Multi-label Node Classification On Graph-Structured Data

Anonymous authors
Paper under double-blind review

Abstract

Graph Neural Networks (GNNs) have shown state-of-the-art improvements in node classification tasks on graphs. While these improvements have been largely demonstrated in a multi-class classification scenario, a more general and realistic scenario in which each node could have multiple labels has so far received little attention. The first challenge in conducting focused studies on multi-label node classification is the limited number of publicly available multi-label graph datasets. Therefore, as our first contribution, we collect and release three real-world biological datasets and develop a multi-label graph generator to generate datasets with tunable properties. While high label similarity (high homophily) is usually attributed to the success of GNNs, we argue that a multi-label scenario does not follow the usual semantics of homophily and heterophily so far defined for a multi-class scenario. As our second contribution, besides defining homophily for the multi-label scenario, we develop a new approach that dynamically fuses the feature and label correlation information to learn label-informed representations. Finally, we perform a large-scale comparative study with 10 methods and 9 datasets which also showcase the effectiveness of our approach. We release our benchmark at `https://anonymous.4open.science/r/LFLF-5D8C/`.

1 Introduction

Most of the existing works on graph node classification deploying Graph Neural Networks (GNNs) focus on a multi-class classification scenario while ignoring a more general and realistic scenario of multi-label classification, in which each node could have multiple labels. This scenario holds, for example in protein-protein interaction networks, in which each protein is labeled with multiple protein functions or associated with different diseases, or in social networks, where each user may carry multiple interest labels. In this work, we focus on multi-label node classification on graph-structured data with graph neural networks. For the sake of brevity, in what follows we will refer to multi-label node classification on graphs simply as multi-label node classification.

Regarding the approaches to deploying GNNs in a multi-label node classification scenario, a common practice is to transform the classification problem into multiple binary classification problems, one per label (Zhang & Zhou, 2013). In other words, $|L|$ binary classifiers are trained, where each classifier j is responsible for predicting the 0/1 association for the corresponding label $\ell_j \in L$. An assumption here, however, is that given the learned feature representations of the nodes, the labels are conditionally independent (Ma et al., 2020). The validity of this assumption cannot be assured in a GNN-based learning approach as GNNs ignore the label correlation among the neighboring nodes and only focus on node feature aggregation during the representation learning step (Jia & Benson, 2020; Ma et al., 2020).

Furthermore, we note that the success of GNNs is widely attributed to feature smoothing over neighborhoods and high label similarity among the neighboring nodes. Graphs with high similarity among labels of neighboring nodes are referred to as having high homophily. Alternatively, in heterophilic graphs, the labels of neighboring nodes usually disagree. Consequently, approaches like H2Gcn (Zhu et al., 2020) have been proposed, which claim a high performance on both homophilic and heterophilic node classification datasets. A subtle point here, however, is that a network with nodes characterized by multiple labels does not obey the crisp separation of homophilic and heterophilic characteristics. As an illustration, consider a friendship network with node labels representing user interests. Each user might share only a very small fraction

of the interests with his friends, which indicates low homophily in the local neighborhood. Yet her/his interests/labels could be fully determined by looking at his one-hop neighbors. Therefore, the network is also not heterophilic in the sense that the connected users have similar interests. Consequently, the solutions taking into account higher-order neighborhoods to tackle low homophily might not always perform well in multi-label networks.

The lack of focused studies on multi-label node classification can also be attributed to the scarcity of available benchmark multi-label graph datasets. As an example, there is a single multi-label classification dataset, namely OGB-Proteins in the Open Graph Benchmark (OGB) (Hu et al., 2020). More so, the OGB-Proteins dataset has around 90% of the nodes unlabeled in the test set. While the OGB leaderboard reflects benchmarking of a large number of methods on OGB-Proteins, the lack of labels in the test set combined with the use of the Area Under the ROC Curve (AUROC) metric leads to overly exaggerated performance scores. In particular, the performance is measured by the average of AUROC scores across the L (L is the total number of labels) binary classification tasks. In such a scenario, the model already achieves a very high score if it outputs a high probability for the negative class for each binary classification task.

Our Contributions. Our work thoroughly investigates the problem of multi-label node classification with GNNs. **Firstly**, we analyze various characteristics of multi-label graph-structured datasets, including label distribution, label induced first- and second-order similarities, that influence the performance of the prediction models. We observe that a large number of nodes in the current datasets only have a single label even if the average number of labels per node is relatively high. Moreover, for the popular OGB-Proteins dataset, around 89.38% of the nodes in the test set and 29.12% of train nodes have no label assigned.

Secondly, to remedy the gap of lack of datasets we build a benchmark of multi-label graph-structured datasets with varying structural properties and label-induced similarities. In particular, we curate 3 biological graph datasets using publicly available data. Besides, we develop a synthetic multi-label graph generator with tunable properties. The possibility to tune certain characteristics allows us to compare various learning methods rigorously.

Thirdly, inspired by the effective structure exploitation strategy of label propagation techniques we develop a simple yet novel approach that dynamically merges the feature and label representations of the nodes and fully utilizes the label correlation information of neighboring nodes. We show that our approach can not only perform well on graphs exhibiting different levels of label similarity but can also effectively exploit the label correlation information to mitigate the effects of bad feature quality.

Finally, we perform a large-scale experimental study evaluating 10 methods over 9 datasets. We observe that simple baselines like DeepWalk sometimes outperform more sophisticated GNNs under some dataset properties. Besides, we show the superior performance of our developed approach on a number of real-world and synthetic graph datasets.

2 Background and Related Work

2.1 Notations and The Problem Setting.

Notations. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph where $\mathcal{V} = \{v_1, \cdots, v_n\}$ is the set of vertices, $\mathcal{E}$ represents the set of links/edges among the vertices. We further denote the adjacency matrix of the graph by $\mathbf{A} \in \{0,1\}^{n \times n}$ and $a_{i,j}$ denotes whether there is an edge between v_i and v_j. $\mathcal{N}(v)$ represents the immediate neighbors of node v in the graph. Furthermore, let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \in \mathbb{R}^{n \times D}$ and $\mathbf{Y} = \{\mathbf{y}_1, \cdots, \mathbf{y}_n\} \in \{0,1\}^{n \times C}$ represent the feature and label matrices corresponding to the nodes in $\mathcal{V}$. In the feature matrix and label matrix, the i-th row represents the feature/label vector of node i. Let $\ell(i)$ denote the set of labels that are assigned to node i. Finally, let $\mathcal{F}$ correspond to the feature set and $\mathcal{L}$ be the set of all labels.

Problem Setting. In this work, we focus on multi-label node classification problems on graph-structured data. In particular, we are given a set of labeled and unlabelled nodes such that each node can have more than one label. We are then interested in predicting labels of unlabelled nodes. We assume that the training nodes are completely labeled. We deal with the transductive setting multi-label node classification problem, where the features and graph structure of the test nodes are present during training.

2.2 Related Work

Multi-label classification, which assigns multiple labels for each instance simultaneously, finds applications in multiple domains ranging from text classification to protein function prediction. In this work we focus on the case when the input data is graph-structured, for example, a protein-protein interaction network or a social network. For completeness, we also discuss the related works on using graph neural networks for multi-label classification on non-graph-structured data in Section 2.2.2. Several other paradigms of multi-label classification including extreme multi-label classification (Liu et al., 2017; Song et al., 2022; Wang et al., 2019), partial multi-label classification (Huynh & Elhamifar, 2020; Jain et al., 2017), multi-label classification with weak supervision (Chu et al., 2018; Hu et al., 2019) (for a complete overview see the recent survey (Liu et al., 2021)). are out of the scope of the current work.

2.2.1 Multi-label Classification On Graph-structured Data

Recent methods designed for multi-label node classification over graph-structured data can be categorized into four groups utilizing (1) node embedding approaches, (2) convolutional neural networks, (3) graph neural networks, and (4) the combination of label propagation and graph neural networks.

Node representation or embedding approaches (Perozzi et al., 2014; Khosla et al., 2019; Ou et al., 2016) usually generate a lookup table of representations such that similar nodes are embedded closer. The learned representations are used as input features for various downstream prediction modules. While different notions of similarities are explored by different approaches a prominent class of method is random walk based which defines similarity among nodes by their co-occurrence frequency in random walks. In this work, we specifically use DeepWalk (Perozzi et al., 2014) as a simple baseline that uses uniform random walks to define node similarity.

Other methods like (Shi et al., 2019; Zhou et al., 2021; Song et al., 2021) use **convolutional neural networks** to first extract node representations by aggregating feature information from its local neighborhood. The extracted feature vectors are then fused with label embeddings to generate final node embeddings. Finally, these node embeddings are used as input for the classification model to generate node labels. In this work, we adopt LANC (Zhou et al., 2021) as a baseline from this category, as previous works (Zhou et al., 2021; Song et al., 2021) have shown its superior performance compared to other commonly used baselines for the multi-label node classification task.

Graph neural networks(GNNs) popularised by graph convolution network (Kipf & Welling, 2016) and its variants compute node representation by recursive aggregation and transformation of feature representations of its neighbors which are then passed to a classification module. Let $\mathbf{x}_i^{(k)}$ be the feature representation of node i at layer k, $\mathcal{N}(i)$ denote the set of its 1-hop neighbors. The $k-th$ layer of a graph convolutional operation can then be described as

$$\mathbf{z}_i^{(k)} = \text{AGGREGATE}\left(\left\{\mathbf{x}_i^{(k-1)}, \left\{\mathbf{x}_j^{(k-1)} \mid j \in \mathcal{N}(i)\right\}\right\}\right), \quad \mathbf{x}_i^{(k)} = \text{TRANSFORM}\left(\mathbf{z}_i^{(k)}\right)$$

For the multi-label node classification, a sigmoid layer is employed as the last layer to predict the class probabilities: $\boldsymbol{y} \leftarrow (\text{sigmoid}(\boldsymbol{z}_i^{(L)}\theta))$, where θ corresponds to the learnable weight matrix in the classification module. GNN models mainly differ in the implementation of the aggregation layer. The simplest model is the graph convolution network (GCN) (Kipf & Welling, 2016) which employs degree-weighted aggregation over neighborhood features. Gat (Veličković et al., 2018) employs several stacked Graph Attention Layers, which allows nodes to attend over their neighborhoods' features. GraphSage (Hamilton et al., 2017) follows a sample and aggregate approach in which only a random sample of the neighborhood is used for the feature aggregation step. GNNs, in general, show better performance on high homophilic graphs in which the connected nodes tend to share the same labels. Recent approaches like H2Gcn (Zhu et al., 2020) show improvement on heterophilic graphs (in the multi-class setting). Specifically, it separates the information aggregated from the neighborhood from that of the ego node. Further, it utilizes higher-order neighborhood information to learn informative node representations.

Label propagation with GNNs. Prior to the advent of GNNs label propagation (LPA) algorithms constituted popular approaches for the task of node classification. Both LPA and GNNs are based on message passing. While GNNs propagate and transform node features, LPA propagates node label information along the edges of the graph to predict the label distribution of the unlabelled nodes. A few recent works (Yang et al., 2021; Wang & Leskovec, 2020) have explored the possibilities of combining LPA and GNNs. (Yang et al., 2021) employs knowledge distillation in which the trained GNN model (teacher model) is distilled into a combination of GNN and parameterized label propagation module. GCN-LPA (Wang & Leskovec, 2020) utilizes LPA serves as regularization to assist the GCN in learning proper edge weights that lead to improved classification performance. Different from our work both of the above works implicitly assume a multi-class setting. Moreover, (Yang et al., 2021) focuses on the interpretable extraction of knowledge from trained GNN and can only discover the patterns learned by the teacher GNN.

2.2.2 Multi-label Classification On Non-Graph-Structured Data Using GNNs.

There has also been an increasing trend to use graph neural networks to exploit the implicit relations between the data and labels in multi-label classification problems on non-graph data. For example, (Saini et al., 2021) models the problem of extreme classification as that of link prediction in a document-label bipartite graph and uses graph neural networks together with the attention mechanism to learn superior node representations by performing graph convolutions over neighborhoods of various orders. ML-GCN(Shi et al., 2019) generates representations for the images using CNN and extracts label correlation by constructing a label-label graph from the label co-occurrence matrix. LaMP(Lanchantin et al., 2019) treats labels as nodes on a label-interaction graph and computes the hidden representation of each label node conditioned on the input using attention-based neural message passing. Likewise, for the task of multi-label image recognition (Chen et al., 2019) builds a directed graph over the object labels, where each label node is represented by word embeddings of the label, and GCN is employed to map this label graph into a set of inter-dependent object classifiers.

Our work is different and complementary to the above works and many more like (Li et al., 2021; Zong & Sun, 2020; Ma et al., 2021a; Zheng et al., 2022; Shi et al., 2020; Xu et al., 2021; Cheng et al., 2021; Pal et al., 2020), in that, we focus on the issues and their resolution of using graph neural networks when the input data is graph-structured when each node can have multiple labels. Our work is of a more fundamental nature in which we focus on enhancing graph neural networks for predicting multiple labels of a node. Our proposed approach can be used in combination with specific techniques proposed in these papers for domain-specific applications.

3 A detailed analysis of existing and new datasets

We commence by analyzing various properties of existing multi-label datasets including label distributions, label similarities, and cross-class neighborhood similarity(CCNS), which could affect the performance of prediction models. In section 3.2 we further curate new real-world biological datasets which to some extent improve the representativeness of multi-label graph datasets. Finally, in Section 4 we propose our synthetic multi-label graph generator to generate multi-label graph datasets with tunable properties. The possibility to control specific influencing properties of the dataset allows us to benchmark various learning methods effectively. We will need the following quantification of label similarity in multi-label datasets.

Label homophily. The performance of GNNs is usually argued in terms of label homophily which quantifies similarity among the neighboring nodes in the graph. In particular, label homophily is defined in (Zhu et al., 2020) as the fraction of the homophilic edges in the graph, where an edge is considered homophilic, if it connects two nodes with the same label. This definition can not be directly used in the multi-label graph datasets, as each node can have more than one label and it is rare in the multi-label datasets that the whole label sets of two connected nodes are the same. Usually, two nodes share a part of their labels. We, therefore, propose a new metric to measure the label homophily h of the multi-label graph datasets as follows.

Definition 1 *Given a multi-label graph $\mathcal{G}$, the label homophily h of $\mathcal{G}$ is defined as the average of the Jaccard similarity of the label set of all connected nodes in the graph:*

$$h = \frac{1}{|\mathcal{E}|} \sum_{(i,j)\in\mathcal{E}} \frac{|\ell(i) \cap \ell(j)|}{|\ell(i) \cup \ell(j)|}.$$

Label homophily is a first-order label-induced similarity in that it quantifies the similarity among neighboring nodes based on their label distributions.

Cross-Class Neighborhood Similarity for Multi-label graphs. Going beyond the label similarity among neighboring nodes, we consider a second order label induced metric which quantifies the similarity among neighborhoods of any two nodes. Ma et al. (2021b) introduced Cross-Class Neighborhood Similarity (CCNS) for multi-class graphs. Using CCNS, (Ma et al., 2021b) attributes the improved performance of GNNs to the higher similarity among neighborhood label distributions of nodes of the same class as compared to different classes. We extend their proposed CCNS measure to analyze multi-label datasets. Given two classes c and c', the CCNS score measures the similarity in the label distributions of neighborhoods of nodes belonging to c and c' and is defined as follows. One can visualize the CCNS scores between all class pairs in an $C \times C$ matrix where C is the total number of classes. Common GNNs are expected to perform better on datasets which higher scores on the diagonal than off-diagonal elements of the corresponding CCNS matrix.

Definition 2 *Given a multi-label graph $\mathcal{G}$ and the set of node labels Y for all nodes, we define the multi-label cross-class neighborhood similarity between classes $c, c' \in C$ is given by*

$$s(c, c') = \frac{1}{|\mathcal{V}_c||\mathcal{V}_{c'}|} \sum_{i\in\mathcal{V}_c, j\in\mathcal{V}_{c'}, i\neq j} \frac{1}{|\ell(i)||\ell(j)|} \cos(\mathbf{d}_i, \mathbf{d}_j), \tag{1}$$

where $\mathcal{V}_c = \{i | c \in \ell(i)\}$ is the set of nodes with one of their labels as c. The vector $\mathbf{d}_i \in \mathbb{R}^C$ corresponds to the empirical histogram (over $|C|$ classes) of node i's neighbors' labels, i.e., the cth entry of $\mathbf{d}_i$ corresponds to the number of nodes in $\mathcal{N}(i)$ that has one of their label as c and the function $cos(.,.)$ measures the cosine similarity and is defined as

$$cos(\mathbf{d}_i, \mathbf{d}_j) = \frac{\mathbf{d}_i \cdot \mathbf{d}_i}{||\mathbf{d}_i||||\mathbf{d}_j||}.$$

Note that as a node in a multi-label dataset could belong to multiple classes we exclude the possibility of comparing a node to itself. By introducing a factor of $|\ell(i)||\ell(j)|$ in the denominator we are able to normalize the contribution of multi-labeled nodes for several class pairs.

3.1 Existing datasets

We start by analyzing the four popular multi-label node classification datasets: (i) BlogCat (Tang & Liu, 2009), in which nodes represent bloggers and edges their relationships, the labels denote the social groups a blogger is a part of, (ii)Yelp (Zeng et al., 2019), in which nodes correspond to the customer reviews and edges to their friendships with node labels representing the types of businesses and (iii) OGB-Proteins (Hu et al., 2020), in which nodes represent proteins, and edges indicate different types of biologically meaningful associations between the proteins, such as physical interactions, co-expression, or homology (Consortium, 2018; D et al., 2019). The labels correspond to protein functions. (iiii) DBLP (Akujuobi et al., 2019), in which nodes represent authors and edges the co-authorship between the authors, and the labels indicate the research areas of the authors.

We report in Table 1 the characteristics of these datasets including the label homophily. In the following, we discuss in detail the various characteristics of these datasets along with their limitations for the effective evaluation of multi-label classification.

Skewed label distributions. Figure 1 illustrates the label distributions in the four datasets. Quantitatively 72.34% nodes in BlogCat only have one label. However, the most labeled data points are assigned with 11

Table 1: Dataset statistics. $|\mathcal{V}|$ and $|\mathcal{E}|$ denote the number of nodes and edges in the graph. $|\mathcal{F}|$ is the dimension of the node features. $clus$ and r_{homo} denote the clustering coefficient and the label homophily. C indicates the size of all labels in the graph. ℓ_{med}, ℓ_{mean}, and ℓ_{max} specify the median, mean, and max values corresponding to the number of labels of a node. '25%', '50%', and '75%' corresponds to the 25th, 50th, and 75th percentiles of the sorted list of the number of labels for a node. "N.A." means the corresponding characteristic is not available in the graph.

Dataset	$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	$\|\mathcal{F}\|$	$clus$	r_{homo}	C	ℓ_{med}	ℓ_{mean}	ℓ_{max}	25%	50%	75%
BlogCat	10K	333K	N.A.	0.46	0.10	39	1	1.40	11	1	1	2
Yelp	716K	7.34M	300	0.09	0.22	100	6	9.44	97	3	6	11
OGB-Proteins	132K	39M	8	0.28	0.15	112	5	12.75	100	0	5	20
DBLP	28K	68K	300	0.61	0.76	4	1	1.18	4	1	1	1

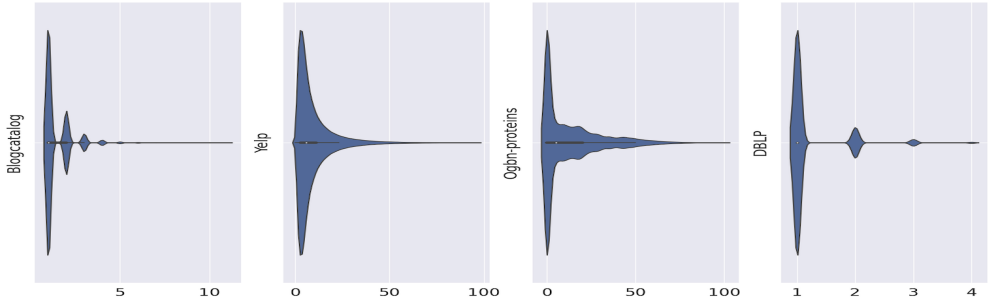


Figure 1: Label distributions. In BlogCat, the majority of the nodes have one label. In OGB-Proteins, around 41% of total nodes have no labels, and only three nodes have the maximum number of 100 labels.

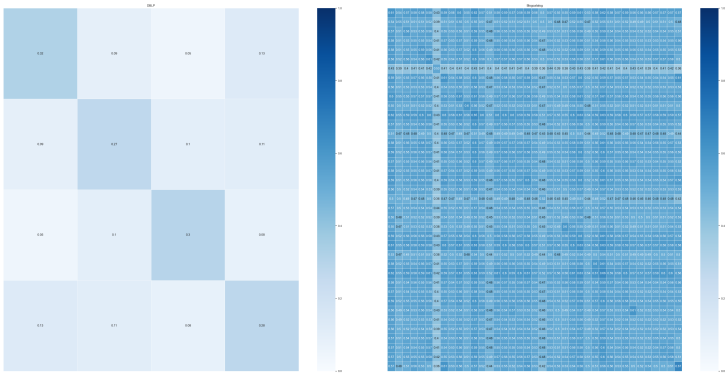
labels. Yelp has a total of 100 labels, the most labeled data points have 97 labels, whereas over 50% of the nodes have equal or less than 5 labels. Nevertheless, Yelp exhibits a high multi-label character with 75% of the nodes with more than 3 labels. OGB-Proteins is an extreme case in which 40.27% of the nodes do not have any label. DBLP is the dataset with the highest portion of nodes with single labels, with the exact percentage of 85.4%.

Issue in evaluation using AUROC scores under high label sparsity. Another so far unreported issue in multi-label datasets is the unlabeled data. In OGB-Proteins, 40.27% of nodes do not have labels. Moreover, 89.4% of the test nodes are unlabelled. More worrying is the use of the AUROC score metric in the OGB leaderboard to benchmark methods for multi-label classification. In particular, a model that assigns "No Label" to each node (i.e. predict negative class corresponding to each of the independent L binary classification tasks) will already show a high AUROC score. We in fact observed that increasing the number of training epochs (which encourage the model to decrease training loss by predicting the negative class) increased the AUROC score whereas other metrics like AP or F1 score dropped or stayed unchanged.

Cross-class neighborhood similarity. In Figure 2 we visualize the cross-class neighborhood similarity matrix for DBLP and BlogCat. The cells on the diagonal reflect the intra-class neighborhood similarity, whereas the other cells indicate the inter-class neighborhood similarity computed using equation 1. The contrast in 2a means that nodes from the same class tend to have similar label distribution in their neighborhood, while nodes from different classes have rather different label distributions in their neighborhoods. We will later see in the experimental section that GNNs indeed benefit from this characteristic to identify correctly the nodes in the same classes in DBLP. On the contrary, the intra- and inter-class similarity are more similar in BlogCat, making it intricate for GNNs to classify the nodes to their corresponding classes.

3.2 New biological interaction datasets

Motivated by the natural applicability of the multi-label classification task in various biological datasets and to improve the representativeness of available datasets, we collect three real-world biological datasets



(a) Cross class Neighborhood Similarity in DBLP

(b) Cross class Neighborhood Similarity in BlogCat

Figure 2: Cross class Neighborhood Similarity in real-world datasets

corresponding to different multi-label classification problems: the PCG dataset for the protein phenotype prediction, the HumLoc, and EukLoc datasets for the human and eukaryote protein subcellular location prediction tasks, respectively. On each dataset, we build a graph in which each protein is modeled as a node. The node label is the corresponding protein's label. An edge represents a known interaction between two proteins retrieved from a public database. The detailed pre-processing steps and the original data sources are discussed in Appendix A.1.1, A.1.2, and A.1.3. Table 2 presents an overview of the three datasets' characteristics.

Table 2: Statistics for new datasets. The column notations are the same as in Table 1.

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{F}\rvert$	$clus$	r_{homo}	C	ℓ_{med}	ℓ_{mean}	ℓ_{max}	25%	50%	75%
PCG	3K	37k	32	0.34	0.17	15	1	1.93	12	1	1	2
HumLoc	3.10k	18K	32	0.13	0.42	14	1	1.19	4	1	1	1
EukLoc	7.70K	13K	32	0.14	0.46	22	1	1.15	4	1	1	1

While all the existing datasets had very low homophily, HumLoc, and EukLoc show higher homophily. Moreover, these datasets improve the representativeness in terms of varying graph structure (reflected in computed clustering coefficient) and in node, edge, and feature sizes. On the downside, these datasets also show a similar low multi-label character, with the majority of nodes in these datasets still having a single label. Among the three datasets, PCG shows a bit more balanced label distribution (see Figure 3) as compared to the other two. Figure 4 provides the cross-class neighborhood similarity scores. All three datasets show different patterns according to CCNS measure which is desirable to analyse the differences in method's performance. While in PCG we see an overall high scores for CCNS, the difference in inter- and intra- class similarities is not prominent. HumLoc shows a slightly more contrasting intra- and inter-class neighborhood similarity. EukLoc, on the other hand, show very small neighborhood label similarities for nodes of same or different classes.

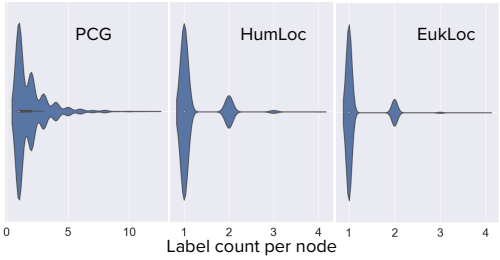
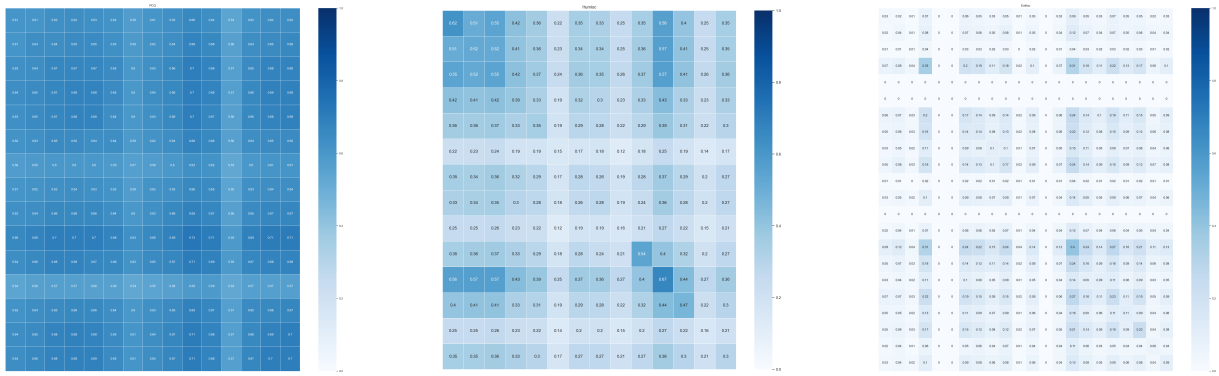


Figure 3: Label distributions in biological datasets. The majority of the nodes in all datasets have one label.



(a) Cross class Neighborhood Similarity in PCG

(b) Cross class Neighborhood Similarity in HumLoc

(c) Cross class Neighborhood Similarity in EukLoc

Figure 4: Cross class Neighborhood Similarity in real-world datasets and proposed biological datasets

4 Multi-label Graph Generator Framework

In the previous sections, we analyzed various real-world dataset properties which could influence a method's performance. We now develop a multi-label graph generator that will allow us to build datasets with tunable properties for a holistic evaluation. With our proposed framework we can build datasets with *high multi-label character, varying feature quality, varying label homophily, and CCNS similarity.* We now describe the two main steps of our multi-label graph generator.

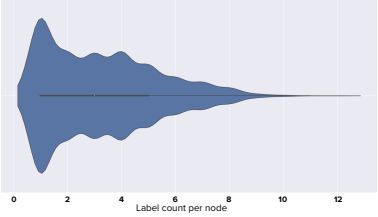
Multi-label generator. In the first step, we generate a multi-label dataset using Mldatagen (Tomás et al., 2014). We start by fixing the total number of labels and features. We then construct a hypersphere, $H \in \mathbb{R}^{|\mathcal{F}|}$ centered at the origin and has a unit radius. Corresponding to each label in set $\mathcal{L}$ we then generate a smaller hypersphere with a random radius but with the condition that it is contained in H. We now start populating the smaller hyperspheres with randomly generated datapoints with $|\mathcal{F}|$ dimensions. Note that each datapoint may lie in a number of overlapping hyperspheres. The labels of the datapoint then correspond to the hyperspheres it lies in.

Graph generator. Having constructed the multi-label dataset, we now construct edges between the data points by using a social distance attachment model (Boguná et al., 2004). In particular, for two given datapoints (nodes) i and j, their corresponding feature vectors are their coordinates given $\mathbf{x}_i$ and $\mathbf{x}_j$ respectively. The corresponding label vectors are denoted by $\mathbf{y}_i$ and $\mathbf{y}_j$. We denote the hamming distance between the label vectors of nodes i and j by $d(\mathbf{y}_i, \mathbf{y}_j)$. We then construct an edge between datapoints (nodes) i and j, (i, j) with probability given by

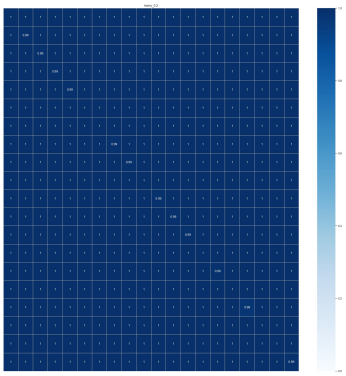
$$p_{ij} = \frac{1}{1 + [b^{-1} d(\mathbf{y}_i, \mathbf{y}_j)]^{\alpha}} \tag{2}$$

where α is a homophily parameter, b is the characteristic distance at which $p_{ij} = \frac{1}{2}$. Note that the edge density is dictated by both the parameters α and b. A larger b would result in denser graphs. A larger homophily parameter α would assign a higher connection probability to the node pairs with shorter distances or nodes with similar labels. In particular, it is a random geometric graph model, which in the limit of large system size (number of nodes) and high homophily (large α) leads to sparsity, non-trivial clustering coefficient and positive degree assortativity (Talaga & Nowak, 2020), properties exhibited by real-world networks. By using different combinations of values of α and b, we can control the connection probability and further the label homophily of the generated synthetic graphs. We perform an extensive empirical analysis to study the relationship between α, b, and the label homophily of the generated datasets. We provide detailed instructions to use our synthetic data generator and our empirical analysis in Appendix A.2.

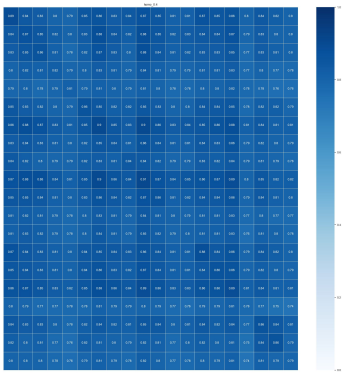
Synthetic datasets with fixed homophily and varying feature quality. For our experimental analysis, we generate synthetic datasets with $3K$ nodes, 10 features, and a total of 20 labels. Towards analyzing the



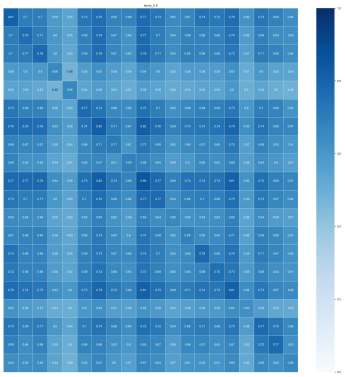
(a) Label distribution in the synthetic dataset is more balanced. A relatively high multi-label character is exhibited with 50% nodes having more than 3 labels.



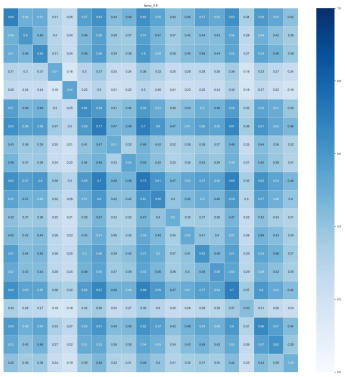
(b) Cross-class Neighborhood Similarity in graph label homophily=0.2



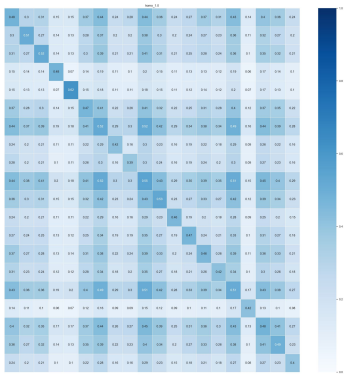
(c) Cross-class Neighborhood Similarity in graph label homophily=0.4



(d) Cross-class Neighborhood Similarity in graph label homophily=0.6



(e) Cross-class Neighborhood Similarity in graph label homophily=0.8



(f) Cross-class Neighborhood Similarity in graph label homophily=1.0

Figure 5: Cross-class Neighborhood Similarity in hypersphere datasets with varying label homophily

variation in the method's performance with variation in *homophily* and *feature quality* first we constructed a dataset with fixed homophily of 0.377 (using $\alpha = 8.8$, $b = 0.12$) and edge set of size $1M$. We refer to this dataset as SYNTHETIC1. We create five variants of SYNTHETIC1 with varying feature quality. In particular, we add 10 random features for every node, which we refer to as *irrelevant* features. We then generate its variants by removing original features such that the ratio of the number of original to that of irrelevant features varies as in $\{1, 0.8, 0.5, 0.2, 0\}$. From the label distribution plot in 5a, we observe the dataset is more multi-label than the real-world datasets because a higher number of nodes now have multiple labels.

Synthetic datasets with varying homophily and CCNS. We also use 5 different pairs of α and b and the same multi-label data from the first step to construct 5 synthetic graphs with label homophily (rounded up) in $\{0.2, 0.4, 0.6, 0.8, 1.0\}$ to conduct the experiment where we test the influence of the label homophily on the performances of the node classification methods. The detailed statistics of the synthetic datasets are provided in Appendix A.2.1 in Table 5. Figure 5 visualizes the cross-class neighborhood similarity in the five hypersphere datasets with varying homophily levels (homophily varies in [0.2, 0.4, 0.6, 0.8, 1.0]). In the synthetic graphs with low label homophily, the intra- and inter-class neighborhood similarities show no significant differences, i.e., the nodes from different classes have similar label distributions in their neighborhoods. We overall observe a high absolute value of neighborhood similarity. The reason here is that similar to BLOGCAT (which also shows overall high CCNS scores) these synthetic low homophily graphs are highly connected and have a high average degree. As the label homophily gets higher, the contrast between the intra- and inter-class similarity becomes more significant.

5 Our Approach: Layerwise Feature Label Fusion

From our dataset analysis in the previous sections we observe that, unlike commonly used multi-class datasets, multi-label datasets usually have low label homophily and a more varied cross-class neighborhood similarity. Moreover, similar to the case of multi-class datasets, node features might not always be available (as in BlogCat) or might be noisy. Overcoming such issues demands explicit modeling of graph structure correlation or neighborhood label patterns for making predictions, a capability which is largely missing in current GNNs Jia & Benson (2020); Ma et al. (2020).

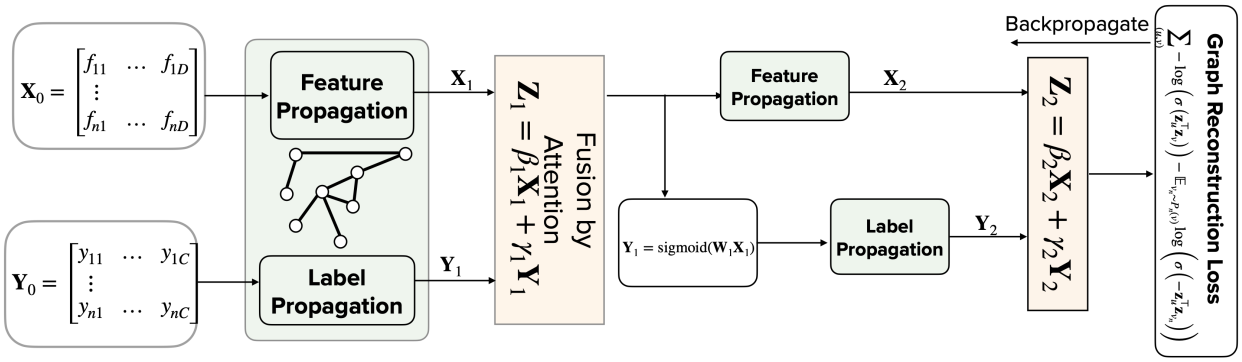


Figure 6: Architecture of LFLF model.

Inspired by Label Propagation based algorithms to utilize the neighborhood label correlation for prediction, we here propose and develop a parameterized approach for *dynamic label propagation*(DLP) (Wang et al., 2013). The key idea of our approach is to dynamically fuse the local similarities based on node features and neighborhood label correlation information at each layer to obtain an enriched representation. Specifically, we design our DLP-based parametric model using 3 modules, namely, *feature propagation, label propagation* and *dynamic fusion* modules.

The model architecture is depicted in Figure 6 and is elaborated in the following. We refer to our approach as **LFLF** (short for **L**ayerwise **F**eature **L**abel **F**usion).

Feature Propagation. The feature propagation module takes as input node features along with the adjacency matrix (graph structure) and outputs feature representations. Specifically, at $(k+1)$th layer, feature representation is computed as: $\mathbf{X}^{(k+1)} = \hat{\mathbf{A}}\mathbf{X}^{(k)}\mathbf{W}^{(k)}$, where $\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I}_n)\mathbf{D}^{-\frac{1}{2}}$, and $\mathbf{D}$ denotes the degree matrix of $\mathbf{A}+\mathbf{I}_n$ and $\mathbf{W}^{(k)}$ corresponds to the weight matrix (parameters) of layer k.

Label Propagation. The label propagation module takes as input (i) label matrix output from the kth layer, $\mathbf{Y}^{(k)}$ and (ii) the corresponding label correlation matrix, $\mathbf{L}^{(k)}_{corr} \in \mathbb{R}^{n\times n}$ projected on node neighborhoods. i.e., the (i,j)-th entry in the label correlation matrix $\mathbf{L}^{(k)}_{corr}$ denoted by $\mathbf{L}^{(k)}_{corr}(i,j)$ is given by

$$\mathbf{L}^{(k)}_{corr}(i,j) = \begin{cases} \mathbf{y}_i \cdot \mathbf{y}_j & \text{if } (i,j) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases}$$

For the input layer, we construct $\mathbf{Y}^{(0)}$ with the ground-truth one-hot encoded label matrix for the training nodes and uniform padding for validation and test nodes. At layer $k+1$ we compute the node label representation, $\mathbf{Y}^{(k+1)}$ by aggregating over the neighborhood labels as:

$$\mathbf{Y}^{(k+1)} = \tilde{\mathbf{L}}^{(k)}_{corr}\mathbf{Y}^{(k)}\mathbf{W}^{(k)}_{LP},$$

where $\tilde{\mathbf{L}}^{(k)}_{corr} = \tilde{\mathbf{D}}^{-\frac{1}{2}}(\mathbf{L}^{(k)}_{corr}+\mathbf{I}_n)\tilde{\mathbf{D}}^{-\frac{1}{2}}$, and $\tilde{\mathbf{D}}$ is the diagonal matrix corresponding to $\mathbf{L}^{(k)}_{corr}+\mathbf{I}_n$, $\mathbf{W}^{(k)}_{LP}$ is the learnable weight matrix for the kth layer in label propagation module. We comment that in our current experimental setup, to save extra computational time, we set the label correlation matrix $\mathbf{L}^{(k)}_{corr} = \mathbf{L}^{(0)}_{corr}$ for all k.

Fusion by Attention. In each layer, the label and feature representations are then merged using an attention mechanism and non-linearity to obtain feature and label-informed representation, $\mathbf{Z}_k$ with its ith row given by

$$\mathbf{Z}_i^{(k)} = \boldsymbol{\beta_i}\mathbf{X}_i^{(k)} + \boldsymbol{\gamma}_i\mathbf{Y}_i^{(k)}, \text{with } \boldsymbol{\beta}_i = \frac{\exp\left(\mathbf{c}_\beta(i)\right)}{\exp\left(\mathbf{c}_\beta(i)\right) + \exp\left(\mathbf{c}_\gamma(i)\right)} \text{ and } \quad \boldsymbol{\gamma}_i = \frac{\exp\left(\mathbf{c}_\gamma(i)\right)}{\exp\left(\mathbf{c}_\beta(i)\right) + \exp\left(\mathbf{c}_\gamma(i)\right)}, \quad (3)$$

Here, $\boldsymbol{\beta} \in \mathbb{R}^{n\times 1}$ and $\boldsymbol{\gamma} \in \mathbb{R}^{n\times 1}$ are importance weights for the feature and label representations, respectively, and the parameters $\mathbf{c}_\beta$ and $\mathbf{c}_\gamma$ are computed as follows.

$$\mathbf{c}_\beta = \tanh\left(\mathbf{X}^{(k)}\mathbf{W}_1^{(k)} + \mathbf{b}_1^{(k)}\right)\mathbf{W}_2^{(k)}, \mathbf{c}_\gamma = \tanh\left(\mathbf{Y}^{(k)}\mathbf{W}_1^{(k)} + \mathbf{b}_1^{(k)}\right)\mathbf{W}_2^{(k)}, \quad (4)$$

where $\mathbf{W}_2^{(k)} \in \mathbb{R}^{h\times 1}$, $\mathbf{W}_1^{(k)} \in \mathbb{R}^{h\times h}$ and $\mathbf{b_1}^{(k)} \in \mathbb{R}^{h\times 1}$ are learnable parameters and h denotes the intermediate embedding dimension.

Intermediate Prediction and Model Training. In each layer, k, the merged representation is used to predict intermediate node labels. This step imitates the kth round of label propagation. The predicted label matrix is then used as input to the label propagation module of the next layer.

One crucial issue of using this propagation rule is to avoid the model from "cheating", as the representation of the training data is generated part from their true labels to perform label prediction. Our solution is to use the unsupervised loss to guide the training process. In particular, instead of supervised loss, we employ a graph reconstruction loss which enforces representations of the connected nodes in the graph to be similar and the disparate nodes to be distinct:

$$\mathcal{J}_\mathcal{G}(u, v) = -\log\left(\sigma\left(\mathbf{z}_u^\top \mathbf{z}_v\right)\right) - \mathbb{E}_{v_n\sim P_n(v)} \log\left(\sigma\left(-\mathbf{z}_u^\top \mathbf{z}_{v_n}\right)\right), \quad (5)$$

where v is sampled from neighbor set $\mathcal{N}(u)$ of u, σ is the sigmoid function defined as $\sigma(x) = \frac{1}{1+\exp(-x)}$, P_n is a negative sampling distribution. Here we use uniform distribution over $\mathcal{V}$ as teh negative sampling distribution. In other words, for each node pair (u, v), we sample some node v_n uniformily at random from the node set $\mathcal{V}$. We approximate the second term in equation 5 by using Q negative samples. While in the current implementation, we choose connected neighbors for the positive sample, one can flexibly incorporate information from higher-order neighborhoods by modifying the choice of positive pairs. The learned representations are then used to train a one-vs-rest logistic regression implemented by LibLinear (Fan et al., 2008) for classification.

Model Variants and Space/Time Complexities. Because of the similarity of our propagation operations to that of GCN we call our model variant LFLF-GCN. Our second model variant, which we refer to as LFLF-GRAPHSAGE uses the aggregation operation of GRAPHSAGE. In principle, we provide a flexible architecture in which a desired aggregation/propagation operation can be used. The space and time complexity of the our approach is then dependent the underlying GNN architecture and a particular training implementation (for example whether using mini-batch or full batch training). In the following we also describe the overheads of computing label correlation matrix and the fused representation as these are the additional components not present in the underlying GNN approach. The computation of label correlation matrix takes time $O(|\mathcal{E}|)$ as we only compute the correlation values for the connected nodes. Moreover, assuming full batch implementation, the computation of the fused matrix using the attention mechanism in any layer requires $O(nh^2)$ where h is the intermediate embedding dimension and space of $O(nh)$ for storing the fused embedding matrix $\mathbf{Z}$. We remark that in our implementation we use mini-batch training with neighborhood sampling to keep our method scalable which results in considerable decrease in both space and runtime requirements.

6 Experiments

We perform a large-scale empirical study comparing 10 methods over 7 real-world multi-label datasets and 2 sets of synthetic datasets with varying homophily and feature quality. Our experiments are designed to

Table 3: Mean performance scores (Average Precision) on real-world datasets. The best score is marked in bold. "OOM" denotes the "Out Of Memory" error

Method	BlogCat	Yelp	OGB-Proteins	DBLP	PCG	HumLoc	EukLoc
Mlp	0.043	0.096	0.026	0.350	0.148	0.170	0.120
DeepWalk	**0.190**	0.096	0.044	0.585	0.229	0.186	0.076
LANC	0.050	OOM	0.045	0.836	0.185	0.132	0.062
Gcn	0.037	0.131	0.054	**0.893**	0.210	0.252	0.152
Gat	0.041	0.150	0.021	0.829	0.168	0.238	0.136
GraphSage	0.045	0.251	0.027	0.868	0.185	0.234	0.124
H2Gcn	0.039	0.226	0.036	0.858	0.192	0.172	0.134
GCN-LPA	0.043	0.116	0.023	0.801	0.167	0.150	0.075
LFLF-Gcn	0.067	0.198	0.230	0.853	0.250	0.251	0.151
LFLF-GraphSage	0.059	**0.294**	**0.329**	0.786	**0.289**	**0.278**	**0.179**

reveal and understand (i) properties of datasets that favor certain methods over others and (ii) the effect of varying feature quality and label homophily on method performance. The training and hyperparameter settings for each model are summarized in the Appendix A.3 in tables 7, 8, and 9. Our code is available at `https://anonymous.4open.science/r/LFLF-5D8C/`

Datasets. We employ 7 real-world datasets including BlogCat, Yelp, OGB-Proteins, DBLP, PCG, HumLoc, EukLoc, and 2 sets of synthetic datasets with varying homophily and feature quality. These datasets are already described in Section 3. For all datasets except OGB-Proteins, HumLoc, and EukLoc we generate 3 random training, validation, and test splits with 60%, 20%, and 20% of the data. For OGB-Proteins, HumLoc, and EukLoc we follow the predefined data splits from (Hu et al., 2020), (Shen & Chou, 2007) and (Chou & Shen, 2007) respectively. As BlogCat has no given node features, we use an identity matrix as the input feature matrix.

Baselines. For a holistic evaluation we include four classes of baseline methods (i) simple baselines, which include Multilayer Perceptron (Mlp), which only uses node features and ignores the graph structure, and DeepWalk, which only uses graph structure and ignores the node features (ii) Convolutional neural networks based which employ convolutional operations to extract representations from node's local neighborhoods and merge them with label embeddings for the final classification. We choose LANC as a baseline from this category, as previous works (Zhou et al., 2021; Song et al., 2021) have shown its superior performance. (iii) graph neural networks including (a) Gcn, Gat, and GraphSage, which are known to perform well for graphs with high label homophily, and (b)H2Gcn, which is designed to perform well both on homophilic and heterophilic graphs and (iiii) GCN-LPA which combines label propagation and GCN for node classification. All these methods are also discussed in Section 2.

Evaluation Metrics. We report the average micro- and macro-F1 score, macro-averaged AUC-ROC score, macro-averaged average precision score, and standard deviation over the three random splits. Due to space constraints, we report average precision (AP) in the main paper, and all detailed results are available in Tables 10, 11, 12 in the Appendix A.4. Our choice of using AP over AUROC as the metric is also motivated in Appendix A.5.

7 Results and Discussion

7.1 Results on real-world datasets.

Table 3 we provide the results for 7 real-world datasets. Overall our approach variant LFLF-GraphSage outperforms all approaches for all datasets except for BlogCat and DBLP in which DeepWalk and Gcn are the corresponding best performing methods.

BlogCat. For BlogCat all GNN approaches as well as LANC and GCN-LPA obtain scores close to that of Mlp which do not use any graph structure. Notably, Mlp uses identity matrix as input features which in principle provides no useful information. The corresponding scores can be seen as the result of a random

assignment. As a method unifying the label and feature propagation, GCN-LPA does not show improvement on BlogCat compared to other baselines. This is because GCN-LPA uses the label propagation to adjust the edge weight and still only generates embedding by aggregating features over the weighted graph, while our models slightly improve over these baselines by additionally incorporating the neighborhood label information exclusively in the form of label representation.

Yelp, OGB-Proteins and DBLP. Yelp has low label homophily and low clustering coefficient but a more balanced label distribution as compared to other real-world datasets. Moreover, it exhibits a high multi-label character, so approaches designed specifically for multi-label classification are expected to perform better for this dataset. Among the GNN-based baselines, H2Gcn which has shown improvements in low homophilic multi-class datasets outperforms Gcn,Gat, and GCN-LPA but is still outperformed by GraphSage. The use of CNNs in LANC to aggregate neighborhood features leads to excessive memory utilization for a graph with a very high maximum degree. This led to the out-of-memory error for Yelp. By utilizing the exact label correlation among the neighboring nodes, the variants of our model achieve a 16% improvement compared to the best-performing baseline.

Another interesting case is that OGB-Proteins where with our label-informed approach we achieve a 6-fold improvement over the best baseline approach. As already mentioned, the previous leaderboard results Hu et al. (2020) on OGB-Proteins are highly exaggerated due to the use of the AUROC metric accompanied by overtraining.

As a co-authorship dataset, DBLP has the highest label homophily and the largest portion of nodes with a single label among all the real-world datasets. As shown in Figure 2a in the section 3.1, the inter-class similarity is much weaker than the intra-class similarity. Besides, the high clustering coefficient indicates that the local neighborhood is highly connected. All of these factors further justify the best performance shown by Gcn. Besides, the relatively poor performances of Mlp and DeepWalk indicate that the features or the structure alone are not sufficient for estimating node labels.

PCG, HumLoc and EukLoc. If the features are highly predictive of the labels, the simple baselines Mlp using only feature information would be a competitive baseline. In the experiments with the datasets, where the features are highly correlated (evident from relatively better performance of Mlp) with the assigned labels, i.e., the biological interaction datasets PCG, HumLoc, and EukLoc (shown in Table 3), GNNs and GCN-LPA tend to have better performance than DeepWalk and LANC which only utilizes graph structure and features from the direct neighborhood, respectively. The effective fusion of feature and label information again allows our models to perform best in this case too.

7.2 Results on synthetic datasets

Table 4 provides results for synthetic datasets with varying feature quality and label homophily. Overall, our approach shows either the best or comparable results with the best-performing method under all conditions of feature quality and label homophily. We provide a detailed analysis and argue about performance differences in the following sections.

7.2.1 Effect of varying feature quality.

As simple baseline Mlp and other GNN-based models use features as input, we assume they will be sensitive to the varying feature quality. As DeepWalk uses the graph structure alone, its performance would not be affected by the varying feature quality. Further, the dynamic fusion process of label and feature information is expected to make LFLF robust to changes in feature quality.

To further validate our hypothesis and test the robustness of the methods to feature quality, we compare the method performances on variants of the generated Synthetic1 dataset. Specifically, we vary the ratio of the original to the irrelevant features as in $\{0, 0.2, 0.5, 0.8, 1.0\}$.

As shown in Table 4, under all levels of feature quality, the performances of DeepWalk do not change, as it generates representations for the nodes solely from the graph structure. The Mlp is unsurprisingly the most sensitive method to the varying feature quality because it completely ignores the graph structure. LANC is

Table 4: Average Precision (mean) on the synthetic datasets with varying levels of feature quality and homophily parameter. r_{ori_feat} and r_{homo} refer to the fraction of original features and the homophily parameter value, respectively.

Method	r_{ori_feat}					r_{homo}				
	0.0	0.2	0.5	0.8	1.0	0.2	0.4	0.6	0.8	1.0
MLP	0.172	0.187	0.220	0.277	0.343	0.343	0.343	0.343	0.343	0.343
DeepWalk	0.487	0.487	0.487	**0.487**	**0.487**	0.181	**0.522**	**0.813**	0.869	0.552
LANC	0.337	0.342	0.365	0.353	0.391	0.190	0.380	0.434	0.481	0.629
GCN	0.313	0.316	0.311	0.301	0.337	0.261	0.343	0.388	0.450	0.493
GAT	0.311	0.339	0.329	0.338	0.360	0.172	0.359	0.390	0.428	0.439
GraphSage	0.300	0.328	0.377	0.393	0.430	0.289	0.426	0.458	0.533	0.553
H2Gcn	0.376	0.401	0.427	0.442	0.467	0.297	0.484	0.512	0.572	0.652
GCN-LPA	0.337	0.333	0.368	0.363	0.391	0.170	0.408	0.495	0.604	0.583
LFLF-Gcn	0.451	0.459	0.449	0.433	0.431	0.355	0.438	0.509	0.600	0.752
LFLF-GraphSage	**0.512**	**0.508**	**0.498**	0.483	0.479	**0.356**	0.482	0.592	**0.891**	**0.906**

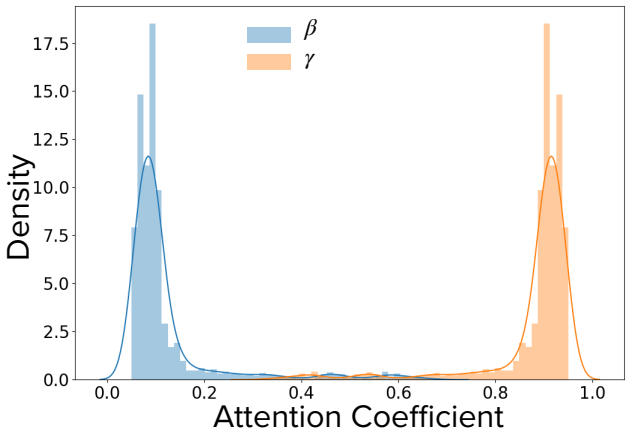


Figure 7: The attention coefficient β and γ learned for the feature propagation and label propagation when only random input features are used for the Synthetic1 dataset.

also sensitive to the change of the feature quality as it extracts feature vectors from the local neighborhood by performing convolutional operations on the stacked feature matrix of the direct neighbors.

Surprisingly GCN-LPA which also employs the label information performs only a little better than GraphSage on the feature-to-noise ratio of 0 and 0.2. H2Gcn, on the other hand, outperforms all GNN-based baselines and GCN-LPA for all levels of feature quality. The Synthetic1 dataset used in this experiment has a label homophily of 0.3768, which is a relatively high label homophily for the multi-label datasets.

As our models effectively use the neighborhood label correlation information to mitigate the effects of low feature quality, we observe high performances of our models even at lower feature quality. In Figure 7, we visualize the distribution of the attention coefficients of each node learned by our model LFLF-GraphSage for the feature and label propagation when the input features are all noise. Thanks to the use of label information as input and the graph reconstruction loss, our models learn to weigh more the label representations and ignore the noisy features.

7.2.2 Effect of varying label homophily.

In this subsection, we test the robustness of the methods to varying homophily and further argue how low label homophily has different semantics on multi-label graphs as it does on multi-class datasets. Specifically, we vary the label homophily as in $\{0.2, 0.4, 0.6, 0.8, 1.0\}$.

As shown in Table 4, the performances of MLP do not change under all level of label homophily, due to the fact that MLP only use the input features.

H2GCN is a method that has been shown to perform well on heterophilic multi-class datasets. However, as shown in Table 4, for low homophilic variants of the synthetic dataset it does not show significant improvements compared to other baselines as it does on multi-class datasets. The reason here is that lower homophily in multi-label datasets does not always imply heterophily or complete dissimilarity among connected nodes. H2GCN is built on the assumption that in low homophilic graphs, higher-order neighborhood information is more important than one-hop neighborhoods, which does not hold in this case.

On the other hand, with most of the attention drawn to developing new complicated methods for the node classification task, we observe simple baselines such as DEEPWALK outperform standard GNNs in several scenarios. On the synthetic datasets with the label homophily of 0.4 and 0.6, DEEPWALK is the best-performing method and still remains a strong baseline in the high label homophily regime. As shown in Table 4, the performance of DEEPWALK drops when the label homophily is 1.0. However, we want to emphasize that this is because we fixed the same walk length for DEEPWALK for all levels of label homophily and the improvement can be shown with a possible hyperparameter tuning.

As mentioned in section 4, the intra-class similarity is significantly stronger than the inter-class similarity in the synthetic graphs with higher label homophily, which helps GNN-based models to better distinguish the nodes from different classes and thus achieve better results in the node classification task. Nevertheless, the variants of our model show improvements across all synthetic graphs with varying label homophily. This is due to the fact that our models dynamically fuse the feature and label representations, resulting in well-informed node representations, even in low homophily scenarios. Overall, our approach variants achieve up to 40% improvement compared to other baselines.

8 Conclusion

We investigate the problem of multi-label node classification on graph-structured data. Filling in the gaps in current literature, we (i) perform in-depth analysis on the commonly used benchmark datasets, create and release several real-world multi-label datasets and a graph generator model to produce synthetic datasets with tunable properties, (ii) develop a new approach which explicitly use label induced node similarities to learn informative representations. Our approach dynamically fuses the feature and label propagation and can be used together with current GNNs. We demonstrate the improvements of our models by conducting large-scale experiments on 9 datasets and 10 methods, and (iii) release our benchmark publicly.

We have novel and compelling insights from our analysis of specific datasets and GNN approaches. For instance, we uncover the pitfalls of the commonly used OGB-Protein dataset for model evaluation. While multi-label graph datasets usually show low homophily, we show that approaches working on low homophilic multi-class datasets cannot trivially work on multi-label datasets which usually have low homophily.

While current graph-based machine learning methods are usually evaluated on multi-class datasets, we demonstrate that the acquired improvements cannot always be translated to the more general scenario when the nodes are characterized by multiple labels. We believe that our work will open avenues for more future work and bring much-deserved attention to multi-label classification on graph-structured data. In future work, we plan to study the interplay of different dataset characteristics (for example edge density and label homophily) on the model performance. Besides, we would like to extend the architecture of our currently proposed two-step LFLF by including supervised as well as other self-learning objectives.

References

Uchenna Akujuobi, Yufei Han, Qiannan Zhang, and Xiangliang Zhang. Collaborative graph walk for semi-supervised multi-label node classification. *CoRR*, abs/1910.09706, 2019. URL `http://arxiv.org/abs/1910.09706`.

Sanmitra Bhattacharya, Viet Ha-Thuc, and Padmini Srinivasan. Mesh: a window into full text for document summarization. *Bioinformatics*, 27(13):i120–i128, 2011.

Marián Boguná, Romualdo Pastor-Satorras, Albert Díaz-Guilera, and Alex Arenas. Models of social networks based on social distance attachment. *Physical review E*, 70(5):056122, 2004.

Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5177–5186, 2019.

Yinlin Cheng, Mengnan Ma, Xingyu Li, and Yi Zhou. Multi-label classification of fundus images based on graph convolutional network. *BMC Medical Informatics and Decision Making*, 21(2):1–9, 2021.

Kuo-Chen Chou and Hong-Bin Shen. Euk-mploc: a fusion classifier for large-scale eukaryotic protein subcellular location prediction by incorporating multiple sites. *Journal of Proteome Research*, 6(5): 1728–1734, 2007.

Hong-Min Chu, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. Deep generative models for weakly-supervised multi-label classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 400–415, 2018.

Gene Ontology Consortium. The gene ontology resource: 20 years and still going strong. 47(D1):D330–D338, 2018.

UniProt Consortium. Uniprot: a hub for protein information. *Nucleic acids research*, 43(D1):D204–D212, 2015.

Szklarczyk D, Gable AL, Junge A Lyon D, Wyder S, Huerta-Cepas J, Simonovic M, Doncheva NT, Morris JH, Bork P, Jensen LJ, and Mering CV. String v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Research*, 47(D1):D607–D613, 2019.

Thi Ngan Dong and Megha Khosla. Towards a consistent evaluation of mirna-disease association prediction models. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 1835–1842. IEEE, 2020.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *the Journal of machine Learning research*, 9:1871–1874, 2008.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *CoRR*, abs/1706.02216, 2017. URL `http://arxiv.org/abs/1706.02216`.

Mengying Hu, Hu Han, Shiguang Shan, and Xilin Chen. Weakly supervised image classification through noise regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11517–11525, 2019.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

Dat Huynh and Ehsan Elhamifar. Interactive multi-label cnn learning with partial labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9423–9432, 2020.

Vikas Jain, Nirbhay Modhe, and Piyush Rai. Scalable generative models for multi-label learning with missing labels. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1636–1644. PMLR, 06–11 Aug 2017. URL `https://proceedings.mlr.press/v70/jain17a.html`.

Junteng Jia and Austion R Benson. Residual correlation in graph neural network regression. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 588–598, 2020.

Samuel Kerrien, Bruno Aranda, Lionel Breuza, Alan Bridge, Fiona Broackes-Carter, Carol Chen, Margaret Duesbury, Marine Dumousseau, Marc Feuermann, Ursula Hinz, et al. The intact molecular interaction database in 2012. *Nucleic acids research*, 40(D1):D841–D846, 2012.

Megha Khosla, Jurek Leonhardt, Wolfgang Nejdl, and Avishek Anand. Node representation learning for directed graphs. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 395–411. Springer, 2019.

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

Jack Lanchantin, Arshdeep Sekhon, and Yanjun Qi. Neural message passing for multi-label classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 138–163. Springer, 2019.

Irene Li, Tianxiao Li, Yixin Li, Ruihai Dong, and Toyotaro Suzumura. Heterogeneous graph neural networks for multi-label text classification. *CoRR*, abs/2103.14620, 2021. URL `https://arxiv.org/abs/2103.14620`.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pp. 115–124, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350228. doi: 10.1145/3077136.3080834. URL `https://doi.org/10.1145/3077136.3080834`.

Weiwei Liu, Haobo Wang, Xiaobo Shen, and Ivor W Tsang. The emerging trends of multi-label learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(11):7955–7974, 2021.

Jiaqi Ma, Bo Chang, Xuefei Zhang, and Qiaozhu Mei. Copulagnn: towards integrating representational and correlational roles of graphs in graph neural networks. *arXiv preprint arXiv:2010.02089*, 2020.

Qianwen Ma, Chunyuan Yuan, Wei Zhou, and Songlin Hu. Label-specific dual graph neural network for multi-label text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3855–3864, 2021a.

Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks? *arXiv preprint arXiv:2106.06134*, 2021b.

Mingdong Ou, Peng Cui, Jian Pei, et al. Asymmetric transitivity preserving graph embedding. In *Proc. of the International Conference on Knowledge Discovery and Data Mining*, pp. 1105–1114, 2016.

Ankit Pal, Muru Selvakumar, and Malaikannan Sankarasubbu. Multi-label text classification using attention-based graph neural network. *arXiv preprint arXiv:2003.11644*, 2020.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proc. of the International Conference on Knowledge Discovery and Data Mining*, 2014.

Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Saüch-Pitarch, Francesco Ronzano, Emilio Centeno, Ferran Sanz, and Laura I Furlong. The disgenet knowledge platform for disease genomics: 2019 update. *Nucleic acids research*, 48(D1):D845–D855, 2020.

Deepak Saini, Arnav Kumar Jain, Kushal Dave, Jian Jiao, Amit Singh, Ruofei Zhang, and Manik Varma. Galaxc: Graph neural networks with labelwise attention for extreme classification. WWW '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383127. doi: 10.1145/3442381.3449937. URL `https://doi.org/10.1145/3442381.3449937`.

Hong-Bin Shen and Kuo-Chen Chou. Hum-mploc: an ensemble classifier for large-scale human protein subcellular location prediction by incorporating samples with multiple sites. *Biochemical and biophysical research communications*, 355(4):1006–1011, 2007.

Min Shi, Yufei Tang, Xingquan Zhu, and Jianxun Liu. Multi-label graph convolutional network representation learning. *CoRR*, abs/1912.11757, 2019. URL http://arxiv.org/abs/1912.11757.

Min Shi, Yufei Tang, Xingquan Zhu, and Jianxun Liu. Multi-label graph convolutional network representation learning. *IEEE Transactions on Big Data*, 2020.

Dezhao Song, Andrew Vold, Kanika Madan, and Frank Schilder. Multi-label legal document classification: A deep learning-based approach with label-attention and domain-specific pre-training. *Inf. Syst.*, 106(C), may 2022. ISSN 0306-4379. doi: 10.1016/j.is.2021.101718. URL https://doi.org/10.1016/j.is.2021.101718.

Zixing Song, Ziqiao Meng, Yifei Zhang, and Irwin King. Semi-supervised multi-label learning for graph-structured data. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 1723–1733, 2021.

Szymon Talaga and Andrzej Nowak. Homophily as a process generating social networks: Insights from social distance attachment model. *Journal of Artificial Societies and Social Simulation*, 23(2):6, 2020. ISSN 1460-7425.

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*, KDD '09, pp. 817–826, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584959. doi: 10.1145/1557019.1557109. URL https://doi.org/10.1145/1557019.1557109.

Jimena Torres Tomás, Newton Spolaôr, Everton Alvares Cherman, and Maria Carolina Monard. A framework to generate synthetic multi-label datasets. *Electronic Notes in Theoretical Computer Science*, 302:155–176, 2014.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. accepted as poster.

Bingyu Wang, Li Chen, Wei Sun, Kechen Qin, Kefeng Li, and Hui Zhou. Ranking-based autoencoder for extreme multi-label classification. *CoRR*, abs/1904.05937, 2019. URL http://arxiv.org/abs/1904.05937.

Bo Wang, Zhuowen Tu, and John K. Tsotsos. Dynamic label propagation for semi-supervised multi-class multi-label classification. In *2013 IEEE International Conference on Computer Vision*, pp. 425–432, 2013. doi: 10.1109/ICCV.2013.60.

Hongwei Wang and Jure Leskovec. Unifying graph convolutional neural networks and label propagation. *CoRR*, abs/2002.06755, 2020. URL https://arxiv.org/abs/2002.06755.

Guanming Wu, Xin Feng, and Lincoln Stein. A human functional protein interaction network and its application to cancer data analysis. *Genome biology*, 11(5):1–23, 2010.

Linli Xu, Sijie Teng, Ruoyu Zhao, Junliang Guo, Chi Xiao, Deqiang Jiang, and Bo Ren. Hierarchical multi-label text classification with horizontal and vertical category correlations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 2459–2468, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.190. URL https://aclanthology.org/2021.emnlp-main.190.

Cheng Yang, Jiawei Liu, and Chuan Shi. Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework. In *Proceedings of The Web Conference 2021 (WWW '21)*. ACM, 2021.

Xiaodi Yang, Shiping Yang, Qinmengge Li, Stefan Wuchty, and Ziding Zhang. Prediction of human-virus protein-protein interactions through a sequence embedding-based machine learning method. *Computational and structural biotechnology journal*, 18:153–161, 2020.

Yang Yang, Ryan N Lichtenwalter, and Nitesh V Chawla. Evaluating link prediction methods. *Knowledge and Information Systems*, 45(3):751–782, 2015.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor K. Prasanna. Graphsaint: Graph sampling based inductive learning method. *CoRR*, abs/1907.04931, 2019. URL `http://arxiv.org/abs/1907.04931`.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2013.

Xiangping Zheng, Xun Liang, and Bo Wu. Capsule graph neural network for multi-label image recognition (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 13117–13118, 2022.

Cangqi Zhou, Hui Chen, Jing Zhang, Qianmu Li, Dianming Hu, and Victor S. Sheng. Multi-label graph node classification with label attentive neighborhood convolution. *Expert Systems with Applications*, 180: 115063, 2021. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2021.115063. URL `https://www.sciencedirect.com/science/article/pii/S0957417421005042`.

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

Daoming Zong and Shiliang Sun. GNN-XML: graph neural networks for extreme multi-label text classification. *CoRR*, abs/2012.05860, 2020. URL `https://arxiv.org/abs/2012.05860`.

A Appendix

Organization. We explain the construction details of the biological datasets in A.1. Furthermore, in Section A.2, we study the parameters of the graph generator and demonstrate how we generate the synthetic datasets with varying label homophily. We also summarize the characteristics of the generated synthetic graphs that were used in Section 6 for the varying homophily and feature quality experiments. In Section A.3, we summarize the hyperparameters of the models we used in this work. In Section A.4, we provide the full original experiment results on all datasets reported in Micro- and Macro- F1, AUROC, and Average Precision score with the standard deviation of the 3 random splits if the dataset is not pre-splitted. Note that for higher precision, the scores are provided in percentages. Last but not least, we provide the motivation for using Average Precision in the main paper in Section A.5.

A.1 Biological Dataset Construction

A.1.1 The Protein phenotype prediction dataset.

A phenotype is any observable characteristic or trait of a disease. Identifying the phenotypes associated with a particular protein could help in clinical diagnostics or finding possible drug targets.

To construct the phenotype prediction dataset, we first retrieve the experimentally validated protein-phenotype associations from the DisGeNET (Piñero et al., 2020) database. We then (i) retain only those protein associations that are marked as "phenotype", (ii) match each disease to its first-level category in the MESH ontology (Bhattacharya et al., 2011), and (iii) remove any (phenotype) label with less than 100 associated proteins. To construct the edges, we acquire the protein functional interaction network from (Wu et al., 2010) (version 2020). We then (i) model each protein as a node in the graph, (ii) retain only the protein-protein interactions between the proteins that have the phenotype labels available, and (iii) remove any isolated nodes from the constructed graph. In the end, our dataset consists of $3,233$ proteins and $37,351$ edges. The node features are the 32-dimensional sequence-based embeddings retrieved from (Consortium, 2015) and (Yang et al., 2020).

A.1.2 The human protein subcellular location prediction dataset (HumLoc).

Proteins might exist at or move between different subcellular locations. Predicting protein subcellular locations can aid the identification of drug targets[1]. We retrieve the human protein subcellular location data from (Shen & Chou, 2007) which contains $3,106$ proteins. Each protein can have one to several labels in 14 possible locations. We then generate the graph multi-label node classification data as follows:

- We model each protein as a node in the graph. We retrieve the corresponding protein sequences from Uniprot (Consortium, 2015). We obtain the corresponding 32-dimensional node feature representation by feeding them to a pre-trained model (Yang et al., 2020) on protein sequences.
- Each node's label is the one-hot encoding (i.e., 14 dimensions) generated from its sub-cellular information. Each value in the label vector represents one sub-cellular location. A value of 1 indicates the corresponding protein exists at the respective location and 0 means otherwise.
- The edge information is generated from the protein-protein interactions retrieved from the IntAct (Kerrien et al., 2012) database. There exists a connection between two nodes in the graph if there exists an interaction between the corresponding proteins in IntAct. For each pair of proteins, more than one interaction of different types might exist. Therefore, we assign each edge a label. The edge label is modeled as a 21-dimensional vector where each value in the vector represents the confidence score for a particular connection type.

In the end, the HumLoc dataset consists of 3,106 nodes and 18,496 edges. Each node can have one to several labels in the 14 possible locations. Among the 3,106 different proteins, 2,580 of them belong only to 1 location; 480 of them belong to 2 locations; 43 of them belong to 3 locations and 3 of them belong to 4 locations. Both the accession numbers and sequences are given. None of the proteins has more than 25% sequence identity to any other in the same subset (subcellular location). For a more detailed description of the original dataset, we refer the readers to (Shen & Chou, 2007).

A.1.3 The eukaryote protein subcellular location prediction dataset (EukLoc).

We retrieve the eukaryote protein subcellular location multi-label data from (Chou & Shen, 2007). We then employ the same data sources and pre-processing strategy as described for the HumLoc dataset to generate the multi-label node classification dataset for eukaryote protein subcellular location prediction. In the end, the final pre-processed data contains 7,766 proteins (nodes) and 13,818 connections(edges). Each protein(node) can receive one to several labels in 22 possible locations.

A.2 Parameter Study of the Graph Generator Model

As mentioned in Section 4, the choice of α and b will directly determine the connection probability $p_{i,j}$ of each pair of nodes i and j and the homophily ratio of the generated synthetic graph. Here, we demonstrate how we choose the value of α and b to generate synthetic graphs with varying homophily ratios. Note that the valid range of α and b may differ when a different distance metric is used.

Firstly, we randomly choose 500 nodes from Synthetic1 dataset and generate a series of small synthetic graphs with varying α and b and observe the relationship between them. Since we have two hyperparameters' ranges to determine, we first fix the value of one and explore the range of the other and then vice versa. To recall, b indicates the characteristic distance at which $p_{ij} = \frac{1}{2}$, and our hamming distance should be in the range of $[0, 1]$, we first fix b to 0.05. We chose a small value of b because the larger value of b would dominate the influence of α and the relationship between the homophily ratio and the change of the value of α becomes unclear. The relationship between the homophily ratio of the generated synthetic graphs and the value of α is shown in Figure 8(a). As shown in the subplot, the label homophily increases monotonically as the value of α increases in the range of $[0, 10]$. As α is interpreted as the homophily parameter in (Boguná et al., 2004), only positive values make sense.

[1] https://en.wikipedia.org/wiki/Protein_subcellular_localization_prediction

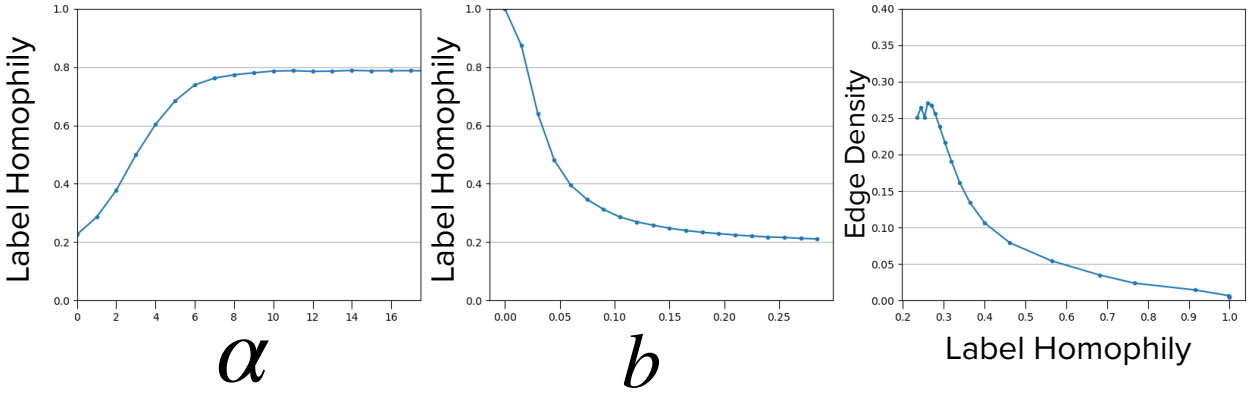


Figure 8: Visualization of the parameter study of the Graph Generator Model. The first two subplots demonstrate the relationships between the value of α and b and the label homophily of the generated synthetic datasets. The last subplot shows the edge density and the label homophily of the generated synthetic graphs. This shows with the same multi-label data, we can generate synthetic graphs with varying label homophily.

Similarly, we then fix α to its middle value in the valid range, i.e. 5, and explore the valid range of b and visualize the relationship of the graph level homophily ratio and the value of b in Figure 8(b). As illustrated in the subfigure, the label homophily decreases as the b increases in the range of $(0, 0.25]$. As b increases, the node pairs with bigger distance would also have 50% of the probability to be connected, the number of edges will increase, and the label homophily ratio will decrease. As b decreases, the node pair with a smaller distance would only have 50% of the probability of being connected. The model becomes cautious about connecting a node pair. The number of edges decreases and only the node pairs, which are alike will be connected, thus, the label homophily ratio increases.

Then, we use combinations of α and b to generate synthetic graphs with specific homophily ratios. We sample 20 αs and bs uniformly from their valid ranges with the increments 0.5 and 0.0125. Since the graph label homophily has an inverse linear relationship with α and b, we arrange the sampled b in reverse order and then form 20 value pairs (alpha, b). We generate 20 synthetic graphs from the multi-label dataset corresponding to Synthetic1 with these value pairs (α, b) and plot the homophily ratio and the edge density of the generated graphs in Figure 8(c). As shown in the subplot, using the same multi-label data, we are able to generate synthetic graphs with varying homophiles. And the edge density decreases when label homophily increases. As in higher label homophily graphs, the generator will only connect the nodes that are highly similar to each other. In contrast, when the label homophily is low, the graph generator will connect every possible node pair in the graph resulting in denser graphs.

A.2.1 Statistics Of The Synthetic Datasets

Here we summarize the characteristics of the generated synthetic graphs with varying label homophily and feature quality. The first row denotes the name of the synthetic graphs, where in the varying homophily experiments, the variants of datasets are named with their label homophily. The Synthetic1 dataset is used in the varying feature quality experiment, where we remove the relevant features to create dataset variants with varying feature quality levels. Note that for all the datasets the label distribution stays the same as they are just different graphs generated from the same multi-label dataset. The statistics on label distribution for these datasets are given in Table 5 and Table 6.

A.3 Hyperparameter Setting

In this section, we summarize all the hyperparameters we used for the experiment section. The detailed setting is listed in Table 7, 8, and 9.

Table 5: The number of edges and clustering coefficient of the synthetic datasets with varying label homophily and Synthetic1. The row of '|E|' denotes the number of edges and the 'clustering coefficient' denotes the clustering coefficient of these datasets

Dataset	0.2	0.4	0.6	0.8	1.0	Synthetic1
|E|	2.37M	598k	298k	79.5k	47.6k	1.00M
Clustering Coefficient	0.53	0.37	0.39	0.49	0.93	0.57

Table 6: The label distribution of the synthetic dataset used in this work. The column notations are same as in Table 1.

	$\|L\|$	$\|L_{med}\|$	$\|L_{mean}\|$	$\|L_{max}\|$	25%	50%	75%
label distribution	20	3	3.23	12	1	3	5

More specifically, for Mlp and all GNN-based methods, we summarize the number of layers, the dimension of the hidden layer, the learning rate, the patience of the Earlystopping, the weight decay, and the number of neighbors we sample for the models that require sampling.

We use the same number of layers and the same hidden size for Mlp and the other GNN-based methods. The learning rate for the synthetic datasets in the varying feature quality and homophily experiments is 0.001 instead of 0.01 as in the other models because the performance of the H2Gcn is further improved. We also use Earlystopping with the patience of 100 epochs to train the models properly. For GraphSage, we sample 25 and 20 one and two hops away neighbors for aggregation. As other GNN-based baselines do not use the sampling method, the corresponding cells are filled with "No".

Table 7: The hyperparameter setting for Mlp and GNN baselines in this work for all datasets

	MLP	GCN, GAT, GraphSAGE	H2GCN	GCN-LPA
Layers	2	2	2	2 GCN 5 LPA
Hidden size	256	256	256	256
Learning rate	0.01	0.01	real-world: 0.01 synthetic: 0.001	0.01
Earlystopping patience	100	100	100	100
Weight decay	5e-4	5e-4	5e-4	1e-4
Sample for aggregation	No	GraphSAGE:[25, 10]	No	No

In the variants of our model, we use two layers where at each of the layers, the feature and label propagation are fused to generate the input features for the next layer. The learning rate and patience for Earlystopping remain the same as in the other GNN-based methods. Specially, we sample 25 one hop away, and 20 two

hops away neighbors for aggregation and for loss calculation, we sample 20 positive neighbors and 60 negative neighbors for the loss function.

Table 8: The hyperparameter setting for LFLF-Gcn, LFLF-GraphSage in this work for all datasets

	Layers	Hidden size	Learning Rate	Patience	Weight Decay	Sample	
						Aggregation	Loss
LFLF	2	256	0.01	100	5e-4	[25, 10]	[20, 60]

For the only random-walk-based method, we deploy the default setting for all the datasets in this work as DeepWalk already shows competitive performance. We perform 10 random walks with the walk length of 10 for each node to generate the sequence and use the window size of 5 for the training pairs, the generated embedding size is 64.

Table 9: The hyperparameter setting for DeepWalk in this work for all datasets

	Number of walks	Walk length	Embedding size	Window size
DeepWalk	10	10	64	5

A.4 The experiment results reported in four metrics

We summarize the experimental results on the real-world datasets and the synthetic datasets in Table 10, Table 11, and Table 12, respectively. For better precision, we report the scores in percentages. Specifically, for the scores reported on OGB-Proteins, the difference between our results and those reported in the benchmark is because 1) we use 2 layer Mlp without Node2Vec features. 2) We use Earlystopping with the patience of 100 epochs to prevent the models from being overtrained. 3) We use sampled local neighborhoods to have a consistent setting for all the datasets using GraphSage. The specific parameters we used are summarized in A.3. Another pre-processing we did was to remove the isolated nodes in PCG. The details are described in Appendix A.1.1.

Table 10: Multi-label Node Classification results on real-world datasets. The results of BlogCat, Yelp, PCG, and DBLP are the mean of three random splits with 60% for train-, 20% validation and 20% test dataset, while the results of OGB-Proteins, HumLoc, EukLoc are reported with the built-in split.

Dataset	**Method**	Micro-F1	Macro-F1	AUC-ROC	AP
Blog-Cat	Mlp	17.11 ± 0.64	2.49 ± 0.18	50.30 ± 1.04	4.25 ± 0.63
	DeepWalk	$\mathbf{35.59 \pm 0.21}$	$\mathbf{19.74 \pm 0.54}$	$\mathbf{73.20 \pm 0.58}$	$\mathbf{18.55 \pm 0.17}$
	LANC	13.95 ± 2.02	4.55 ± 0.82	52.34 ± 0.91	5.03 ± 0.07
	Gcn	16.69 ± 0.47	2.63 ± 0.08	47.85 ± 0.06	3.69 ± 0.04
	Gat	17.22 ± 0.52	2.48 ± 0.08	50.88 ± 1.45	4.05 ± 0.09
	GraphSage	16.18 ± 0.31	2.38 ± 0.27	52.73 ± 0.82	4.50 ± 0.12
	H2Gcn	16.86 ± 0.34	2.60 ± 0.16	49.83 ± 1.08	3.92 ± 0.05
	GCN-LPA	17.15 ± 0.68	2.55 ± 0.19	51.35 ± 0.67	4.33 ± 0.31
	LFLF-Gcn	$\underline{19.69 \pm 0.57}$	$\underline{4.93 \pm 0.56}$	58.97 ± 0.56	$\underline{6.70 \pm 0.41}$
	LFLF-GraphSage	18.24 ± 0.72	4.39 ± 0.69	$\underline{59.45 \pm 0.96}$	5.88 ± 0.48

Table 10: Multi-label Node Classification results on real-world datasets. The results of BlogCat, Yelp, PCG, and DBLP are the mean of three random splits with 60% for train-, 20% validation and 20% test dataset, while the results of OGB-Proteins, HumLoc, EukLoc are reported with the built-in split.

Dataset	Method	Micro-F1	Macro-F1	AUC-ROC	AP
Yelp	Mlp	26.04 ± 0.09	18.55 ± 0.20	50.17 ± 0.01	9.58 ± 0.01
	DeepWalk	49.78 ± 0.07	24.98 ± 0.04	50.67 ± 0.08	9.60 ± 0.02
	LANC	–	–	–	–
	Gcn	52.21 ± 0.07	27.60 ± 0.04	53.81 ± 0.13	13.14 ± 0.06
	Gat	51.24 ± 0.08	26.66 ± 0.06	67.80 ± 0.05	15.00 ± 0.07
	GraphSage	$\mathbf{56.06 \pm 0.10}$	31.26 ± 0.10	$\mathbf{81.05 \pm 0.25}$	25.09 ± 0.31
	H2Gcn	54.12 ± 0.01	30.52 ± 0.11	75.25 ± 0.46	22.57 ± 0.51
	GCN-LPA	50.31 ± 0.29	25.68 ± 0.43	61.09 ± 2.27	11.62 ± 0.74
	LFLF-Gcn	46.03 ± 0.05	30.93 ± 0.05	63.47 ± 0.37	19.79 ± 0.14
	LFLF-GraphSage	48.52 ± 0.99	$\mathbf{31.53 \pm 0.32}$	77.66 ± 1.70	$\mathbf{29.37 \pm 0.87}$
OGB-Proteins	Mlp	2.55	2.40	54.05	2.59
	DeepWalk	**2.88**	**2.75**	68.75	4.41
	LANC	2.35	2.21	68.03	4.48
	Gcn	2.77	2.63	71.48	5.36
	Gat	2.55	2.40	50.64	2.14
	GraphSage	2.59	2.43	55.83	2.68
	H2Gcn	2.55	2.39	62.75	3.61
	GCN-LPA	2.56	2.41	53.22	2.33
	LFLF-Gcn	1.98	1.90	81.31	22.99
	LFLF-GraphSage	2.14	2.03	**90.66**	**32.85**
DBLP	Mlp	42.14 ± 0.27	32.04 ± 0.65	54.47 ± 0.07	34.97 ± 0.14
	DeepWalk	63.27 ± 0.34	59.11 ± 0.36	74.81 ± 0.16	58.49 ± 0.25
	LANC	81.93 ± 0.29	80.39 ± 0.42	91.76 ± 0.36	83.55 ± 0.98
	Gcn	$\mathbf{87.03 \pm 0.20}$	$\mathbf{85.80 \pm 0.38}$	94.15 ± 0.16	$\mathbf{89.27 \pm 0.24}$
	Gat	83.06 ± 0.17	81.26 ± 0.14	92.57 ± 0.07	82.93 ± 0.16
	GraphSage	85.22 ± 0.23	83.89 ± 0.21	$\mathbf{94.32 \pm 0.02}$	86.84 ± 0.18
	H2Gcn	83.99 ± 0.92	82.56 ± 0.86	92.14 ± 0.57	85.82 ± 0.64
	GCN-LPA	82.88 ± 0.31	81.31 ± 0.34	90.17 ± 0.43	80.07 ± 1.24
	LFLF-Gcn	82.63 ± 0.15	81.41 ± 0.23	91.90 ± 0.06	85.31 ± 0.28
	LFLF-GraphSage	79.42 ± 0.18	77.54 ± 0.19	89.61 ± 0.21	78.56 ± 0.27
PCG	Mlp	38.04 ± 1.20	18.03 ± 1.29	51.07 ± 0.63	14.78 ± 0.60
	DeepWalk	$\mathbf{42.26 \pm 1.37}$	$\mathbf{31.49 \pm 0.90}$	63.58 ± 0.87	22.86 ± 1.00
	LANC	36.28 ± 0.34	20.50 ± 1.15	56.58 ± 0.69	18.53 ± 1.14
	Gcn	41.46 ± 1.21	25.59 ± 0.92	59.54 ± 0.82	21.03 ± 0.34
	Gat	36.91 ± 1.75	19.24 ± 0.75	56.33 ± 4.64	16.75 ± 2.17
	GraphSage	38.89 ± 1.17	24.44 ± 1.74	58.57 ± 0.08	18.45 ± 0.29
	H2Gcn	39.05 ± 0.99	24.38 ± 2.17	58.10 ± 0.14	19.19 ± 0.49
	GCN-LPA	39.57 ± 1.12	22.90 ± 1.33	54.74 ± 0.95	16.71 ± 0.14
	LFLF-Gcn	40.64 ± 0.78	26.80 ± 0.31	64.30 ± 0.31	24.99 ± 0.72
	LFLF-GraphSage	41.28 ± 1.19	28.97 ± 1.13	$\mathbf{66.45 \pm 0.66}$	$\mathbf{28.92 \pm 1.03}$
Hum-Loc	Mlp	42.12	18.04	66.04	16.95
	DeepWalk	45.26	23.30	65.67	18.58
	LANC	39.25	11.51	59.65	13.24
	Gcn	51.67	25.57	67.28	25.15
	Gat	47.10	17.49	72.47	23.75
	GraphSage	48.05	21.22	70.30	23.42
	H2Gcn	45.39	18.35	64.31	17.23

Table 10: Multi-label Node Classification results on real-world datasets. The results of BlogCat, Yelp, PCG, and DBLP are the mean of three random splits with 60% for train-, 20% validation and 20% test dataset, while the results of OGB-Proteins, HumLoc, EukLoc are reported with the built-in split.

Dataset	Method	Micro-F1	Macro-F1	AUC-ROC	AP
	GCN-LPA	45.73	18.15	62.40	14.96
	LFLF-Gcn	49.56	23.03	75.46	25.07
	LFLF-GraphSage	**52.08**	**27.36**	**76.75**	**27.84**
Eu-k-Loc	Mlp	43.58	11.13	66.83	12.00
	DeepWalk	34.67	6.74	56.12	7.58
	LANC	36.08	4.55	51.13	6.16
	Gcn	45.86	12.27	70.53	15.15
	Gat	41.58	6.76	71.65	13.59
	GraphSage	44.65	11.96	69.04	12.44
	H2Gcn	44.93	11.80	69.45	13.35
	GCN-LPA	36.72	5.93	56.65	7.45
	LFLF-Gcn	44.28	12.57	76.31	15.06
	LFLF-GraphSage	**46.53**	**14.10**	**78.68**	**17.86**

A.5 Challenge of the Evaluation Metrics

The Area under the ROC curve (AUC) and the Area under the Precision-Recall curve (AUPR) are two widely accepted non-parametric measurement scores used by existing works. Nevertheless, as discussed in (Yang et al., 2015), the AUC score is sometimes misleading for highly imbalanced datasets like those in our work. In addition, AUPR might lead to over-estimation of the models' performance when the number of thresholds (or unique prediction values) is limited (Dong & Khosla, 2020). For such reasons, as suggested in (Dong & Khosla, 2020), we instead use the Average Precision (AP) score as our primary evaluation metric. Following previous works, we also report the F1 score. As it is a threshold-based metric we emphasize that it might be biased when the benchmarked models have different prediction score ranges.

A.6 Cross-class Neighborhood Similarity Plots

In this section, we put the heat maps of the cross-class neighborhood similarity for all the datasets used in this work. We use color coding, where a darker shade in the cell indicates a stronger cross-class neighborhood similarity.

Table 11: Multi-label Node Classification results on Synthetic dataset with varying feature quality. All results are the mean of three random splits. The Ratios of the relevant and the irrelevant features are $[0, 0.2, 0.5, 0.8, 1.0]$.

Feature Ratios	**Method**	Micro-F1	Macro-F1	AUC-ROC	AP
0	Mlp	67.05 ± 0.91	25.15 ± 0.89	50.99 ± 1.11	17.17 ± 0.42
	DeepWalk	$\mathbf{86.22 \pm 0.39}$	$\mathbf{47.80 \pm 0.31}$	84.15 ± 0.56	48.70 ± 0.51
	LANC	69.67 ± 0.26	26.68 ± 1.16	75.36 ± 1.17	33.68 ± 0.88
	Gcn	67.09 ± 0.95	25.12 ± 1.09	77.17 ± 0.92	31.27 ± 0.66
	Gat	66.67 ± 0.71	25.09 ± 1.24	61.96 ± 3.97	31.10 ± 3.17
	GraphSage	67.90 ± 1.40	25.87 ± 1.51	66.47 ± 0.31	30.01 ± 0.71
	H2Gcn	71.12 ± 1.26	27.45 ± 1.18	80.83 ± 0.98	37.64 ± 1.72
	GCN-LPA	70.26 ± 2.00	26.70 ± 1.47	70.58 ± 2.04	33.65 ± 2.61
	LFLF-Gcn	80.37 ± 1.43	40.12 ± 4.47	$\mathbf{87.03 \pm 1.04}$	45.07 ± 2.01
	LFLF-GraphSage	84.72 ± 0.49	45.88 ± 1.34	84.32 ± 0.35	$\mathbf{51.22 \pm 1.38}$
0.2	Mlp	68.37 ± 0.55	26.61 ± 0.47	54.47 ± 0.19	18.70 ± 0.69
	DeepWalk	$\mathbf{86.22 \pm 0.39}$	$\mathbf{47.80 \pm 0.31}$	84.15 ± 0.56	48.70 ± 0.51
	LANC	70.84 ± 1.75	27.62 ± 0.74	74.42 ± 2.02	34.24 ± 1.42
	Gcn	67.42 ± 1.07	25.31 ± 1.09	77.47 ± 0.87	31.59 ± 0.63
	Gat	67.07 ± 1.07	25.03 ± 1.16	64.55 ± 0.60	33.90 ± 0.30
	GraphSage	70.14 ± 0.60	27.99 ± 1.07	69.44 ± 1.09	32.84 ± 0.79
	H2Gcn	73.21 ± 1.07	28.85 ± 1.23	81.78 ± 1.24	40.12 ± 4.24
	GCN-LPA	69.67 ± 1.67	26.14 ± 1.53	71.02 ± 1.71	33.33 ± 1.31
	LFLF-Gcn	81.04 ± 1.55	41.45 ± 4.86	$\mathbf{85.83 \pm 1.36}$	45.86 ± 1.51
	LFLF-GraphSage	84.78 ± 0.49	45.80 ± 1.32	84.51 ± 0.43	$\mathbf{50.82 \pm 1.32}$
0.5	Mlp	68.83 ± 0.78	27.57 ± 1.10	61.63 ± 0.74	21.95 ± 0.40
	DeepWalk	$\mathbf{86.22 \pm 0.39}$	$\mathbf{47.80 \pm 0.31}$	84.15 ± 0.56	48.70 ± 0.51
	LANC	73.21 ± 2.79	30.02 ± 2.01	77.05 ± 1.16	36.45 ± 1.30
	Gcn	67.55 ± 1.08	25.39 ± 1.08	77.25 ± 0.85	31.14 ± 0.61
	Gat	67.76 ± 1.75	25.49 ± 1.46	64.40 ± 1.64	32.92 ± 0.88
	GraphSage	74.70 ± 0.49	31.25 ± 0.89	76.24 ± 0.47	37.65 ± 0.65
	H2Gcn	76.16 ± 0.47	32.85 ± 0.28	$\mathbf{84.96 \pm 0.55}$	42.70 ± 0.27
	GCN-LPA	72.11 ± 1.89	27.80 ± 0.77	74.39 ± 1.26	36.84 ± 1.02
	LFLF-Gcn	80.30 ± 0.47	40.02 ± 1.63	84.79 ± 1.82	44.88 ± 3.20
	LFLF-GraphSage	83.99 ± 0.66	44.80 ± 1.54	83.64 ± 0.30	$\mathbf{49.78 \pm 1.50}$
0.8	Mlp	70.17 ± 0.81	29.37 ± 0.99	69.53 ± 0.46	27.65 ± 0.77
	DeepWalk	$\mathbf{86.22 \pm 0.39}$	$\mathbf{47.80 \pm 0.31}$	84.15 ± 0.56	$\mathbf{48.70 \pm 0.51}$
	LANC	72.67 ± 3.56	29.67 ± 3.13	74.73 ± 1.07	35.32 ± 2.06
	Gcn	69.56 ± 1.14	25.79 ± 1.14	77.03 ± 0.99	30.10 ± 0.71
	Gat	67.93 ± 0.75	25.40 ± 1.17	66.41 ± 3.55	33.78 ± 0.96
	GraphSage	75.66 ± 0.51	32.39 ± 1.50	78.31 ± 0.49	39.25 ± 0.86
	H2Gcn	78.68 ± 0.38	36.17 ± 0.56	$\mathbf{86.01 \pm 0.64}$	44.21 ± 0.31
	GCN-LPA	72.62 ± 1.46	27.74 ± 0.67	73.99 ± 1.06	36.28 ± 1.57
	LFLF-Gcn	79.19 ± 2.29	37.76 ± 3.79	84.11 ± 1.90	43.32 ± 2.60
	LFLF-GraphSage	83.78 ± 0.60	44.43 ± 0.68	82.99 ± 0.32	48.25 ± 1.25
1.0	Mlp	72.90 ± 0.27	31.52 ± 0.52	75.23 ± 0.63	34.29 ± 0.07
	DeepWalk	$\mathbf{86.22 \pm 0.39}$	$\mathbf{47.80 \pm 0.31}$	84.15 ± 0.56	$\mathbf{48.70 \pm 0.51}$
	LANC	74.13 ± 1.61	30.38 ± 1.29	75.26 ± 0.93	37.47 ± 0.27
	Gcn	70.74 ± 0.80	26.57 ± 0.97	79.22 ± 0.82	33.70 ± 0.79
	Gat	70.22 ± 1.27	26.35 ± 0.93	66.52 ± 1.03	36.01 ± 0.72
	GraphSage	78.71 ± 0.59	35.77 ± 1.40	80.77 ± 0.17	43.05 ± 0.22
	H2Gcn	79.97 ± 0.19	38.73 ± 0.91	$\mathbf{87.09 \pm 0.12}$	46.71 ± 0.37
	GCN-LPA	76.28 ± 1.27	31.91 ± 1.07	76.38 ± 0.36	39.10 ± 2.03
	LFLF-Gcn	78.52 ± 2.41	38.23 ± 2.27	83.90 ± 2.21	43.09 ± 1.62
	LFLF-GraphSage	83.70 ± 0.55	44.13 ± 0.74	83.07 ± 0.55	47.90 ± 0.79

Table 12: Multi-label Node Classification results on Synthetic dataset with varying label homophily. All results are the mean of three random splits. The label homophiles (rounded up) are $[0.2, 0.4, 0.6, 0.8, 1.0]$

Label Homophily	**Method**	Micro-F1	Macro-F1	AUC-ROC	AP
0.2	Mlp	72.90 ± 0.27	31.52 ± 0.52	$\mathbf{75.23 \pm 0.63}$	34.29 ± 0.07
	DeepWalk	66.62 ± 0.62	26.02 ± 0.55	52.19 ± 0.91	18.07 ± 0.71
	LANC	67.05 ± 0.92	25.12 ± 2.20	54.49 ± 0.53	18.95 ± 0.43
	Gcn	67.28 ± 0.67	25.21 ± 1.04	66.59 ± 0.58	26.06 ± 1.07
	Gat	65.09 ± 0.89	24.61 ± 1.41	51.43 ± 0.33	17.05 ± 0.36
	GraphSage	73.57 ± 0.20	31.22 ± 0.65	71.72 ± 0.30	28.94 ± 1.09
	H2Gcn	$\mathbf{73.80 \pm 0.59}$	$\mathbf{31.86 \pm 0.89}$	73.24 ± 0.08	29.69 ± 0.53
	GCN-LPA	67.02 ± 0.89	25.24 ± 1.31	49.92 ± 0.93	17.02 ± 0.31
	LFLF-Gcn	68.24 ± 1.90	25.73 ± 1.58	72.49 ± 1.73	35.46 ± 1.82
	LFLF-GraphSage	67.87 ± 1.77	25.40 ± 1.23	70.26 ± 0.54	$\mathbf{35.62 \pm 1.84}$
0.4	Mlp	72.90 ± 0.27	31.52 ± 0.52	75.23 ± 0.63	34.29 ± 0.07
	DeepWalk	$\mathbf{88.79 \pm 0.20}$	$\mathbf{54.74 \pm 1.73}$	85.74 ± 0.33	$\mathbf{52.15 \pm 1.41}$
	LANC	75.16 ± 0.77	33.69 ± 0.94	76.53 ± 0.86	37.99 ± 0.74
	Gcn	72.16 ± 1.70	26.95 ± 1.18	80.19 ± 0.75	34.29 ± 0.88
	Gat	71.50 ± 1.30	27.58 ± 1.11	69.96 ± 1.69	35.86 ± 0.59
	GraphSage	79.09 ± 0.33	36.05 ± 1.56	81.00 ± 0.35	42.57 ± 0.49
	H2Gcn	81.30 ± 0.25	40.54 ± 0.74	$\mathbf{87.91 \pm 0.04}$	48.36 ± 0.44
	GCN-LPA	78.11 ± 1.38	32.71 ± 1.47	78.00 ± 0.70	40.80 ± 0.20
	LFLF-Gcn	82.94 ± 0.44	38.47 ± 0.43	85.44 ± 0.71	43.77 ± 1.08
	LFLF-GraphSage	85.14 ± 0.60	45.69 ± 0.57	82.55 ± 0.66	48.21 ± 0.03
0.6	Mlp	72.90 ± 0.27	31.52 ± 0.52	75.23 ± 0.63	34.29 ± 0.07
	DeepWalk	$\mathbf{95.94 \pm 0.05}$	$\mathbf{82.58 \pm 0.19}$	$\mathbf{95.32 \pm 0.80}$	$\mathbf{81.34 \pm 0.95}$
	LANC	80.36 ± 1.81	39.65 ± 3.14	81.85 ± 2.66	43.42 ± 3.61
	Gcn	75.47 ± 0.46	29.43 ± 0.43	84.08 ± 0.72	38.78 ± 0.33
	Gat	75.13 ± 0.39	29.26 ± 0.56	72.85 ± 0.79	39.01 ± 0.23
	GraphSage	82.46 ± 0.67	40.46 ± 1.67	83.24 ± 0.89	45.79 ± 0.91
	H2Gcn	83.40 ± 1.94	44.32 ± 1.59	89.98 ± 0.86	51.19 ± 1.92
	GCN-LPA	85.45 ± 2.42	47.67 ± 4.72	83.96 ± 2.22	49.54 ± 3.54
	LFLF-Gcn	86.88 ± 0.22	47.92 ± 1.67	88.72 ± 0.66	50.88 ± 2.06
	LFLF-GraphSage	91.36 ± 0.56	62.40 ± 0.49	88.53 ± 1.39	59.18 ± 1.02
0.8	Mlp	72.90 ± 0.27	31.52 ± 0.52	75.23 ± 0.63	34.29 ± 0.07
	DeepWalk	96.53 ± 0.61	89.25 ± 1.96	95.81 ± 0.47	86.93 ± 1.92
	LANC	79.35 ± 0.97	46.03 ± 2.24	87.75 ± 0.60	48.10 ± 1.17
	Gcn	80.72 ± 0.55	36.60 ± 1.31	86.82 ± 0.62	44.98 ± 0.40
	Gat	77.35 ± 0.49	31.63 ± 1.10	83.10 ± 0.69	42.82 ± 0.86
	GraphSage	84.22 ± 0.38	48.16 ± 1.42	87.37 ± 0.34	53.34 ± 0.84
	H2Gcn	86.00 ± 2.63	55.85 ± 4.92	91.61 ± 1.64	57.17 ± 2.57
	GCN-LPA	88.96 ± 0.68	64.09 ± 4.18	89.53 ± 1.07	60.44 ± 4.41
	LFLF-Gcn	89.04 ± 0.76	61.11 ± 4.28	93.29 ± 0.91	59.99 ± 3.13
	LFLF-GraphSage	$\mathbf{96.97 \pm 0.66}$	$\mathbf{89.73 \pm 2.86}$	$\mathbf{97.82 \pm 0.08}$	$\mathbf{89.14 \pm 3.27}$
1.0	Mlp	72.90 ± 0.27	31.52 ± 0.52	75.23 ± 0.63	34.29 ± 0.07
	DeepWalk	80.25 ± 0.11	62.80 ± 0.46	83.83 ± 1.08	55.16 ± 0.67
	LANC	83.37 ± 0.96	60.77 ± 3.51	92.53 ± 1.27	62.85 ± 3.01
	Gcn	81.61 ± 0.25	42.19 ± 0.45	86.48 ± 0.31	49.32 ± 0.92
	Gat	79.37 ± 0.64	34.67 ± 1.87	87.20 ± 2.15	43.90 ± 2.90
	GraphSage	82.15 ± 0.49	46.06 ± 0.99	89.73 ± 0.45	55.25 ± 0.42
	H2Gcn	91.59 ± 1.74	76.09 ± 5.96	93.94 ± 1.03	65.20 ± 5.71
	GCN-LPA	86.92 ± 1.13	60.71 ± 3.01	90.75 ± 0.46	58.28 ± 3.02
	LFLF-Gcn	88.42 ± 0.66	71.09 ± 3.86	96.02 ± 0.86	75.21 ± 4.06
	LFLF-GraphSage	$\mathbf{93.53 \pm 0.43}$	$\mathbf{86.57 \pm 0.87}$	$\mathbf{98.29 \pm 0.52}$	$\mathbf{90.55 \pm 1.73}$

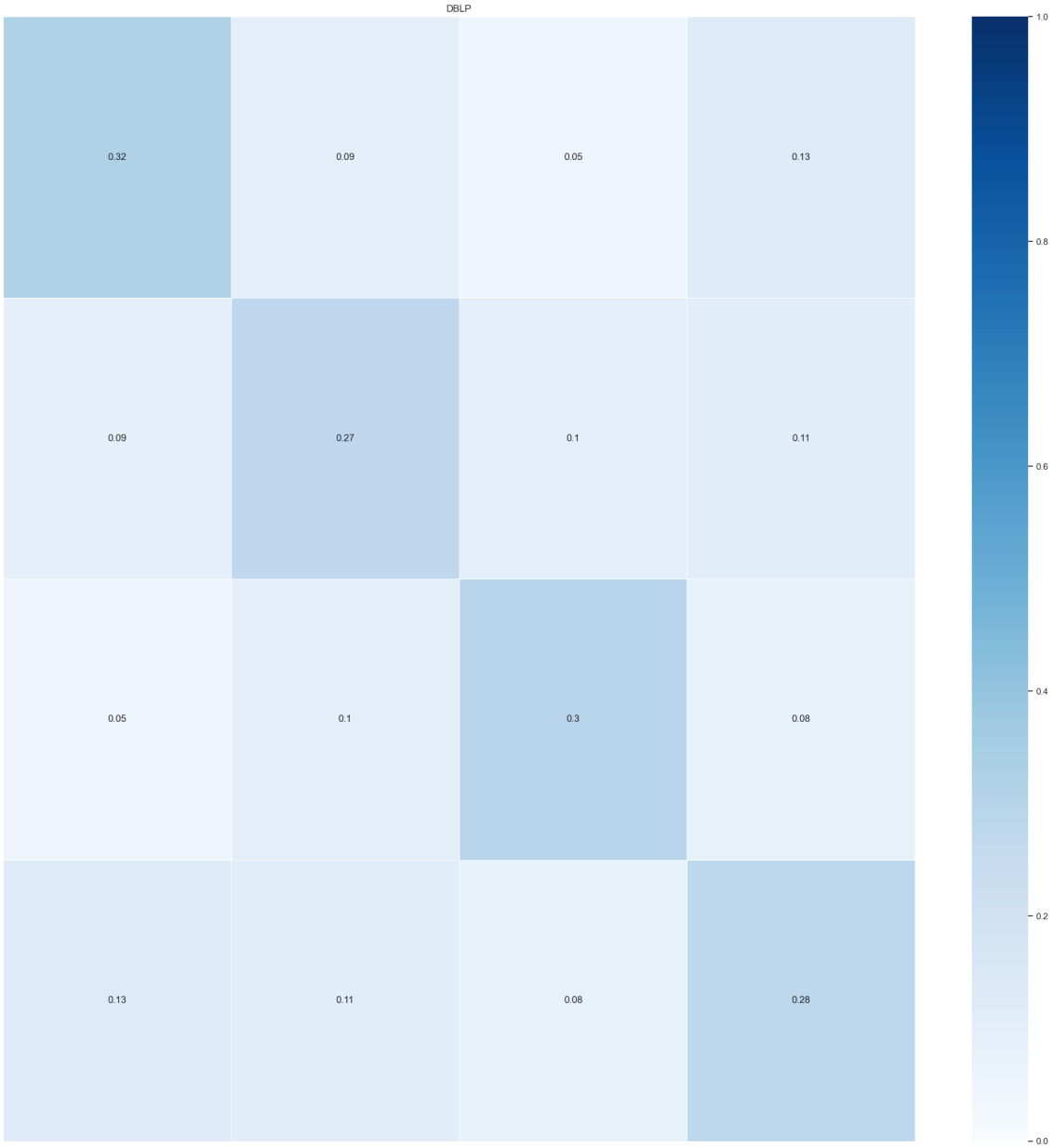


Figure 9: Cross class Neighborhood Similarity in DBLP

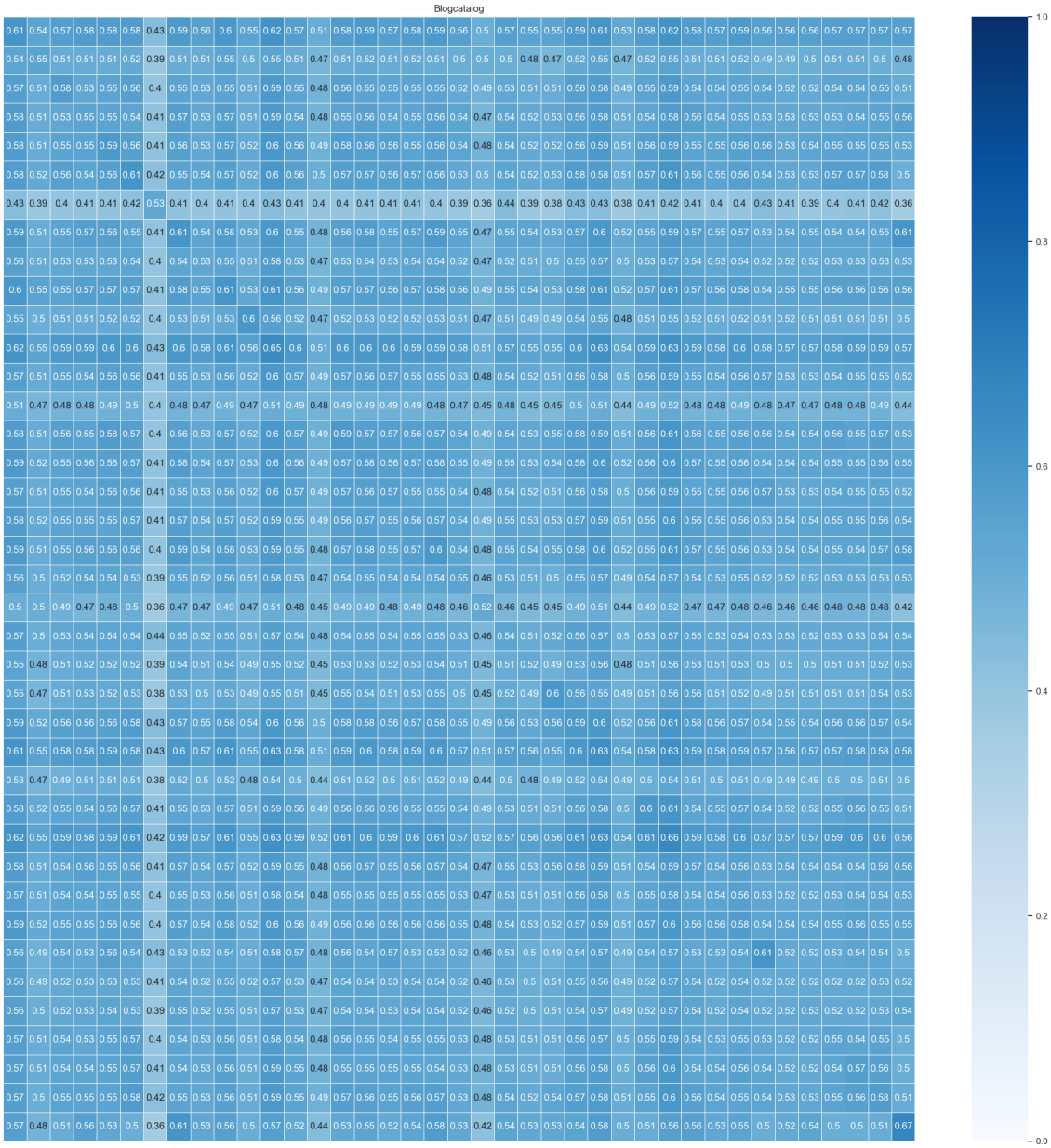


Figure 10: Cross class Neighborhood Similarity in BlogCat

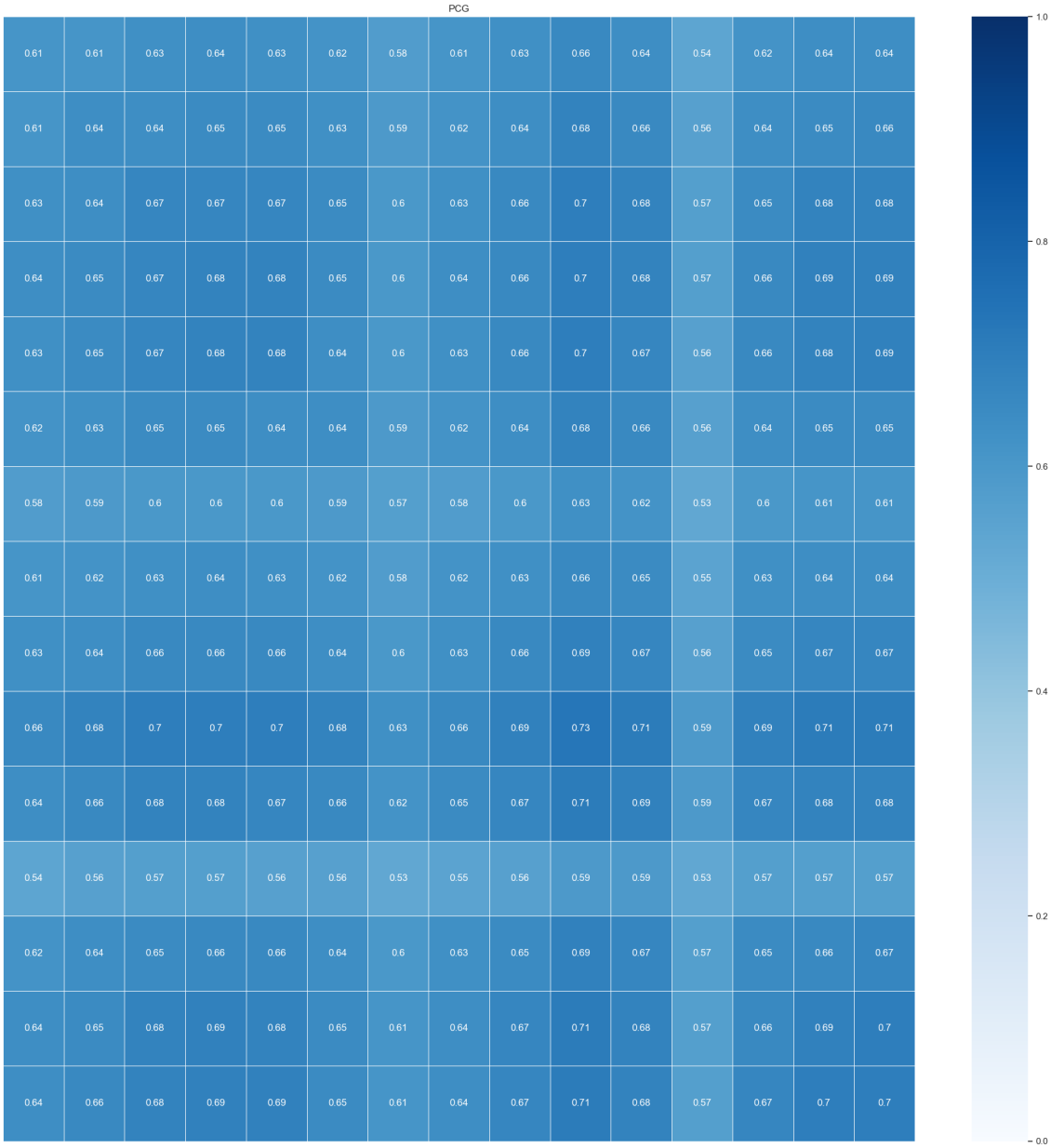


Figure 11: Cross class Neighborhood Similarity in PCG

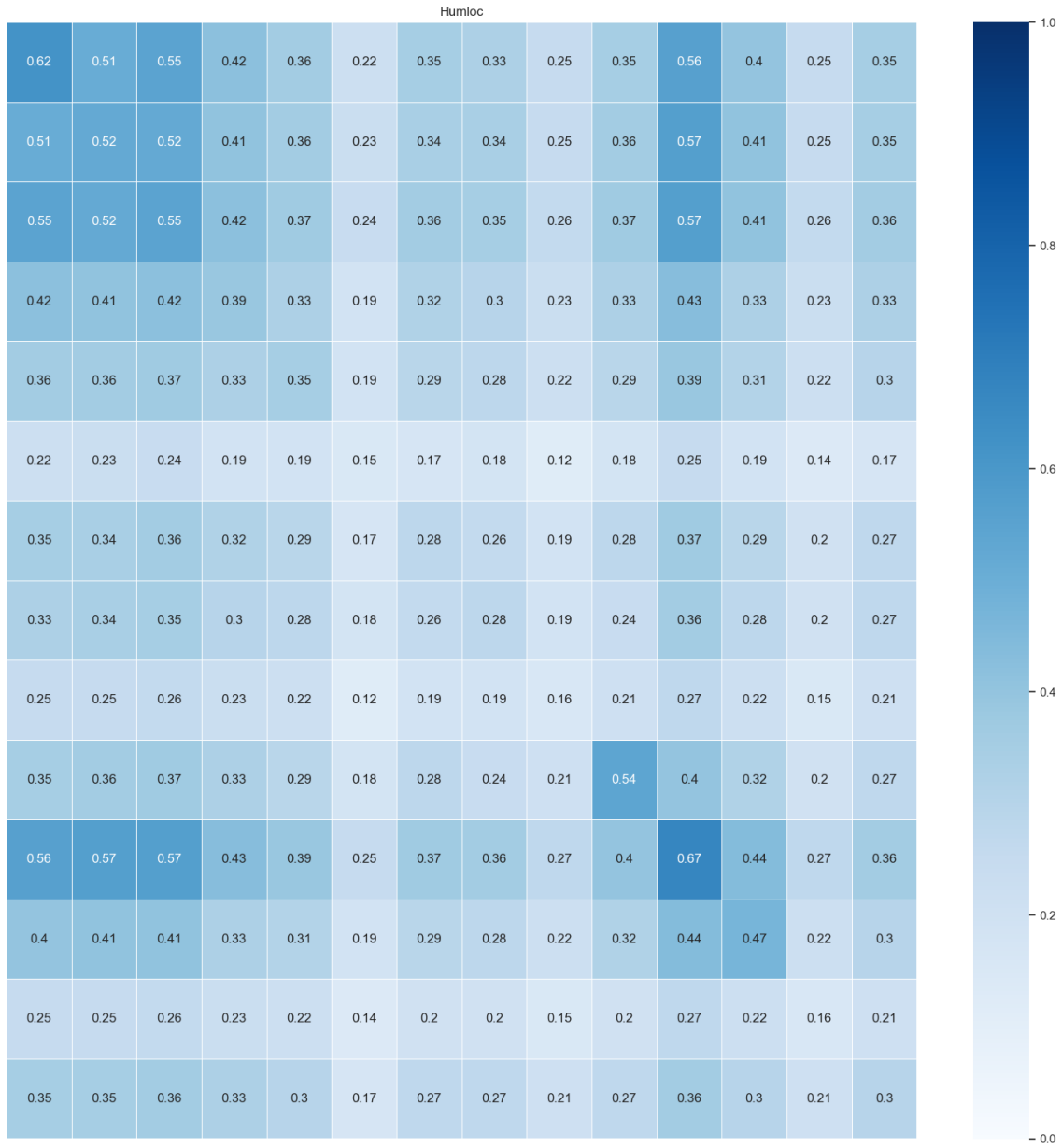


Figure 12: Cross class Neighborhood Similarity in HumLoc

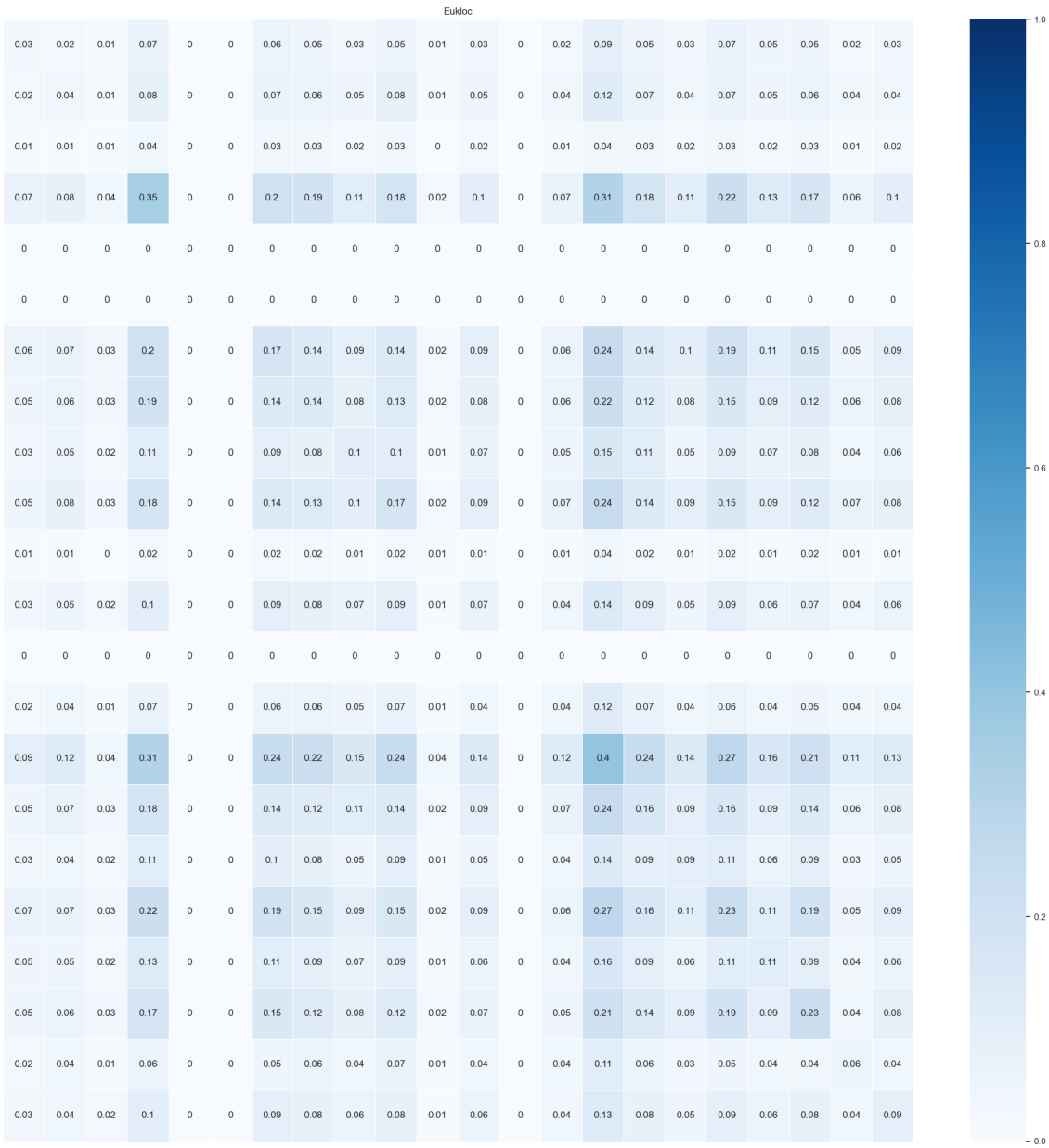


Figure 13: Cross class Neighborhood Similarity in EukLoc

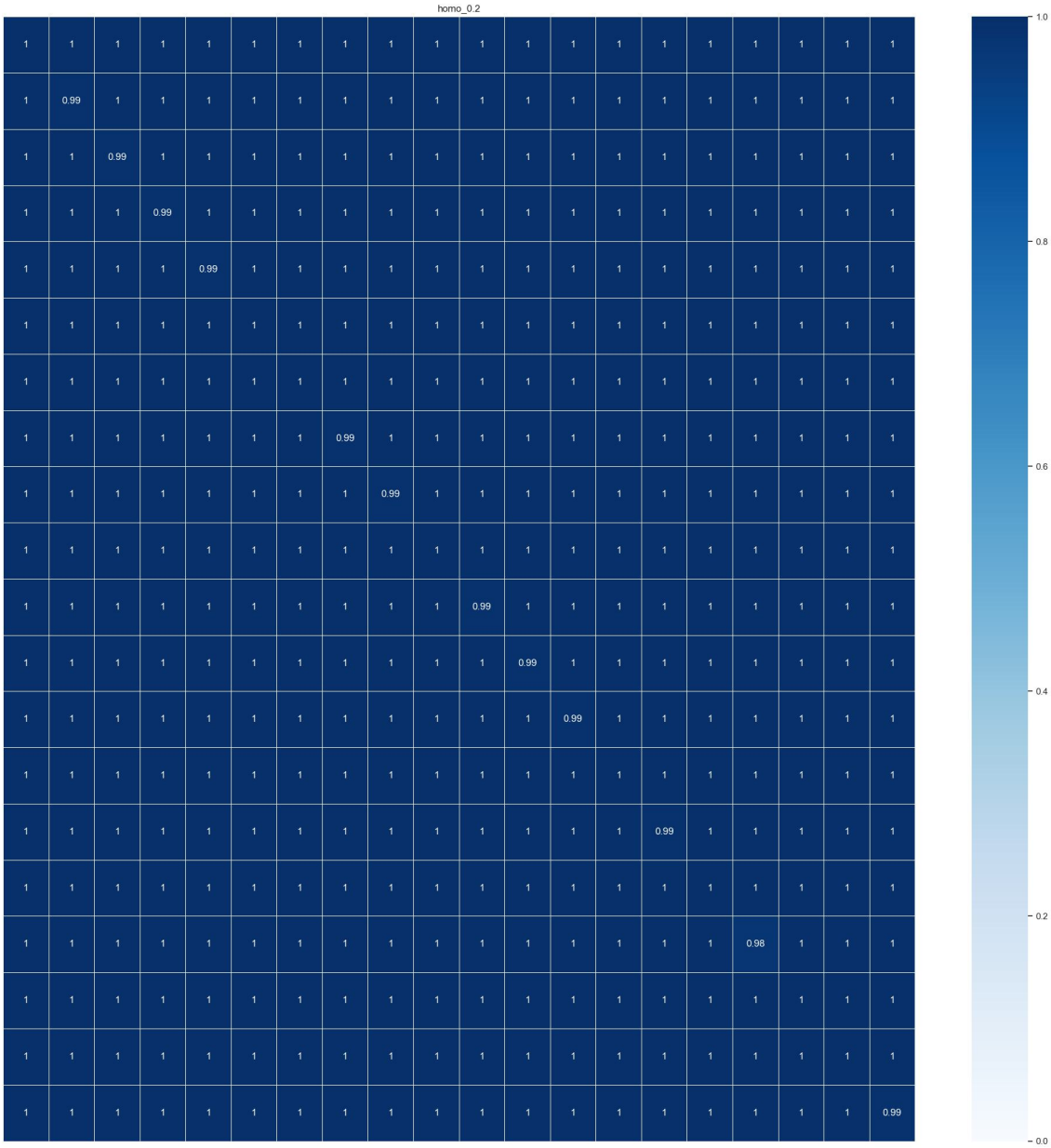


Figure 14: Cross-class Neighborhood Similarity in graph label homophily=0.2

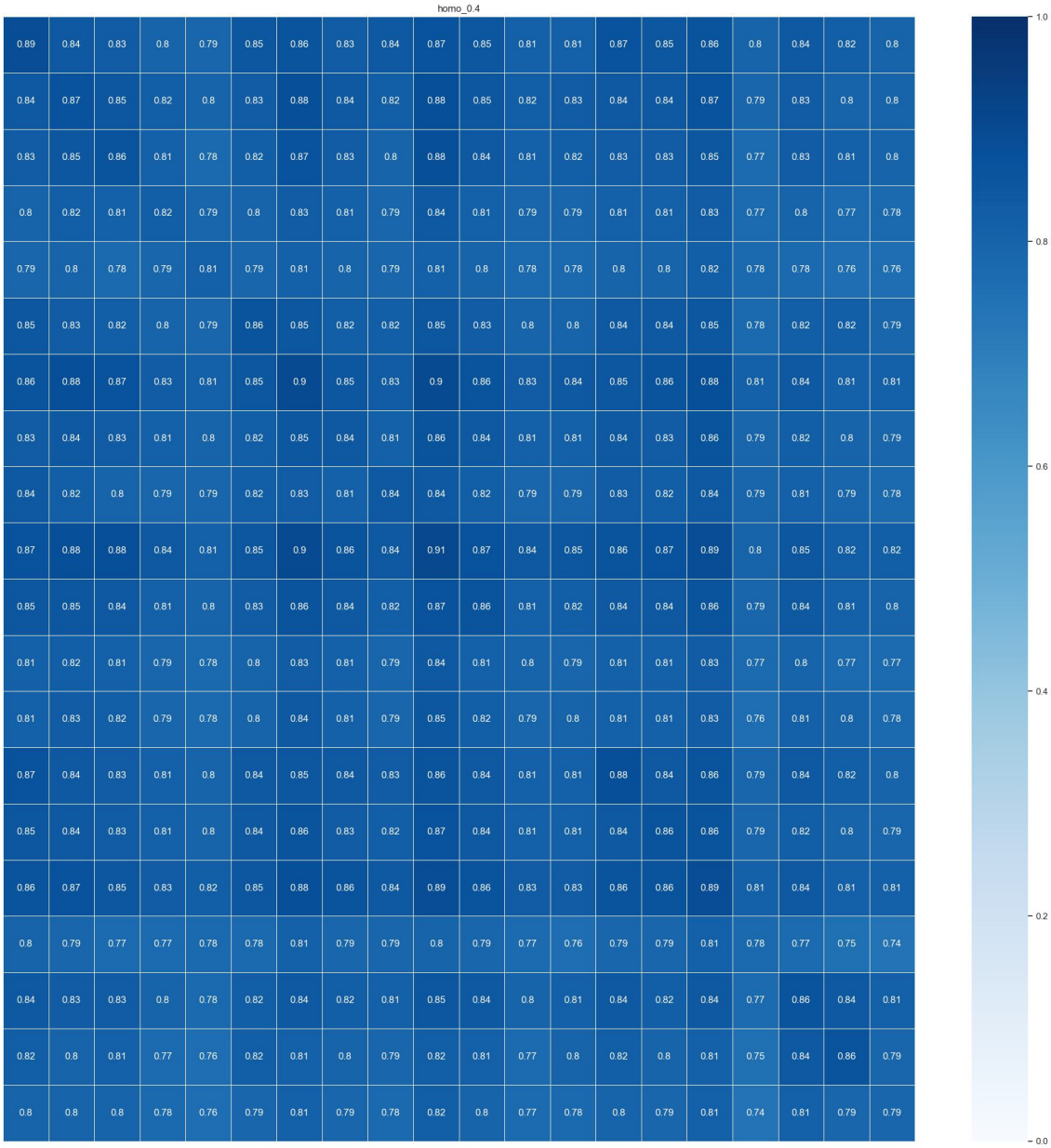


Figure 15: Cross-class Neighborhood Similarity in graph label homophily=0.4

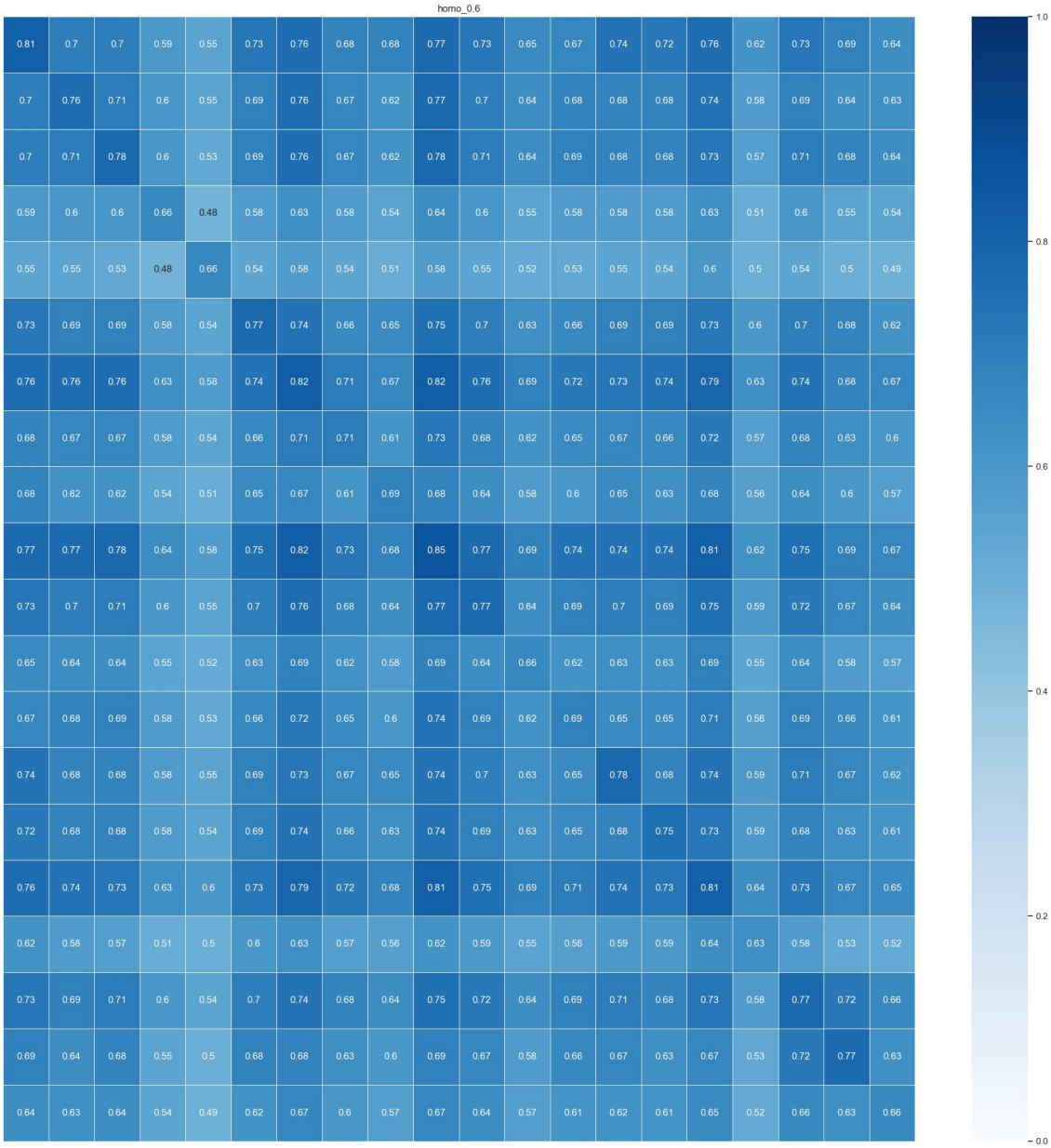


Figure 16: Cross-class Neighborhood Similarity in graph label homophily=0.6

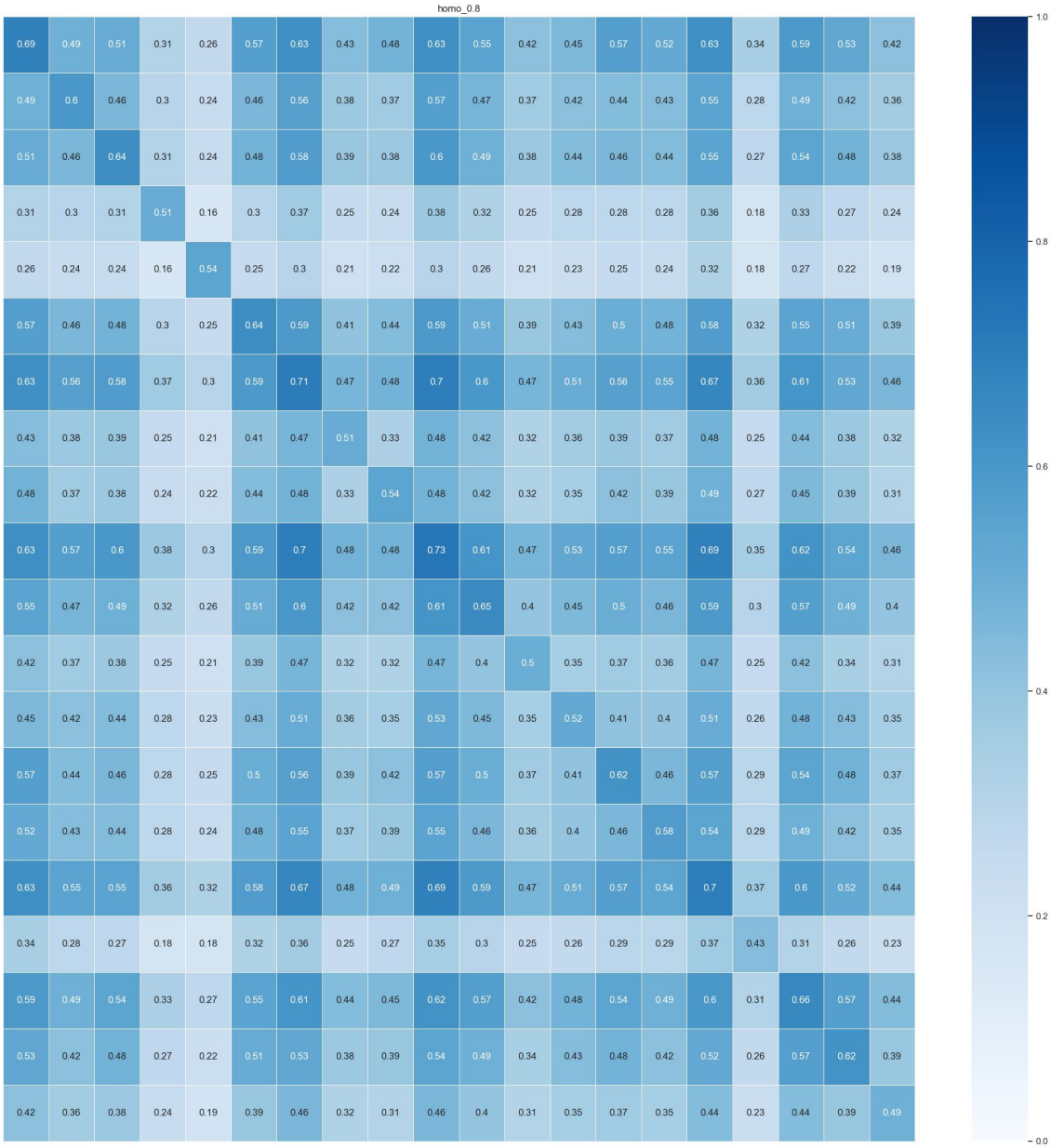


Figure 17: Cross-class Neighborhood Similarity in graph label homophily=0.8

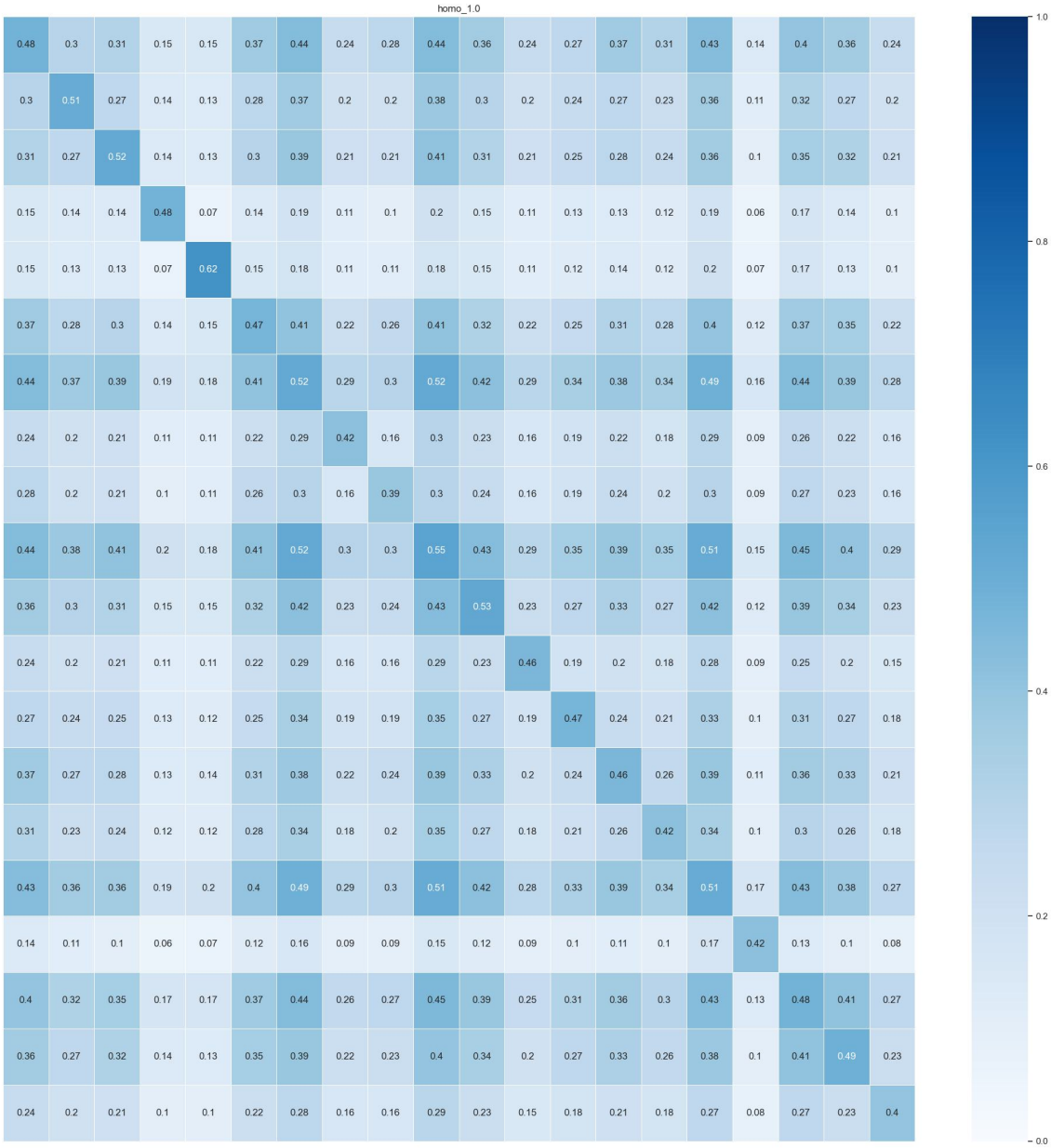


Figure 18: Cross-class Neighborhood Similarity in graph label homophily=1.0