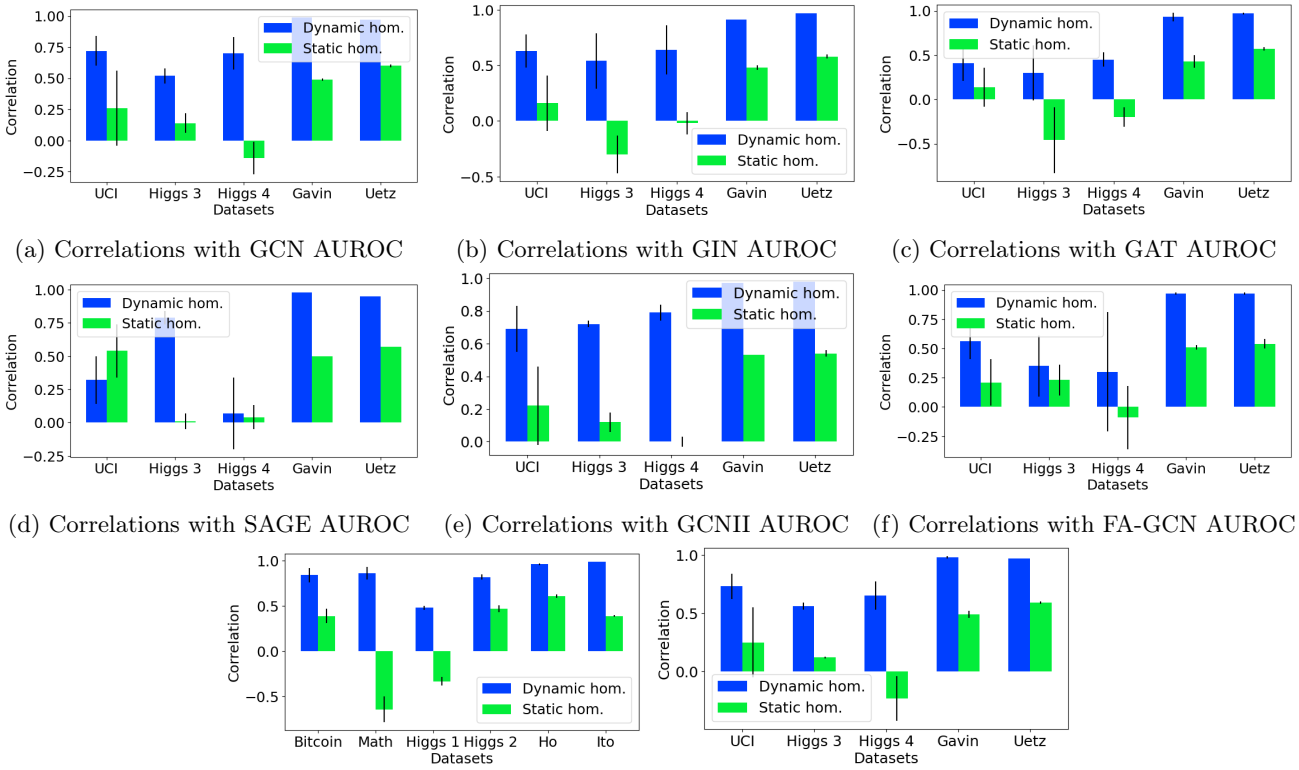
(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. **Not Applicable**

Supplementary Material

Contents

A REMAINING EXPERIMENTAL RESULTS

In Figure 5, we present the average correlations obtained by dynamic and static homophily with GNN performance on the remaining GNN and dataset combinations. We observe consistent trends from the main paper and for most combinations dynamic homophily obtains a higher correlation than static homophily. In Table 2, we compare the average performance obtained by all GNNs on all dynamic graph datasets. We observe consistent trends from the main paper and generally GNNs with heterophilous designs outperform GNNs with homophilous designs.



(a) Correlations with GCN AUROC (b) Correlations with GIN AUROC (c) Correlations with GAT AUROC

(d) Correlations with SAGE AUROC (e) Correlations with GCNII AUROC (f) Correlations with FA-GCN AUROC

(g) Correlations with SGC AUROC (h) Correlations with SGC AUROC

Figure 5: Mean ± standard deviations of Spearman's rank correlation coefficient between GNN AUROC and homophily measures across graphs in the test set for the remaining dynamic graph datasets. For most GNN and dynamic graph combinations, dynamic homophily has a higher correlation with GNN performance compared to static homophily.

Table 2: Mean ± standard deviation of GNN AUROC across time for each dynamic graph in the test set. In general, GNNs with heterophilous designs (SAGE, GCNII, FA-GCN) outperform GNNs with homophilous designs (SGC, GCN, GIN, GAT), suggesting that heterophilous design choices can help alleviate low dynamic homophily in the dynamic setting.

	Epidemiological Dynamic Graphs			Higgs Dynamic Graphs				Protein-protein Dynamic Graphs			
GNN	**UCI**	**Bitcoin**	**Math**	**Higgs 1**	**Higgs 2**	**Higgs 3**	**Higgs 4**	**Gavin**	**Ho**	**Ito**	**Uetz**
SGC (*hom.*)	0.84 ± 0.01	0.85 ± 0.01	0.82 ± 0.01	0.54 ± 0.02	0.53 ± 0.01	**0.53 ± 0.00**	0.53 ± 0.01	0.60 ± 0.00	0.60 ± 0.00	0.67 ± 0.00	0.65 ± 0.00
GCN (*hom.*)	0.84 ± 0.01	0.85 ± 0.01	0.82 ± 0.01	0.56 ± 0.01	0.54 ± 0.01	0.52 ± 0.01	**0.53 ± 0.00**	0.61 ± 0.00	0.60 ± 0.00	0.67 ± 0.00	0.65 ± 0.00
GIN (*hom.*)	0.85 ± 0.01	**0.88 ± 0.01**	0.86 ± 0.01	0.49 ± 0.00	0.52 ± 0.01	0.52 ± 0.00	0.52 ± 0.00	0.54 ± 0.00	0.53 ± 0.00	0.63 ± 0.00	0.64 ± 0.00
GAT (*hom.*)	0.79 ± 0.01	0.84 ± 0.01	0.81 ± 0.01	0.60 ± 0.02	0.51 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	0.63 ± 0.03	0.60 ± 0.01	**0.68 ± 0.00**	**0.67 ± 0.00**
SAGE (*het.*)	**0.86 ± 0.01**	0.87 ± 0.01	**0.88 ± 0.01**	**0.61 ± 0.00**	**0.55 ± 0.00**	0.50 ± 0.01	0.49 ± 0.00	**0.68 ± 0.00**	0.68 ± 0.00	**0.68 ± 0.00**	**0.67 ± 0.00**
GCNII (*het.*)	0.84 ± 0.01	**0.88 ± 0.01**	**0.88 ± 0.01**	0.56 ± 0.01	0.53 ± 0.00	0.52 ± 0.01	**0.53 ± 0.00**	**0.68 ± 0.00**	**0.69 ± 0.00**	**0.68 ± 0.00**	**0.67 ± 0.00**
FA-GCN (*het.*)	0.79 ± 0.01	0.83 ± 0.01	0.80 ± 0.01	0.57 ± 0.01	0.54 ± 0.01	0.52 ± 0.00	0.52 ± 0.00	**0.68 ± 0.00**	**0.69 ± 0.00**	**0.68 ± 0.00**	**0.67 ± 0.00**

B PROOFS OF MAIN RESULTS

B.1 Characterization of AUROC after l GCN Layers

Lemma B.1. *The expected AUROC of $f^{(l)}$ at timestep t can be written as follows,*

$$\mathbb{E}[A_t(f^{(l)})] = 1 - \Phi\left(-\frac{\mathbb{E}_{i|i\in V_{t+1}^+}\left[\mathbf{h}_t^{(l)}(i)\right] - \mathbb{E}_{j|j\in V_{t+1}^-}\left[\mathbf{h}_t^{(l)}(j)\right]}{\mathbb{V}_{i,j|i\in V_{t+1}^+, j\in V_{t+1}^-}[\mathbf{h}_t^{(l)}(i) + \mathbf{h}_t^{(l)}(j)]}\right) \tag{16}$$

where Φ is the cumulative distribution function of the Gaussian distribution, and V_{t+1}^+ and V_{t+1}^- are the nodes in the positive and negative classes at time $t+1$, respectively.

Proof. The expected AUROC of $f^{(l)}$ at timestep t is equivalent to the probability that $f^{(l)}$ ranks a random positive node higher than a random negative node. Thus, the expected AUROC of $f^{(l)}$ at timestep t can also be expressed as,

$$\mathbb{E}[A_t(f^{(l)})] = \mathbb{P}(f^{(l)}(i) > f^{(l)}(j) \mid i \in V_{t+1}^+, j \in V_{t+1}^-) \tag{17}$$

$$= \mathbb{P}(f^{(l)}(i) - f^{(l)}(j) > 0 \mid i \in V_{t+1}^+, j \in V_{t+1}^-) \tag{18}$$

Defining random variable Z_t as the linear combination $Z_t = f^{(l)}(i) - f^{(l)}(j)$ given a random node $i \in V_{t+1}^+$ and random node $j \in V_{t+1}^-$, the expected AUROC amounts to the quantity:

$$\mathbb{E}[A_t(f^{(l)})] = 1 - \Phi_{Z_t}(0) \tag{19}$$

where Φ is the cumulative distribution function of the random variable Z_t. Furthermore, since Z_t is Gaussian distributed, we can apply the definition of the CDF for Gaussian distributions and express the expected AUROC at timestep t as the following:

$$\mathbb{E}[A_t(f^{(l)})] = 1 - \Phi_{Z_t}(0) = 1 - \Phi\left(-\frac{\mathbb{E}[Z_t]}{\mathbb{V}[Z_t]}\right) \tag{20}$$

Substituting the definition of Z_t into Equation 20 completes the proof. □

B.2 Expected Distance for l layer GNNs

Theorem B.2. *At time t the difference in expected node representations between a future positive and negative node after l layers of a GCN can be expressed as:*

$$\mathbb{E}_{i|i\in V_{t+1}^+}\left[\mathbf{h}_t^{(l)}(i)\right] - \mathbb{E}_{j|j\in V_{t+1}^-}\left[\mathbf{h}_t^{(l)}(j)\right] = 2 \cdot \mu_t \cdot (h_t^+ + h_t^- - 1)^l. \tag{21}$$

where μ_t is the magnitude of the mean for all $\mathbf{x}_t(i)$, h_t^+ is the probability node i's label at time $t+1$ is the same as its neighbor's label at time t given node i's label is positive at $t+1$ such that:

$$h_t^+ = \mathbb{P}(y_{t+1}(i) = y_t(j) \mid j \in \hat{\mathcal{N}}_t(i), y_{t+1}(i) = +1), \tag{22}$$

and h_t^- is the probability node i's label at time $t+1$ is the same as its neighbor's label at time t given node i's label is negative at $t+1$ such that:

$$h_t^- = \mathbb{P}(y_{t+1}(i) = y_t(j) \mid j \in \hat{\mathcal{N}}_t(i), y_{t+1}(i) = -1). \tag{23}$$

We denote h_t^+ and h_t^- as the positive and negative dynamic homophily levels, respectively.

As a proof sketch for Theorem 3.2, we first show that the expected node representations after 1 GCN layer is a scaling of the expected initial node representation, where the scaling factor is an function of the positive and negative dynamic homophily levels. As a result, the difference in the expected node representations between a randomly sampled positive node and a randomly sampled negative node after 1 layer of a GCN is a function of dynamic homophily. This proves the theorem for $l = 1$. We prove the result for the general l layer case using induction.

Proof. The first step in proving Theorem 3.2 is to measure the expected node representation after one layer of a GNN with mean aggregation, starting with the negative case in which $i \in V_{t+1}^-$. Formally, we express this quantity as $\mathbb{E}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(1)}(i)\right]$, where the expectation is taken over all nodes $i \in V_{t+1}$ for which $y_{t+1}(i) = -1$. The first step in measuring this expectation is to use the propagation rule for a GNN with mean aggregation:

$$\mathbb{E}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(1)}(i)\right] = \sum_{j\in\hat{\mathcal{N}}_t(i)} \mathbb{E}_{i,j|i\in V_{t+1}^-}\left[\frac{\mathbf{x}_t(j)}{d_t(i)+1}\right] \tag{24}$$

The above tell us that the expected node representation is a sum over the neighbors of node i. Now, the key step in expressing the expected node representation is to *partition* the neighbors of i into those whose label at time t is the same as or opposes i's label at timestep $t+1$:

$$\begin{aligned}\mathbb{E}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(1)}(i)\right] = &\left(\mathbb{E}_{i,j|i\in V_{t+1}^-, j\in V_t^-}\left[\mathbf{x}_t(j)\right]\cdot\mathbb{P}(y_{t+1}(i)=y_t(j)\mid j\in\hat{\mathcal{N}}_t(i), i\in V_{t+1}^-)\right)\\ &+\left(\mathbb{E}_{i,j|i\in V_{t+1}^-, j\in V_t^+}\left[\mathbf{x}_t(j)\right]\cdot(1-\mathbb{P}(y_{t+1}(i)=y_t(j)\mid j\in\hat{\mathcal{N}}_t(i), i\in V_{t+1}^-))\right)\end{aligned} \tag{25}$$

In fact, this partition over the set of neighbors is determined precisely by negative dynamic homophily. Applying the definition of negative dynamic homophily to (25) produces the following:

$$\mathbb{E}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(1)}(i)\right] = \mathbb{E}_{i,j|i\in V_{t+1}^-, j\in V_t^-}\left[\mathbf{x}_t(j)\right]\cdot h_t^- + \mathbb{E}_{i,j|i\in V_{t+1}^-, j\in V_t^+}\left[\mathbf{x}_t(j)\right]\cdot(1-h_t^-) \tag{26}$$

Applying our assumption that for all $t\in[0,T]$ and for all i, $y_t(i)\in\{-1,+1\}$ and $\mathbf{x}_t(i)\sim N(y_t(i)\cdot\mu_t,\sigma^2)$ produces:

$$\mathbb{E}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(1)}(i)\right] = \mathbb{E}_{j|j\in V_t^+}\left[\mathbf{x}_t(j)\right]\cdot -h_t^- + \mathbb{E}_{j|j\in V_t^+}\left[\mathbf{x}_t(j)\right]\cdot(1-h_t^-) = (1-2h_t^-)\cdot\mu_t \tag{27}$$

Following a similar derivation, we measure the expected node representation for the positive case in which $i\in V_{t+1}^+$:

$$\mathbb{E}_{i|i\in V_{t+1}^+}\left[\mathbf{h}_t^{(1)}(i)\right] = \mathbb{E}_{j|j\in V_t^+}\left[\mathbf{x}_t(j)\right]\cdot h_t^+ + \mathbb{E}_{j|j\in V_t^+}\left[\mathbf{x}_t(j)\right]\cdot(h_t^+-1) = (2h_t^+-1)\cdot\mu_t \tag{28}$$

Subtracting (27) from (28), yields our desired result, in which the distance between the expected node representations of the opposing classes becomes a function of the average dynamic homophily. We have proven the base case where $l=1$. In order to prove the induction step, we assume Theorem 3.2 holds at layer l, and show it holds at layer $l+1$. At layer $l+1$ the key to expressing the difference in expected node representations is to do so in a recursive manner as follows:

$$\begin{aligned}&\mathbb{E}_{i|i\in V_{t+1}^+}\left[\mathbf{h}_t^{(l+1)}(i)\right] - \mathbb{E}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(l+1)}(i)\right]\\ &\quad= \left(\mathbb{E}_{j|j\in V_t^-}\left[\mathbf{h}_t^{(l)}(j)\right]\cdot h_t^+ + \mathbb{E}_{j|j\in V_t^+}\left[\mathbf{h}_t^{(l)}(j)\right]\cdot(1-h_t^+)\right)\\ &\qquad- \left(\mathbb{E}_{j|j\in V_t^-}\left[\mathbf{h}_t^{(l)}(j)\right]\cdot h_t^- + \mathbb{E}_{j|j\in V_t^+}\left[\mathbf{h}_t^{(l)}(j)\right]\cdot(1-h_t^-)\right)\end{aligned} \tag{29}$$

Applying our assumption that for all $t\in[0,T]$ and for all i, $y_t(i)\in\{-1,+1\}$, and $\mathbf{x}_t(i)\sim\mathcal{N}(y_t(i)\cdot\mu_t,\sigma^2)$ and grouping like terms produces:

$$\mathbb{E}_{i|i\in V_{t+1}^+}\left[\mathbf{h}_t^{(l+1)}(i)\right] - \mathbb{E}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(l+1)}(i)\right] = \left(\mathbb{E}_{i|i\in V_t^+}\left[\mathbf{h}_t^{(l)}(i)\right] - \mathbb{E}_{i|i\in V_t^-}\left[\mathbf{h}_t^{(l)}(i)\right]\right)\cdot(2h_t^D-1). \tag{30}$$

Applying the inductive hypothesis completes the proof. □

B.3 Concentration of the Expected Distance

Theorem B.3. *For any $\epsilon > 0$, the probability that at time t the distance between the empirical and expected difference after l GCN layers is larger than ϵ is bounded as follows:*

$$\mathbb{P}\left(\left|(\mu^{(l)}_{V^+_{t+1}} - \mu^{(l)}_{V^-_{t+1}}) - (\mathbb{E}_{i|i\in V^+_{t+1}}[\mathbf{h}^{(l)}_t(i)] - \mathbb{E}_{i|i\in V^-_{t+1}}[\mathbf{h}^{(l)}_t(i)])\right| \geq \epsilon\right) \leq \mathcal{O}\left(e^{-\epsilon^2 L^{(l)}_{t,+}} + e^{-\epsilon^2 L^{(l)}_{t,-}}\right), \tag{31}$$

where $\mu^{(l)}_{V^+_{t+1}}$ and $\mu^{(l)}_{V^-_{t+1}}$ are the empirical mean representations after l GCN layers over positive and negative nodes at time $t+1$ respectively, and

$$L^{(l)}_{t,+} = \frac{\left|V^+_{t+1}\right|^2}{\sigma^4 \cdot \sum_{i\in V^+_{t+1}} \left(\sum_{j\in\hat{\mathcal{N}}_t(i)} \frac{l}{d_t(j)^l}\right)^2}, \quad L^{(l)}_{t,-} = \frac{\left|V^-_{t+1}\right|^2}{\sigma^4 \cdot \sum_{i\in V^-_{t+1}} \left(\sum_{j\in\hat{\mathcal{N}}_t(i)} \frac{l}{d_t(j)^l}\right)^2}. \tag{32}$$

Before proving Theorem 3.3, we first state the following relevant definitions and lemmas pertaining to the concentration of Gaussian random variables and Lipschitz functions.

Definition B.4 (Lipschitz functions)**.** Let (X, d_x) and (Y, d_Y) be metric spaces. A function $f : X \to Y$ is called Lipschitz if there exists $L \in \mathbb{R}$ such that:

$$d_Y(f(u), f(v)) \leq L \cdot d_X(u, v) \quad \text{for every } u, v \in X \tag{33}$$

The infimum of all L is called the Lipschitz norm of f and is denoted $||f||_{\text{Lip}}$.

Lemma B.5 (Wainwright (2019))**.** *Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is L-Lipschitz with respect to Euclidean distance, and let $\mathbf{x} = (x_0, \ldots, x_n) \sim \mathcal{N}(0, 1)$. Then for all $\epsilon \in \mathbb{R}$,*

$$\mathbb{P}(|f(\mathbf{x}) - \mathbb{E}[f(x)]| \leq \epsilon) \leq 2exp\left(\frac{-\epsilon^2}{2L^2}\right) \tag{34}$$

Lemma B.6 (Vershynin (2018))**.** *Every differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is Lipschitz, and*

$$||f||_{Lip} \leq sup_{x\in\mathbb{R}^n} ||\nabla f(x)||_2 \tag{35}$$

Proof. We aim to bound the difference between the expected difference in node representation and their empirical difference. To prove the bound, we use a similar derivation as in Theorem 3.3. We first decompose the difference terms and treat each separately as follows:

$$\left|\Big(\frac{1}{\left|V^+_{t+1}\right|}\sum_{i\in V^+_{t+1}} \mathbf{h}^{(l)}_t(i) - \frac{1}{\left|V^-_{t+1}\right|}\sum_{i\in V^-_{t+1}} \mathbf{h}^{(l)}_t(i)\Big) - \Big(\mathbb{E}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right] - \mathbb{E}_{i|i\in V^-_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right]\Big)\right| \tag{36}$$

$$= \left|\Big(\frac{1}{\left|V^+_{t+1}\right|}\sum_{i\in V^+_{t+1}} \mathbf{h}^{(l)}_t(i) - \mathbb{E}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right]\Big) - \Big(\mathbb{E}_{i|i\in V^-_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right] - \frac{1}{\left|V^-_{t+1}\right|}\sum_{i\in V^-_{t+1}} \mathbf{h}^{(l)}_t(i)\Big)\right| \tag{37}$$

$$\leq \left|\Big(\frac{1}{\left|V^+_{t+1}\right|}\sum_{i\in V^+_{t+1}} \mathbf{h}^{(l)}_t(i) - \mathbb{E}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right]\Big)\right| - \left|\Big(\frac{1}{\left|V^-_{t+1}\right|}\sum_{i\in V^-_{t+1}} \mathbf{h}^{(l)}_t(i) - \mathbb{E}_{i|i\in V^-_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right]\Big)\right| \tag{38}$$

where the last inequality follows from the triangle inequality. Here, the first term amounts to the difference between the mean representation for positive nodes and the expected representation of postive nodes, while the second term is the difference between the mean representation for negative nodes and the expected representation of negative nodes. We proceed to bound each separately. For the first term, we obtain the following *recursive*

formulation for the hidden representation:

$$\frac{1}{\left|V_{t+1}^{+}\right|}\sum_{i\in V_{t+1}^{+}}\mathbf{h}_t^{(l)}(i)-\mathbb{E}_{i|i\in V_{t+1}^{+}}\left[\mathbf{h}_t^{(l)}(i)\right]=\frac{1}{\left|V_{t+1}^{+}\right|}\sum_{i\in V_{t+1}^{+}}\frac{1}{d_t(i)}\sum_{j\in\hat{\mathcal{N}}_t(i)}\mathbf{h}_t^{(l-1)}(j)-\mathbb{E}_{i|i\in V_{t+1}^{+}}\left[\mathbf{h}_t^{(l)}(i)\right]$$
$$=\frac{1}{\left|V_{t+1}^{+}\right|}\sum_{i\in V_{t+1}^{+}}\frac{1}{d_t(i)}\sum_{j\in\hat{\mathcal{N}}_t(i)}\cdots\frac{1}{d_t(i)}\sum_{j\in\hat{\mathcal{N}}_t(i)}\phi(\mathbf{x_t'(j)})-\mathbb{E}_{i|i\in V_{t+1}^{+}}\left[\mathbf{h}_t^{(l)}(i)\right],\quad(39)$$

where $\phi(\mathbf{x_t'(j)})=\sigma_t^2(j)\mathbf{x_t(j)}+\mu_t(j)$, $\mu_t(j)$ is the mean of node j, and $\sigma_t^2(j)$ is the variance of node j at time t. First notice that each $\mathbf{x_t'(j)}$ are *standard* gaussian random variables. We have achieved this result by introducing function ϕ, which transforms standard gaussian random variables to gaussian random variables with mean $\mu_t(j)$ and variance $\sigma_t^2(j)$. Second, we have that the empirical mean representation for the positive class is a function $f_t^{+}:\mathbb{R}^n\to\mathbb{R}$ of standard gaussian random variables. Thus, we can use Lemmas B.5 and B.6 to bound the distance between the empirical mean representation for positive nodes and the expected representation for positive nodes.

$$f_t^{(l),+}(\mathbf{x_t})=\frac{1}{\left|V_{t+1}^{+}\right|}\sum_{i\in V_{t+1}^{+}}\mathbf{h}_t^{(l)}(i)=\frac{1}{\left|V_{t+1}^{+}\right|}(h_t^{(l)}(0)+\cdots+h_t^{(l)}(\left|V_{t+1}^{+}\right|))\quad(40)$$

$$||\nabla f_t^{(l),+}||_2^2=\sum_{i\in V_{t+1}^{+}}\left(\frac{\sigma^2}{\left|V_{t+1}^{+}\right|}\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2=\frac{\sigma^4}{\left|V_{t+1}^{+}\right|^2}\sum_{i\in V_{t+1}^{+}}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2\quad(41)$$

where equation (41) follows from the fact that $\mathbf{x}_t(i)$ appears in the summation over all positive nodes precisely $d_t(i)\cdot l$ times, l times for each of its neighbors $j\in\hat{\mathcal{N}}_t(i)$. Following a similar derivation for the second term where we aim to bound the difference between the mean negative node representation and the expected negative representation at layer l yields:

$$f_t^{(l),-}(\mathbf{x_t})=\frac{1}{\left|V_{t+1}^{-}\right|}\sum_{i\in V_{t+1}^{+}}\mathbf{h}_t^{(l)}(i)=\frac{1}{\left|V_{t+1}^{-}\right|}(h_t^{(l)}(0)+\cdots+h_t^{(l)}(\left|V_{t+1}^{-}\right|))\quad(42)$$

$$||\nabla f_t^{(l),-}||_2^2=\sum_{i\in V_{t+1}^{+}}\left(\frac{\sigma^2}{\left|V_{t+1}^{-}\right|}\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2=\frac{\sigma^4}{\left|V_{t+1}^{-}\right|^2}\sum_{i\in V_{t+1}^{+}}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2\quad(43)$$

By Lemma 1, we obtain the following bounds for (1) the distance between the empirical positive node representation and the expected positive representation at layer l and (2) the distance between the empirical negative node representation and the expected negative representation at layer l:

$$\mathbb{P}\left(\left|\frac{1}{\left|V_{t+1}^{+}\right|}\sum_{i\in V_{t+1}^{+}}\mathbf{h}_t^{(l)}(i)-\mathbb{E}_{i|i\in V_{t+1}^{+}}\left[\mathbf{h}_t^{(l)}(i)\right]\right|\geq\epsilon\right)\leq\mathcal{O}\Big(\exp\Big(\frac{-\epsilon^2\left|V_{t+1}^{+}\right|^2}{\sigma^4\sum_{i\in V_{t+1}^{+}}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2}\Big)\Big)\quad(44)$$

$$\mathbb{P}\left(\left|\frac{1}{\left|V_{t+1}^{-}\right|}\sum_{i\in V_{t+1}^{-}}\mathbf{h}_t^{(l)}(i)-\mathbb{E}_{i|i\in V_{t+1}^{-}}\left[\mathbf{h}_t^{(l)}(i)\right]\right|\geq\epsilon\right)\leq\mathcal{O}\Big(\exp\Big(\frac{-\epsilon^2\left|V_{t+1}^{-}\right|^2}{\sigma^4\sum_{i\in V_{t+1}^{-}}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{l}{d_t(j)^l}\right)^2}\Big)\Big)\quad(45)$$

Finally, we apply a union bound to conclude the proof. □

B.4 Expected AUROC for l layer GNNs

Theorem B.7. *The expected AUROC of $f^{(l)}$ at timestep t can be upper bounded as follows,*

$$\mathbb{E}[A_t(f^{(l)})]\leq 1-\Phi\left(-\frac{2\cdot\mu_t\cdot(h_t^{+}+h_t^{-}-1)^l}{v_{t+1}^{+}(l)+v_{t+1}^{-}(l)}\right).\quad(46)$$

where $v^+_{t+1}(l)$ and $v^-_{t+1}(l)$ are the lower bounds of the variances of the future positive and negative nodes after l GCN layers, respectively, and are defined recursively in terms of the dynamic homophily levels as follows,

$$v^+_{t+1}(l) = {h^+_t}^2 \cdot v^+_{t+1}(l-1) + (1-h^+_t)^2 \cdot v^-_{t+1}(l-1) \tag{47}$$

$$v^-_{t+1}(l) = {h^-_t}^2 \cdot v^-_{t+1}(l-1) + (1-h^-_t)^2 \cdot v^+_{t+1}(l-1) \tag{48}$$

$$v^+_{t+1}(0) = v^-_{t+1}(0) = \sigma^2. \tag{49}$$

Moreover the probability that at time t the distance between the empirical AUROC and the expected AUROC of a l layer GCN is larger than ϵ is bounded as follows,

$$\mathbb{P}(\left|A_t(f^{(l)}) - \mathbb{E}[A_t(f^{(l)})]\right| \geq \epsilon) \leq e^{\frac{-2\cdot\left|V^+_{t+1}\right|\cdot\left|V^-_{t+1}\right|\cdot\epsilon^2}{\left|V^+_{t+1}\right|+\left|V^-_{t+1}\right|}} \tag{50}$$

In order to prove Theorem 3.4, our main goal will be to solve for the variances of the representations for the positive class and negative class respectively. Once we solve for the variances, we can use Lemma B.1 and Theorem 3.2 to obtain the full expression.

Proof. We prove Theorem 3.4 with induction. The first step is to show that the node representations after 1 GCN layer can be expressed as a linear combination of independent Gaussian random variables. Beginning with 1-layer GNNs on the positive case, the variance terms can be written,

$$\mathbb{V}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(1)}_t(i)\right] = \mathbb{V}_{i|i\in V^+_{t+1}}\left[\sum_{j\in\hat{\mathcal{N}}(i)}\frac{\mathbf{x}_t(j)}{d_t(i)+1}\right] \tag{51}$$

Equation 51 indeed tells us that after 1-layer of GCN, the node representation for the positive class is a linear combination of independent Gaussian random variables. Since there are only two different types of Gaussian random variables, one centered at μ_t and the other at $-\mu_t$, we can solve for the variance by determining the ratio of nodes from each of the distributions. To do so, we can apply the same partition trick as in Theorem 3.2, treating $\mathbf{h}^{(1)}_t(i)$ as a weighted sum of the Gaussian random variables where the weights are the dynamic homophily levels. Conditioning on $i \in V^+_{t+1}$, we have,

$$\mathbb{V}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(1)}_t(i)\right] = \mathbb{V}_{i,j,k|i\in V^+_{t+1},j\in V^+_t,k\in V^-_t}\left[\mathbf{x}_t(j)\cdot h^+_t + \mathbf{x}_t(k)\cdot(1-h^+_t)\right] \tag{52}$$

$$= {h^+_t}^2\cdot\sigma^2 + (1-h^+_t)^2\sigma^2 \tag{53}$$

where the last line follows from $\mathbf{x}_t(j)$ and $\mathbf{x}_t(k)$ being Gaussian random variables with variance σ^2. Following a similar derivation for the negative case and conditioning on $i \in V^-_{t+1}$, we have,

$$\mathbb{V}_{i|i\in V^-_{t+1}}\left[\mathbf{h}^{(1)}_t(i)\right] = \mathbb{V}_{i,j,k|i\in V^-_{t+1},j\in V^+_t,k\in V^-_t}\left[\mathbf{x}_t(k)\cdot h^-_t + \mathbf{x}_t(j)\cdot(1-h^-_t)\right] \tag{54}$$

$$= {h^-_t}^2\cdot\sigma^2 + (1-h^-_t)^2\cdot\sigma^2 \tag{55}$$

The variances of the node representations for the one-layer case can then be written as,

$$\mathbb{V}_{i,j|i\in V^+_t,j\in V^-_t}[\mathbf{h}^{(1)}_t(i) - \mathbf{h}^{(1)}_t(j)] = {h^+_t}^2\cdot\sigma^2 + (1-h^+_t)^2\sigma^2 + {h^-_t}^2\cdot\sigma^2 + (1-h^-_t)^2\cdot\sigma^2, \tag{56}$$

which satisfies the base case. We now consider $l+1$-layer GCNs. In the multilayer case conditioning on positive nodes, the key in determining the variance is to apply the partition trick as follows,

$$\begin{aligned}\mathbb{V}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(l+1)}_t(i)\right] &= \mathbb{V}_{i,j,k|i\in V^+_{t+1},j\in V^+_t,k\in V^-_t}\left[\mathbf{h}^{(l)}_t(j)\cdot h^+_t + \mathbf{h}^{(l)}_t(k)\cdot(1-h^+_t)\right] \\ &= {h^+_t}^2\cdot\mathbb{V}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right] + (1-h^+_t)^2\cdot\mathbb{V}_{j|j\in V^-_{t+1}}\left[\mathbf{h}^{(l)}_t(j)\right] + c\cdot\mathrm{Cov}_{i,j|i\in V^+_{t+1},j\in V^-_{t+1}}(\mathbf{h}^{(l)}_t(i),\mathbf{h}^{(l)}_t(j)), \\ &\geq {h^+_t}^2\cdot\mathbb{V}_{i|i\in V^+_{t+1}}\left[\mathbf{h}^{(l)}_t(i)\right] + (1-h^+_t)^2\cdot\mathbb{V}_{j|j\in V^-_{t+1}}\left[\mathbf{h}^{(l)}_t(j)\right]\end{aligned} \tag{57}$$

where $c = 2{h_t^+}^2(1-h_t^+)^2$ and in the first line we have partitioned $\mathbf{h}_t^{(l+1)}(i)$ into a weighted sum of $\mathbf{h}_t^{(l)}(j)$ and $\mathbf{h}_t^{(l)}(k)$ based on the positive dynamic homophily. In the second line, we have an additional covariance term since $\mathbf{h}_t^{(l)}(i)$ and $\mathbf{h}_t^{(l)}(j)$ are not necessarily independent and the two hidden representations may share nodes. We obtain the lower bound by noticing that the covariance term must be positive due to the linearity of covariance. In essence, $\mathbf{h}_t^{(l)}(i)$ and $\mathbf{h}_t^{(l)}(j)$ are weighted sums of independent gaussian random variables. Expanding the covariance through linearity, the only nonzero terms are the variances of the shared nodes between the two representations and since the variance must be positive, the covariance must also be positive. Now, conditioning on negative nodes, we have,

$$\begin{aligned}\mathbb{V}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(l+1)}(i)\right] &= \mathbb{V}_{i,j,k|i\in V_{t+1}^+, j\in V_t^+, k\in V_t^-}\left[\mathbf{h}_t^{(l)}(k)\cdot h_t^- + \mathbf{h}_t^{(l)}(j)\cdot(1-h_t^-)\right] \\ &= {h_t^-}^2\cdot\mathbb{V}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(l)}(i)\right] + (1-h_t^-)^2\cdot\mathbb{V}_{j|j\in V_{t+1}^+}\left[\mathbf{h}_t^{(l)}(j)\right] + c\cdot\mathrm{Cov}_{i,j|i\in V_{t+1}^+, j\in V_{t+1}^-}(\mathbf{h}_t^{(l)}(i),\mathbf{h}_t^{(l)}(j)) \\ &\geq {h_t^-}^2\cdot\mathbb{V}_{i|i\in V_{t+1}^-}\left[\mathbf{h}_t^{(l)}(i)\right] + (1-h_t^-)^2\cdot\mathbb{V}_{j|j\in V_{t+1}^+}\left[\mathbf{h}_t^{(l)}(j)\right] \quad (58)\end{aligned}$$

where $c = 2{h_t^-}^2(1-h_t^-)^2$. Adding the variances in Equations 57 and 58 and applying the induction hypothesis satisfies our recursive definitions of the variance lower bounds for $l+1$-layer GCNs. In order to obtain the probability bound, we apply Theorem 5 in Agarwal et al. (2005), bounding the distance between the expected AUROC and the empirical AUROC.

□

B.5 Expected Distance and Concentration in Multiclass Classification

Theorem B.8. *Given graph $\boldsymbol{G}_{0:T}$ at timestep t, the Euclidean distance of the expected difference between a randomly sampled node of class c_m and a randomly sampled node of class c_n after 1 layer of a GCN can be expressed as:*

$$\left|\left|\mathbb{E}_{i|y_{t+1}(i)=c_m}\left[\mathbf{h}_t^{(1)}(i)\right] - \mathbb{E}_{i|y_{t+1}(i)=c_n}\left[\mathbf{h}_t^{(1)}(i)\right]\right|\right|_2 = \mu_t\cdot\left(\sum_{k=1}^{K}(\mathbf{T}_t[c_m,c_k]-\mathbf{T}_t[c_n,c_k])^2\right)^{\frac{1}{2}}. \quad (59)$$

Furthermore, for any $\epsilon > 0$, the probability that the distance between the empirical mean representation and the expected representation is larger than ϵ is bounded as follows:

$$\begin{aligned}&\mathbb{P}\left(\left|\left|\left(\frac{1}{\left|V_{t+1}^{c_m}\right|}\sum_{i\in V_{t+1}^{c_m}}\mathbf{h}_t^{(1)}(i) - \frac{1}{\left|V_{t+1}^{c_n}\right|}\sum_{i\in V_{t+1}^{c_n}}\mathbf{h}_t^{(1)}(i)\right) - \left(\mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(1)}(i)\right] - \mathbb{E}_{i|i\in V_{t+1}^{c_n}}\left[\mathbf{h}_t^{(1)}(i)\right]\right)\right|\right|_2 \geq \epsilon\right) \\ &\qquad \leq \mathcal{O}\left(exp\left(\frac{-\epsilon^2\left|V_{t+1}^{c_m}\right|^2|C|}{\sum_{i\in V_{t+1}^{c_m}}\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{1}{d_t(j)}}^2\right) + exp\left(\frac{-\epsilon^2\left|V_{t+1}^{c_n}\right|^2|C|}{\sum_{i\in V_{t+1}^{c_n}}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{1}{d_t(j)}\right)^2}\right)\right) \quad (60)\end{aligned}$$

Proof. In the multiclass classification case, rather than partitioning the neighbors into those whose label at timestep t is the same as i's label at timestep $t+1$, we instead partition the neighbors into their respective classes at timestep t. Now, the partition is determined by entries of the dynamic compatibility matrix. First assume $y_{t+1}(i) = c_m$:

$$\begin{aligned}\mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(1)}(i)\right] &= \sum_{k=1}^{K}\mathbb{E}_{j|y_t(j)=c_k}\left[\mathbf{x}_t(j)\right]\cdot\mathbb{P}(y_t(j)=c_k \mid j\in\hat{\mathcal{N}}_t(i), y_{t+1}(i)=c_m) \quad (61)\\ &= \sum_{k=1}^{K}\mathbb{E}_{j|y_t(j)=c_k}\left[\mathbf{x}_t(j)\right]\cdot\mathbf{T}_t[c_m,c_k]. \quad (62)\end{aligned}$$

We follow a similar derivation assuming $y_{t+1}(i) = c_n$, and we obtain the following:

$$\mathbb{E}_{i|i\in V_{t+1}^{c_n}}\left[\mathbf{h}_t^{(1)}(i)\right] = \sum_{k=1}^{K}\mathbb{E}_{j|y_t(j)=c_k}\left[\mathbf{x}_t(j)\right]\cdot\mathbb{P}(y_t(j)=c_k \mid j\in\hat{\mathcal{N}}_t(i), y_{t+1}(i)=c_n) \tag{63}$$

$$= \sum_{k=1}^{K}\mathbb{E}_{j|y_t(j)=c_k}\left[\mathbf{x}_t(j)\right]\cdot\mathbf{T}_t[c_n,c_k]. \tag{64}$$

Measuring the l^2 norm of the difference between (64) and (62), and applying the assumption that for all i, $y_t(i)\in\mathcal{C}$ and $\mathbf{x}_t(i)\sim\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{t}\mathbf{Y}_t(i)},\sigma^2)$, yields:

$$\left|\left|\mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(1)}(i)\right] - \mathbb{E}_{i|i\in V_{t+1}^{c_n}}\left[\mathbf{h}_t^{(1)}(i)\right]\right|\right|_2 = \mu_t\cdot\left|\left|\sum_{k=1}^{K}(\mathbf{T}_t[c_m,c_k]-\mathbf{T}_t[c_n,c_k])\right|\right|_2 \tag{65}$$

$$= \mu_t\cdot\left(\sum_{k=1}^{K}(\mathbf{T}_t[c_m,c_k]-\mathbf{T}_t[c_n,c_k])^2\right)^{\frac{1}{2}}. \tag{66}$$

We aim to bound the difference between the expected difference in node representation and their empirical difference. To prove the bound, we use a similar derivation as in Theorems 3.3. We first decompose the difference terms and treat each separately as follows:

$$\left|\left|\Big(\frac{1}{\left|V_{t+1}^{c_m}\right|}\sum_{i\in V_{t+1}^{c_m}}\mathbf{h}_t^{(1)}(i) - \frac{1}{\left|V_{t+1}^{c_n}\right|}\sum_{i\in V_{t+1}^{c_n}}\mathbf{h}_t^{(1)}\Big) - \Big(\mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(1)}(i)\right] - \mathbb{E}_{i|i\in V_{t+1}^{c_n}}\left[\mathbf{h}_t^{(1)}(i)\right]\Big)\right|\right|_2$$

$$\leq \left|\left|\Big(\frac{1}{\left|V_{t+1}^{c_m}\right|}\sum_{i\in V_{t+1}^{c_m}}\mathbf{h}_t^{(1)}(i) - \mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(1)}(i)\right]\Big)\right|\right|_2 - \left|\left|\Big(\frac{1}{\left|V_{t+1}^{c_n}\right|}\sum_{i\in V_{t+1}^{c_n}}\mathbf{h}_t^{(1)}(i) - \mathbb{E}_{i|i\in V_{t+1}^{c_n}}\left[\mathbf{h}_t^{(1)}(i)\right]\Big)\right|\right|_2 \tag{67}$$

where the last inequality follows from the triangle inequality. Again, the first term amounts to the difference between the mean representation for nodes in $V_{t+1}^{c_m}$ and the expected representation of nodes in nodes in $V_{t+1}^{c_m}$, while the second term is the difference between the mean representation for nodes in nodes in $V_{t+1}^{c_n}$ and the expected representation of nodes in nodes in $V_{t+1}^{c_m}$. We proceed to bound each separately.

The key in proving the upper bound in Theorem C.2 is to handle each dimension of the node representations separately. Using our assumption that node representations are drawn from a multivariate gaussian with a diagonal covariance, each dimension of the representation is independent and equivalent to a univariate gaussian. For dimension arbitrary dimension k of the representation, we obtain the following *recursive formulation* for the hidden representation:

$$\frac{1}{\left|V_{t+1}^{c_m}\right|}\sum_{i\in V_{t+1}^{c_m}}\mathbf{h}_t^{(l)}(i)[k] - \mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(l)}(i)[k]\right]$$

$$= \frac{1}{\left|V_{t+1}^{c_m}\right|}\sum_{i\in V_{t+1}^{c_m}}\frac{1}{d_t(i)}\sum_{j\in\hat{\mathcal{N}}_t(i)}\mathbf{x}_t^{(1)}(j)[k] - \mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(1)}(i)[k]\right]$$

$$= \frac{1}{\left|V_{t+1}^{c_m}\right|}\sum_{i\in V_{t+1}^{c_m}}\frac{1}{d_t(i)}\sum_{j\in\hat{\mathcal{N}}_t(i)}\phi(\mathbf{x}'_{\mathbf{t}}(\mathbf{j})[\mathbf{k}]) - \mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(l)}(i)[k]\right] \tag{68}$$

where $\phi(\mathbf{x}'_{\mathbf{t}}(\mathbf{j}))$, $\mu_t(j)$, and $\sigma_t^2(j)$ are as defined in Theorem 3.2. Again, each $\mathbf{x}'_{\mathbf{t}}(\mathbf{j})[\mathbf{k}]$ are *standard* gaussian random variables, and the empirical mean representation for the positive class at layer l is a function $f_t^{(l),+}:\mathbb{R}^n\to\mathbb{R}$ of standard gaussian random variables. Thus, we can use Lemmas B.5 and B.6 to bound the distance between the

empirical mean representation for positive nodes and the expected representation for positive nodes.

$$f_t^{(1),c_m}(\mathbf{x_t}) = \frac{1}{\left|V_{t+1}^{c_m}\right|} \sum_{i \in V_{t+1}^{c_m}} \mathbf{h}_t^{(1)}(i)[k] = \frac{1}{\left|V_{t+1}^{c_m}\right|} (h_t^{(1)}(0)[k] + \cdots + h_t^{(1)}(\left|V_{t+1}^{c_m}\right|)[k]) \tag{69}$$

$$||\nabla f_t^{(l),c_m}||_2^2 = \sum_{i \in V_{t+1}^{c_m}} \left(\frac{\sigma^2}{\left|V_{t+1}^{c_m}\right|} \sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{1}{d_t(j)} \right)^2 = \frac{\sigma^4}{\left|V_{t+1}^{c_m}\right|^2} \sum_{i \in V_{t+1}^{c_m}} \left(\sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{1}{d_t(j)} \right)^2 \tag{70}$$

Following a similar derivation for the second term where we aim to bound the difference between the mean negative node representation and the expected negative representation at layer l yields:

$$f_t^{(1),c_n}(\mathbf{x_t}) = \frac{1}{\left|V_{t+1}^{c_n}\right|} \sum_{i \in V_{t+1}^{c_n}} \mathbf{h}_t^{(1)}(i)[k] = \frac{1}{\left|V_{t+1}^{c_n}\right|} (h_t^{(1)}(0)[k] + \cdots + h_t^{(l)}(\left|V_{t+1}^{c_n}\right|\,[k])) \tag{71}$$

$$||\nabla f_t^{(1),c_n}||_2^2 = \sum_{i \in V_{t+1}^{c_n}} \left(\frac{\sigma^2}{\left|V_{t+1}^{c_n}\right|} \sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{1}{d_t(j)} \right)^2 = \frac{\sigma^4}{\left|V_{t+1}^{c_n}\right|^2} \sum_{i \in V_{t+1}^{c_m}} \left(\sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{l}{d_t(j)} \right)^2 \tag{72}$$

By Lemma 1, we obtain the following bounds for (1) the distance between the empirical positive node representation and the expected positive representation at layer l and (2) the distance between the empirical negative node representation and the expected negative representation at layer l:

$$\mathbb{P}\left(\left| \frac{1}{\left|V_{t+1}^{c_m}\right|} \sum_{i \in V_{t+1}^{c_m}} \mathbf{h}_t^{(1)}(i) - \mathbb{E}_{i | i \in V_{t+1}^{c_m}} \left[\mathbf{h}_t^{(1)}(i) \right] \right| \geq \epsilon \right) \leq \mathcal{O}\Big(\exp\Big(\frac{-\epsilon^2 \left|V_{t+1}^{c_m}\right|^2}{\sigma^4 \sum_{i \in V_{t+1}^{c_m}} \left(\sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{l}{d_t(j)^l} \right)^2} \Big) \Big) \tag{73}$$

$$\mathbb{P}\left(\left| \frac{1}{\left|V_{t+1}^{c_n}\right|} \sum_{i \in V_{t+1}^{c_n}} \mathbf{h}_t^{(1)}(i) - \mathbb{E}_{i | i \in V_{t+1}^{c_n}} \left[\mathbf{h}_t^{(1)}(i) \right] \right| \geq \epsilon \right) \leq \mathcal{O}\Big(\exp\Big(\frac{-\epsilon^2 \left|V_{t+1}^{c_n}\right|^2}{\sigma^4 \sum_{i \in V_{t+1}^{c_n}} \left(\sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{l}{d_t(j)^l} \right)^2} \Big) \Big) \tag{74}$$

Notice, we have obtained our bounds assuming arbitrary dimension k. It suffices to apply a second union bound over all $k \in |C|$ in order to obtain the final bound in Theorem C.2. □

C ADDITIONAL THEORY, EMPIRICAL RESULTS, AND DISCUSSIONS

C.1 Dynamic Homophily and Multiclass Classification

In the multiclass classification case, dynamic homophily alone does not represent the distance between nodes of different classes. Thus, we introduce the dynamic compatibility matrix as an extension of the class compatibility matrix, allowing us to estimate the node representations of GCNs and measure the expected distance in node representations.

Why is static homophily insufficient? We first consider why static homophily is insufficient in multiclass classification on a static graph. In multiclass classification, each class can be associated with $|\mathcal{C}|$ probabilities, where the k-th probability for class m denotes the probability a node of class m forms an edge with a node of class k. These probabilities are essential for estimating node representations because they capture the full neighbor label distribution across all node classes. Intuitively, each class must have a unique neighbor distribution in order for GNNs to discriminate between the classes. Notably, here static homophily can be low, yet the distributions can be unique, leading to high GNN performance (Lim et al., 2021; Ma et al., 2022; Zhu et al., 2023).

In the the static setting, the class compatibility matrix captures the entire neighbor label distribution for each class. Thus, we aim to develop a similar quantity in the dynamic setting that captures the same intuition as dynamic class homophily in binary classification. To this end, we propose the dynamic compatibility matrix formally defined as follows:

Definition C.1 (Dynamic Compatibility Matrix)**.** Given dynamic graph $\boldsymbol{G}_{0:T}$, the dynamic compatibility matrix $\mathbf{T}_t$ at timestep t is a $|\mathcal{C}| \times |\mathcal{C}|$ matrix where entry $\mathbf{T}_t[c_m, c_n]$ is defined:

$$\mathbf{T}_t[c_m, c_n] = \mathbb{P}(y_t(j) = c_n \mid j \in \hat{\mathcal{N}}_t(i), y_{t+1}(i) = c_m). \tag{75}$$

We now proceed to measure the node representations for specific classes under multiclass node classification in the dynamic setting. Here, we show that the expected Euclidean distance between node representations of different classes is a function of the Euclidean distance between the neighbor label distributions of the two classes. In our analysis we make the following assumptions:

Assumptions (Multiclass Classification): Let $\boldsymbol{G}_{0:T}$ be a dynamic graph, and $\mathcal{C} = \{c_1, \ldots, c_K\}$ be the set of node classes, where $K > 2$. For all $t \in [0, T]$ and for all $i \in V_t$, $y_t(i) \in \mathcal{C}$ and $\mathbf{x}_t(i) \sim N(\boldsymbol{\mu}_{\boldsymbol{t}y_t(i)}, \sigma^2)$ where N is a mutivariate normal distribution with mean $\boldsymbol{\mu}_{\boldsymbol{t}y_t(i)} = \mu_t \cdot \text{one-hot}(y_t(i)) \in \mathbb{R}^K$ and covariance $\sigma^2 = \text{Diag}(\boldsymbol{\sigma}^{\mathbf{2}}) \in \mathbb{R}^{K \times K}$.

Theorem C.2. *The Euclidean distance of the expected difference between a randomly sampled node of class c_m and a randomly sampled node of class c_n after 1 layer of a GCN can be expressed as:*

$$\left|\left|\mathbb{E}_{i|y_{t+1}(i)=c_m}\left[\mathbf{h}_t^{(1)}(i)\right] - \mathbb{E}_{i|y_{t+1}(i)=c_n}\left[\mathbf{h}_t^{(1)}(i)\right]\right|\right|_2 = \mu_t \cdot \left(\sum_{k=1}^{K}(\mathbf{T}_t[c_m, c_k] - \mathbf{T}_t[c_n, c_k])^2\right)^{\frac{1}{2}}. \tag{76}$$

Furthermore, for any $\epsilon > 0$, the probability that the distance between the empirical mean representation and the expected representation is larger than ϵ is bounded as follows:

$$\begin{aligned}\mathbb{P}\left(\left|\left|\left(\frac{1}{\left|V_{t+1}^{c_m}\right|}\sum_{i\in V_{t+1}^{c_m}}\mathbf{h}_t^{(1)}(i) - \frac{1}{\left|V_{t+1}^{c_n}\right|}\sum_{i\in V_{t+1}^{c_n}}\mathbf{h}_t^{(1)}(i)\right) - \left(\mathbb{E}_{i|i\in V_{t+1}^{c_m}}\left[\mathbf{h}_t^{(1)}(i)\right] - \mathbb{E}_{i|i\in V_{t+1}^{c_n}}\left[\mathbf{h}_t^{(1)}(i)\right]\right)\right|\right|_2 \geq \epsilon\right) \\ \leq \mathcal{O}\left(exp\left(\frac{-\epsilon^2\left|V_{t+1}^{c_m}\right|^2|C|}{\sum_{i\in V_{t+1}^{c_m}}\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{1}{d_t(j)}}^2\right) + exp\left(\frac{-\epsilon^2\left|V_{t+1}^{c_n}\right|^2|C|}{\sum_{i\in V_{t+1}^{c_n}}\left(\sum_{j\in\hat{\mathcal{N}}_t(i)}\frac{1}{d_t(j)}\right)^2}\right)\right)\end{aligned} \tag{77}$$

Theorem C.2 tells us that the expected Euclidean distance between node representations of different classes depend on the Euclidean distance between the neighbor label distributions of the two classes. Moreover, the expected distance is close to the empirical distance with high probability. As the neighbor label distributions for two classes become more similar, the Euclidean distance between their expected node representation decreases, and we expect 1-layer GCNs to perform worse in discriminating between the two classes.

C.2 Extension of Dynamic homophily to Continuous Dynamic Graphs

In the main paper, we propose dynamic homophily in the context of discrete dynamic graphs. While there are many examples of discrete dynamic graphs such as social networks and protein-protein interaction networks, there are also continuous representations of graphs where the graph is defined as a sequence of edges rather than a sequence of static graphs. A straightforward way to extend dynamic homophily to continuous graphs is to first specify a window size k, then define the dynamic homophily at time t as the dynamic homophily measured within the window size $[t, t + k]$. We obtain the sequence of dynamic homophily levels by measuring it along the windows $[tk, (t + 1)k]$ for all t. This strategy aligns with most tasks defined on continuous graphs, where performance is measured along time slices of the continuous graph and the window size is determined by the application (Kumar et al., 2019; Huang et al., 2023).

In practice, the evaluation of dynamic homophily in continuous settings is challenging since most publicly available continuous graphs are heterogeneous, where nodes are of different types (Kumar et al., 2019; Huang et al., 2023). For example, Kumar et al. (2019) considers a Reddit network where nodes are users and subreddits, while edges connect users to subreddits. Another example can be found in Huang et al. (2023) where the dataset is a cryptocurrency network where nodes are users and tokens, while edges connect users to tokens. In these cases, it is not straightforward to measure homophily since it is difficult to compare user nodes to token or subreddit nodes, and there may be no interpretation for a user node to be similar to a token or subreddit node. Thus, due to the heterogeneity of these datasets, we leave the further exploration of heterogenous continuous dynamic graphs and dynamic homophily for future work.

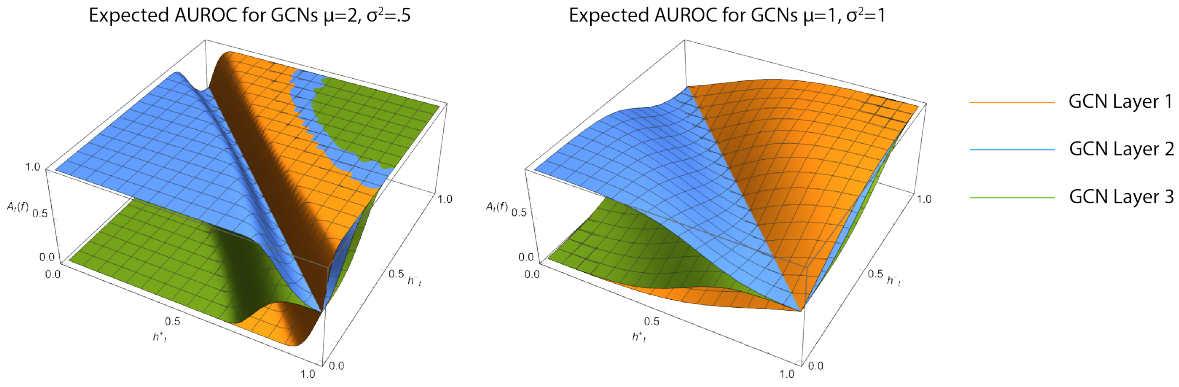


Figure 6: Expected AUROC across GCN layers as a function of dynamic homophily levels for different μ_t and σ.

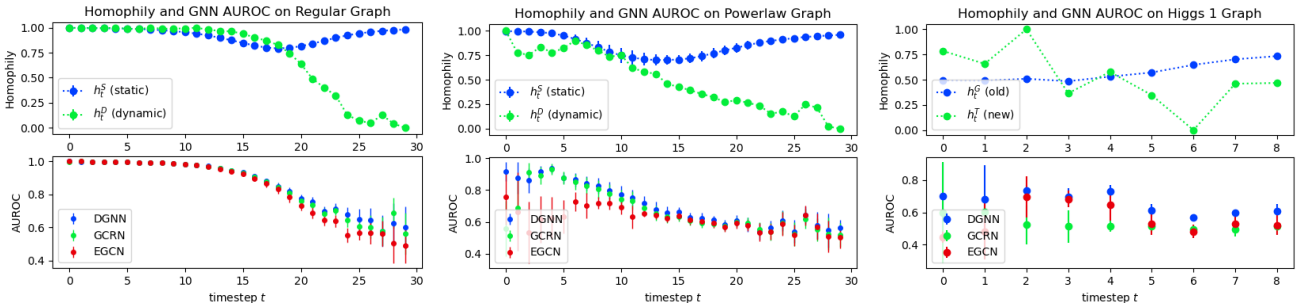
C.3 Further Discussion of Theorem 3.4

Here, we provide additional discussion of Theorem 3.4. In Figure 6, we visualize the expected AUROC at different GCN layers on the same plot for different choices of μ_t and σ^2. We find that indeed optimal AUROC for different dynamic homophily levels is obtained for different GCN layers. In particular, when $\mu_t = 2$ and $\sigma^2 = .5$, deeper odd-layered GCNs lead to the best performance when both dynamic homophily levels are high, while deeper even-layered GCNs lead to the best performance when both dynamic homophily levels are low. When $\mu_t = 1$ and $\sigma^2 = 1$, deeper GCNs perform poorly since smoothing is not as beneficial. Here, the optimal GCNs are 1-layer and 2-layer GCNs when dynamic homophily levels are both high and low, respectively. In light of these results, a natural solution in obtaining the best performance across a dynamic graph with changing dynamic and dynamic homophily levels is to combine the representations across layers such that a GNN leverages the most useful ones across the spectrum of dynamic and dynamic homophily levels.

C.4 Dynamic Homophily and Dynamic GNNs

Following the same training and evaluation setup described in the main paper, we also test dynamic homophily's correlation with performance for various dynamic GNNs. Generally, dynamic GNNs find a way to combine an RNN component with a message passing module in order to leverage the temporal signal present in the dynamic graph. However, since in our synthetic experimental setup the SI model makes the Markov assumption, a dynamic GNN that leverages the full history should simply learn to ignore all timesteps prior to t, making them equivalent to static GNNs on these datasets. On the Higgs networks, intuitively, the information from timestep t should be enough information to predict the spread of the signal at time $t + 1$. Lastly, existing results in Fu and He (2022) on the protein-protein interaction networks suggest that dynamic GNNs leveraging the temporal signal perform just as well as static GNNs that do not leverage the temporal data. Given these arguments, we do not expect dynamic GNNs to perform better than static GNNs on our chosen datasets. Nevertheless, for completeness purposes, we test the following representative dynamic GNNs from the literature on a subset of the datasets in the main paper: **DGNN** (Manessi et al., 2020; Narayan and Roe, 2018; Chen et al., 2022), **GCRN** (Seo et al., 2018), and **EvolveGCN** (Pareja et al., 2020). We provide descriptions of all dynamic GNNs in the context of their dynamic components in Appendix D.

Following our evaluation procedure in the main paper, we compute the average correlation between dynamic homophily and dynamic GNN performance. Across all combinations of synthetic graphs and GNN approaches, the average correlation between dynamic homophily and GNN performance exceeds the correlation between static homophily and GNN performance (Table 3). To fully capture changes to dynamic GNN performance, we evaluate at each time step for each of the datasets (Figure 7). More specifically we measure AUROC, dynamic homophily, and static homophily at each timestep for the Regular, Powerlaw, and Higgs 1 graph. While dynamic homophily exhibits the same trends as GNN performance, static homophily does not since it stays high across all three graphs for all timesteps. Comparing the performance of dynamic GNNs to static GNNs, we find generally that dynamic GNNs perform worse than static ones, confirming our hypothesis that GNNs leveraging the full temporal signal do not provide benefits on our chosen datasets (Table 4).



(a) Regular dynamic graph (b) Powerlaw dynamic graph (c) Higgs 1 dynamic graph

Figure 7: Mean and standard deviations (error bars) of dynamic homophily, static homophily, and AUROC across all graphs in the test set for the Regular, Powerlaw, and Higgs 1 dynamic graph. For all three dynamic graphs, static homophily stays relatively high across time, while dynamic homophily and GNN performances show different trends.

Table 3: The mean ± standard deviation of Spearman's rank correlation coefficient between GNN AUROC and homophily measures across all graphs in the test set. For all combinations of GNN and dynamic graph, dynamic homophily tends to have a higher correlation with dynamic GNN performance compared to static homophily.

		Epidemiological Dynamic Graphs		Higgs Dynamic Graphs	
GNN	**Homophily**	**Regular**	**Powerlaw**	**Higgs 1**	**Higgs 2**
DGNN	(*static*) h_t^S	0.52 ± 0.09	0.09 ± 0.15	-0.55 ± 0.32	OOM
	(*dynamic*) h_t^D	**0.90 ± 0.04**	**0.83 ± 0.08**	**0.60 ± 0.35**	OOM
GCRN	(*static*) h_t^S	0.51 ± 0.09	0.13 ± 0.13	-0.24 ± 0.26	OOM
	(*dynamic*) h_t^D	**0.91 ± 0.04**	**0.85 ± 0.07**	**0.22 ± 0.09**	OOM
EvolveGCN	(*static*) h_t^S	0.50 ± 0.09	-0.18 ± 0.23	-0.05 ± 0.27	OOM
	(*dynamic*) h_t^D	**0.91 ± 0.03**	**0.77 ± 0.13**	**-0.03 ± 0.44**	OOM

Table 4: Mean ± standard deviation of GNN AUROC across time for each dynamic graph in the test set. Dynamic graphs tend to perform worse than static ones, indicating leveraging the full timeseries on our chosen datasets do not improve performance.

	Epidemiological Dynamic Graphs		Higgs Dynamic Graphs	
GNN	**Regular**	**Powerlaw**	**Higgs 1**	**Higgs 2**
GCN	0.96 ± 0.02	0.81 ± 0.01	0.56 ± 0.01	0.54 ± 0.01
GAT	0.95 ± 0.02	0.78 ± 0.01	0.50 ± 0.01	0.50 ± 0.01
SAGE	**0.96 ± 0.01**	**0.82 ± 0.01**	0.61 ± 0.00	**0.55 ± 0.00**
DGNN	0.86 ± 0.01	0.69 ± 0.01	**0.66 ± 0.07**	OOM
GCRN	0.86 ± 0.01	0.67 ± 0.02	0.53 ± 0.02	OOM
EvolveGCN	0.83 ± 0.01	0.62 ± 0.01	0.56 ± 0.05	OOM

D EXPERIMENTAL DETAILS

D.1 Pseudo-synthetic Dataset Details

We first describe how we utilize the SI model in generating the dynamic graph for the pseudo-synthetic dataset. Formally, at timestep t susceptible node i can become infected by an infected neighbor j with probability $\alpha_j \cdot \beta_i$, where $\alpha_j \in [0, 1]$ is the infectivity of j and $\beta_i \in [0, 1]$ is the susceptibility of i. Then, at timestep t for susceptible node i its label at timestep $t+1$ is sampled from a Bernoulli parameterized by $1-\prod_{j \in \mathcal{N}_t(i)}(1-\alpha_j \cdot \beta_i \cdot \mathbb{I}(y_t(j) == +1))$.

Given the graph structure, labels are generated by first assigning to each node hidden parameters α_i and β_i, sampled from unique beta distributions specified prior to assignment that do not change over time. Parameters for the beta distributions are selected such that all nodes in the dynamic graph are infected at the final timestep. We then randomly sample a small set of nodes at timestep $t = 0$ as initially infected, determining $\mathbf{y}_0$. Then,

we use the SI model to generate the label sequence $\{\mathbf{y}_1, \ldots, \mathbf{y}_T\}$. At time t, node i's feature vector is given by, $\mathbf{x}_t(i) = [y_t(i)||\alpha(i)||\beta(i)]$.

We now present the pseudocode for generation of each dynamic graph in Algorithm 1. Below, we summarize the label and feature generation process for an arbitrary node i:

$$\alpha_i \sim \mathcal{B}eta(\theta_{1,\text{inf}}, \theta_{2,\text{inf}}) \tag{78}$$

$$\beta_i \sim \mathcal{B}eta(\theta_{1,\text{sus}}, \theta_{2,\text{sus}}) \tag{79}$$

$$\mathbf{x}_t(i) = [y_t(i)||\alpha(i)||\beta(i)] \tag{80}$$

$$y_{t+1}(i) \sim \begin{cases} B(p_{\text{init}}), & \text{if } t = 0 \\ B\Big(1 - \prod\limits_{j \in \tilde{\mathcal{N}}_{t-1}(i)} (1 - \alpha_j \cdot \beta_i \cdot \mathbb{I}(y_t(j) = +1))\Big), & \text{if } t > 0 \end{cases} \tag{81}$$

where, $\mathcal{B}eta(\theta_{1,\text{inf}}, \theta_{2,\text{inf}})$ and $\mathcal{B}eta(\theta_{1,\text{sus}}, \theta_{2,\text{sus}})$ represents the beta distributions for the infectivity and susceptibility parameters, and B represents the bernoulli distribution. Importantly, the choice of parameters for the beta distributions affect the behavior of the infection. If the underlying susceptibilities of all nodes are too high or the underlying infectivity of all nodes are too low, infection does not spread to each node in the dynamic graphs. Ideally, we want to avoid degenerate infection cases where infection does not spread. Thus, in order to generate our dynamic graphs we ensure infection spreads throughout the entire graph by the final timestep by choosing the parameters for the beta distributions appropriately.

Algorithm 1 SI Model of Infectious Disease

procedure SI($V_{0:T}$, $\mathbf{A}_{0:T}$, $\mathcal{B}eta(\theta_{1,\text{inf}}, \theta_{2,\text{inf}})$, $\mathcal{B}eta(\theta_{1,\text{sus}}, \theta_{2,\text{sus}})$, p_{init})
  for $i \in V_{0:T}$ **do** # Initialize infectivity parameter, susceptibility parameter, and $y_0(i)$ for all nodes
    if i has not been assigned parameters α and β **then**
      $\alpha_i \sim \mathcal{B}eta(\theta_{1,\text{inf}}, \theta_{2,\text{inf}})$
      $\beta_i \sim \mathcal{B}eta(\theta_{1,\text{sus}}, \theta_{2,\text{sus}})$
    end if
  end for
  for $i \in V_0$ **do**
    $y_0(i) \sim B(p_{\text{init}})$
  end for
  for $t \leftarrow 0, T$ **do** # For all timesteps, for all nodes in V_t, simulate SI infection
    for $i \in V_t$ **do**
      if $y_t(i) == 0$ **then**
        for $j \in \mathcal{N}_t(i)$ **do**
          $y_t(i) \sim B(\alpha_j \cdot \beta_i \cdot \mathbb{I}(y_t(j) = +1))$
          if $y_t(i) == +1$ **then**
            break
          end if
        end for
      end if
    end for
  end for
  for $t \leftarrow 0, T$ **do** # Gather features for all nodes for all timesteps
    for $i \in V_t$ **do**
      $\mathbf{x_t(i)} \leftarrow [y_t(i), \alpha_i, \beta_i]$
    end for
  end for
  return $G_{0:T} = (V_t, \mathbf{A}_t, \mathbf{X}_t, \mathbf{y}_t)$
end procedure

Below we provide additional details about each dataset used in the main paper including specific parameters for both synthetic and real-world networks. We begin with the synthetic datasets, each of which are generated in Networkx 3.2 (Hagberg and Conway, 2020),

Table 5: Statistics for dynamic graphs. N is the number of dynamic graphs provided in each of the train, validation, and test set.

Dynamic graph	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	T	N
Regular	1,000	45,000	30	20
Powerlaw	1,000	90,000	30	20
Block	1,000	150,000	30	20
UCI	1,899	59,835	29	20
Bitcoin	5,881	35,592	35	20
Math	24,818	506,550	20	20
Higgs Day 1	3,124	16,781	24	1
Higgs Day 2	12,075	127,662	24	1
Higgs Day 3	23,853	304,255	24	1
Higgs Day 4	30,822	459,707	24	1
Gavin	2,541	140,040	36	1
Ho	1,548	42,220	36	1
Ito	2,856	8,638	36	1
Uetz	922	2,159	36	1

- **Regular**: a 3-regular graph on $n = 1000$ nodes.
- **Powerlaw**: an albert barabasi graph with $n = 1000$ nodes grown by attaching new nodes each with $m = 3$ edges that are preferentially attached to existing nodes with high degree.
- **Block**: a stochastic block model on $n = 1000$ nodes with $c = 20$ communities of equal size where a node forms an edge within its community with probability $p_{\text{in}} = 0.10$ and an edge outside of its community with probability $p_{\text{out}} = 0.001$.

We provide the dataset details for our real datasets. For each real dataset, we discretize the dynamic graph by setting the static graph at a particular timestep as the tuple of nodes and edges that lie within the timestep's associated time interval. More specifically, the static graph at time t is composed of all nodes and edges that lie within the interval $[tw,\, t(w+1)]$, where w is the interval size. For each dataset, we list the specific choice of w.

- **UCI**: a dynamic network of private messages sent on an online social network at the University of California Irvine. Nodes represent users, and edges represent private messages between users. There are a total of $|\mathcal{E}| = 59835$ temporal edges, and we set the interval size equal to $w = 2000$.
- **Bitcoin**: a dynamic network of transactions on the Bitcoin platform Bitcoin OTC. Nodes represent users and edges represent transactions. There are a total of $|\mathcal{E}| = 35592$ temporal edges, and we set the interval size equal to $w = 1000$.
- **Math**: a dynamic network of interactions on the stack exchange web site Math Overflow. Nodes represent users, and edges represent various types of interactions including answering questions, commenting on questions, and commenting on answers. There are a total of $|\mathcal{E}| = 506550$ temporal edges, and we set the interval size equal to $w = 12000$

D.2 Real Datasets

The Higgs dataset is a social network of twitter where nodes are users and edges indicate friendship and follower statuses. The dataset records tweets, replies, and mentions of the announcement of the discovery of the Higgs boson between users. As described in the main paper, we use four separate daily graphs each divided into 24 hours. At each timestep, the task is to predict which nodes become positive at the next timestep. At time t, we construct features for each node based on a learnable embedding matrix concatenated with the positive statuses of the nodes at timestep t. At time t, future labels for nodes are are the positive statuses of nodes at timestep $t + 1$.

The protein-protein interaction networks are from a biological repository of dynamic protein-protein interaction networks (DPPIN) (Fu and He, 2022). As described in the main paper, we consider four separate dynamic graphs

each spanning 36 timestamps where timesteps represent 3 successive metabolic cycles of yeast cells at different resolutions. We split the graph chronologically into 10 train graphs, 10 validation graphs, and 16 test graphs. At time t we construct features for each node based on the node's protein type concatenated with the active statuses of the nodes at timestep t. At time t, future labels for nodes are the active statuses of the nodes at time $t+1$. We summarize statistics of all datasets in Table 5.

D.3 Model Details

We provide a summary of all GNNs used in the main paper below. First, we describe the static GNNs in the context of homophily, including their aggregation scheme and additional designs that aim to alleviate specific heterophilous settings. Next, we describe the dynamic GNNs in the context of their dynamic components.

- **SGC** (homophilous) uses symmetric-normalized mean aggregation without any intermediate weights or nonlinearities between layers (Wu et al., 2019).
- **GCN** (homophilous) uses symmetric-normalized mean aggregation, including intermediate weights and nonlinearities between layers (Kipf and Welling, 2017).
- **GIN** (homophilous) is theoretically more expressive than GCNs, leveraging sum aggregation (Xu et al., 2019).
- **GAT** (homophilous) uses self-attention aggregation (Veličković et al., 2018). GAT is a popular choice included in our experiments for completeness.
- **SAGE** (heterophilous) uses mean aggregation, but incorporates the heterophilous design choice of separating the ego-representations from aggregated neighbor representations (Hamilton et al., 2017).
- **GCNII** (heterophilous) uses symmetric-normalized mean aggregation, but incorporates the heterophilous design choice of the addition of residual connections (Li et al., 2018).
- **FA-GCN** (heterophilous) uses attention-based aggregation, but incorporates the heterophilous design choices of both SAGE and GCNII (Bo et al., 2021).

In our dynamic GNN experiments, we train the following dynamic GNNs described below in the context of their dynamic components.

- **DGNN** combines a GCN and RNN by first applying the GCN to the initial node representations for all nodes across all timesteps. Next, an RNN is applied to each timeseries of node representations. Finally, a readout layer is applied to obtain the prediction for all nodes across all timesteps. This particular formulation for dynamic GCNs has been widely used in many dynamic graph applications (Manessi et al., 2020; Narayan and Roe, 2018; Chen et al., 2022).
- **GCRN** combines a GCN and RNN by replacing each weight update in the RNN with a GCN. The new RNN layer, GCRN, is applied to each timeseries of node representations, and a readout layer is applied to obtain the the prediction for all nodes across all timesteps (Seo et al., 2018).
- **EvolveGCN** combines a GCN and RNN by using an RNN evolve the weights of GCN layers across timesteps (Pareja et al., 2020).

We implement and train our models on a GeForce GTX 1080.

D.4 Training and Evaluation Details

We train all GNNs using the Adam optimizer and in full batch mode where each batch consists of a single static graph (Kingma and Ba, 2015). We perform a hyperparameter search over number of layers in the range [0, 3], learning rate in the range [.1, .0001], and the size of hidden dimension in the range [32, 128]. Training is stopped when either 200 epochs are reached or when validation performance no longer improves after 50 epochs. We report the mean test AUROC and compute correlations over the top 4 models with the best validation AUROC.

We now describe the computation of static and dynamic homophily. At a particular timestep t, we compute static homophily as the mean over all static local homophily levels in the static graph at time t. Formally,

$$h_t^S = \frac{1}{|V_t|} \sum_{i \in V_t} h_t^S(i) = \frac{1}{|V_t|} \sum_{i \in V_t} \sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{\mathbb{I}[y_t(i) == y_t(j)]}{\left|\hat{\mathcal{N}}_t(i)\right|}, \tag{82}$$

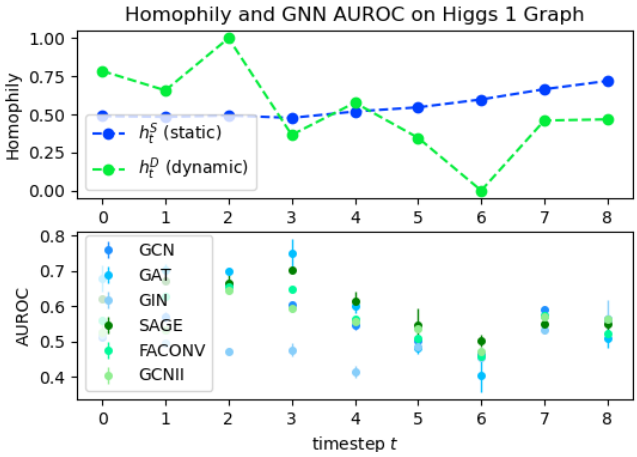
where $h_t^S(i)$ is the local static homophily for node i defined as the ratio of neighbors with the same label as itself at time t. Following Definition 1, we compute dynamic homophily as the following:

$$h_t^D = \frac{1}{2}\left(\frac{1}{|V_{t+1}^+|} \sum_{i \in V_{t+1}^+} h_t^D(i) + \frac{1}{|V_{t+1}^-|} \sum_{i \in V_{t+1}^-} h_t^D(i)\right) \tag{83}$$

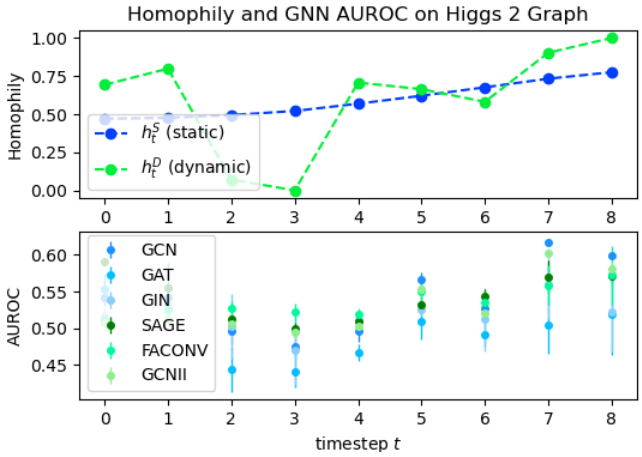
$$= \frac{1}{2}\left(\frac{1}{|V_{t+1}^+|} \sum_{i \in V_{t+1}^+} \sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{\mathbb{I}[y_{t+1}(i) == y_t(j)]}{\left|\hat{\mathcal{N}}_t(i)\right|} + \frac{1}{|V_{t+1}^-|} \sum_{i \in V_{t+1}^-} \sum_{j \in \hat{\mathcal{N}}_t(i)} \frac{\mathbb{I}[y_{t+1}(i) == y_t(j)]}{\left|\hat{\mathcal{N}}_t(i)\right|}\right). \tag{84}$$

E ADDITIONAL PLOTS

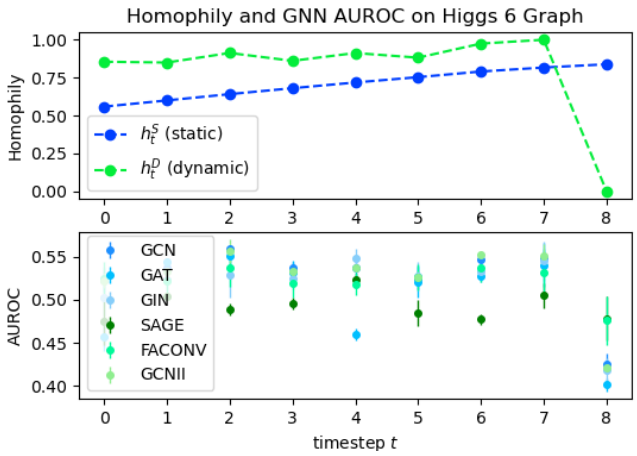
Below we include additional plots on all combinations of GNNs and dynamic graphs.



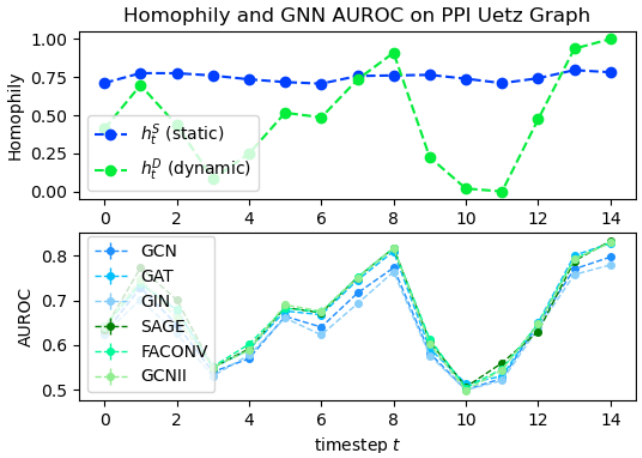
(a) Higgs 1 dynamic graph



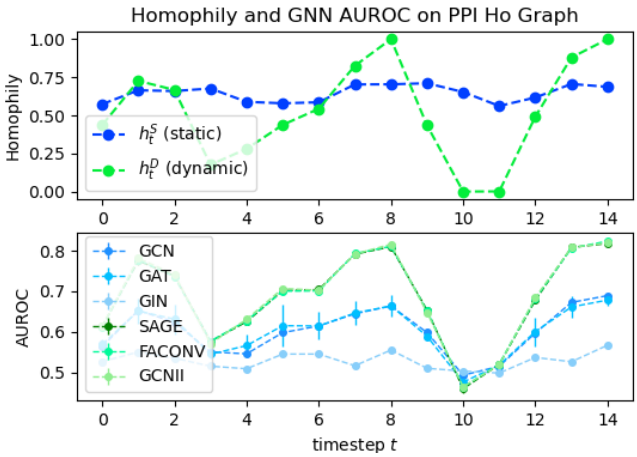
(b) Higgs 2 dynamic graph



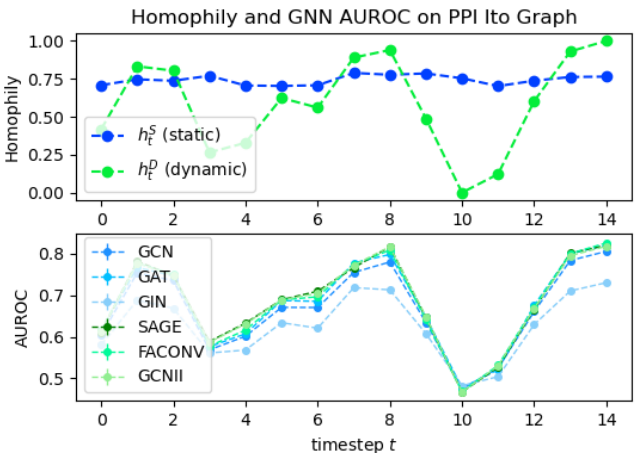
(a) Higgs 6 dynamic graph



(b) DPPIN Uetz dynamic graph



(a) DPPIN Ho dynamic graph



(b) DPPIN Ito dynamic graph

F SOCIETAL IMPACT

As our work aims to understand the performance of GNNs in general dynamic node classification settings, we do not foresee any immediate *negative* societal outcomes. For particular applications of dynamic node classification settings such as predicting the spread of infectious disease or misinformation, our work sheds light on current GNN limitations, potentially guiding the design of future GNNs aimed at better solving these tasks.