A CRITICAL LOOK AT EVALUATION OF GNNS UNDER HETEROPHILY: ARE WE REALLY MAKING PROGRESS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Node classification is a classical graph representation learning task on which Graph Neural Networks (GNNs) have recently achieved strong results. However, it is often believed that standard GNNs only work well for *homophilous* graphs, i.e., graphs where edges tend to connect nodes of the same class. Graphs without this property are called *heterophilous*, and it is typically considered that specialized methods are required to achieve strong performance on such graphs. Many GNN models for heterophilous graphs have recently been proposed in the literature. However, these models are typically evaluated on the same set of six heterophilous datasets. In this work, we challenge this evaluation setting. First, we show that the popular heterophilous benchmarks have serious drawbacks, the most significant being a large number of duplicate nodes in `squirrel` and `chameleon` datasets, which leads to train-test data leakage. We show that removing duplicate nodes strongly affects GNN performance on these datasets. Then, we propose a set of heterophilous graphs of varying properties that we believe can serve as a better benchmark for testing the performance of GNNs under heterophily. We show that, at this moment, standard GNNs achieve competitive results on heterophilous graphs outperforming most of the specialized models.

1 INTRODUCTION

The field of machine learning on graph-structured data has recently attracted a lot of attention, with Graph Neural Networks (GNNs) proving to be a particularly powerful graph representation learning method. Thus, using GNNs has become a de-facto standard approach to graph machine learning, and a lot of versions of GNNs have been proposed in the literature Kipf & Welling (2017); Hamilton et al. (2017); Veličković et al. (2017); Xu et al. (2018), most of them falling under a general Message Passing Neural Networks (MPNNs) framework Gilmer et al. (2017). MPNNs learn node representations by an iterative neighborhood-aggregation process, where each layer updates each node's representation by combining previous-layer representations of the node itself and its neighbors. The node feature vector is used as the initial node representation. Thus, MPNNs combine node features with graph topology, allowing them to learn complex dependencies between nodes.

In many real-world networks, edges tend to connect similar nodes. This property is called *homophily*. Typical examples of homophilous networks are social networks, where users tend to connect to users with similar interests, and citation networks, where papers mostly cite works from the same research area. The opposite of homophily is called *heterophily*: this property describes the preference of network nodes to connect to nodes not similar to them. For example, in financial transaction networks, fraudsters often perform transactions with non-fraudulent users, and in dating networks, most connections are between people of opposite genders.

Early works on GNNs mostly evaluated their models on homophilous graphs. This has led to claims that GNNs implicitly use the homophily of a graph and are thus not suitable for heterophilous graphs Zhu et al. (2021; 2020); He et al. (2022); Wang et al. (2022). Recently, many works have proposed new GNN models specifically designed for heterophilous graphs that are claimed to outperform standard GNNs. However, these models are typically evaluated on the same six heterophilous graphs introduced in Pei et al. (2020). In this work, we challenge this evaluation setting. We highlight several downsides of the standard heterophilous datasets, such as low diversity, small size, extreme class imbalance of some datasets, and, most importantly, the presence of a large number of

duplicate nodes in squirrel and chameleon datasets. We show that models rely on duplicated nodes to achieve strong results, and removing these nodes significantly affects their performance. In particular, standard GNNs achieve very strong results on the cleaned-up datasets outperforming all or almost all heterophily-specific methods.

Motivated by the shortcomings of currently used heterophilous benchmarks, we collect a set of diverse heterophilous graphs and propose to use them as a better benchmark. The proposed datasets come from different domains, all of them have a low value of homophily, and we also verify that they have diverse structural properties. We evaluate a wide range of GNNs, both standard, and heterophily-specific, on the proposed benchmark. In doing so, we uncover that the standard baselines usually outperform heterophily-specific models. Thus, the progress in learning under heterophily can be limited to the standard datasets used for their evaluation. Our results also show that there is, however, a trick that is useful for learning on heterophilous graphs — separating ego- and neighbor-embeddings, which was proposed in Zhu et al. (2020). This trick consistently improves the baselines (such as GAT or Graph Transformer) and allows to achieve the best results. We hope that the proposed benchmark will be useful for further progress in learning under heterophily.

2 Related work

Measuring homophily. While much effort has been put into developing graph representation learning methods for heterophilous graphs, there is no universally agreed upon measure of homophily. Homophily measures typically used in the literature are *edge homophily* Abu-El-Haija et al. (2019); Zhu et al. (2020), which is simply the fraction of edges that connect nodes of the same class, and *node homophily* Pei et al. (2020), which computes the proportion of neighbors that have the same class for all nodes and then averages these values across all nodes. These two measures are simple and intuitive; however, as shown in Lim et al. (2021); Platonov et al. (2022), they are sensitive to the number of classes and their balance, which makes these measures hard to interpret and compare across different datasets. To fix these issues, Lim et al. (2021) propose another homophily measure. However, Platonov et al. (2022) show that it also can provide unreliable results. To solve the issues with existing measures, Platonov et al. (2022) propose *adjusted homophily* which corrects the number of edges between classes by their expected value. Thus, adjusted homophily becomes insensitive to the number of classes and their balance. Platonov et al. (2022) show that adjusted homophily satisfies a number of desirable properties, which makes it appropriate for comparing homophily levels between different datasets. Thus, in our work, we will use adjusted homophily for measuring homophily of graphs.

Graph datasets. Early works on GNNs mostly evaluated their models on highly homophilous graphs. The most popular of them are three citation networks: cora, citeseer, and pubmed Giles et al. (1998); McCallum et al. (2000); Namata et al. (2012); Sen et al. (2008); Yang et al. (2016). Examples of other graph datasets for node classification that appear in the literature include citation networks coauthor-cs, coauthor-physics from Shchur et al. (2018), co-purchasing networks amazon-computers, amazon-photo also from Shchur et al. (2018), discussion network reddit from Hamilton et al. (2017). All of these datasets also have high levels of homophily. Recently, Open Graph Benchmark Hu et al. (2020) was proposed in order to provide challenging large-scale graphs for evaluating GNN performance, however, the proposed datasets such as ogbn-arxiv, ogbn-products, ogbn-papers100M are also highly homophilous Zhu et al. (2020); Platonov et al. (2022).

As for heterophilous graphs, the datasets used in most studies dedicated to learning under heterophily are limited to the six graphs proposed in Pei et al. (2020): squirrel, chameleon, actor, texas, cornell, and wisconsin. These graphs have become the de-facto standard benchmark for evaluating heterophily-specific models and were used in Zhu et al. (2021; 2020); Chien et al. (2020); Yan et al. (2021); Maurya et al. (2022); Li et al. (2022); He et al. (2022); Wang & Zhang (2022); Wang et al. (2022); Du et al. (2022); Suresh et al. (2021); Bo et al. (2021); Ma et al. (2021); Luan et al. (2021); Bodnar et al. (2022). We further discuss these datasets in Section 3. Recently, a set of large-scale heterophilous graph datasets has been proposed in Lim et al. (2021). However, due to their size, these datasets are mostly suitable for evaluating scalable graph methods rather than GNNs and thus have not seen wide adoption in the GNN community yet.

Specialized methods for learning under heterophily. Many methods designed for achieving good results either specifically under heterophily or in both homophily and heterophily settings have been recently proposed. In this paragraph, we briefly describe some of these methods. Pei et al. (2020) were the first to attract attention to learning in the heterophily setting. Their proposed approach (Geom-GCN) precomputes unsupervised node embeddings and defines convolution in the latent space of these embeddings. Zhu et al. (2020) is also one of the first works to consider the problem of learning under heterophily. The authors identified three designs in existing GNNs that allow the models to generalize to the heterophily setting: ego- and neighbor-embedding separation, aggregation across higher-order neighborhoods, and combining intermediate representations from different layers for final node representation. Zhu et al. (2021) further proposed a new architecture CPGNN that incorporates a learnable class compatibility matrix in the GNN aggregation step that can model different levels of homophily. Chien et al. (2020) proposed a Generalized PageRank-inspired architecture (GPR-GNN) with learnable weights designed to adapt to various node label patterns. Yan et al. (2021) related learning under heterophily to the problem of oversmoothing and proposed two modifications to the GCN architecture: degree corrections and signed messages. Maurya et al. (2022) proposed decoupling feature aggregation from GNN layers and using soft feature selection. Li et al. (2022) proposed GloGNN and GloGNN++ models that aggregate information from global nodes in the graph. He et al. (2022) proposed block-modeling guided GNN architecture that can learn different aggregation rules for different nodes. Wang & Zhang (2022) proposed JacobiConv — a spectral GNN that is supposed to achieve strong results on both homophilous and heterophilous graphs. Wang et al. (2022) designed a new propagation mechanism that can adaptively change propagation and aggregation for different nodes. Du et al. (2022) proposed GBK-GNN that uses bi-kernel feature transformation and a selection gate to capture useful information in both homophily and heterophily settings. Suresh et al. (2021) proposed transforming the input graph into a computation graph based on both proximity and structural information. Bo et al. (2021) proposed a self-gating mechanism that allows their model to adaptively integrate both low-frequency and high-frequency signals. Luan et al. (2021) introduced the Adaptive Channel Mixing (ACM) framework to address those cases of heterophily that are harmful for GNN performance. Bodnar et al. (2022) proposed neural sheaf diffusion models that learn cellular sheaves from data to achieve strong results on heterophilous graphs.

Performance of standard GNNs under heterophily. While it is widely considered that standard GNNs do not perform well under heterophily Zhu et al. (2021; 2020); He et al. (2022); Wang et al. (2022), recently, there have been several works that show that standard GNNs can achieve strong results on some heterophilous graphs Ma et al. (2021); Luan et al. (2021). However, these results were primarily obtained on synthetic or semi-synthetic datasets. Platonov et al. (2022) explain these observations by high *label informativeness* of the considered graphs: mutual information between neighbors' labels can be high even when these neighbors have different labels. In contrast, we show that standard GNNs often achieve strong results and outperform specialized methods on real-world graphs with low label informativeness.

3 Issues with popular heterophilous datasets

In this section, we revisit datasets commonly used for heterophilous node classification. As discussed in Section 2, there are six datasets that are the most popular: Wikipedia networks `squirrel` and `chameleon`, actor co-occurrence in Wikipedia pages (`actor`), and WebKB datasets `texas`, `wisconsin`, and `cornell`. The standard preprocessing of these datasets is done by Pei et al. (2020). First, we note that these datasets only come from three sources; thus, they do not provide sufficient coverage of different heterophilous patterns that can be found in real data, and more diverse datasets are required for a fair evaluation of models under heterophily. However, that is not the only problem with this benchmark. In this section, we show that some of these datasets have certain drawbacks that may highly affect the evaluation results.

3.1 Squirrel and Chameleon

These datasets are initially collected by Rozemberczki et al. (2021): nodes represent articles from the English Wikipedia (December 2018), and edges reflect mutual links between them. Node features

indicate the presence of particular nouns in the articles. Nodes are grouped by Pei et al. (2020) into five categories based on the average monthly traffic of the web page.

While analyzing these datasets, we noticed many groups of nodes with exactly the same neighborhood and the same label. For instance, in `squirrel`, there is a group of 48 nodes that all have the same 15 neighbors and the same label. In `chameleon`, there is a group of 92 nodes with the same 18 neighbors and the same label. For brevity, we further call such nodes 'duplicates'. Since duplicates from the same group appear in the train, validation, and test parts, they create a train-test data leakage: for duplicates from the test set, their labels can be predicted by simply matching the node's neighborhood to neighborhoods of train nodes. We further show that removing this leakage strongly affects the performance of GNNs.

If we look at the original dataset from Rozemberczki et al. (2021), we see that all duplicates from the same group have exactly the same monthly traffic, which seems suspicious. Since the source code for collecting the datasets is unavailable, we can only hypothesize that such duplicates can correspond to previous versions of the same Wikipedia page. This hypothesis agrees with the following observations: 1) the duplicate nodes may have different features, 2) all the edges of such duplicates are outgoing, 3) for (almost) each such group of duplicates, there is a unique node in the dataset with the same outgoing edges and the same monthly traffic, but with some additional incoming edges. The latter observation may mean that there is an 'actual' version of the article that, in contrast to older ones, gets all incoming links. There are a few exceptions for the latter observation, which may mean that some pages are not cited by other Wikipedia articles in the dataset, i.e., they are 'actual' versions of zero incoming degrees.

Since all the duplicates do not have any incoming edges, we performed a simple filtering of `squirrel` and `chameleon` by removing all the nodes without incoming edges. While this procedure can potentially remove some 'actual' not cited articles, we checked that the number of such nodes is only 18 for `squirrel` (0.35% of the original dataset) and 26 for `chameleon` (1.1%). We provide the statistics of `squirrel` and `chameleon` datasets before and after filtering in Table 6 in Appendix B.

Table 1 shows the number of nodes in original `squirrel` and `chameleon`, as well as the number of duplicates and non-duplicates in these datasets. We see that the duplicates constitute more than half of each dataset. In the same table, we report the accuracy of GraphSAGE Hamilton et al. (2017) on duplicates and non-duplicates separately. We can see a significant difference in performance on these two types of nodes, confirming that the model implicitly relies on data leakage to make predictions. We additionally show the distribution of duplicates across classes in Table 5 in Appendix B.

Table 1: Duplicates in Wikipedia datasets

	squirrel	chameleon
number of nodes	5201	2277
number of duplicates	2996	1413
number of non-duplicates	2205	864
GraphSAGE accuracy		
on duplicates	51.21 ± 01.82	73.36 ± 02.54
on non-duplicates	36.31 ± 02.11	45.72 ± 03.69

We further evaluate several models on the original and filtered datasets; see Table 2 for the results. We refer to Section 5.1 and Appendix A for the description of the models and evaluation setup. First, we see a significant performance drop for all models except for ResNet on `squirrel` (this exception is not surprising — ResNet is a graph-agnostic model and therefore cannot utilize data leakage based on node neighborhoods). This performance drop confirms that the models implicitly rely on leaked data to achieve strong results on unfiltered datasets. Moreover, we see that the exact drop in performance significantly differs between models, and thus the ranking of models on the filtered datasets is very different from the ranking on the original datasets. This suggests that different models have different capacity to utilize data leakage. To better illustrate the difference in rankings, we report model ranks on both original and filtered datasets in Table 2. Some models have particularly strong performance changes. For example, FSGNN is the best model by a significant margin on both original datasets; however, on filtered `squirrel` and `chameleon` it achieves only 12th and 13th places, respectively. Such a substantial shake-up raises concerns about the validity of conclusions made in previous works that rely on analyzing the performance of different models on these datasets.

Table 2: Accuracy of models on original and filtered `squirrel` and `chameleon`. The column 'ranks' reports the positions in the ranked list of models on the original and filtered datasets.

	squirrel			chameleon		
	accuracy on original dataset	accuracy on filtered dataset	ranks	accuracy on original dataset	accuracy on filtered dataset	ranks
ResNet	34.60 ± 1.26	37.26 ± 1.49	18 / 6	47.37 ± 2.41	37.56 ± 3.09	20 / 14
ResNet+SGC	35.40 ± 1.45	37.23 ± 2.76	17 / 7	53.55 ± 1.54	39.22 ± 3.91	18 / 9
ResNet+adj	59.82 ± 1.54	39.88 ± 2.53	3 / 2	68.18 ± 1.33	40.02 ± 2.35	6 / 6
GCN	57.08 ± 1.20	40.70 ± 1.35	6 / 1	68.00 ± 2.41	41.44 ± 4.09	7 / 5
GraphSAGE	44.87 ± 1.27	36.46 ± 2.16	10 / 10	63.07 ± 2.19	39.11 ± 5.05	9 / 10
GAT	62.34 ± 1.42	35.54 ± 1.99	2 / 15	69.14 ± 2.10	41.69 ± 4.20	5 / 3
GAT-sep	45.85 ± 1.78	35.81 ± 2.11	9 / 14	62.57 ± 1.93	38.90 ± 3.40	11 / 11
GT	58.10 ± 1.55	36.40 ± 2.38	4 / 11	69.96 ± 1.91	39.76 ± 4.92	2 / 7
GT-sep	44.70 ± 1.31	36.95 ± 1.72	11 / 8	62.17 ± 1.98	39.67 ± 3.61	12 / 8
Geom-GCN-S	36.30 ± 1.11	36.87 ± 2.05	16 / 9	60.28 ± 1.84	37.40 ± 2.06	14 / 16
Geom-GCN-I	33.10 ± 1.02	38.15 ± 2.39	19 / 4	59.93 ± 1.47	37.40 ± 3.23	15 / 15
Geom-GCN-P	38.17 ± 1.28	36.16 ± 1.42	13 / 13	60.88 ± 2.41	35.91 ± 2.65	13 / 17
H_2GCN	36.44 ± 1.81	34.74 ± 1.11	15 / 16	57.11 ± 1.49	35.20 ± 3.08	16 / 18
CPGNN	29.29 ± 1.06	32.50 ± 1.65	20 / 19	55.59 ± 2.13	34.63 ± 2.74	17 / 20
GPR-GNN	50.35 ± 1.80	33.57 ± 3.21	8 / 18	69.52 ± 1.77	34.89 ± 5.45	4 / 19
FSGNN	72.88 ± 1.66	36.24 ± 1.47	1 / 12	77.17 ± 1.34	38.14 ± 2.80	1 / 13
GloGNN	57.54 ± 1.39	30.30 ± 1.92	5 / 20	69.78 ± 2.42	38.43 ± 3.74	3 / 12
FAGCN	38.11 ± 1.66	34.12 ± 1.19	14 / 17	62.81 ± 1.86	48.38 ± 1.78	10 / 1
GBK-GNN	38.47 ± 1.09	37.91 ± 1.22	12 / 5	52.08 ± 1.71	41.57 ± 2.76	19 / 4
JacobiConv	50.79 ± 2.29	39.43 ± 1.95	7 / 3	67.74 ± 2.22	45.21 ± 1.59	8 / 2

3.2 Cornell, Texas, Wisconsin

`Cornell`, `texas`, and `wisconsin` were introduced by Pei et al. (2020). These are three sub-datasets of the WebKB[1] webpage dataset collected from computer science departments of various universities. In these datasets, nodes are web pages, and edges are hyperlinks between them. Node features are the bag-of-words representation of web pages. The target is a category of the web page: 'student', 'project', 'course', 'staff', or 'faculty'. We first note that these datasets are very small (183-251 nodes and 295-499 edges), which can lead to unstable and statistically insignificant results. Indeed, from results of various models reported in previous works it can be seen that the standard deviation on these datasets is very high. Moreover, these datasets have very imbalanced classes, to the point that the `texas` dataset has a class that consists of only one node, which makes using this class for training and evaluation meaningless. We report the number of nodes in different classes of these datasets in Table 7 in Appendix B. We note that all the previous works that use these datasets report accuracy on them, however, this metric is not designed for measuring performance in the setting with strong class imbalance and can provide misleading results in this setting.

4 New datasets with heterophily

Motivated by the observations described in the previous section, we collected several new datasets for evaluating GNNs under heterophily. We aim to obtain a set of datasets satisfying the following conditions:

- Datasets should be heterophilous. We evaluate this using the adjusted homophily measure; see formal definition below.
- Graph structure should be helpful for the task. To check this, we compare the performance of graph-agnostic methods with GNN methods. We expect GNNs to have a noticeable gain in performance.

[1] `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/wwkb`

Table 3: Statistics of the new heterophilous datasets

	wiki-cooc	roman-empire	amazon-ratings	minesweeper	workers	questions
nodes	10000	22662	24492	10000	11758	48921
edges	2243042	32927	93050	39402	519000	153540
average degree	448.61	2.91	7.60	7.88	88.28	6.28
global clustering	0.23	0.29	0.32	0.43	0.23	0.02
avg local clustering	0.55	0.39	0.58	0.44	0.53	0.03
diameter	3	6824	46	99	11	16
node features	100	300	300	7	10	301
classes	5	18	5	2	2	2
edge homophily	0.34	0.05	0.38	0.68	0.59	0.84
adjusted homophily	-0.03	-0.05	0.14	0.01	0.09	0.02
LI	0.02	0.11	0.04	0.00	0.01	0.00

- Datasets should be diverse, i.e., have different structural properties and come from various domains. Thus, for each dataset, we report several characteristics that we describe below.

Further in this section, we describe new datasets proposed in this paper. For each dataset, we report its basic characteristics, such as the number of nodes, edges, and features, as well as various graph statistics which we now define.

First, we measure homophily. As discussed above, we focus on adjusted homophily, but we also report edge homophily to be comparable with previous studies reporting this measure. Formally, edge homophily is

$$h_{edge} = \frac{|(u,v) \in E : y_u = y_v\}|}{|E|},$$

where y_u is the label of a node u and E is the set of edges. Adjusted homophily is based on the edge homophily and can be computed as follows:

$$h_{adj} = \frac{h_{edge} - \sum_{k=1}^{C} D_k^2/(2|E|)^2}{1 - \sum_{k=1}^{C} D_k^2/(2|E|)^2},$$

where $D_k := \sum_{v\,:\,y_v=k} d(v)$ and $d(v)$ denotes the degree of a node v. Adjusted homophily was proposed in Platonov et al. (2022), where the authors show that it satisfies a number of desirable properties, which makes it appropriate for comparing datasets with different number of classes and class balance.

We also report *label informativeness* (LI) introduced by Platonov et al. (2022) and shown to be a better predictor of GNN performance than homophily. Label informativeness quantifies how much information a neighbor's label gives about the node's label. To formally define this measure, we let $(\xi, \eta) \in E$ be an edge sampled uniformly at random among all edges and define

$$\text{LI} := I(y_\xi, y_\eta)/H(y_\xi)\,.$$

Here y_ξ and y_η are (random) labels of ξ and η, $H(y_\xi)$ is the entropy of y_ξ and $I(y_\xi, y_\eta)$ is the mutual information of ξ and η.

We also report several standard graph characteristics such as diameter and clustering coefficient. In the literature, there are two popular definitions of the clustering coefficient (Boccaletti et al., 2014). The global clustering coefficient is defined as the ratio between the number of triangles and the number of pairs of adjusted edges. To get the average local clustering coefficient, we first compute the clustering for each node and then average the obtained values across all nodes.

Table 3 provides statistics of the five new datasets that we propose for evaluating GNN performance under heterophily. One can see that these datasets have diverse properties. `Wiki-cooc` is the densest graph with an average node degree above 400, while the rest of the graphs are sparse, `roman-empire` being the sparsest one. `Questions` has very low values of clustering coefficients compared to other graphs, which shows that it has a small proportion of closed node triplets. `Roman-empire` is the only graph in our benchmark with a value of label informativeness significantly larger than zero. Below we will describe each of the new datasets in more detail.

Wiki-cooc This dataset represents word co-occurrence in the English Wikipedia. Nodes are unique words, their classes are parts of speech, and edges connect frequently co-occurring words. For nodes, we take 5000 nouns, 2500 verbs, 1500 adjectives, 500 adverbs, and 500 numerals. In all cases, we select the most frequent words of the required part of speech. We create an edge between two words if they co-occur at least 30 times in the text. We use the English Wikipedia 2022.03.01 dump from Lhoest et al. (2021). For node features, we use FastText word embeddings Grave et al. (2018). To make the task more challenging, we only use the first 100 dimensions of the 300-dimensional word embeddings.

Roman-empire This dataset is based on the Roman Empire article from English Wikipedia, which was selected since it is one of the longest articles on Wikipedia. The text was retrieved from the English Wikipedia 2022.03.01 dump from Lhoest et al. (2021). Each node in the graph corresponds to one word in the text. In contrast to the `wiki-cooc` dataset, here we consider *non-unique* words. Thus, the number of nodes in the graph is equal to the article's length. Two words are connected with an edge if at least one of the following two conditions holds: either these words follow each other in the text, or these words are connected in the dependency tree of the sentence (one word depends on the other). Thus, the graph is a line graph with additional shortcut edges corresponding to syntactic dependencies between words. The class of a node is its syntactic role (we select the 17 most frequent roles as unique classes and collapse all the other roles into the 18th class). For node features, we use FastText word embeddings (Grave et al., 2018). While this task can probably be better solved with models from the field of NLP, we adapt it to evaluate GNNs in the setting of low homophily, sparse connectivity, and potential long-range dependencies.

This graph has 22.7K nodes and 32.9K edges. By construction, the structure of this dataset is chain-like; thus, it has the smallest average degree (2.9) and the largest diameter (6824). This graph is heterophilous, $h_{adj} = -0.05$. Interestingly, this dataset has a larger value of label informativeness compared to all heterophilous datasets analyzed by Platonov et al. (2022). This means that there are non-trivial connectivity patterns specific for this dataset.

Amazon-ratings This dataset is based on the Amazon product co-purchasing network metadata dataset[2] from SNAP Datasets (Leskovec & Krevl, 2014). Nodes are products (books, music CDs, DVDs, VHS video tapes), and edges connect products that are frequently bought together. The task is to predict the average rating given to a product by reviewers. We divided possible rating values into five classes. For node features, we use the mean of FastText embeddings Grave et al. (2018) for words in the product description. To reduce the size of the graph, we only consider the largest connected component of the 5-core of the graph.

Minesweeper This dataset is inspired by the Minesweeper game, and it is the only synthetic dataset in our benchmark. The graph is a regular 100x100 grid where each node (cell) is connected to eight neighboring cells. 20% of the nodes are randomly selected as mines. The task is to predict which nodes are mines. The node features are one-hot-encoded numbers of neighboring mines. However, for randomly selected 50% of the nodes, the features are unknown, which is indicated by a separate binary feature.

The structure of this graph is significantly different from other datasets due to its regularity. The average degree is 7.88 since almost all the nodes have exactly eight neighbors. Since mines are placed randomly, both adjusted homophily and label informativeness are close to zero. Note that edge homophily in this dataset equals 0.68, which again shows that this measure cannot reflect homophily for datasets with unbalanced classes.

Workers This dataset is based on data from a crowdsourcing platform. The nodes represent workers, and an edge connects two workers if they have worked on the same task on the platform. The task is to predict which workers have been banned in one of the projects. Node features are based on the workers' profile information and task performance statistics.

This graph has 11.8K nodes, with the average degree of 88.28. Thus, this graph is significantly more dense than all the other graphs except for `wiki-cooc`. About 22% of the workers in this dataset have been banned.

[2] `https://snap.stanford.edu/data/amazon-meta.html`

Questions This dataset is based on data from a question-answering website. Nodes are users, and an edge connects two nodes if one user answered the other user's question during a given time interval. To restrict the size of the dataset, we consider only users interested in the topic 'medicine'. The task is to predict which users remained active on the website (were not deleted or blocked) after the data on which the graph is based was collected. For node features, we use the mean of FastText embeddings Grave et al. (2018) for words in the user description. Since some users (15%) do not have descriptions, we use an additional binary feature that indicates such users.

The obtained dataset has 48.9K nodes, and the average degree is 6.28. We note that the classification task is highly unbalanced: 97% of the users are in the active class. These causes high edge homophily, but the adjusted homophily indicates that the graph is heterophilous: $h_{adj} = 0.02$. This dataset has the smallest clustering coefficients among the proposed ones, which means it has a small fraction of closed node triplets.

5 BENCHMARKING EXISTING ALGORITHMS

5.1 SETUP

Baselines We choose several representative neural architectures as our baselines. First, we take a **ResNet**-like model He et al. (2016) as a graph-agnostic baseline. This model treats all nodes as independent samples and does not have access to the graph topology. Thus, if graph topology provides useful information for the task, we expect other models to outperform ResNet. Further, we use two simple node feature augmentation strategies to provide ResNet with some information about the graph structure. One strategy is multiplying the initial node feature matrix with a power of normalized graph adjacency matrix, which smooths node features along graph edges. This strategy was proposed in Wu et al. (2019) (their proposed model SGC is a linear classifier on top of preprocessed features, while we use a ResNet-like model on top of preprocessed features — we name this model **ResNet+SGC**). Another strategy is augmenting node features with the rows of the adjacency matrix, thus directly providing information about the graph connectivity. This approach is inspired by LINK Zheleva & Getoor (2009) — a linear model using the adjacency matrix rows as features — and is very similar to the recently proposed LINKX model Lim et al. (2021), which combines node features and adjacency matrix rows and uses a custom ResNet-like model. We name this version of our model **ResNet+adj**.

Further, we use 2 classic GNN architectures: **GCN** Kipf & Welling (2017) and **GraphSAGE** Hamilton et al. (2017). For GraphSAGE, we use the version with the mean aggregation function and we do not use the node sampling technique used in the original paper.

As a more advanced GNN architecture, we take **GAT** Veličković et al. (2017), which uses attention-based aggregation. However, GAT uses a very simple attention mechanism and, as a result, can only compute a limited kind of attention — for instance, the ranking of the attention scores does not depend on the query node Brody et al. (2022). To overcome this limitation, we also use a model with a more powerful attention mechanism — Graph Transformer (**GT**) Shi et al. (2020), which is an adaptation of the popular Transformer architecture Vaswani et al. (2017) to graphs.

Zhu et al. (2020) shows that separating ego- and neighbor-embeddings in the GNN aggregation step (as done in GraphSAGE, where the node's embedding is concatenated to the mean of its neighbors' embeddings instead of being summed with them) is beneficial when learning under heterophily. Thus, we add this simple architectural modification to GAT and GT models, which originally do not separate ego- and neighbor embeddings. We name these model modifications **GAT-sep** and **GT-sep**.

We augment all our baseline models with skip connections He et al. (2016), and layer normalization Ba et al. (2016), which are standard neural architecture elements in modern deep learning.

Heterophily-specific models We use eight models specifically designed for node classification under heterophily: **H_2GCN** (Zhu et al., 2020), **CPGNN** (Zhu et al., 2021), **GPR-GNN** (Chien et al., 2020), **FSGNN** (Maurya et al., 2022), **GloGNN** (Li et al., 2022), **FAGCN** (Bo et al., 2021), **GBK-GNN** Du et al. (2022), and **JacobiConv** Wang & Zhang (2022). To our knowledge, this is the largest comparison of heterophily-specific models in the literature. For our comparison on original and filtered `squirrel` and `chameleon`, we also use **Geom-GCN** with three embedding

Table 4: The performance of models on the proposed datasets: ROC AUC for `minesweeper`, `workers`, and `questions` and accuracy for the remaining datasets

	wiki-cooc	roman-empire	amazon-ratings	minesweeper	workers	questions
ResNet	89.36 ± 0.71	65.71 ± 0.44	45.70 ± 0.69	50.95 ± 1.12	73.08 ± 1.28	70.10 ± 00.75
ResNet+SGC	90.44 ± 0.59	73.91 ± 0.47	49.83 ± 0.61	70.95 ± 0.85	80.59 ± 0.82	75.45 ± 00.82
ResNet+adj	92.84 ± 0.29	52.08 ± 0.61	51.36 ± 0.47	50.25 ± 0.69	78.71 ± 1.05	75.71 ± 01.40
GCN	91.01 ± 0.69	70.51 ± 0.75	47.77 ± 0.69	89.47 ± 0.69	83.14 ± 1.11	75.97 ± 01.23
GraphSAGE	93.60 ± 0.31	85.80 ± 0.69	52.77 ± 0.54	93.53 ± 0.50	82.34 ± 0.19	76.52 ± 00.93
GAT	92.44 ± 0.80	81.02 ± 0.46	47.95 ± 0.53	92.10 ± 0.67	83.98 ± 0.57	77.42 ± 01.23
GAT-sep	93.96 ± 0.38	88.54 ± 0.48	52.11 ± 0.28	93.90 ± 0.35	83.84 ± 0.47	76.89 ± 00.76
GT	92.52 ± 0.61	86.29 ± 0.57	51.24 ± 0.39	92.21 ± 0.76	83.38 ± 0.86	77.63 ± 00.57
GT-sep	92.82 ± 0.61	87.06 ± 0.49	52.09 ± 0.52	92.35 ± 0.53	82.87 ± 0.97	77.84 ± 01.00
H_2GCN	89.24 ± 0.32	68.09 ± 0.29	41.36 ± 0.47	89.95 ± 0.38	81.76 ± 0.68	66.66 ± 1.84
CPGNN	84.84 ± 0.66	63.78 ± 0.50	44.69 ± 0.35	71.27 ± 1.14	72.44 ± 0.80	67.09 ± 2.63
GPR-GNN	91.90 ± 0.78	73.37 ± 0.68	43.90 ± 0.48	81.79 ± 0.98	70.59 ± 1.15	73.41 ± 1.24
FSGNN	98.03 ± 0.37	72.46 ± 0.60	47.75 ± 0.74	81.58 ± 0.42	80.32 ± 0.90	75.45 ± 1.24
GloGNN	88.49 ± 0.45	63.85 ± 0.49	37.28 ± 0.66	62.53 ± 1.34	73.90 ± 0.95	67.15 ± 1.92
FAGCN	91.88 ± 0.37	70.53 ± 0.99	46.32 ± 2.50	89.69 ± 0.60	81.87 ± 0.94	77.04 ± 1.56
GBK-GNN	97.81 ± 0.32	75.87 ± 0.43	43.47 ± 0.51	83.56 ± 0.84	78.06 ± 0.91	72.98 ± 1.05
JacobiConv	92.66 ± 0.22	73.75 ± 1.11	44.00 ± 0.41	88.62 ± 1.51	73.55 ± 1.22	70.25 ± 5.53

methods (Isomap (Geom-GCN-I), Poincare (Geom-GCN-P), and struc2vec (Geom-GCN-S)) (Pei et al., 2020) (we do not use these models for other datasets since the code for precomputing node representations is not available).

We provide details about our training setup and hyperparameters selection in Appendix A. The proposed datasets and the code of our experiments are available.[3]

5.2 Results

Table 4 shows the performance of different models on our datasets. We can see that on all datasets except for `wiki-cooc`, the best results are achieved by baselines rather than heterophily-specific models. With the exception of `wiki-cooc`, none of the heterophily-specific models get in the top-3 best results. Occasionally some heterophily-specific models perform even worse than the graph-agnostic ResNet baseline. We believe this shows that the progress in learning under heterophily made in recent years was mostly illusionary: standard GNNs generally outperform specialized models.

As for standard GNNs, we notice that the best results are almost always (with the exception of `workers`) achieved by models that separate ego- and neighbor-embeddings (GraphSAGE, GAT-sep, GT-sep). GAT-sep and GT-sep typically outperform their versions without embedding separation, which shows that this trick proposed in Zhu et al. (2020) is indeed helpful for learning under heterophily.

6 Conclusion

In this paper, we uncover significant issues with most datasets typically used to evaluate heterophily-specific GNN models. In particular, we show that the duplicates present in the `squirrel` and `chameleon` datasets lead to a train-test data leakage, and removing these duplicates drastically changes the relative performance of different models.

Motivated by this issue, we propose several new datasets of different nature and diverse structural properties that can form a better benchmark. We evaluate baselines and heterophily-specific models on these datasets and show that baselines generally outperform specialized models. We hope that the proposed benchmark will be useful for further progress in learning under heterophily.

[3] https://github.com/heterophily-submit/

REFERENCES

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

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

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

Stefano Boccaletti, Ginestra Bianconi, Regino Criado, Charo I Del Genio, Jesús Gómez-Gardenes, Miguel Romance, Irene Sendina-Nadal, Zhen Wang, and Massimiliano Zanin. The structure and dynamics of multilayer networks. *Physics reports*, 544(1):1–122, 2014.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Lio, and Michael M Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022.

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

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

C Lee Giles, Kurt D Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries*, pp. 89–98, 1998.

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

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism, 2020. URL https://arxiv.org/abs/2003.02218.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 175–184, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.emnlp-demo.21.

Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. Finding global homophily in graph neural networks when meeting heterophily. *arXiv preprint arXiv:2205.07308*, 2022.

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

Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Is heterophily a real nightmare for graph neural networks to do node classification? *arXiv preprint arXiv:2109.05641*, 2021.

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

Sunil Kumar Maurya, Xin Liu, and Tsuyoshi Murata. Simplifying approach to node classification in graph neural networks. *Journal of Computational Science*, pp. 101695, 2022.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

Galileo Namata, Ben London, Lise Getoor, Bert Huang, and UMD EDU. Query-driven active surveying for collective classification. In *10th International Workshop on Mining and Learning with Graphs*, volume 8, pp. 1, 2012.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 2020.

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

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

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

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*, 2020.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

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

Xiyuan Wang and Muhan Zhang. How powerful are spectral graph neural networks. *arXiv preprint arXiv:2205.11172*, 2022.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pp. 6861–6871. PMLR, 2019.

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

Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. *arXiv preprint arXiv:2102.06462*, 2021.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pp. 40–48. PMLR, 2016.

Elena Zheleva and Lise Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th international conference on World wide web*, pp. 531–540, 2009.

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

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

Table 5: Distribution of duplicates across classes in Wikipedia datasets

	class 1	class 2	class 3	class 4	class 5
			squirrel		
number of nodes	1042	1040	1039	1040	1040
number of duplicates	291	528	644	722	811
number of non-duplicates	751	512	395	318	229
			chameleon		
number of nodes	456	460	453	521	387
number of duplicates	219	328	251	359	256
number of non-duplicates	237	132	202	162	131

A TRAINING DETAILS AND HYPERPARAMETERS SELECTION

In this section, we describe the details of our training setup for experiments in Section 3.1 and Section 5. We treat all graphs as undirected. For `squirrel` and `chameleon` datasets we using the 10 existing standard train/validation/test splits. For filtered versions of these datasets we use the same splits with duplicates removed. For each of our new proposed datasets, we fix 10 random 50%/25%/25% train/validation/test splits. We train each model on each split once, reporting mean performance and standard deviation. For binary classification datasets (`minesweeper`, `workers`, and `questions`) we report ROC AUC, and for all the other datasets we report accuracy.

We found that our baseline models are quite robust to hyperparameter values, so we use the same hyperparameters for them and do not perform any model-specific or dataset-specific tuning. Namely, we use the following hyperparameter values: the number of layers is set to 5, the hidden dimension is set to 512, and the dropout probability is set to 0.2. For GAT and Graph Transformer models, the number of attention heads is set to 8. We use the Adam optimizer (Kingma & Ba, 2014) with learning rate of $3 \cdot 10^{-5}$. We train each model for 1000 steps and select the best step based on the performance on the validation dataset.

Unlike the baselines, heterophily-specific models turned out to be quite sensitive to the particular choice of hyperparameters. Namely, the choice of learning rate and weight decay may significantly impact the model performance, while the model performance is quite robust to dropout rate and size of the model — width and depth. Higher learning rates are known to improve generalization (Lewkowycz et al., 2020), and we use the Adam optimizer with the learning rate 0.05 in most experiments or 0.01 if the training diverged (i.e., for GloGNN). We have run experiments with 5 different values of weight decay $\{10^{-5}, 3 \cdot 10^{-5}, 10^{-4}, 3 \cdot 10^{-4}, 10^{-3}\}$ and selected the run with the best validation performance to be reported. For GloGNN and FSGNN, we have also searched over the parameter defining the furthest k-hop neighborhood used in the model. For $\mathbf{H_2}$GCN and CPGNN we use the parameter grid recommended by the original papers. Finally, for the original `squirrel` and `chameleon`, we take the parameters used by the authors of the corresponding models.

B ADDITIONAL DATASET STATISTICS

In Table 5, we show the distribution of duplicates across classes in `squirrel` and `chameleon` datasets. We can see that there is a large number of duplicates in all classes, however, their distribution is not even.

In Table 6, we provide the statistics of `squirrel` and `chameleon` datasets before and after filtering.

In Table 7, we report the distribution of nodes across classes in `texas`, `cornell`, and `wisconsin` datasets.

Table 6: Statistics of `squirrel` and `chameleon` before and after filtering

	squirrel	squirrel-filtered	chameleon	chameleon-filtered
number of nodes	5201	2205	2277	864
number of edges	198353	46557	31371	7754
average degree	76.27	42.23	27.55	17.95
global clustering	0.35	0.64	0.31	0.61
average local clustering	0.42	0.46	0.48	0.58
diameter	10	9	11	10
number of node features	2089	2089	2325	2325
number of classes	5	5	5	5
edge homophily	0.22	0.21	0.23	0.23
adjusted homophily	0.01	0.01	0.03	0.03
label informativeness	0.00	0.00	0.05	0.01

Table 7: Number of nodes in different classes of `texas`, `cornell`, and `wisconsin` datasets

	class 1	class 2	class 3	class 4	class 5
texas	33	1	18	101	30
cornell	38	16	30	82	17
wisconsin	10	70	118	32	21

C Comparison to the benchmark proposed in Lim et al. (2021)

Recently, a benchmark of large-scale heterophilous graph datasets has been proposed in Lim et al. (2021). This section describes how this benchmark differs from our proposed datasets. The first and most important difference is the size of the graphs. Lim et al. (2021) specifically collect large datasets to evaluate the performance of scalable graph methods under heterophily. However, this prevents them from comparing to many standard GNNs designed for heterophilous graphs since such GNNs are often compute and memory intensive and thus cannot scale to the size of the graphs proposed by Lim et al. (2021). In contrast, for our benchmark, we purposefully collect graphs with less than 50K nodes, allowing us to compare many models for learning under heterophily proposed in the literature.

Another difference lies in the domains from which the datasets come. Graphs are a natural way to represent data from different fields; thus, a comprehensive graph benchmark should cover a wide variety of them. Our graphs come from different and more diverse domains than the ones proposed in Lim et al. (2021). Specifically, Lim et al. (2021) use social networks (`penn94`, `pokec`, `genius`, `twitch-gamers`), citation networks (`arxiv-year`, `snap-patents`), and a web graph (`wiki`). In contrast, our datasets are a word co-occurrence graph (`wiki-cooc`), a word dependency graph (`roman-empire`), a product co-purchasing network (`amazon-ratings`), a synthetic graph emulating the game of minesweeper (`minesweeper`), a crowdsourcing platform worker network (`workers`), and a question-answering website interaction network (`questions`).