Node Mutual Information: Enhancing Graph Neural Networks for Heterophily

Seongjin Choi[1*] **Gahee Kim**[2*] **Se-Young Yun**[2]
[1]POSTECH [2]KAIST
jincs10@postech.ac.kr
{gaheekim,yunseyoung}@kaist.ac.kr

Abstract

Graph neural networks (GNNs) have achieved great success in graph analysis by leveraging homophily, where connected nodes share similar properties. However, GNNs struggle on heterophilic graphs where connected nodes tend to differ. Some of the existing methods use neighborhood expansion which is intractable for large graphs. This paper proposes utilizing node mutual information (MI) to capture dependencies between nodes in heterophilic graphs for use in GNNs. We first define a probability space associated with the graph and introduce k^{th} node random variables to partition the graph based on node distances. The MI between two nodes' random variables then quantifies their dependency regardless of distance by considering both direct and indirect connections. We propose k^{th} MIGNN where the k^{th} MI values are used as weights in the message aggregation function. Experiments on real-world datasets with varying heterophily ratios show the proposed method achieves competitive performance compared to baseline GNNs. The results demonstrate that leveraging node mutual information effectively captures complex node dependencies in heterophilic graphs.

1 Introduction

Graphs have a wide range of applications in various fields, such as data analysis [25], chemistry [32], biology [6, 7], and sociology [11, 27]. Within these fields, tasks such as graph isomorphism problems, link predictions, and node classifications are particularly important [30, 33, 24]. The use of neural networks in analyzing graph data has led to significant advancements [23, 9, 8, 14, 4], thanks in part to the homophily assumption that neighboring nodes should share similar features or labels. As nodes are updated at each layer of Graph Neural Networks (GNNs), their features are based on those of their adjacent nodes.

However, some graphs do not follow the homophily principle, termed heterophilic graphs, where adjacent nodes are more likely to exhibit dissimilar features or labels. A significant portion of real-world graphs is heterophilic in nature, including examples like molecules and webpages [35, 19]. The direct application of GNNs to these heterophilic graphs has yielded unsatisfactory results due to the absence of homophily, a fundamental assumption underlying the performance of GNNs.

To suitably process heterophilic graphs, adaptations to GNNs are necessitated. Prior studies have introduced neighborhood extension methods that consider not just the characteristics of adjacent nodes, but also incorporate other node features representative of heterophily when aggregated [20, 35, 15, 12, 13, 26, 18, 31, 10, 28]. However, the neighborhood extension methods used in previous studies are intractable for large datasets. In addition, recent studies[21][22] have identified problems with existing heterophily benchmark datasets[20], such as duplication of nodes and instability of

[*]Equally contributed

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

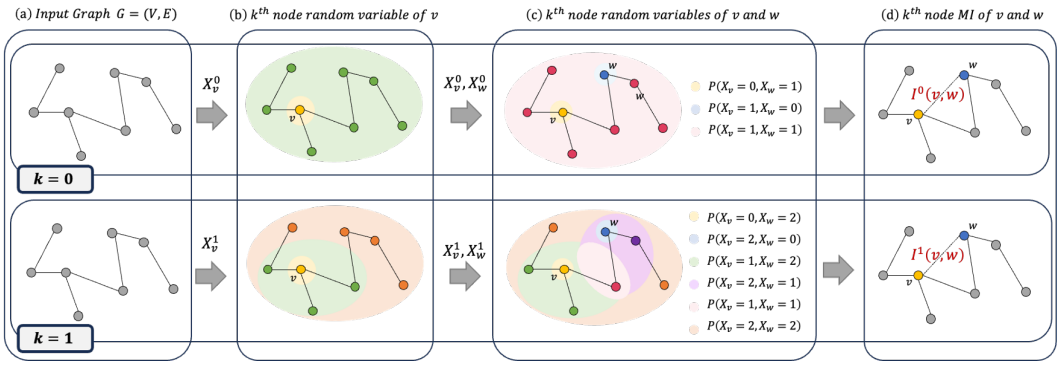


Figure 1: (a) The input graph G is given with a set of nodes V and a set of edges E. (b) Node random variables are defined according to k. The 0^{th} node random variable of v only separates v itself from the rest of the nodes. On the other hand, the 1^{st} node random variable of v divides the nodes in the graph into v, its 1-hop neighbors, and the rest of the nodes. (c) Two node random variables generate different joint probability distributions. (d) Node mutual information from the probability distribution induced by the k^{th} node random variables. Node mutual information is a measure of the dependency between two random variables. It can be used in GNNs architecture as edge weight.

results due to small size. These studies have proposed a new heterophily dataset that addresses these problems. Previous studies have performed poorly on this new dataset. To address this problem without neighborhood extension, we propose a novel GNN that incorporates k^{th} node mutual information edge weights to capture the dependency between nodes in a graph.

We first consider a graph as a probability space, where the probability measure of a node indicates how much it is connected to other nodes in the graph. We then define the independence of two nodes as the independence of the subsets generated by their 0-hop neighborhoods. We can refine the notion of independence by considering the l-hop neighborhoods of node v for l from 0 to k. We can do this by setting a k^{th} node random variable associated with node v, which assigns different values to other nodes according to their shortest path distance from node v.

Figure 1(b) shows how the node random variable of v is defined according to k, and how the node random variable divides the probability space in each case. For example, the 0^{th} node random variable only considers the nodes that are directly connected to v. Figure 1(c) shows the joint probability distribution generated by the node random variables of two nodes v and w to capture the dependencies of the two nodes. Since the 1^{st} node random variable considers the 1-hop neighborhood of node v, it can be seen that it considers a more refined dependency than the 0^{th} node random variable case.

In real-world data, it is almost impossible for two nodes to be independent, even if $k = 0$. This is because nodes in real-world graphs are often connected in complex ways[17][22], and there is always some degree of dependency between them. Therefore, we need to capture how much two nodes are dependent with k^{th} degree accuracy. We can do this by using mutual information, which is a measure of the dependency between two k^{th} node random variables. MI is a good measure of dependency because it takes into account both the direct and indirect connections between two nodes. For example, if two nodes are directly connected, then they will have a high MI value. However, even if two nodes are not directly connected, they may still have a high MI value if they are indirectly connected through other nodes.

The proposed GNN uses MI to weigh the edges in the graph. This means that edges between nodes that are more dependent will be given higher weights. This helps the GNN to learn more accurate representations of the nodes, which results in better performance in heterophilic graphs.

Our contributions can be summarized as follows:

- We propose a novel method for measuring the node mutual information between two nodes by introducing a degree-based probability measure and subgraph-based random variables.
- We proposed a new message-passing method in GNNs that leverages node mutual information as the weight of message aggregation.

- We conduct experiments on real-world datasets to validate the effectiveness of our proposed methods on heterophilic datasets. The experimental results demonstrate the efficacy of our proposed node mutual information method.

The remainder of the paper is organized as follows: Section 2 provides background on graph heterophily and mutual information through a probabilistic lens. In section 3 and 4, we define node mutual information within a graph and discuss how node mutual information differs from some common node similarity indices. Section 5 describes how to utilize node mutual information in message aggregation. Section 6 presents the experimental results obtained from real-world datasets. Finally, Section 7 concludes the findings of the paper.

2 Preliminaries

2.1 Homophily and heterophily

Homophily refers to the property that connected nodes in a graph tend to have similar properties. Various metrics have been proposed to quantify homophily in a graph. Node homophily[20] measures the ratio of nodes that share the same class among adjacent nodes in the entire graph. It is defined as:

$$\mathcal{H}_{node}(G) = \frac{1}{|V|} \sum_{v \in V} \frac{|\{u | u \in N_v, y_u = y_v\}|}{|N_v|},$$

where $G = (V, E)$ is the graph with a set of nodes V and a set of edges E. N_v denotes the neighbors of node v, and y_v is the label of node v. Edge homophily[35][2] refers to the ratio of edges that connect nodes of the same class among all edges in the graph. It is defined as:

$$\mathcal{H}_{edge}(G) = \frac{|\{(u, v) | (u, v) \in E, y_u = y_v\}|}{|E|},$$

where (u, v) denotes an edge between node u and v. While intuitive, these measures are sensitive to the number or balance of classes. Class homophily[17] evaluates homophily at the class level regardless of class imbalance. It is defined as:

$$\mathcal{H}_{class}(G) = \frac{1}{C} \sum_{k=1}^{C} \left[h_k - \frac{|\{v | y_v = C_k\}|}{|V|} \right]_+, \; h_k = \frac{\sum_{x \in \{v | y_v = C_k\}} |\{u | u \in N_x, y_u = C_k\}|}{\sum_{x \in \{v | y_v = C_k\}} |N_x|},$$

where C_k is the k^{th} labels, $[a]_+ = max(a, 0)$ and h_k is the class-wise homophily metric.

However, those homophily in the same graphs can vary. [21] addressed this by characterizing homophily and proposed adjusted homophily. It is defined as:

$$\mathcal{H}_{adjusted}(G) = \frac{\mathcal{H}_{edge}(G) - \sum_{k=1}^{C} D_k^2/(2|E|)^2}{1 - \sum_{k=1}^{C} D_k^2/(2|E|)^2},$$

where $D_k := \sum_{v: y_v = k} d(v)$ and $d(v)$ denotes the degree of a node v. A lower homophily indicates a more heterophilic graph. This paper uses these metrics to analyze heterophilic graphs.

2.2 Mutual information

A probability space $(\Omega, \mathcal{F}, \mathbb{P})$ is a triple where Ω is a sample space, $\mathcal{F}$ is an event space, and $\mathbb{P}$ is a probability measure. A discrete random variable X is a function $X : \Omega \to \mathbb{R}$ such that $X(\Omega)$ is at most countable. We can compute the probability of an experiment X with values in $A \subset X(\Omega)$ by $\mathbb{P}\left(X^{-1}(A)\right)$. One of the important concepts is the independence of two random variables X, Y. Two random variables are independent if $\mathbb{P}\left(X^{-1}(p) \cap Y^{-1}(q)\right) = \mathbb{P}\left(X^{-1}(p)\right) \cdot \mathbb{P}\left(Y^{-1}(q)\right)$ for any $p, q \in \mathbb{R}$. If they are not independent, we can measure how much two random variables are dependent, called the mutual information of X, Y.

We can compute the mutual information of X, Y, $I(X, Y)$, by computing the entropy of X, Y and the joint entropy of X and Y. The formula for $I(X, Y)$ for discrete random variables X, Y is given by

$$I(X, Y) = -\sum_{i,j} \mathbb{P}\left(X^{-1}(p_i) \cap Y^{-1}(q_j)\right) \cdot \log\left(\frac{\mathbb{P}\left(X^{-1}(p_i)\right) \cdot \mathbb{P}\left(Y^{-1}(q_j)\right)}{\mathbb{P}\left(X^{-1}(p_i) \cap Y^{-1}(q_j)\right)} \right)$$

where the images of X, Y are $\{p_i\}_{i=1,\ldots,m}, \{q_j\}_{j=1,\ldots,n}$. $0 \le I(X,Y) \le H(X,Y)$ and $I(X,Y) = 0$ for independent X, Y. Please refer to Appendix A for more details.

3 Node Mutual Information

In this section, we establish a formal definition for k^{th} node mutual information between two nodes. First, we give a degree-based probability measure on a graph.

Definition 3.1. *(Probability space associated with the graph) Given a graph $G = (V, E)$, a probability space associated to the graph is a triple $(\Omega, \mathcal{F}, \mathbb{P})$ with $\Omega := V$, $\mathcal{F} := 2^V$, and $\mathbb{P}(A) := \frac{\sum_{v\in A} d(v)}{\sum_{w\in V} d(w)}$ for $A \in \mathcal{F}$, where $d(v)$ is the outgoing degree of v.*

It can be verified that $(V, 2^V, \mathbb{P})$ is indeed a probability space (See the AppendixA). $\mathbb{P}(A)$ encapsulates the presence of nodes through the originating edges from A. We refer to $\mathbb{P}(A)$ as the impact of A.

To say about the degree of the dependence of two nodes, we must partition V according to the shortest path distance for a given node. A k^{th} node random variable associated with v exactly captures this information.

Definition 3.2. *(k^{th} Node random variable associated to v for $k = 0, 1, \ldots$) Let $N_{-1}(v) := \varnothing$, $N_0(v)$ be the ego-node $\{v\}$ and $N_l(v)$ be the l-hop neighborhood of v for $l \ge 1$.*
A 0^{th} node random variable X_v^0 associated to v is a measurable function $X_v^0 : V \to \mathbb{R}$ defined by

$$X_v^0(w) := \begin{cases} 0 & \text{if } w \in N_0(v) \\ 1 & \text{if } w \in V \backslash N_0(v). \end{cases} \tag{1}$$

A k^{th}($k \ge 1$) node random variable X_v^k associated to v is a measurable function $X_v^k : V \to \mathbb{R}$ defined by

$$X_v^k(w) := \begin{cases} 0 & \text{if } w \in N_0(v) \\ j & \text{if } w \in N_j(v) \backslash N_{j-1}(v) \text{ for } j = 1, \ldots, k \\ k+1 & \text{if } w \in V \backslash N_k(v). \end{cases} \tag{2}$$

X_v^k is automatically measurable (See the AppendixA), so we can talk about $\mathbb{P}\left((X_v^k)^{-1}(p)\right)$ for any $p \in \mathbb{R}$. Moving forward, we introduce the notions of k^{th} node entropy and k^{th} joint node entropy.

Definition 3.3. *(k^{th} Entropy of the node, k^{th} joint entropy of two nodes for $k = 0, 1, \ldots$) Suppose the probability space $(V, 2^V, \mathbb{P})$ associated to the graph $G = (V, E)$ is given. Let v, w be two nodes of G.*

- *A k^{th} entropy of the node v, $H^k(v)$, is the entropy of random variable X_v^k*

$$H^k(v) := H(X_v^k) = -\sum_{i=0,1,\ldots,k+1} \mathbb{P}\left((X_v^k)^{-1}(i)\right) \cdot \log\left(\mathbb{P}\left((X_v^k)^{-1}(i)\right)\right) \tag{3}$$

- *A k^{th} joint entropy of two nodes v and w, $H^k(v, w)$, is the joint entropy of two random variables X_v^k and X_w^k*

$$\begin{aligned} H^k(v,w) &:= H(X_v^k, X_w^k) \\ &= -\sum_{i,j=0,1,\ldots,k+1} \mathbb{P}\left((X_v^k)^{-1}(i) \cap (X_w^k)^{-1}(j)\right) \cdot \log\left(\mathbb{P}\left((X_v^k)^{-1}(i) \cap (X_w^k)^{-1}(j)\right)\right) \end{aligned} \tag{4}$$

The k^{th} node entropy characterizes the distribution of impacts among $N_j(v) \backslash N_{j-1}(v)$ for $j = 0, \ldots, k$ and $V \backslash N_k(v)$. For instance, if their impacts are uniformly distributed on $\{N_j(v) \backslash N_{j-1}(v), V \backslash N_k(v)\}_{j=0,\ldots,k}$ with each having an impact of $\frac{1}{k+1}$, then $H^k(v)$ would be $\log(k+1)$. On the other hand, if one of the impacts is 1 while others are 0, then $H^k(v)$ would be 0. In general, the impacts are distributed in a certain manner, and the node entropy captures this information. The k^{th} joint entropy $H^k(v, w)$ reflects how the impacts within the partition $\{(X_v^k)^{-1}(i) \cap (X_w^k)^{-1}(j)\}_{i,j=0,1,\ldots,k+1}$ are distributed. By leveraging the concept of entropy, we

can define the mutual information between nodes and employ it to quantify the distance between two nodes based on their mutual information.

Definition 3.4. *(k^{th} Node mutual information, k^{th} distance induced by the mutual information) Suppose we have two nodes v, w with node random variables X_v^k, X_w^k.*

- *A k^{th} node mutual information of v and w, $I^k(v,w)$, is a mutual information between two random variables X_v^k and X_w^k*

$$I^k(v,w) := H^k(v) + H^k(w) - H^k(v,w). \tag{5}$$

- *A k^{th} distance of two nodes induced by the mutual information, $D^k(v,w)$, is defined by*

$$D^k(v,w) := 1 - \frac{I^k(v,w)}{H^k(v,w)}. \tag{6}$$

- *A k^{th} normalized node mutual information of v, w, $(I')^k(v,w)$, is defined by*

$$(I')^k(v,w) := 1 - D^k(v,w) = \frac{H^k(v) + H^k(w) - H^k(v,w)}{H^k(v,w)}. \tag{7}$$

The k^{th} node mutual information $I^k(v,w)$ quantifies the dependence of $\{(X_v^k)^{-1}(i) \cap (X_w^k)^{-1}(j)\}_{i,j=0,1,\ldots,k+1}$. The k^{th} distance $D^k(v,w)$ is a metric that arises from the node mutual information, with values ranging from 0 to 1 (See the AppendixA). If two nodes have a strong interaction, their distance will be small. For example, $D^k(v,v) = 0$. To assign a larger weight to nodes with significant interaction, we introduce a modified measure $(I')^k(v,w)$ defined as $1 - D^k(v,w)$. Consequently, $(I')^k(v,v) = 1$. Notably, $(I')^k(v,w) > (I')^k(v',w')$ implies that the interaction between nodes v and w is more active than the interaction between nodes v' and w'. We refer to the k^{th} normalized node mutual information as MI.
We mainly consider $k = 1$, so we omit $k = 1$ and write $X_v, H(v), H(v,w), I(v,w), D(v,w)$ and $I'(v,w)$ from now on.

4 Comparison to Node Similarity Indices

In this section, we will explore the difference between node mutual information and other similarity metrics. First, we will introduce the existing node similarity indices, followed by a discussion of simple examples and heterophilic graphs that highlight the unique characteristics of node mutual information.

4.1 Node similarity indices

[16] proposed common neighbor, Jaccard coefficient, preferential attachment, and Adamic-Adar index to measure how two nodes are similar. Common neighbor quantifies shared neighbors, while adjacency determines direct connections between nodes. The Jaccard coefficient normalizes common neighbors by neighborhood unions. Preferential attachment considers node degrees, and Adamic-Adar measures reciprocal logarithmic sums of shared neighbor degrees. Resource allocation[34] is similar to Adamic-Adar but without logarithms. Table 1 compares the similarities of nodes u and v measured by the existing indices and MI in the six different graphs in Figure 2. The results indicate that MI better reflects the configuration of the graph compared to existing similarity indices.

Adjacency index only takes into account the connection information between two nodes. It does not consider the number of common neighbors or the structural configuration of the subgraph. In Figure 2, the adjacency index assigns a value of 1 to example (a), indicating a perfect similarity between the nodes. However, in all other cases where there is a shared node, the adjacency index assigns a value of 0 uniformly.

Common neighbor and Jaccard index are more sophisticated measures of similarity that take into account the number of common neighbors. However, it does not consider the structural configuration

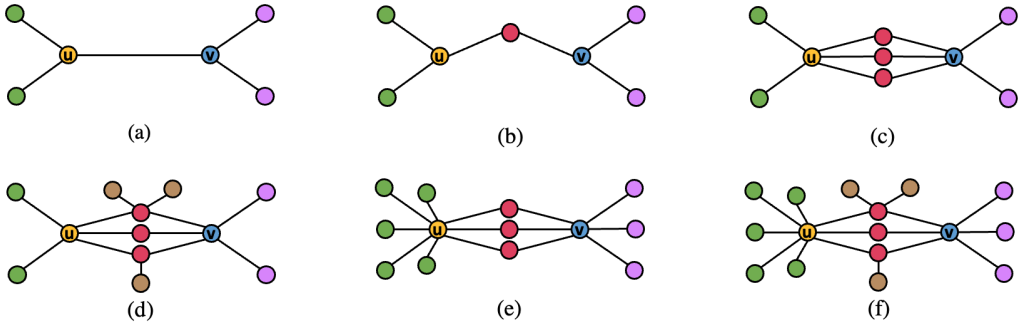


Figure 2: Six different examples

	MI	Adjacency	Common Neighbor	Jaccard	Preferential Attachment	Adamic-Adar	Resource Allocation
(a)	**0.51**	1	0	0.00	9	0.00	0.00
(b)	**0.36**	0	1	0.20	9	1.44	0.50
(c)	**0.43**	0	3	0.43	25	4.33	1.50
(d)	**0.27**	0	3	0.43	25	3.07	1.08
(e)	**0.37**	0	3	0.27	48	4.33	1.50
(f)	**0.23**	0	3	0.27	48	3.07	1.08

Table 1: Comparison of node similarity indices for examples

of the subgraph of sharing nodes. As a result, the Common Neighbor and Jaccard index can be misleading in graphs with different shared neighborhood configurations. For example, in Figure 2 (c-d) and (e-f), nodes u and v have the same number of common neighbors, but their shared neighborhoods have different structural configurations. The Jaccard index will assign them the same similarity score, even though they are not structurally similar.

Preferential Attachment is a measure of similarity that is based on the idea that nodes with more connections are more likely to be connected to other nodes. However, it does not consider the number of common neighbors or the structural configuration of the subgraph. In Figure 2 (a-b), (c-d), and (e-f), nodes u and v have the same degree, but they are not structurally similar. The Preferential Attachment will assign them the same similarity score, even though their connectivity(a-b) and configuration ((c-d) and (e-f)) are different.

Adamic-Adar and Resource Allocation are more sophisticated measures of similarity that take into account the number of common neighbors and the structural configuration of the subgraph. However, they are still not as accurate as MI. For example, in Figure 2 (c-e) and (d-f), nodes u and v have the same number of common neighbors, and their shared neighborhoods have the same structural configuration. The Adamic-Adar and Resource Allocation will assign them the same similarity score, even though they are not structurally similar.

None of these existing methods can capture both the connectivity and configuration of the subgraph of u and v. MI takes into account all of the configuration of the graph, which allows it to distinguish between structurally similar nodes even if they have the same number of common neighbors or the same degree.

5 Architecture

We propose architectures that aim to enhance Message Passing Neural Networks (MPNNs) for heterophily by leveraging node mutual information. In MPNNs, message passing for each layer comprises a message function M_t and a node feature update function U_t. The node feature $h_v^{(t+1)}$ of node

v at step $t+1$ is updated using the message m_{t+1} generated at step $t+1$: $h_v^{(t+1)} = U_t(h_v^{(t)}, m_v^{(t+1)})$. The message at step $t+1$, m_{t+1}, is computed by aggregating the neighborhood features from the previous step: $m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^{(t)}, h_w^{(t)})$ [8]. Therefore, the node representation can be expressed as:

$$\mathbf{h}_v^{(t+1)} = U_t\left(\mathbf{h}_v^{(t)}, \sum_{w \in \mathcal{N}(v)} M_t(\mathbf{h}_w^{(t)}, \mathbf{h}_v^{(t)})\right) \tag{8}$$

Node mutual information for message passing The proposed architecture is designed by utilizing MI in the message function M_t of the MPNNs. The MI value of two nodes increases as the impact of the shared node increases and depends on whether the two nodes are adjacent. We devised a method for utilizing MI as a weights of the message aggregation function. We can rewrite Equation 8 as follow:

$$\mathbf{h}_v^{(t+1)} = \sigma\left(\mathbf{h}_v^{(t)}, \mathbf{W}^{(t+1)} \sum_{w \in V} (I')^k(v, w) \cdot \mathbf{h}_w^{(t)}\right) \tag{9}$$

We refer to our method as k-MIGNN when it utilizes k^{th} node mutual information in the message aggregation function.

6 Experiments

We have discussed the definition and significance of node mutual information. We proceeded to conduct experiments to validate the effectiveness of node mutual information in GNNs. Evaluations were performed on six real-world datasets and one synthetic dataset with varying homophily ratios, measuring the mean accuracy and ROC AUC of the node classification task.

6.1 Experimental settings

6.1.1 Dataset

The most commonly used benchmark datasets for evaluating heterophily are WebKB and WikipediaNetwork, which were proposed by [20]. However, [22] recently discovered that WikipediaNetwork contains many duplicate nodes with identical neighborhoods and labels. This introduces train-test leakage, which allows models to achieve high performance without actually learning heterophily. To address this issue, [22] revised the WikipediaNetwork dataset by removing the duplicate nodes.

[22] also noted that existing benchmarks have limited diversity since they originate from a small number of domains, resulting in datasets with similar properties within the same domain. Therefore, [22] proposed five new datasets from different unique sources to better cover real-world scenarios. This includes the roman-empire, amazon-rating, minesweeper, tolokers, and questions. The roman-empire graph is based on the Wikipedia article on the Roman Empire with 22.7K word nodes and semantic edges. The amazon-ratings graph contains product nodes connected by co-purchase relationships with the task of predicting ratings. A synthetic minesweeper graph on a 100x100 grid provides node classification to identify mine locations. A real-world tolokers graph involving 11.8K crowd-sourcing platform users linked by common tasks aims to predict banned workers. Also, a medical question-answer website graph with 48.9K user nodes and answered-user edges poses the challenge of identifying active users.

The newly proposed datasets and modified WikipediaNetwork datasets exhibit low adjusted homophily and are thus considered heterophily benchmarks. Among these, minesweeper, tolokers, and questions are homophilic in terms of node and edge homophily but heterophilic regarding class and adjusted homophily. More details on the datasets are provided in Table 2.

6.1.2 Baselines

We conducted a thorough evaluation of our algorithm by comparing it to seven baseline methods. GCN[14] represents classical GNNs using convolutional operations. H2GCN [35] employs multi-hop message passing to aggregate information from potential neighborhoods. CPGNN [36] introduces a learnable compatibility matrix to model connectivity patterns between node classes. GloGNN[15] obtains representations using a coefficient matrix optimized to account for group effects. GPR-GNN

	squirrel	chameleon	roman-empire	amazon-ratings	minesweeper	tolokers	questions
$\mathcal{H}$(adjusted)	0.01	0.03	-0.05	0.14	0.01	0.09	0.02
$\mathcal{H}$(class)	0.03	0.06	0.02	0.13	0.01	0.17	0.09
$\mathcal{H}$(edge)	0.21	0.24	0.05	0.38	0.68	0.59	0.84
$\mathcal{H}$(node)	0.19	0.23	0.05	0.32	0.68	0.59	0.57
nodes	2223	890	22662	24492	10000	11758	48921
edges	46998	8854	32927	93050	39402	519000	153540
classes	5	5	18	5	2	2	2

Table 2: Dataset Statistics.

	squirrel	chameleon	roman-empire	amazon-ratings	minesweeper	tolokers	questions	Avg.Acc.
GCN	39.47 ± 1.47	40.89 ± 4.12	73.69 ± 0.74	48.70 ± 0.63	89.75 ± 0.52	**83.64 ± 0.67**	76.09 ± 1.27	64.60
H2GCN	35.10 ± 1.15	26.75 ± 3.64	60.11 ± 0.52	36.47 ± 0.23	89.71 ± 0.31	73.35 ± 1.01	63.59 ± 1.46	55.01
CPGNN	30.04 ± 2.03	33.00 ± 3.15	63.96 ± 0.62	39.79 ± 0.77	52.03 ± 5.46	73.36 ± 1.01	65.96 ± 1.95	51.16
GPR-GNN	38.95 ± 1.99	39.93 ± 3.30	64.85 ± 0.27	44.88 ± 0.34	86.24 ± 0.61	72.94 ± 0.97	55.48 ± 0.91	57.61
GloGNN	35.11 ± 1.24	25.09 ± 3.58	59.63 ± 0.69	36.89 ± 0.14	51.08 ± 1.23	73.39 ± 1.17	65.74 ± 1.19	49.56
FAGCN	41.08 ± 2.27	41.90 ± 2.72	65.22 ± 0.56	44.12 ± 0.30	88.17 ± 0.73	77.75 ± 1.05	**77.24 ± 1.26**	62.21
JacobConv	29.71 ± 1.66	39.00 ± 4.20	71.14 ± 0.42	43.55 ± 0.48	89.66 ± 0.40	68.66 ± 0.65	73.88 ± 1.16	59.37
0-MIGNN	**41.73 ± 2.58**	41.91 ± 3.98	86.92 ± 0.57	48.86 ± 0.48	84.13 ± 0.57	80.79 ± 0.82	73.10 ± 0.92	65.35
1-MIGNN	39.70 ± 1.76	42.83 ± 4.04	91.53 ± 0.47	**49.25 ± 0.66**	90.59 ± 0.64	82.53 ± 1.12	76.46 ± 1.24	67.56
2-MIGNN	40.70 ± 1.69	**44.05 ± 4.21**	**91.91 ± 0.40**	48.92 ± 0.59	**91.63 ± 0.67**	82.27 ± 1.06	75.97 ± 1.26	**67.92**

Table 3: Node classification results on seven datasets are reported. The highest accuracy for each dataset is in bold text and the second highest accuracy is underlined. Mean accuracy scores are reported for four datasets: squirrel, chameleon, roman-empire and amazon-ratings. ROC AUC scores are used to evaluate three other datasets: minesweeper, tolokers, and questions. The rightmost column shows the average accuracy across all datasets, calculated as the mean of the results to summarize overall performance.

[5] learns hidden features and then propagates them with learnable generalized pagerank weights that can adapt to the graph structure. FAGCN[3] aggregates different frequency signals flexibly based on network structure. JacobConv[28] uses Jacobi polynomials as an optimized linear spectral filter basis for expressive representations.

6.1.3 Hyperparameter settings

For our network architecture, we considered three hyperparameters: the number of layers, the dimension of hidden representations, and the dropout ratio. Additionally, we utilized two hyperparameters for the optimizer: the learning rate and weight decay. To determine suitable hyperparameter values, we conducted a grid search. For detailed information on the hyperparameter settings, see Appendix B.

6.2 Results on node classification

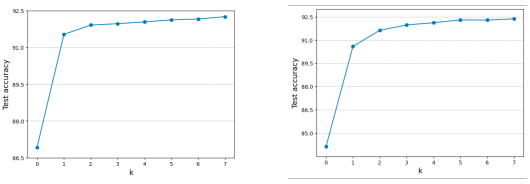
The competitive results demonstrate the effectiveness of MIGNN. When compared to the baseline, MIGNN achieves high overall performance. In particular, MIGNN outperforms GCN by almost 20 percentage points on the roman-empire dataset. Roman-empire is the most heterophilic dataset in terms of the four homophily ratios, which confirms that the proposed method works well on heterophilic graphs. Furthermore, MIGNN demonstrates exceptional performance, particularly on datasets characterized by low homophily ratios across all four types, such as the squirrel, chameleon, and Amazon rating datasets. Meanwhile, as mentioned above, the minesweeper, tolokers, and questions datasets exhibit disagreement between homophily ratios. These three datasets can also be considered homophilic graphs based on node and edge homophily.

From another perspective, roman-empire, squirrel, chameleon, and amazon-ratings have more classes than minesweeper, tolokers, and questions. Roman-empire has the most classes with 18, and our model tends to perform better when the number of classes is large.

We conducted node classification experiments for MI with k values of 0, 1, and 2 on all datasets. We observed that performance generally improved as k increased, supporting our hypothesis that the k^{th} node random variable forms a finer set in the graph probability space, allowing us to refine the dependency between two nodes.

6.3 MI for higher k

We conducted experiments on the roman-empire and minesweeper datasets to study the impact of k^{th} MI on node classification performance. The range of k is from 0 to 7, and the hyper-parameters other than k are fixed. Figure 3 shows the results of the node classification task on both datasets as k increases. Classification accuracy increased with increasing k in both datasets. This is because as k increases, the k^{th} node random variables take into account more distant neighbors at each node. This allows for a more precise representation of the dependency between two nodes, which in turn allows k^{th}-MIGNN to learn a better representation of each node. We believe that several factors, such as the diameter of the dataset and the number of classes in the dataset, will influence the determination of the optimal k for each dataset. Finding the optimal k value is a potential area for future work.



(a) roman-empire (b) minesweeper

Figure 3: Change in accuracy with respect to k

6.4 Number of layers

To evaluate the effect of changing the number of layers on performance, we fixed k to 1 in the roman-empire dataset and observed how the node classification performance changed as the number of layers increased from 2 to 6. Table 4 shows the results according to the number of layers of GCN and MIGNN. For GCN, performance peaked at 2 layers and then decreased as more layers were added, reaching between 2 and 6 layers. This is because GCN aggregates features from all neighboring nodes equally, regardless of connection strength. As layers increase, node features begin to resemble each other, hindering the ability of the model to classify nodes accurately. In contrast, MI saw continuous improvement in performance as the number of layers grew. MI encodes information about both the presence and importance of relationships between nodes during feature aggregation. Larger MI values represent stronger ties, so nodes with strong connections have a greater influence on updating representations. This allows nodes to maintain relatively unique embeddings based on their most important relationships. By incorporating connectivity strengths through MI, the model can leverage additional layers to gradually distill more elaborate node representations without causing excessive loss of discriminative power between nodes. Therefore, when MI is utilized in the aggregation function, performance gains can result from increasing network depth.

# of layer	2	3	4	5	6
GCN	**78.61**	78.38	77.74	76.94	76.81
1-MIGNN	88.72	89.96	90.90	91.21	**91.53**

Table 4: Change in accuracy with respect to number of layers

7 Conclusion

In this paper, we propose k^{th} node mutual information of two nodes to capture the dependence of two nodes together with their k-hop neighborhoods. We found that using the node mutual information in the edge weight of GNNs achieved competitive results for $k = 0, 1, 2$. The performance also increased as k increased, indicating that the node mutual information can capture more information about the relationships between nodes as the neighborhood size increases. Interestingly, we found that the performance of GNNs with node mutual information increased with the number of layers, while GCN plateaued after a few layers. This is because node mutual information encodes the importance of nodes during feature aggregation which allows the GNN to learn more complex relationships between nodes as the number of layers increases. Overall, node mutual information can capture the dependence of nodes together with their neighborhoods, which is important for understanding the relationships between nodes in heterophilic datasets. Using the node mutual information in GNNs can improve their performance on heterophilic datasets.

Acknowledgments and Disclosure of Funding

This work was supported by Samsung Electronics Co., Ltd (IO201209-07881-01) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program(KAIST)).

References

[1] *Probability: A Graduate Course*. Springer New York, 2005. doi: 10.1007/b138932. URL `https://doi.org/10.1007/b138932`.

[2] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pages 21–29. PMLR, 2019.

[3] 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.

[4] Benjamin Paul Chamberlain, Sergey Shirobokov, Emanuele Rossi, Fabrizio Frasca, Thomas Markovich, Nils Hammerla, Michael M Bronstein, and Max Hansmire. Graph neural networks for link prediction with subgraph sketching. *arXiv preprint arXiv:2209.15486*, 2022.

[5] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. *arXiv preprint arXiv:2006.07988*, 2020.

[6] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

[7] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. *Advances in neural information processing systems*, 30, 2017.

[8] 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.

[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[10] Dongxiao He, Chundong Liang, Huixin Liu, Mingxiang Wen, Pengfei Jiao, and Zhiyong Feng. Block modeling-guided graph convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4022–4029, 2022.

[11] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

[12] Di Jin, Zhizhi Yu, Cuiying Huo, Rui Wang, Xiao Wang, Dongxiao He, and Jiawei Han. Universal graph convolutional networks. *Advances in Neural Information Processing Systems*, 34:10654–10664, 2021.

[13] Wei Jin, Tyler Derr, Yiqi Wang, Yao Ma, Zitao Liu, and Jiliang Tang. Node similarity preserving graph convolutional networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 148–156, 2021.

[14] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[15] 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.

[16] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, 2003.

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

[18] Meng Liu, Zhengyang Wang, and Shuiwang Ji. Non-local graph neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 44(12):10270–10276, 2021.

[19] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *Proceedings of the 16th international conference on World Wide Web*, pages 201–210, 2007.

[20] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

[21] Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. Characterizing graph datasets for node classification: Beyond homophily-heterophily dichotomy. *arXiv preprint arXiv:2209.06177*, 2022.

[22] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: are we really making progress? *arXiv preprint arXiv:2302.11640*, 2023.

[23] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[24] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

[25] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

[26] Susheel Suresh, Vinith Budde, Jennifer Neville, Pan Li, and Jianzhu Ma. Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns. *arXiv preprint arXiv:2106.06586*, 2021.

[27] Lei Tang and Huan Liu. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 817–826, 2009.

[28] Tao Wang, Di Jin, Rui Wang, Dongxiao He, and Yuxiao Huang. Powerful graph convolutional networks with adaptive propagation mechanism for homophily and heterophily. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4210–4218, 2022.

[29] David Williams. *Probability with Martingales*. Cambridge University Press, February 1991. doi: 10.1017/cbo9780511813658. URL `https://doi.org/10.1017/cbo9780511813658`.

[30] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[31] Tianmeng Yang, Yujing Wang, Zhihan Yue, Yaming Yang, Yunhai Tong, and Jing Bai. Graph pointer neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8832–8839, 2022.

[32] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31, 2018.

[33] Muhan Zhang and Yixin Chen. Inductive matrix completion based on graph neural networks. *arXiv preprint arXiv:1904.12058*, 2019.

[34] Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71:623–630, 2009.

[35] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in Neural Information Processing Systems*, 33:7793–7804, 2020.

[36] 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.

A Basic measure theory

In this section, we provide basic measure theory (see [29], [1]). We prove that $(V, 2^V, \mathbb{P})$ associated to the graph $G = (V, E)$ is a probability space, any function $X : (V, 2^V, \mathbb{P}) \to \mathbb{R}$ is measurable and the mutual information is non-negative.

Definition A.1. *(Measurable space, Borel space) Let Ω be a set.*

1. *A collection $\mathcal{F} \subset 2^\Omega$ is called a σ-algebra on Ω if*
   (a) *$\mathcal{F}$ contains $\varnothing, \Omega$*
   (b) *$\mathcal{F}$ is closed under the complement (i.e. $A \in \mathcal{F} \implies \Omega \backslash A \in \mathcal{F}$).*
   (c) *$\mathcal{F}$ is closed under the countable union (i.e.$\{A_i\}_{i\in\mathbb{N}} \subset \mathcal{F} \implies \bigcup_i A_i \in \mathcal{F}$).*

   We call a pair $(\Omega, \mathcal{F})$ as a measurable space. An element $A \in \mathcal{F}$ is called a measurable set.
2. *A collection $\mathcal{T} \subset 2^\Omega$ is called a topology on Ω if*
   (a) *$\mathcal{T}$ contains $\varnothing, \Omega$*
   (b) *$\mathcal{T}$ is closed under arbitrary union (i.e. $\{U_\alpha\}_{\alpha\in\Lambda} \subset \mathcal{T} \implies \bigcup_{\alpha\in\Lambda} U_\alpha \in \mathcal{T}$).*
   (c) *$\mathcal{T}$ is closed under finite intersection (i.e. $\{U_i\}_{i=1,\dots,m} \subset \mathcal{T} \implies \bigcap_{i=1,\dots,m} U_i \in \mathcal{T}$).*

   We call a pair $(\Omega, \mathcal{T})$ as a topological space. An element $U \in \mathcal{T}$ is called an open set.
3. *For a collection $\mathcal{G} \subset 2^\Omega$, define $\mathcal{F}_\mathcal{G}$ as the collection of all possible countable unions, intersections, and complements of elements in $\mathcal{G}$. It is clearly the σ-algebra by construction and call it as a σ-algebra generated by $\mathcal{G}$.*
4. *Given a topological space $(\Omega, \mathcal{T})$, call $\mathcal{F}_\mathcal{T}$ as a Borel σ-algebra. We call $(\Omega, \mathcal{T}, \mathcal{F}_\mathcal{T})$ as a Borel space. An element of the Borel σ-algebra is called a Borel set.*

Measurable sets in $\mathcal{F}$ are the sets that can be assigned a "size" in a consistent way. Open sets in $\mathcal{T}$ are the sets that can be used to define the "nearness" of points in Ω. Given any collection of subsets, we can find the smallest σ-algebra containing them and call the σ-algebra generated by them. We can always make a topological space into measurable space by the Borel σ-algebra. A Borel space is a measurable space in which every open set is measurable.
Borel space we will use is the Euclidean space $\mathbb{R}$. $\mathbb{R}$ has the standard metric defined by $d(x, y) := |x - y|$ and the metric induces a canonical metric topology. With respect to the metric topology $\mathcal{T}_d$, $(\mathbb{R}, \mathcal{T}_d, \mathcal{F}_{\mathcal{T}_d})$ becomes a Borel space. We simply denote $(\mathbb{R}, \mathcal{T}_d, \mathcal{F}_{\mathcal{T}_d})$ as $\mathbb{R}$.

Definition A.2. *(Probability space)*

1. *Let $(\Omega, \mathcal{F})$ be a measurable space. A measure μ is a function from $\mathcal{F}$ to $\mathbb{R} \cup \{\infty\}$ satisfying*
   (a) *$\mu(\varnothing) = 0$*
   (b) *$\mu(A) \geq 0$ for any $A \in \mathcal{F}$*
   (c) *For any pairwise disjoint countable collection $\{B_i\}_{i\in\mathbb{N}} \subset \mathcal{F}$, $\mu(\biguplus_i B_i) = \sum_i \mu(B_i)$.*

   We call a triple $(\Omega, \mathcal{F}, \mu)$ as a measure space.
2. *We call a measure space $(\Omega, \mathcal{F}, \mathbb{P})$ as a probability space if $\mathbb{P}(\Omega) = 1$. In this case, Ω is called a sample space and its element is called an outcome. $\mathcal{F}$ is called an event space and its element is called an event.*

The intuition is that we want to consider Ω as the set of all possible outcomes. Then, $\mathcal{F}$ is the set of events, and $\mathbb{P}$ measures the probability of an event.

Lemma A.1. *A triple $(V, 2^V, \mathbb{P})$ associated to the graph $G = (V, E)$ is a probability space.*

Proof. Since 2^V is closed under arbitrary union, intersection, and complement, it is σ-algebra. Hence it suffices to show that $\mathbb{P}$ is a probability measure on the measurable space $(V, 2^V)$.

1. $\mathbb{P}(\varnothing) = 0$ by the definition of degree.

2. $\mathbb{P}(A) \geq 0$ for any $A \in 2^V$ since the degree is non-negative function.

3. For any pairwise disjoint countable collection $\{B_i\}_{i\in\mathbb{N}} \subset 2^V$,

$$\mathbb{P}\left(\bigcup_i^{\cdot} B_i\right) = \frac{\sum_{v\in\bigcup_i B_i} d(v)}{\sum_{w\in V} d(w)} = \frac{\sum_i \sum_{v\in B_i} d(v)}{\sum_{w\in V} d(w)} = \sum_i \left(\frac{\sum_{v\in B_i} d(v)}{\sum_{w\in V} d(w)}\right) = \sum_i \mathbb{P}(B_i). \tag{10}$$

4. $\mathbb{P}(V) = \frac{\sum_{v\in V} d(v)}{\sum_{w\in V} d(w)} = 1$.

Therefore, $\mathbb{P}$ is a probability measure on $(V, 2^V)$. □

Definition A.3. *(Random variable, σ-algebra generated by random variable, independence of two random variables)*

1. *A random variable $X : (\Omega, \mathcal{F}, \mathbb{P}) \to \mathbb{R}$ is a function such that*

$$X^{-1}(A) := \{p \in V \mid X(p) \in A\} \in \mathcal{F} \text{ for any Borel set } A \text{ in } \mathbb{R}. \tag{11}$$

2. *A σ-algebra $\mathcal{A}_X$ generated by the random variable X is the σ-algebra generated by $\{X^{-1}(p)\}_{p\in\mathbb{R}}$.*

3. *A random variable X is called discrete if $X(\Omega)$ is at most countable.*

4. *Two random variables $X, Y : (\Omega, \mathcal{F}, \mathbb{P}) \to \mathbb{R}$ are called independent if*

$$\mathbb{P}\left(X^{-1}(A) \cap Y^{-1}(B)\right) = \mathbb{P}\left(X^{-1}(A)\right) \cdot \mathbb{P}\left(Y^{-1}(B)\right) \tag{12}$$

   for any Borel sets A, B in $\mathbb{R}$.

5. *Two sub σ-algebras $\mathcal{A}, \mathcal{B} \subset \mathcal{F}$ are called independent if*

$$\mathbb{P}(A \cap B) = \mathbb{P}(A) \cdot \mathbb{P}(B) \tag{13}$$

   for any $A \in \mathcal{A}, B \in \mathcal{B}$.

A random variable assigns a value to each outcome of an experiment. We can compute the probability of an experiment with values in A by $\mathbb{P}\left(X^{-1}(A)\right)$ for any Borel set $A \subset \mathbb{R}$. The σ-algebra generated by the random variable is the σ-algebra generated by all preimages of X. Independence of X and Y means that the two experiments X and Y are not correlated at all. We can check that independence of X, Y and independence of $\mathcal{A}_X, \mathcal{A}_Y$ are equivalent.

Lemma A.2. *Suppose $(V, 2^V, \mathbb{P})$ is a probability space associated to the graph $G = (V, E)$. Then any function $X : V \to \mathbb{R}$ is a random variable.*

Proof. $X^{-1}(A) = \{p \in V \mid X(p) \in A\}$ is a subset of V for any $A \subset \mathbb{R}$, so $X^{-1}(A) \in 2^V$ for any $A \subset \mathbb{R}$. Hence $X^{-1}(A) \in 2^V$ for any Borel set $A \subset \mathbb{R}$. □

Next, we will define a quantity called mutual information to measure the degree to which two random variables are dependent. To define mutual information, we first need to define entropy.

Definition A.4. *(Entropy and joint entropy)*

1. *Let $X : (\Omega, \mathcal{F}, \mathbb{P}) \to \mathbb{R}$ be a discrete random variable with the image $\{p_i\}_{i=1,\dots,m}$. The entropy of X, $H(X)$, is defined by*

$$H(X) := -\sum_{i=1,\dots,m} \mathbb{P}\left(X^{-1}(p_i)\right) \cdot \log\left(\mathbb{P}\left(X^{-1}(p_i)\right)\right). \tag{14}$$

2. *Let $X, Y : (\Omega, \mathcal{F}, \mathbb{P}) \to \mathbb{R}$ be two discrete random variables with the images $\{p_i\}_{i=1,\dots,m}, \{q_j\}_{j=1,\dots,n}$, respectively. The joint entropy of X, Y, $H(X, Y)$, is defined by*

$$H(X, Y) := -\sum_{i,j} \mathbb{P}\left(X^{-1}(p_i) \cap Y^{-1}(q_j)\right) \cdot \log\left(\mathbb{P}\left(X^{-1}(p_i) \cap Y^{-1}(q_j)\right)\right). \tag{15}$$

The entropy $H(X)$ characterizes the distribution $\{\mathbb{P}\left(X^{-1}(p_i)\right)\}_{i=1,\dots,m}$. Similarly, the joint entropy $H(X,Y)$ characterizes the distribution $\{\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)\}_{i,j}$.

Definition A.5. *(Mutual information of two discrete random variables) Let X,Y be two discrete random variables X,Y with the images $\{p_i\}_{i=1,\dots,m}, \{q_j\}_{j=1,\dots,n}$, respectively.*

1. *A mutual information of X,Y, $I(X,Y)$, is defined by*

$$I(X,Y) := H(X) + H(Y) - H(X,Y) \tag{16}$$

$$= -\sum_{i,j}\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)\cdot\log\left(\frac{\mathbb{P}\left(X^{-1}(p_i)\right)\cdot\mathbb{P}\left(Y^{-1}(q_j)\right)}{\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)}\right). \tag{17}$$

$$\tag{18}$$

   If $\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right) = 0$, then set

$$\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)\cdot\log\left(\frac{\mathbb{P}\left(X^{-1}(p_i)\right)\cdot\mathbb{P}\left(Y^{-1}(q_j)\right)}{\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)}\right) := 0. \tag{19}$$

2. *A distance of two random variables induced by the mutual information, $D(X,Y)$, is defined by*

$$D(X,Y) := 1 - \frac{I(X,Y)}{H(X,Y)} \tag{20}$$

3. *A normalized mutual information of X,Y, $I'(X,Y)$, is defined by*

$$I'(X,Y) := 1 - D(X,Y) = \frac{H(X)+H(Y)-H(X,Y)}{H(X,Y)} \tag{21}$$

Mutual information of X,Y quantifies the failure of independence of X,Y. If two random variables are independent, $I(X,Y) = 0$ by its definition. On the other hand, if $X = Y$ then $I(X,Y) = H(X)$.

Lemma A.3. *A mutual information $I(X,Y)$ is non-negative.*

Proof. Since $-\log x \geq 1 - x$ for $0 < x < 1$,

$$I(X,Y) = -\sum_{i,j}\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)\cdot\log\left(\frac{\mathbb{P}\left(X^{-1}(p_i)\right)\cdot\mathbb{P}\left(Y^{-1}(q_j)\right)}{\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)}\right) \tag{22}$$

$$\geq \sum_{i,j}\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)\cdot\left(1 - \frac{\mathbb{P}\left(X^{-1}(p_i)\right)\cdot\mathbb{P}\left(Y^{-1}(q_j)\right)}{\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)}\right) \tag{23}$$

$$= \sum_{i,j}\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right) - \sum_{i,j\,\&\,\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right)\neq 0}\mathbb{P}\left(X^{-1}(p_i)\right)\cdot\mathbb{P}\left(Y^{-1}(q_j)\right) \tag{24}$$

$$\geq \sum_{i,j}\mathbb{P}\left(X^{-1}(p_i)\cap Y^{-1}(q_j)\right) - \sum_{i,j}\mathbb{P}\left(X^{-1}(p_i)\right)\cdot\mathbb{P}\left(Y^{-1}(q_j)\right) \tag{25}$$

$$= 1 - \sum_{i}\left(\mathbb{P}\left(X^{-1}(p_i)\right)\right)\cdot\sum_{j}\left(\mathbb{P}\left(Y^{-1}(q_j)\right)\right) = 1 - 1\cdot 1 = 0. \tag{26}$$

□

Similar argument shows that $I(X,Y) \leq H(X,Y)$, so $D(X,Y), I'(X,Y) \in [0,1]$. In particular, $D(X,X) = 0$ and $D(X,Y) = 1$ for independent random variables X,Y.

B Hyeperparameter settings

All experiments were conducted on an NVIDIA GeForce RTX 3090 GPU. This GPU has 10,496 CUDA cores with 24GB of memory and a GPU clock speed of 3.1 GHz. The Table 5, Table 6 and Table 7 provide the details of the hyperparameters used in all the experiments.

	Dataset	squirrel	chameleon	roman-emprie	amazon-ratings	minesweeper	tolokers	questions
0-MIGNN	num layer	5	5	6	5	6	2	6
	hidden dimension	256	512	256	512	256	64	256
	dropout	0.8	0.2	0.4	0.2	0.4	0.6	0.4
	weight decay	0	0	0	0	0	5.00E-06	0
	learning rate	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	0.01	3.00E-05
	result	41.73 ± 2.58	41.91 ± 3.98	86.92 ± 0.57	48.86 ± 0.48	84.13 ± 0.57	80.79 ± 0.82	73.10 ± 0.92
1-MIGNN	num layer	5	5	6	5	6	2	6
	hidden dimension	256	512	256	512	256	64	256
	dropout	0.8	0.2	0.4	0.2	0.4	0.6	0.4
	weight decay	0	0	0	0	0	5.00E-06	0
	learning rate	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	0.01	3.00E-05
	result	39.70 ± 1.76	42.83 ± 4.04	91.53 ± 0.47	49.25 ± 0.66	90.59 ± 0.64	82.53 ± 1.12	76.46 ± 1.24
2-MIGNN	num layer	5	5	6	5	6	2	6
	hidden dimension	256	512	256	512	256	64	256
	dropout	0.8	0.2	0.4	0.2	0.4	0.6	0.4
	weight decay	0	0	0	0	0	5.00E-06	0
	learning rate	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	0.01	3.00E-05
	result	40.70 ± 1.69	44.05 ± 4.21	91.91 ± 0.40	48.92 ± 0.59	91.63 ± 0.67	82.27 ± 1.06	75.97 ± 1.26

Table 5: Hyperparameter for each dataset.

	k	0	1	2	3	4	5	6	7
roman-emprie	num layer	6	6	6	6	6	6	6	6
	hidden dimension	256	256	256	256	256	256	256	256
	dropout	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
	weight decay	0	0	0	0	0	0	0	0
	learning rate	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05
	result	86.92 ± 0.57	91.53 ± 0.47	91.91 ± 0.40	91.97 ± 0.36	92.04 ± 0.37	92.12 ± 0.36	92.16 ± 0.37	92.25 ± 0.33
minsweeper	num layer	6	6	6	6	6	6	6	6
	hidden dimension	256	256	256	256	256	256	256	256
	dropout	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
	weight decay	0	0	0	0	0	0	0	0
	learning rate	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05
	result	84.13 ± 0.57	90.59 ± 0.64	91.63 ± 0.67	91.98 ± 0.67	92.12 ± 0.60	92.30 ± 0.54	92.29 ± 0.54	92.37 ± 0.53

Table 6: Hyperparameter for k.

		num layer	2	3	4	5	6
roman-empire	GCN	hidden dimension	512	512	512	512	512
		dropout	0.2	0.2	0.2	0.2	0.2
		weight decay	0	0	0	0	0
		learning rate	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05
		result	78.61 ± 0.46	78.38 ± 0.35	77.74 ± 0.52	76.94 ± 0.74	76.81 ± 0.33
	1-MIGNN	hidden dimension	256	256	256	256	256
		dropout	0.4	0.4	0.4	0.4	0.4
		weight decay	0	0	0	0	0
		learning rate	3.00E-05	3.00E-05	3.00E-05	3.00E-05	3.00E-05
		result	88.72 ± 0.56	89.96 ± 0.59	90.90 ± 0.53	91.21 ± 0.42	91.53 ± 0.47

Table 7: Hyperparameter for layer