When Heterophily Meets Heterogeneity: Challenges and a New Large-Scale Graph Benchmark

Junhong Lin
junhong@mit.edu
Massachusetts Institute of Technology
Cambridge, MA, United States

Xiaojie Guo
Xiaojie.Guo@ibm.com
IBM Research
Yorktown Heights, NY, United States

Shuaicheng Zhang
zshuai8@vt.edu
Virginia Tech
Blacksburg, VA, United States

Yada Zhu
yzhu@us.ibm.com
IBM Research
Yorktown Heights, NY, United States

Julian Shun
jshun@mit.edu
Massachusetts Institute of Technology
Cambridge, MA, United States

Abstract

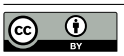
Graph mining has become crucial in fields such as social science, finance, and cybersecurity. Many large-scale real-world networks exhibit both heterogeneity, where multiple node and edge types exist in the graph, and heterophily, where connected nodes may have dissimilar labels and attributes. However, existing benchmarks primarily focus on either heterophilic homogeneous graphs or homophilic heterogeneous graphs, leaving a significant gap in understanding how models perform on graphs with both heterogeneity and heterophily. To bridge this gap, we introduce $\mathcal{H}^2$GB, a large-scale node-classification graph benchmark that brings together the complexities of both the heterophily and heterogeneity properties of real-world graphs. $\mathcal{H}^2$GB encompasses 9 real-world datasets spanning 5 diverse domains, 28 baseline models, and a unified benchmarking library with a standardized data loader, evaluator, unified modeling framework, and an extensible framework for reproducibility. We establish a standardized workflow supporting both model selection and development, enabling researchers to easily benchmark graph learning methods. Extensive experiments across 28 baselines reveal that current methods struggle with heterophilic and heterogeneous graphs, underscoring the need for improved approaches. Finally, we present a new variant of the model, $\mathcal{H}^2$G-former, developed following our standardized workflow, that excels at this challenging benchmark. Both the benchmark and the framework are publicly available at Github and PyPI, with documentation hosted at https://junhongmit.github.io/H2GB.

CCS Concepts

• **Information systems** → **Data mining**; **Digital libraries and archives**.

Keywords

Graph Mining, Graph Transformers, Graph Neural Networks, Large-scale Graphs, Heterogeneous Graphs, Graph Heterophily



KDD '25, Toronto, ON, Canada

ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3737421

ACM Reference Format:
Junhong Lin, Xiaojie Guo, Shuaicheng Zhang, Yada Zhu, and Julian Shun. 2025. When Heterophily Meets Heterogeneity: Challenges and a New Large-Scale Graph Benchmark. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3711896.3737421

1 Introduction

Graphs are commonly used to model complex relationships across various domains, such as finance [50], social science [26, 48] and cybersecurity [16, 52]. Many real-world graphs contain millions or even billions of nodes and edges, making scalable learning methods essential. Graph neural networks (GNNs) [15, 23] have achieved state-of-the-art performance on graph learning tasks. However, they were designed primarily for *homogeneous homophilic graphs*, where the nodes and edges are of a single type [11, 64], and connected nodes are similar, as shown in Figure 1(a).

As real-world graphs grow in scale, they increasingly exhibit heterogeneity and heterophily. ***Heterogeneity*** arises from multiple entity and relation types, adding structural and semantic complexity. This diversity, in turn, intensifies ***heterophily***, the tendency for connected nodes to have dissimilar labels or attributes. For example, financial networks (Figure 1(d)) [2, 45] contain diverse node types (e.g., person, business) and edge types (e.g., wire transfer, check transaction). Furthermore, fraudsters tend to have different labels than their innocent neighbors, making these networks both heterogeneous and heterophilic. These properties, common in domains such as e-commerce [32], academia [19, 59], and cybersecurity [3, 25], pose significant challenges to GNN performance.

In recent years, researchers have actively explored methods to overcome these challenges in two separate directions. First, to handle graphs with heterophily, there has been a recent line of research on developing heterophilic graph benchmarks [4, 30] and heterophily-centered GNNs [4, 35, 43, 63, 64] that incorporate long-range relationships and distinct aggregation mechanisms, such as distant node exploration [1, 29, 43, 64], signed aggregation [4, 35, 63], and local grouping [29]. However, these heterophilic GNNs are restricted to homogeneous graphs, as illustrated in Figure 1(b). Second, heterogeneous GNNs have been proposed to handle the diverse information present in heterogeneous graphs [9, 17, 21, 46, 51, 58]. However, most heterogeneous GNNs are implicitly built upon the

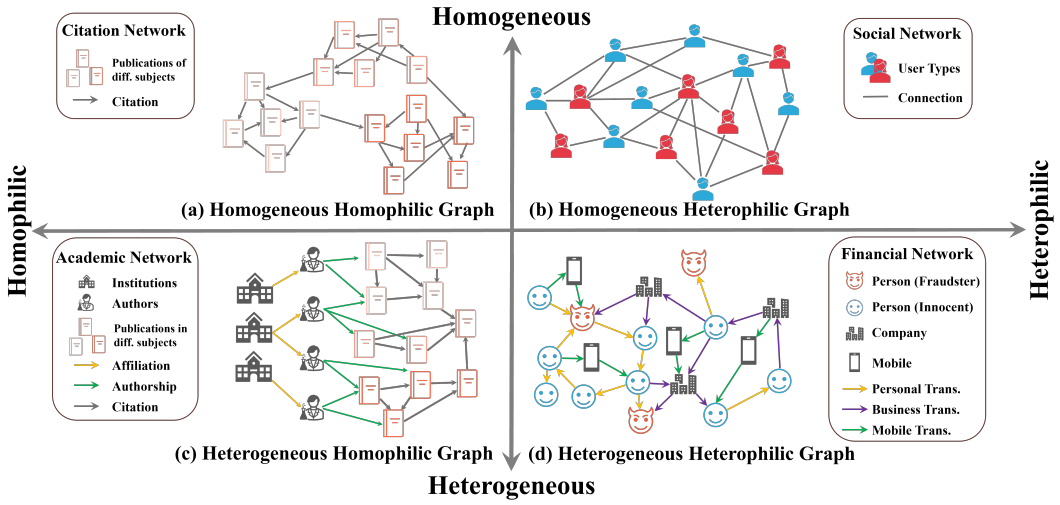


Figure 1: Examples of graphs with different levels of heterophily and heterogeneity. Nodes with different class labels and edges of different types are represented with different colors (e.g., publications in different subjects or different kinds of financial transactions).

homophily assumption, as illustrated in Figure 1(c), and exhibit poor performance on heterophilic graphs [14].

While there has been recent progress on handling heterogeneity and heterophily separately, many large real-world graphs exhibit both properties simultaneously. A recent research effort, the *Heterophily Graph Learning Handbook* [34], explicitly highlights this gap, emphasizing that previous research primarily evaluated models on graphs that focused on either only heterophily or only heterogeneity. The following challenges arise when exploring graph learning in heterophilic and heterogeneous settings. **(1) Lack of benchmarks for graphs with both heterophily and heterogeneity [34]**: Existing benchmarks either focus exclusively on homogeneous graphs, neglecting the diversity of node and edge types found in real-world graphs, or on heterogeneous graphs while assuming homophily. **(2) Limited understanding of heterophily in heterogeneous Graphs [34]**: Heterophily has been largely studied in homogeneous graphs, leaving its impact on heterogeneous structures under-explored. This gap limits our understanding of how heterophilic patterns interact with diverse node and edge types. Guo et al. [14] found that heterogeneous GNNs often degrade in performance under heterophily, highlighting the need for better modeling strategies. **(3) Inadequacy of heterophilic GNNs on large-scale heterogeneous graphs**: Heterophilic GNNs are typically designed for homogeneous graphs, making them ineffective in heterogeneous settings where node and edge types vary. They also struggle to scale with graph size, as many were developed for small graphs, limiting their applicability to large real-world networks.

To address these challenges, we introduce the Heterophilic and Heterogeneous Graph Benchmark ($\mathcal{H}^2$GB), the first, novel and comprehensive graph benchmark designed to evaluate graph learning methods on large-scale heterophilic and heterogeneous graphs across multiple real-world domains. As shown in Figure 2, $\mathcal{H}^2$GB provides the following contributions:

- **Diverse Real-World Datasets**: $\mathcal{H}^2$GB consists of 4 applications, and 9 real-world datasets spanning 5 domains: academia, finance, e-commerce, social science, and cybersecurity.
- **Standardized Benchmarking**: $\mathcal{H}^2$GB establishes a standardized evaluation framework for node classification, providing an extensive comparison of *28 baseline models* implemented through our previously built modular graph learning framework, UnifiedGT [31], including message-passing GNNs, graph transformers, and non-GNN baselines, under a unified experimental setup.
- **Standardized Workflow**: We introduce a standard workflow supporting both model selection and development. In particular, we demonstrate a case study using $\mathcal{H}^2$GB for the development of a new model in Section 5.3.
- **New Heterophily Measure**: Existing metrics (e.g., edge heterophily) provide limited insights into heterogeneous graph structures. We introduce a *new heterophily measure, the $\mathcal{H}^2$ index,* which better captures complex heterophilic interactions, addressing a key limitation identified in prior literature [34].
- **Scalability Focus**: $\mathcal{H}^2$GB emphasizes scalability by evaluating graph learning methods on large-scale heterophilic and heterogeneous graphs. Most of our datasets are large, containing millions of nodes and tens of millions of edges (see Table 1), which are orders of magnitude larger than existing heterophilic benchmarks [30, 61]. $\mathcal{H}^2$GB evaluates models across large-scale graphs, identifies the performance bottlenecks of existing GNNs, and encourages the development of scalable heterophilic graph learning methods.
- **Open-Source Benchmarking Library**: $\mathcal{H}^2$GB is released as an extensible and user-friendly Python library consisting of a unified data loader and evaluator, making it easy to access datasets, evaluate methods, and compare performance.

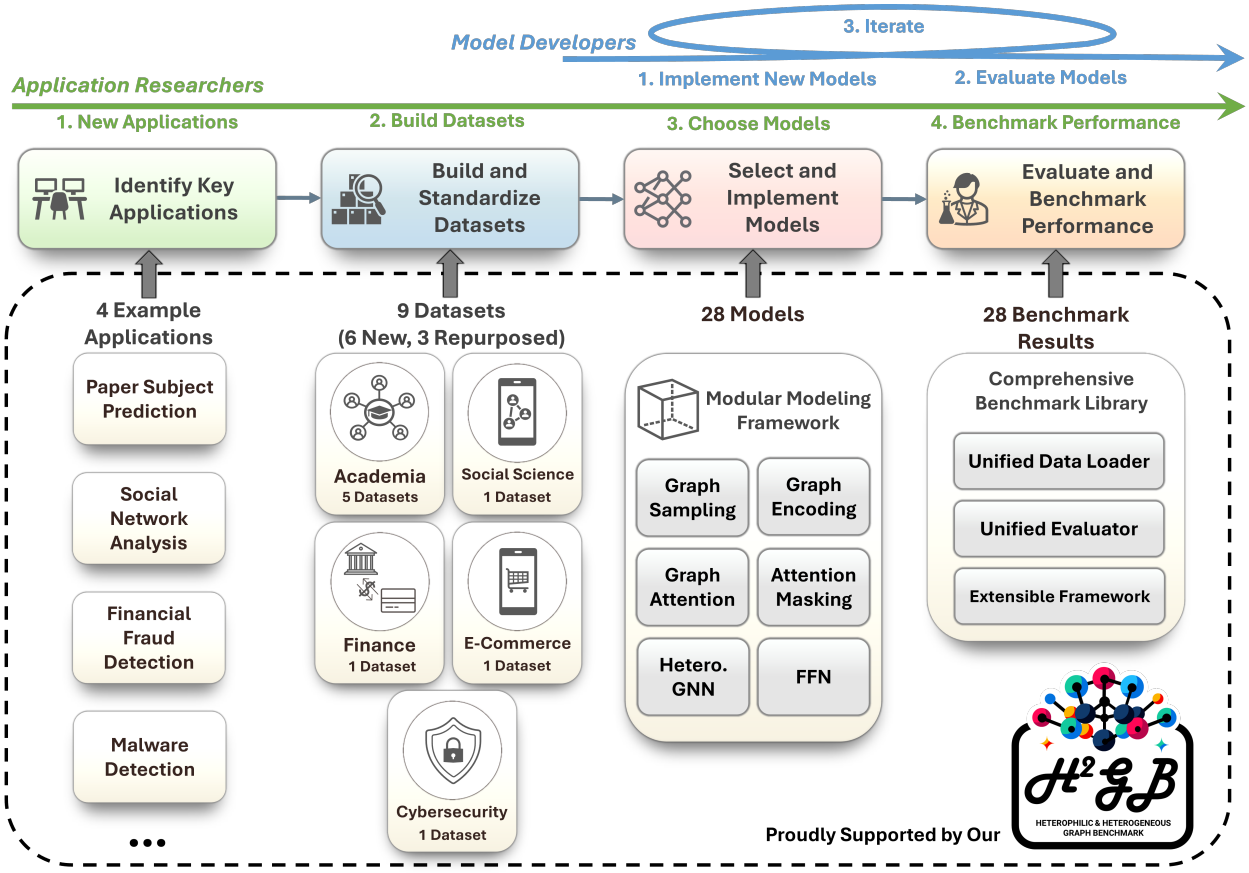


Figure 2: $\mathcal{H}^2$GB offers a complete benchmark workflow for heterophilic and heterogeneous graph learning, featuring a diverse dataset suite (Section 3), a modular modeling framework (Section 4), and a comprehensive benchmark library, making it easy to evaluate and compare different methods (Section 5). The green and blue arrows on top highlight two standard workflows for users to interact with $\mathcal{H}^2$GB.

Through comprehensive experiments on our datasets, we draw the following insights: (1) homogeneous heterophilic GNNs underperform heterogeneous homophilic GNNs due to their inability to account for diverse node and edge types; (2) performance varies significantly among heterogeneous homophilic GNNs, likely due to differences in their architectural robustness when exposed to heterophily; and (3) non-scalable GNNs struggle on our large-scale heterogeneous heterophilic benchmark. Lastly, following our established standard workflow, we develop $\mathcal{H}^2$G-former, a new effective model variant by incorporating several new components including masked label embedding, heterogeneous attention, k-hop attention mask, and type-specific FFNs, significantly improving performance on datasets in $\mathcal{H}^2$GB.

2 Preliminaries and Related Work

Definition 1 (Graph Heterogeneity)**.** A heterogeneous graph is a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R})$, where each node $v \in \mathcal{V}$ and edge $e \in \mathcal{E}$ has a type given by $\tau(v) : V \to \mathcal{A}$ and $\phi(e) : E \to \mathcal{R}$. Here, $\mathcal{A}$ and $\mathcal{R}$ are the set of node and edge types, respectively.

Definition 2 (Metapath-Induced Subgraphs)**.** A metapath is a sequence of edges, defined as $\mathcal{P} = A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots A_n \xrightarrow{R_n} A_{n+1}$, where $A_i \in \mathcal{A}$ and $R_i \in \mathcal{R}$. Given a metapath $\mathcal{P}$, we can construct a metapath-induced subgraph $\mathcal{G}_\mathcal{P}$, which includes edge (u, v) in $\mathcal{G}_\mathcal{P}$ if and only if there exists at least one length-n path between u and v following the metapath $\mathcal{P}$ in the original graph $\mathcal{G}$.

Definition 3 (Graph Heterophily)**.** Graph heterophily quantifies the dissimilarity between connected nodes based on their attributes or labels. Common metrics such as edge heterophily [64] and node heterophily [43] are designed for homogeneous graphs, quantifying the proportion of connected nodes that have different labels.

Definition 4 (Node Classification Task)**.** Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R})$, only a subset of nodes of a specific type $\mathcal{V}_T \subseteq \mathcal{V}$ (*task entities*) are labeled. The task is to learn a function $f : (\mathcal{G}, v) \mapsto y_v$ that predicts the label y_v for unlabeled nodes $v \in \mathcal{V}_T$.

Graph Learning for Heterogeneous and Heterophilic Graphs. Existing heterogeneous GNNs are classified into *metapath-based* methods, which extract structural information from homogeneously-typed subgraphs by predefined metapaths to capture diverse semantic data [9, 46, 51, 58], and *metapath-free* methods, which process

Table 1: Statistics of $\mathcal{H}^2$GB datasets. #C is the number of classes, with imbalance ratios provided for binary classification. The training/validation/test split ratio is indicated under the Split Scheme.

Dataset	# Nodes (types)	# Edges (types)	# Feat.	# C (Ratio)	Label	Split Scheme (Ratio [%])	Metric
ogbn-mag	1,939,743 (4)	42,182,144 (7)	128	349	paper venue	Time (85/9/6)	Accuracy
mag-year	1,939,743 (4)	42,182,144 (7)	128	5	publication year	Random (50/25/25)	Accuracy
oag-cs	1,112,691 (4)	27,537,448 (22)	768	3,514	paper venue	Time (80/9/11)	Accuracy
oag-eng	929,315 (4)	12,346,854 (22)	768	3,956	paper venue	Time (88/10/2)	Accuracy
oag-chem	1,918,881 (4)	38,098,014 (22)	768	2,985	paper venue	Time (90/8/2)	Accuracy
RCDD	13,806,619 (7)	157,814,864 (14)	256	2 (11:1)	risk commodity	Time (70/15/15)	F1 score
IEEE-CIS-G	153,880 (12)	2,873,472 (22)	4823	2 (12:1)	fraud transaction	Time (80/10/10)	F1 score
H-Pokec	1,731,977 (16)	51,774,836 (31)	66	2 (1:1)	gender	Random (50/25/25)	Accuracy
PDNS	1,173,558 (2)	76,797,104 (4)	10	2 (1:2)	malicious domain	Time (70/20/10)	F1 score

structural and semantic information simultaneously, enhancing message aggregation by incorporating node and edge types without relying on predefined paths [17, 21, 36, 65]. While these approaches take heterogeneity into account, they generally maintain the homophily assumption. In contrast, existing heterophilic GNNs have been tailored primarily for homogeneous graphs and lack mechanisms to address heterogeneity [1, 4, 30]. Recent works aim to bridge this gap by improving heterophilic learning on heterogeneous graphs through augmented graphs and disentangled loss functions [14, 28]; however, they primarily focus on enhancing existing models rather than introducing fundamentally new solutions optimized for both heterophily and heterogeneity.

Current Datasets. Recent evaluations of heterophilic graph learning primarily use small-scale datasets from Pei et al. [43]. Lim et al. [30] have compiled larger non-homophilic graph datasets, which have become the standard for evaluating heterophilic GNNs, but their datasets are limited to homogeneous graphs. Several heterogeneous academic network datasets have been introduced, including DBLP [36], ACM [36], ogbn-mag [20], MAG240M [19], and IGB [22]. However, these datasets have not been tested with heterophilic GNN methods. Moreover, the pure focus on academic networks narrows their use in addressing graph learning challenges in other domains.

Conventional Heterophily Metrics. Typical heterophily metrics, such as edge heterophily ($\mathcal{H}_{\text{edge}}$) [64], node heterophily ($\mathcal{H}_{\text{node}}$) [43], and adjusted heterophily ($\mathcal{H}_{\text{adj}}$) [44], are designed for homogeneous graphs, quantifying different aspects of label mixing among connected nodes. While edge and node heterophily directly reflect label differences along edges or within local neighborhoods, they are sensitive to class imbalance [30]. Adjusted heterophily mitigates this issue by normalizing based on class distributions. A common approach to extend these metrics to heterogeneous graphs is to disregard node and edge types, treating the graph as homogeneous. Yet, this simplification overlooks structural dependencies across different node types. Traditional metrics typically assess heterophily only among nodes of the same type, failing to account for homophily that may emerge along metapath-based structures. Guo et al. [14] empirically showed that heterogeneous GNNs perform better when metapath-induced subgraphs are homophilic, a factor not captured by typical heterophily measures. Consequently, these metrics can misrepresent a model's true ability to handle heterophilic relationships in heterogeneous graphs.

This limitation underscores the need for a better heterophily measure designed for heterogeneous graphs. Recent works [14, 32] have proposed the metapath-based label heterophily (MLH) measure, which extends edge heterophily, $\mathcal{H}_{\text{edge}}$, to a metapath-induced subgraph $\mathcal{G}_{\mathcal{P}}$, and is formulated as follows:

$$\text{MLH}(\mathcal{G}) = \text{Agg}(\mathcal{H}_{\text{edge}}(\mathcal{G}_{\mathcal{P}}) | \mathcal{P} \in \mathcal{M}_k), \tag{1}$$

where $\mathcal{M}_k$ denotes a k-hop metapath set, and $\text{Agg} \in \{\text{mean}, \max\}$. However, it suffers from class imbalance [30], leading to artificially low values (indicating homophily) in datasets that are inherently heterophilic. For instance, as shown in Table 2, the RCDD and IEEE-CIS-G datasets demonstrate significant class imbalance, which contributes to deceptively low MLH values.

3 Heterophilic and Heterogeneous Graph Benchmark ($\mathcal{H}^2$GB)

In this section, we present $\mathcal{H}^2$GB, a benchmark consisting of 9 large-scale datasets (6 new ones and 3 from existing work), shown in Table 1, spanning 5 diverse domains (Figure 2): academia, e-commerce, finance, social science, and cybersecurity. We also introduce a new heterophily measure that better captures the heterophilic properties of heterogeneous graphs. The benchmark standardizes data loading, data splitting, feature encoding, and performance evaluation, which together enable open and reproducible research on heterophilic and heterogeneous GNNs.[1]

3.1 Key Applications

Real-world graphs exhibit a diverse range of applications, many of which inherently involve both heterophily and heterogeneity. We identify four representative key real-world applications where such graph structures naturally arise: paper venue classification, social network analysis, financial fraud detection, and malware detection. They span diverse domains—academia, finance, e-commerce, cybersecurity, and social science—each presenting unique challenges that demand robust graph learning methods, ensuring that $\mathcal{H}^2$GB captures the complexities of large-scale real-world heterophilic and heterogeneous graphs across multiple domains.

[1] As a special case, $\mathcal{H}^2$GB can also be useful for systematically evaluating homogeneous GNNs (by simply applying a learnable type-dependent feature projection and then ignoring the type information on nodes and edges).

Table 2: Heterophily measures on each dataset. A value near 0 indicates homophily, where nodes primarily connect to others of the same class, while values around 1 suggest heterophily, where nodes prefer connections to different classes. y_v is the label of node v, C denotes the number of classes, $d(v)$ is the in-degree of node v, $D_k = \sum_{v:y_v=k} d(v)$ is the total in-degree of class k nodes, $\mathcal{M}_k$ is a k-hop metapath set, and Agg $\in$ {mean, max} is an aggregation function.

Heterophily Metric		ogbn-mag	mag-year	oag-cs	oag-eng	oag-chem	RCDD	IEEE-CIS-G	H-Pokec	PDNS
Edge Heterophily	$\mathcal{H}_{\text{edge}} = \frac{\lvert\{(u,v) \in \mathcal{E} : y_u \neq y_v\}\rvert}{\lvert\mathcal{E}\rvert}$	0.9205	0.7909	0.9835	0.9586	0.9457	0.5001	0.5917	0.5663	0.4990
Node Heterophily	$\mathcal{H}_{\text{node}} = \frac{1}{\lvert\mathcal{V}\rvert} \sum_{v \in \mathcal{V}} \frac{\lvert\{u \in \mathcal{N}(v) : y_v \neq y_u\}\rvert}{\lvert\mathcal{N}(v)\rvert}$	0.9539	0.7946	0.9880	0.9748	0.9696	0.5005	0.5839	0.5667	0.4992
Adjusted Heterophily	$\mathcal{H}_{\text{adj}} = 1 - \frac{1 - \sum_{k=1}^{C} D_k^2/(2\lvert\mathcal{E}\rvert)^2 - \mathcal{H}_{\text{edge}}}{1 - \sum_{k=1}^{C} D_k^2/(2\lvert\mathcal{E}\rvert)^2}$	0.9312	0.9977	0.9847	0.9612	0.9496	0.8398	1.3151	1.1350	1.0027
Metapath-based Label Heterophily	$\text{MLH} = \text{Agg}\left(\mathcal{H}_{\text{edge}}(\mathcal{G}_{\mathcal{P}}) \mid \mathcal{P} \in \mathcal{M}_k\right)$	0.8731	0.7718	0.9623	0.8689	0.8724	0.4912	0.1352	0.3922	0.3916
$\mathcal{H}^2$ **Index (Ours)**	$\mathcal{H}^2 = \text{Agg}\left(\mathcal{H}_{\text{adj}}(\mathcal{G}_{\mathcal{P}}) \mid \mathcal{P} \in \mathcal{M}_k\right)$	0.8773	0.9654	0.9652	0.8729	0.8858	0.9776	0.9846	0.9488	0.7866

Paper Venue Classification. In academic networks, papers are often connected through citations, co-authorships, or shared topics. While prior studies typically assume a homophilic structure where related papers belong to the same venue, real-world academic graphs exhibit heterophily—papers from the same author often span multiple venues and disciplines. We study this using one existing dataset, `ogbn-mag`[20], and 4 new datasets: `mag-year`, which re-labels `ogbn-mag` based on publication years to highlight temporal label shifts, and `oag-cs`, `oag-eng`, and `oag-chem`, which are newly constructed from the Open Academic Graph[59], and reflect disciplinary diversity.

Social Network Analysis. Social networks provide another example of graphs with both heterophily and heterogeneity. Unlike traditional homophilic assumptions, where friends tend to share similar attributes, real-world social structures reveal connections across diverse demographic and interest groups. Our new `H-Pokec` dataset, derived from the Pokec social network [27], introduces heterophilic relationships influenced by user demographics and personal affiliations, such as shared hobbies or cultural interests.

Financial Fraud Detection. Fraudulent activities in financial transactions and e-commerce platforms often follow heterophilic patterns: fraudsters attempt to disguise themselves by mimicking normal behaviors with innocent nodes while still forming distinct interaction patterns. Meanwhile, financial networks are inherently heterogeneous, consisting of multiple entity types such as users, businesses, and transactions. Our dataset collection includes a new `IEEE-CIS-G` graph dataset (developed from a Kaggle tabular dataset [18] for credit card fraud detection in the finance domain) and a repurposed `RCDD` [32] dataset (for risk commodity detection in e-commerce domain), both of which capture the heterophilic and heterogeneous nature of financial interactions.

Malware Detection. Malicious entities on the Internet, such as botnets and phishing domains, do not always form homophilic clusters—they attempt to infiltrate and blend in with legitimate entities. The repurposed `PDNS` dataset [25] models such behaviors in cybersecurity by representing domain name system (DNS) interactions as a heterogeneous graph, where malicious and benign domains interact with different network entities, making detection a challenging task.

3.2 Data Standardization

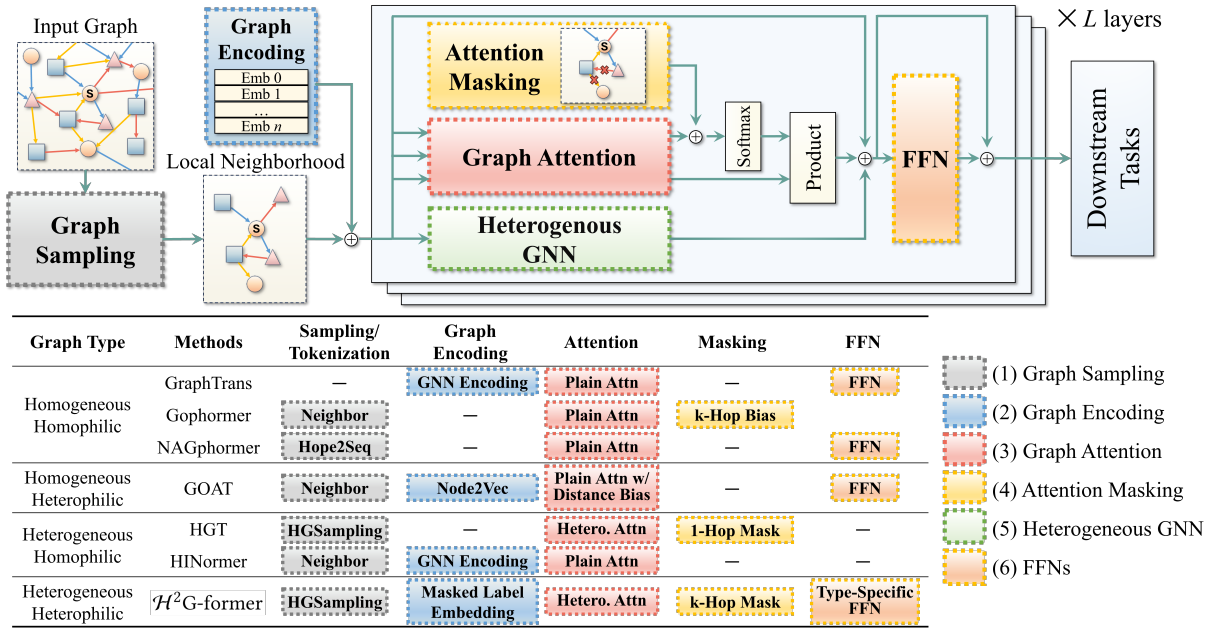
To ensure consistency, we clean, preprocess, and format all datasets in $\mathcal{H}^2$GB following a standardized pipeline. We encapsulate each dataset in the widely used `HeteroData` object format, supported by the PyTorch Geometric (PyG) library, ensuring seamless compatibility with existing heterogeneous graph learning frameworks. Dataset details are provided in Table 1 and Appendix B.2.

3.2.1 ***Data Formatting and Structure.*** Each dataset is carefully processed to maintain diverse node/edge types and meaningful graph structures. We ensure that (1) **node features are consistently structured**, meaning they share a common representation format across datasets (e.g., numerical embeddings or categorical encodings), facilitating cross-dataset comparisons and model training; and (2) **heterogeneous graph information is retained**, with explicit node and edge type definitions stored in the widely used PyG `HeteroData` format, ensuring compatibility with heterogeneous GNN models. To facilitate reproducibility and extensibility, new datasets can be integrated into $\mathcal{H}^2$GB using our dataset construction script templates, allowing users to format and pre-process data consistently within the framework.

3.2.2 ***Splitting Strategy.*** For most datasets, we employ a temporal split scheme, ensuring that the training set precedes the validation set in time, and the validation set precedes the test set. This strategy aligns with real-world prediction scenarios, where models must generalize to future data rather than relying on randomly shuffled samples. Two exceptions are `mag-year`, where publication year is the prediction target and thus unsuitable for temporal splitting, and `H-Pokec`, which lacks timestamp information.

3.3 Data Quantification ($\mathcal{H}^2$ Index)

To better characterize the structural properties of our datasets, we systematically quantify heterophily in heterogeneous contexts using several standard heterophily metrics and our new metric, the $\mathcal{H}^2$ index (Table 2).



Graph Type	Methods	Sampling/ Tokenization	Graph Encoding	Attention	Masking	FFN
Homogeneous Homophilic	GraphTrans	—	GNN Encoding	Plain Attn	—	FFN
	Gophormer	Neighbor	—	Plain Attn	k-Hop Bias	
	NAGphormer	Hope2Seq	—	Plain Attn	—	FFN
Homogeneous Heterophilic	GOAT	Neighbor	Node2Vec	Plain Attn w/ Distance Bias	—	FFN
Heterogeneous Homophilic	HGT	HGSampling	—	Hetero. Attn	1-Hop Mask	—
	HINormer	Neighbor	GNN Encoding	Plain Attn	—	—
Heterogeneous Heterophilic	$\mathcal{H}^2$G-former	HGSampling	Masked Label Embedding	Hetero. Attn	k-Hop Mask	Type-Specific FFN

(1) Graph Sampling
(2) Graph Encoding
(3) Graph Attention
(4) Attention Masking
(5) Heterogeneous GNN
(6) FFNs

Figure 3: The modular modeling framework (UnifiedGT) provided by $\mathcal{H}^2$GB. We choose several example models from the 28 baselines to demonstrate how they can be reproduced via the modular components provided by the modeling framework.

3.3.1 ***New Heterogeneous Heterophily Metric.*** Inspired by the adjusted heterophily metric [44, 47], we propose the *class-adjusted heterogeneous heterophily index* $\mathcal{H}^2$, formulated as follows:

$$\mathcal{H}^2(\mathcal{G}) = \text{Agg}\left(\mathcal{H}_{\text{adj}}(\mathcal{G}_{\mathcal{P}}) \,|\mathcal{P} \in \mathcal{M}_k\right), \tag{2}$$

where $\mathcal{G}_{\mathcal{P}}$ denotes a metapath-induced subgraph, $\mathcal{H}_{\text{adj}}$ is the adjusted heterophily, $\mathcal{M}_k$ is a k-hop metapath set, and Agg $\in$ {mean, max} is an aggregation function. Intuitively, the adjusted heterophily $\mathcal{H}_{\text{adj}}$ quantifies the degree of heterophily relative to what would be expected in a random graph. Under the random graph configuration model described in [40], where for every node v we create $d(v)$ copies of it and then find a random matching among all nodes, the likelihood of a given edge endpoint connecting to a node of class k is approximately $D_k/(2|\mathcal{E}|)$ (as assumed in [44]). Thus, the expected heterophily is the likelihood that two edge endpoints are in different classes, which is $1 - \sum_{k=1}^{C} D_k(D_k-1)/((2|\mathcal{E}|)(2|\mathcal{E}|-1)) \approx 1 - \sum_{k=1}^{C} D_k^2/(2|\mathcal{E}|)^2$. As a result, a value of $\mathcal{H}^2$ close to 0 indicates that nodes predominantly connect to other nodes of the same class, exhibiting homophily. A value approaching or exceeding 1 suggests that nodes are more likely to connect to nodes of different classes, demonstrating heterophily. The set of all possible metapaths $\mathcal{P}$ can potentially be large, and so we introduce an additional constraint where only length-2 metapaths are considered. We select the mean function as the aggregation function to reflect the general heterophily across all metapaths. The $\mathcal{H}^2$ value for each dataset is presented in Table 2.

3.4 Standard Workflow

We establish a standard workflow for model developers and application researchers to use $\mathcal{H}^2$GB, as shown in Figure 2.

- *Application Researchers* can search for effective models for their new dataset/application domain as follows:
  (1) **Identify new applications** requiring heterophilic and heterogeneous graph learning.
  (2) **Build and integrate dataset** into $\mathcal{H}^2$GB.
  (3) **Choose models** from our modeling framework.
  (4) **Benchmark performance.**
- *Model Developers* can perform model development as follows:
  (1) **Implement new models** by modifying models in $\mathcal{H}^2$GB.
  (2) **Evaluate models** to understand performance gaps.
  (3) **Iterate** to refine scalable heterophilic and heterogeneous learning approaches.

4 Modular Modeling Framework

To facilitate standardized benchmarking, $\mathcal{H}^2$GB incorporates UnifiedGT [31], a modular modeling framework that we previously designed that is capable of expressing various GNN architectures, as shown in Figure 3. UnifiedGT provides a structured approach to decomposing graph learning models into modular components, including graph sampling, encoding, attention mechanisms, heterogeneous GNN, and feedforward networks (FFN), allowing flexible integration of different modeling techniques.

The modeling framework enables flexible experiments and performance comparisons across 28 state-of-the-art baseline models, reducing implementation variability and simplifying the process of integrating new models into $\mathcal{H}^2$GB. The modeling framework provides simple baselines and three categories of state-of-the-art GNN and graph transformer models. The simple baselines include models that only consider node features, such as MLP [12], and models that only consider graph topology, such as label propagation

Table 3: Benchmark results of various GNN methods. Standard deviations are calculated over 5 runs with different random seeds. We highlight the first and second best results. Label propagation (LP) has deterministic results. Out-of-memory (OOM) indicates the method ran out of memory on an Nvidia V100 GPU with 32GB of memory. §§: Heterogeneous Heterophilic.

	Datasets→ ($\mathcal{H}^2$ Index)	Avg. Rank	Accuracy						F1 score		
	Methods↓		ogbn-mag (0.8773)	mag-year (0.9654)	oag-cs (0.9652)	oag-eng (0.8729)	oag-chem (0.8858)	H-Pokec (0.9488)	RCDD (0.9776)	IEEE-CIS-G (0.9846)	PDNS (0.7866)
	MLP	23.2	27.27 ± 0.50	26.52 ± 0.64	09.26 ± 0.51	20.18 ± 0.92	13.61 ± 0.41	62.75 ± 0.34	75.87 ± 1.38	04.26 ± 8.52	73.92 ± 0.66
Graph Only	LP+1Hop	18.9	38.36	26.61	19.79	36.07	22.48	45.42	67.07	0.00	81.53
Graph Only	LP+2Hop	14.8	37.38	39.45	20.98	36.73	21.54	76.72	67.84	0.00	82.13
Graph Only	SGC+1Hop	24.6	16.46 ± 0.24	26.48 ± 0.17	06.42 ± 0.17	10.93 ± 3.18	07.02 ± 1.72	52.91 ± 0.43	05.47 ± 6.92	13.04 ± 3.53	74.24 ± 1.90
Graph Only	SGC+2Hop	25.2	14.28 ± 0.28	26.46 ± 0.05	06.09 ± 0.50	08.77 ± 1.22	05.00 ± 1.10	59.55 ± 1.75	06.07 ± 5.29	07.98 ± 8.54	61.34 ± 1.14
Homogeneous Homophilic	GCN	14.3	42.90 ± 0.50	32.91 ± 0.50	18.22 ± 0.60	29.09 ± 0.52	18.57 ± 1.06	70.63 ± 0.36	85.81 ± 0.87	28.79 ± 1.07	81.22 ± 0.30
Homogeneous Homophilic	GraphSAGE	8.4	40.80 ± 0.56	36.28 ± 0.19	22.92 ± 0.29	36.16 ± 0.20	24.66 ± 0.48	77.29 ± 0.30	85.02 ± 0.83	31.49 ± 1.23	91.44 ± 0.32
Homogeneous Homophilic	GAT	11.7	48.60 ± 0.29	33.50 ± 0.62	19.12 ± 0.25	28.74 ± 0.60	14.05 ± 0.44	70.89 ± 0.20	86.71 ± 1.27	28.51 ± 0.45	93.97 ± 0.27
Homogeneous Homophilic	GIN	15.6	37.32 ± 0.33	31.15 ± 0.54	16.33 ± 1.34	29.62 ± 1.15	17.86 ± 0.62	74.72 ± 0.32	84.22 ± 0.34	28.53 ± 0.54	87.91 ± 0.46
Homogeneous Homophilic	APPNP	18.3	37.64 ± 0.31	29.79 ± 0.61	17.90 ± 0.60	28.63 ± 0.40	17.19 ± 1.06	57.27 ± 1.22	82.95 ± 0.67	27.27 ± 1.47	80.70 ± 0.73
Homogeneous Homophilic	NAGphormer	11.7	42.47 ± 0.74	32.60 ± 0.06	16.49 ± 0.55	31.85 ± 0.80	23.78 ± 0.35	80.59 ± 0.15	85.46 ± 0.50	17.07 ± 0.34	92.37 ± 0.22
Homogeneous Homophilic	GraphTrans	13.7	47.25 ± 1.54	36.14 ± 0.41	02.39 ± 0.22	06.55 ± 3.53	02.23 ± 0.20	77.80 ± 0.17	86.00 ± 0.56	30.53 ± 1.60	93.00 ± 0.39
Homogeneous Homophilic	Gophormer	16.6	42.87 ± 0.64	35.17 ± 0.27	03.68 ± 1.24	10.42 ± 3.73	04.26 ± 2.85	71.55 ± 2.04	80.56 ± 6.13	30.79 ± 1.06	91.58 ± 0.05
Homogeneous Heterophilic	MixHop	6.4	46.99 ± 0.41	36.36 ± 0.28	23.04 ± 0.24	36.88 ± 0.73	25.03 ± 0.90	78.78 ± 0.27	85.43 ± 1.22	30.13 ± 0.86	92.78 ± 0.18
Homogeneous Heterophilic	LINKX	12.0	40.83 ± 0.18	42.81 ± 0.14	15.32 ± 0.08	32.85 ± 0.38	22.98 ± 0.24	79.66 ± 0.94	OOM	31.42 ± 1.20	87.74 ± 0.52
Homogeneous Heterophilic	FAGCN	20.9	33.06 ± 0.59	27.10 ± 0.66	10.46 ± 0.44	22.75 ± 0.94	13.01 ± 0.44	67.15 ± 0.09	81.06 ± 1.24	10.09 ± 5.09	82.84 ± 1.07
Homogeneous Heterophilic	ACM-GCN	21.1	33.50 ± 1.13	23.20 ± 1.21	11.23 ± 0.75	22.27 ± 0.77	13.81 ± 0.43	66.69 ± 0.09	75.52 ± 1.74	16.98 ± 0.29	88.48 ± 0.48
Homogeneous Heterophilic	LSGNN	13.6	38.87 ± 0.83	40.47 ± 0.58	15.20 ± 0.60	29.43 ± 0.74	19.96 ± 0.69	78.37 ± 0.49	83.84 ± 0.91	14.68 ± 1.86	88.91 ± 0.17
Homogeneous Heterophilic	GOAT	10.3	41.59 ± 0.09	32.92 ± 0.41	20.74 ± 0.39	35.82 ± 0.52	21.75 ± 0.17	76.55 ± 0.71	87.13 ± 0.45	30.31 ± 0.73	91.71 ± 0.27
Homogeneous Heterophilic	PolyFormer	17.2	35.58 ± 0.24	31.13 ± 0.50	09.22 ± 0.24	21.4 ± 0.62	15.26 ± 0.50	70.74 ± 0.10	83.61 ± 0.69	17.26 ± 0.07	94.81 ± 0.09
Heterogeneous Homophilic	R-GCN	5.3	46.93 ± 0.46	35.60 ± 0.48	23.10 ± 1.09	37.10 ± 0.49	25.80 ± 0.32	78.05 ± 0.28	87.00 ± 1.35	31.44 ± 0.96	92.55 ± 0.44
Heterogeneous Homophilic	R-GraphSAGE	6.0	50.94 ± 0.44	38.07 ± 0.41	22.81 ± 0.63	36.11 ± 0.45	26.00 ± 0.59	77.00 ± 0.32	86.81 ± 1.74	29.85 ± 0.47	92.81 ± 0.37
Heterogeneous Homophilic	R-GAT	11.0	41.51 ± 0.47	35.40 ± 0.88	21.03 ± 0.59	35.90 ± 0.60	26.14 ± 0.34	67.17 ± 0.24	80.37 ± 0.62	22.09 ± 0.94	94.29 ± 0.16
Heterogeneous Homophilic	HAN	19.1	39.00 ± 0.22	29.66 ± 0.43	13.14 ± 1.96	27.81 ± 0.69	17.03 ± 0.66	54.04 ± 2.17	78.56 ± 1.42	23.15 ± 0.43	84.58 ± 0.76
Heterogeneous Homophilic	HGT	5.9	50.23 ± 0.48	39.47 ± 1.66	22.51 ± 0.40	35.51 ± 0.52	25.48 ± 0.76	78.91 ± 0.43	86.05 ± 1.01	30.89 ± 0.80	92.76 ± 0.15
Heterogeneous Homophilic	HINormer	27.7	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM
Heterogeneous Homophilic	SHGN	11.0	43.39 ± 0.28	34.43 ± 1.23	22.03 ± 0.46	36.93 ± 0.67	24.07 ± 0.94	50.50 ± 0.89	79.67 ± 2.53	31.66 ± 0.86	89.33 ± 0.21
§§	$\mathcal{H}^2$G-former	1.1	55.67 ± 0.35	52.55 ± 0.66	28.47 ± 0.93	46.63 ± 0.65	30.62 ± 0.31	82.45 ± 0.19	87.35 ± 0.80	31.55 ± 0.92	96.43 ± 0.21

(LP, one and two hops) [42, 62], as well as a simple GNN model that focuses on aggregation of neighborhood information with reduced nonlinearities and weight matrices, SGC [53]. The first class of GNN baselines, designed for *homogeneous homophilic* graphs, includes GCN [23], GraphSAGE [15], GAT [49], GIN [55], APPNP [10], NAGphormer [5], GraphTrans [54], and Gophormer [60]. The second class of baselines, optimized for *homogeneous heterophilic* graphs, includes MixHop [1], LINKX [30], FAGCN [4], ACM-GCN [35], LSGNN [6], GOAT [24], and PolyFormer [37]. The third class of baselines, designed for *heterogeneous homophilic* graphs, includes relational GCN (R-GCN) [46], GraphSAGE (R-GraphSAGE), GAT (R-GAT), HAN [51], HGT [21], HINormer [38], and SHGN [36]. Lastly, we present a new model $\mathcal{H}^2$G-former designed for *heterogeneous heterophilic* graphs, developed following our established workflow (Section 5.3). The detailed descriptions of each model can be found in Appendix C.1.

5 Experiments

In this section, we conduct comprehensive experiments to evaluate existing and proposed methods in $\mathcal{H}^2$GB using an Nvidia V100 GPU with 32GB of memory. The homogeneous methods ignore the node and edge types.

5.1 General Setup

5.1.1 ***Training and Evaluation.*** The dataset splits can be found at Table 1, where most of the split strategy is based on timestamps on the nodes. Test performance is reported for the learned parameters corresponding to the highest validation performance. We use F1 score as the metric for the datasets with large class imbalance, as it is less sensitive to class imbalance than accuracy. For the other datasets, we use classification accuracy as the metric.

5.1.2 ***Minibatching Sampling.*** Most existing heterophilic GNNs are designed for small graphs and struggle to scale to large graphs.

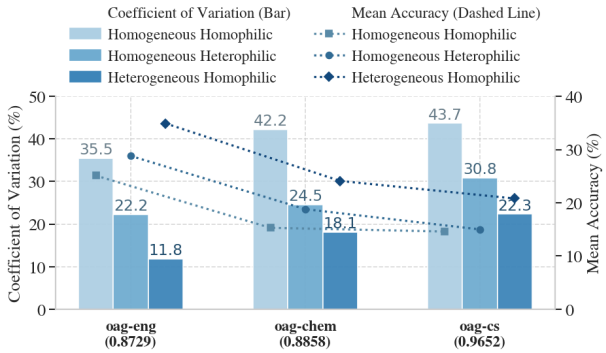


Figure 4: Model group performance versus heterophily. The coefficient of variation is the standard deviation of model accuracy in each group normalized by the mean accuracy. The $\mathcal{H}^2$ index is indicated under each dataset name.

To enable training on large graphs, our framework supports optional minibatching, where models process sampled local neighborhoods instead of the full graph. In our experiments, we adopt minibatching for scalability, using a consistent sampling strategy across all models within each dataset to ensure fair comparison. While some models may benefit from specialized sampling, varying strategies would introduce confounding factors that obscure model-level effects.

5.2 Experimental Results

Table 3 lists the results of each method across the datasets proposed in $\mathcal{H}^2$GB. We make the following observations:

(1) **$\mathcal{H}^2$G-former consistently outperforms baselines across diverse graph structures.** It achieves the best average rank (1.1) and consistently outperforms or matches the existing methods on all of the datasets. This highlights its ability to effectively capture both heterophilic and heterogeneous structures, reinforcing the need for models tailored to such real-world graphs.

(2) **Homogeneous heterophilic GNNs struggle with heterogeneous graphs.** While methods like MixHop and GOAT outperform homogeneous homophilic GNNs in our benchmark, achieving a better average rank, their advantage diminishes when compared to heterogeneous homophilic GNNs. This performance degradation primarily stems from their inability to effectively incorporate diverse node and edge types. For example, the semantic meaning of each type of node can be different, resulting in different distributions in the node features. These homogeneous heterophilic GNNs cannot adjust their parameters to learn from node features of different distributions.

(3) **Performance of heterogeneous homophilic GNNs depends on their ability to handle heterophily.** The performance of heterogeneous models varies significantly, likely due to differences in their architectural robustness when exposed to heterophily. For instance, models relying on local attention mechanisms (e.g., R-GAT, HAN, and SHGN compute attention over 1-hop neighbors) generally underperform. We quantitatively illustrate this in Figure 4, where we select three datasets from a single domain (academic networks), with similar heterogeneity (number of nodes/edge types) but different heterophily. We evaluate the performance variations within each model group, and can clearly observe that datasets with higher heterophily (e.g., `oag-cs`) show greater variations across models within the group. Consistent with observation (2), we also observe that heterogeneous models perform better, with lower variations and higher mean accuracy, emphasizing the importance of effectively handling the different node and edge types in achieving good task performance. Building on this insight, our $\mathcal{H}^2$G-former incorporates k-hop attention, instead of 1-hop attention, and considers the graph heterogeneity, leading to improved performance.

(4) **Scalability issues in existing GNNs.** A significant gap exists between the best and worst-performing homogeneous heterophilic GNNs, particularly as the graph size increases. Many of these GNNs were designed for small-scale datasets and full-graph training and struggle when trained on large-scale graphs using mini-batching. For example, FAGCN and ACM-GCN show degraded performance, consistent with observations in the previous work [30]. This underscores the need for scalable architectures that can handle both heterophily and heterogeneity.

(5) **Dataset-specific insights: how performance varies by domain.** Our results demonstrate that certain model types perform well in specific domains but fail in others, emphasizing the importance of a diverse benchmark. In academic networks (e.g., `ogbn-mag` and `oag-cs`), R-GraphSAGE and R-GCN perform well, leveraging hierarchical information from paper-author-affiliation relationships. Homogeneous heterophilic models struggle, as they lack relational reasoning over entity types. In e-commerce and security networks (e.g., `RCDD` and `PDNS`), GOAT and PolyFormer perform well, suggesting that effective handling of long-range dependencies and robust graph structure encoding are crucial in fraud and security applications. In social networks (e.g., `H-Pokec`), the homophilic model NAGphormer performs surprisingly well, likely due to its ability to aggregate information from multi-hop neighborhoods, effectively capturing long-range homophilic signals. We also observe that models leveraging heterophilic signals, such as MixHop and LSGNN, achieve relatively strong performance by addressing heterophily among labeled users. However, they still underperform compared to $\mathcal{H}^2$G-former, as they fail to exploit the rich metapath information embedded in the graph.

5.3 Case Study: $\mathcal{H}^2$GB for Model Development

$\mathcal{H}^2$GB provides user-friendly examples (Figure 5) and facilitates research following our standardized workflow. We present a case study on the construction of the `oag-cs` dataset and the development of the $\mathcal{H}^2$G-former model.

- **Step 1: Identifying Application.** We aim to predict which venue a computer science paper will be published in, a challenging task due to the diverse paper-author-affiliation interactions and interdisciplinary nature of research.
- **Step 2: Building and Standardizing Dataset.** Using the Open Academic Graph (OAG), we extract papers in the computer science field to construct an academic network. We represent

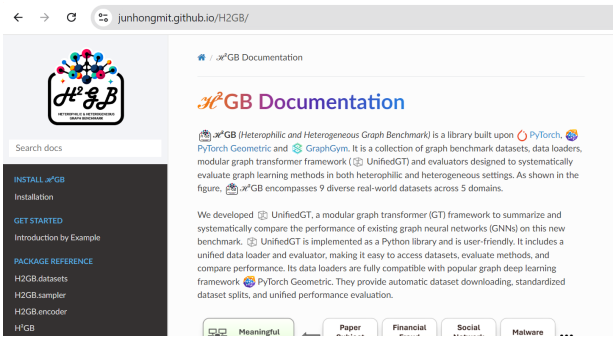


Figure 5: $\mathcal{H}^2$GB has a user-friendly website and provides an introduction with examples.

node features using paper abstract embeddings and define multiple node types, including papers, authors, affiliations, and topics, along with their interactions as edge types. Publication venues serve as node labels. This dataset is integrated into $\mathcal{H}^2$GB as `oag-cs` and made accessible through our standardized data loader.

- **Step 3: Evaluating Baselines and Identifying Limitations.** We evaluated all baselines and found the best accuracy to be 23.10%, meaning that fewer than a quarter of papers are correctly classified. This suggests room for improvement.
- **Step 4: Iterative Model Development using Modular Components.** To demonstrate how our benchmark can facilitate principled model design, we use UnifiedGT to systematically enhance a strong baseline, HGT (22.51%). As shown in Figure 3, HGT consists of: *HGSampling* (Graph Sampling), *Heterogeneous Attention* (Graph Attention), and *1-Hop Mask* (Attention Masking). We experiment with component-level modifications: replacing the 1-Hop Mask with *k-Hop Mask* (enabling better context capture), enhancing graph encoding with *masked label embeddings* (which assists in predicting node labels), and introducing a *Type-Specific FFN* since HGT lacks a dedicated FFN before the output. This modular modification process results in our new method, $\mathcal{H}^2$G-former, illustrating how $\mathcal{H}^2$GB enables targeted model development through interpretable architecture changes.
- **Step 5: Results.** With these modifications, the accuracy improves to 28.47% shown in Table 3, a 5.37% improvement over the best baseline. This demonstrates how $\mathcal{H}^2$GB enables systematic model evaluation and component-wise experimentation, making it a powerful toolbox for benchmarking and research.

6 Conclusion

We introduce $\mathcal{H}^2$GB, a comprehensive benchmark for evaluating graph learning models on large-scale real-world heterophilic and heterogeneous graphs. We provide a unified benchmarking library with a standardized data loader, evaluator, and extensible framework for systematic experimentation. Our comprehensive benchmarking on 28 baseline models highlights the challenges posed by heterophilic and heterogeneous graphs and provides insights into model performance. Through a case study, we demonstrate how $\mathcal{H}^2$GB facilitates model selection and guides the development of improved methods such as $\mathcal{H}^2$G-former. We believe $\mathcal{H}^2$GB serves as a vital resource for advancing scalable and realistic graph learning research. Directions for future work include incorporating more datasets into $\mathcal{H}^2$GB and extending datasets and models to other tasks such as link prediction and node regression.

Acknowledgments

This work is funded by the MIT-IBM AI Watson Lab, NSF awards #CCF-1845763, #CCF-2316235, and #CCF-2403237, Google Faculty Research Award, and Google Research Scholar Award. We thank Dawei Zhou (Virginia Tech) for his valuable feedback and guidance.

References

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. 2019. Mixhop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing. In *International Conference on Machine Learning (ICML)*. PMLR, 21–29.
[2] Erik Altman, Jovan Blanuša, Luc Von Niederhäusern, Béni Egressy, Andreea Anghel, and Kubilay Atasu. 2024. Realistic Synthetic Financial Transactions for Anti-Money Laundering Models. *Advances in Neural Information Processing Systems (NeurIPS)* 36 (2024).
[3] M. Aravind, VG Sujadevi, Manu R Krishnan, Prem Sankar Au, Soumajit Pal, Anu Vazhayil, Geetapriya Sridharan, and Prabaharan Poornachandran. 2022. Malicious Node Identification for DNS Data Using Graph Convolutional Networks. In *IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, Vol. 7. 104–109.
[4] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. 2021. Beyond Low-Frequency Information in Graph Convolutional Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 35. 3950–3957.
[5] Jinsong Chen, Kaiyuan Gao, Gaichao Li, and Kun He. 2022. NAGphormer: A Tokenized Graph Transformer for Node Classification in Large Graphs. In *The International Conference on Learning Representations (ICLR)*.
[6] Yuhan Chen, Yihong Luo, Jing Tang, Liang Yang, Siya Qiu, Chuan Wang, and Xiaochun Cao. 2023. LSGNN: Towards General Graph Neural Network in Node Classification by Local Similarity. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. 3550–3558.
[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. 4171–4186.
[8] Matthias Fey and Jan Eric Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.
[9] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. MAGNN: Metapath Aggregated Graph Neural Network for Heterogeneous Graph Embedding. In *Proceedings of the International Conference on World Wide Web (WWW)*. 2331–2341.
[10] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict Then Propagate: Graph Neural Networks Meet Personalized PageRank. In *International Conference on Learning Representations*.
[11] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural Message Passing for Quantum Chemistry. In *International Conference on Machine Learning (ICML)*. 1263–1272.
[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.
[13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning. *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), 21271–21284.
[14] Jiayan Guo, Lun Du, Wendong Bi, Qiang Fu, Xiaojun Ma, Xu Chen, Shi Han, Dongmei Zhang, and Yan Zhang. 2023. Homophily-Oriented Heterogeneous Graph Rewiring. In *Proceedings of the International Conference on World Wide Web (WWW)*. 511–522.
[15] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).
[16] Haoyu He, Yuede Ji, and H Howie Huang. 2022. Illuminati: Towards Explaining Graph Neural Networks for Cybersecurity Analysis. In *IEEE European Symposium on Security and Privacy (EuroS&P)*. 74–89.
[17] Huiting Hong, Hantao Guo, Yucheng Lin, Xiaoqing Yang, Zang Li, and Jieping Ye. 2020. An Attention-Based Graph Neural Network for Heterogeneous Structural

Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 34. 4132–4139.
[18] Addison Howard, Bernadette Bouchon-Meunier, IEEE CIS, John Lei, Lynn@Vesta, Marcus2010, and Hussein Abbass. 2019. IEEE-CIS Fraud Detection. Kaggle. https://www.kaggle.com/competitions/ieee-fraud-detection
[19] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. 2021. OGB-LSC: A Large-Scale Challenge for Machine Learning on Graphs. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*.
[20] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open Graph Benchmark: Datasets for Machine Learning on Graphs. *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), 22118–22133.
[21] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. In *Proceedings of the International Conference on World Wide Web (WWW)*. 2704–2710.
[22] Arpandeep Khatua, Vikram Sharma Mailthody, Bhagyashree Taleka, Tengfei Ma, Xiang Song, and Wen-mei Hwu. 2023. IGB: Addressing The Gaps In Labeling, Features, Heterogeneity, and Size of Public Graph Datasets for Deep Learning Research. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.
[23] Thomas N Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.
[24] Kezhi Kong, Jiuhai Chen, John Kirchenbauer, Renkun Ni, C Bayan Bruss, and Tom Goldstein. 2023. GOAT: A Global Transformer on Large-Scale Graphs. In *Proceedings of the International Conference on Machine Learning (ICML)*. 17375–17390.
[25] Udesh Kumarasinghe, Fatih Deniz, and Mohamed Nabeel. 2022. PDNS-Net: A Large Heterogeneous Graph Benchmark Dataset of Network Resolutions for Graph Learning. *arXiv preprint arXiv:2203.07969* (2022).
[26] Jure Leskovec and Julian McAuley. 2012. Learning to Discover Social Circles in Ego Networks. *Advances in Neural Information Processing Systems (NeurIPS)* 25 (2012).
[27] Jure Leskovec and Rok Sosič. 2016. SNAP: A General-Purpose Network Analysis and Graph-Mining Library. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 1 (2016), 1–20.
[28] Jintang Li, Zheng Wei, Jiawang Dan, Jing Zhou, Yuchang Zhu, Ruofan Wu, Baokun Wang, Zhang Zhen, Changhua Meng, Hong Jin, et al. 2023. Hetero2Net: Heterophily-Aware Representation Learning on Heterogeneous Graphs. *arXiv preprint arXiv:2310.11664* (2023).
[29] Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. 2022. Finding Global Homophily in Graph Neural Networks When Meeting Heterophily. In *International Conference on Machine Learning (ICML)*. 13242–13256.
[30] Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser Nam Lim. 2021. Large Scale Learning on Non-Homophilous Graphs: New Benchmarks and Strong Simple Methods. *Advances in Neural Information Processing Systems (NeurIPS)* 34 (2021), 20887–20902.
[31] Junhong Lin, Xiaojie Guo, Shuaicheng Zhang, Dawei Zhou, Yada Zhu, and Julian Shun. 2024. UnifiedGT: Towards a Universal Framework of Transformers in Large-Scale Graph Learning. In *Proceedings of the 2024 IEEE International Conference on Big Data (IEEE Big Data 2024)*.
[32] Yijian Liu, Hongyi Zhang, Cheng Yang, Ao Li, Yugang Ji, Luhao Zhang, Tao Li, Jinyu Yang, Tianyu Zhao, Juan Yang, et al. 2023. Datasets and Interfaces for Benchmarking Heterogeneous Graph Neural Networks. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*. 5346–5350.
[33] Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic Gradient Descent With Warm Restarts. In *International Conference on Learning Representations (ICLR)*.
[34] Sitao Luan, Chenqing Hua, Qincheng Lu, Liheng Ma, Lirong Wu, Xinyu Wang, Minkai Xu, Xiao-Wen Chang, Doina Precup, Rex Ying, et al. 2024. The heterophilic graph learning handbook: Benchmarks, models, theoretical analysis, applications and challenges. *arXiv preprint arXiv:2407.09618* (2024).
[35] Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. 2022. Revisiting Heterophily for Graph Neural Networks. *Advances in Neural Information Processing Systems (NeurIPS)* 35 (2022), 1362–1375.
[36] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are We Really Making Much Progress? Revisiting, Benchmarking and Refining Heterogeneous Graph Neural Networks. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 1150–1160.
[37] Jiahong Ma, Mingguo He, and Zhewei Wei. 2024. Polyformer: Scalable node-wise filters via polynomial graph transformer. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2118–2129.
[38] Qiheng Mao, Zemin Liu, Chenghao Liu, and Jianling Sun. 2023. Hinormer: Representation Learning on Heterogeneous Information Networks with Graph Transformer. In *Proceedings of the International Conference on World Wide Web (WWW)*. 599–610.
[39] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[40] Michael Molloy and Bruce A Reed. 1995. A Critical Point for Random Graphs with a Given Degree Sequence. *Random Structures & Algorithms* 6 (1995), 161–180.
[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).
[42] Leto Peel. 2017. Graph-Based Semi-Supervised Learning for Relational Networks. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*. 435–443.
[43] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.
[44] Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. 2024. Characterizing Graph Datasets for Node Classification: Homophily-Heterophily Dichotomy and Beyond. *Advances in Neural Information Processing Systems (NeurIPS)* 36 (2024).
[45] Susie Xi Rao, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang. 2021. xFraud: Explainable Fraud Transaction Detection. *Proceedings of the VLDB Endowment (PVLDB)* 15, 3 (2021), 427–436.
[46] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *European Semantic Web Conference (ESWC)*. 593–607.
[47] Susheel Suresh, Vinith Budde, Jennifer Neville, Pan Li, and Jianzhu Ma. 2021. Breaking the Limit of Graph Neural Networks by Improving the Assortativity of Graphs with Local Mixing Patterns. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 1541–1551.
[48] Lubos Takac and Michal Zabovsky. 2012. Data Analysis in Public Social Networks. In *International Scientific Conference and International Workshop Present Day Trends of Innovations*, Vol. 1.
[49] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*.
[50] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. 2019. A Semi-Supervised Graph Attentive Network for Financial Fraud Detection. In *IEEE International Conference on Data Mining (ICDM)*. 598–607.
[51] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous Graph Attention Network. In *Proceedings of the International Conference on World Wide Web (WWW)*. 2022–2032.
[52] Dana Warmsley, Alex Waagen, Jiejun Xu, Zhining Liu, and Hanghang Tong. 2022. A Survey of Explainable Graph Neural Networks for Cyber Malware Analysis. In *IEEE International Conference on Big Data (Big Data)*. 2932–2939.
[53] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying Graph Convolutional Networks. In *International Conference on Machine Learning (ICML)*. 6861–6871.
[54] Zhanghao Wu, Paras Jain, Matthew Wright, Azalia Mirhoseini, Joseph E Gonzalez, and Ion Stoica. 2021. Representing Long-Range Context for Graph Neural Networks with Global Attention. *Advances in Neural Information Processing Systems (NeurIPS)* 34 (2021), 13266–13279.
[55] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *International Conference on Learning Representations (ICLR)*.
[56] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).
[57] Jiaxuan You, Zhitao Ying, and Jure Leskovec. 2020. Design Space for Graph Neural Networks. *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), 17009–17021.
[58] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous Graph Neural Network. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 793–803.
[59] Fanjin Zhang, Xiao Liu, Jie Tang, Yuxiao Dong, Peiran Yao, Jie Zhang, Xiaotao Gu, Yan Wang, Bin Shao, Rui Li, et al. 2019. OAG: Toward Linking Large-Scale Heterogeneous Entity Graphs. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 2585–2595.
[60] Jianan Zhao, Chaozhuo Li, Qianlong Wen, Yiqi Wang, Yuming Liu, Hao Sun, Xing Xie, and Yanfang Ye. 2021. Gophormer: Ego-Graph Transformer for Node Classification. *arXiv preprint arXiv:2110.13094* (2021).
[61] Xin Zheng, Yixin Liu, Shirui Pan, Miao Zhang, Di Jin, and Philip S Yu. 2022. Graph Neural Networks for Graphs with Heterophily: A Survey. *arXiv preprint*

arXiv:2202.07082 (2022).
[62] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. 2003. Learning with Local and Global Consistency. *Advances in Neural Information Processing Systems (NeurIPS)* 16 (2003).
[63] Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. 2021. Graph Neural Networks with Heterophily. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 35. 11168–11176.
[64] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), 7793–7804.
[65] Shichao Zhu, Chuan Zhou, Shirui Pan, Xingquan Zhu, and Bin Wang. 2019. Relation Structure-Aware Heterogeneous Graph Neural Network. In *IEEE International Conference on Data Mining (ICDM)*. 1534–1539.

A Dataset Documentation, Metadata, and Intended Use

All datasets in $\mathcal{H}^2$GB are intended for academic use, and their corresponding licenses are described in Appendix B.1. We release our $\mathcal{H}^2$GB as an open-source library under the MIT license. For ease of access, we provide the following links to the $\mathcal{H}^2$GB benchmark suite and UnifiedGT framework:

- The open-source library is at https://github.com/junhongmit/H2GB/.
- The $\mathcal{H}^2$GB `Python` package is at https://pypi.org/project/H2GB.
- Datasets and documentation are at https://junhongmit.github.io/H2GB/.

Croissant Metadata. Croissant metadata records documenting each dataset can be found at

- `ogbn-mag`, `mag-year`: Croissant metadata.
- `oag-cs`, `oag-eng`, `oag-chem`: Croissant metadata.
- `RCDD`: Croissant metadata.
- `IEEE-CIS`: Croissant metadata.
- `H-Pokec`: Croissant metadata.
- `PDNS`: Croissant metadata.

B Additional Dataset Details

B.1 Licenses

In this section, we indicate the licenses of the collected datasets:

- `ogbn-mag`, `mag-year`, `oag-cs`, `oag-eng`, `oag-chem`: **ODC-BY**. Licensed via Open Graph Benchmark [20] and Open Academic Graph [59].
- `RCDD`: **CC BY 4.0**. Publicly released [32]. Node/edge type names are redacted for confidentiality; features are numeric.
- `IEEE-CIS`: Released via the IEEE CIS Kaggle challenge [18], with anonymized transaction records and numeric-only features. To the best of our knowledge, it was not released with a license.
- `Pokec`: **BSD**. Provided via SNAP [27, 48]. Text features are removed; only numeric features are retained for privacy.
- `PDNS`: Publicly released [25], with anonymized graphs and numeric-only features. To the best of our knowledge, the dataset was not released with a license.

B.2 Dataset Details.

All datasets in $\mathcal{H}^2$GB are formatted as `HeteroData` objects compatible with PyTorch Geometric. We summarize each dataset below.

- `ogbn-mag` [20]: A heterogeneous academic graph with papers, authors, institutions, and fields of study, connected via four relation types. Paper nodes have 128-dimensional Word2Vec [39] features; others are initialized via mean aggregation. Labels denote paper venues. We adopt the official temporal split: training (pre-2018), validation (2018), testing (post-2018).
- `mag-year` [20]: Same structure as `ogbn-mag`, but paper labels correspond to publication year buckets (5 balanced classes).
- `oag-cs`, `oag-eng`, and `oag-chem` [59]: Subsets of OAG for computer science, engineering, and chemistry, respectively. Entities and relations match `ogbn-mag`. Paper nodes use 768-dim XLNet [56] embeddings of their titles. Labels are paper venues. We apply a temporal split: train (pre-2017), val (2017), test (post-2017).
- `RCCD` (Risk Commodity Detection Dataset) [32]: A large-scale heterogeneous e-commerce graph from Alibaba. Node/edge types (except for items) are anonymized. Item nodes have 256-dimemsional features (BERT [7] + BYOL [13]). Labels indicate risk commodities (binary). We follow the official split, where the validation set is split from the training set, and the test set is obtained over time.
- `IEEE-CIS-G` [18]: A bipartite financial graph from a Kaggle fraud detection dataset. Nodes include transactions and 11 types of transaction metadata (e.g., card info, email domains). Edges link transactions to metadata (22 relation types). Each transaction has a 4823-dimensional feature vector. Fraud labels are binary; 4% are positive. A temporal split is used for evaluation.
- `H-Pokec` [48]: A social network graph with users and hobby club entities. Edges capture friendships and affiliations. User nodes have 66-dimensional profile-based features and gender labels. We apply a random split.
- `P-DNS` [25]: A cybersecurity graph of domain and IP nodes from passive DNS logs. Edges include resolutions and domain similarity. Domain nodes have 10-dimensional features (e.g., subdomain count, impersonation flags) and binary labels for maliciousness. We use a temporal split based on resolution time.

Figure 6 illustrates the heterogeneous graph schema for each dataset. Each schema is a type-level graph, where nodes represent node types and edges denote relation types. Legends indicate the number of nodes and edges per type.

C Experiment Setup

Experiments are implemented in Python 3.9 using PyTorch 2.0.1 [41] (**BSD-3 license**) and PyTorch Geometric 2.5.0 [8] (**MIT license**). UnifiedGT builds on GraphGym [57] (**MIT license**), offering modular components and flexible configuration. We provide experiment configurations for full reproducibility. All training and preprocessing were conducted on an Nvidia V100 GPU (32GB memory).

C.1 Additional Details of Baselines

Baselines include five groups: (1) node-only methods, (2) structure-only methods, (3) homogeneous homophilic GNNs, (4) homogeneous heterophilic GNNs, and (5) heterogeneous homophilic GNNs.

(1) ***Node-only.*** **MLP** [12] ignores the graph structure.

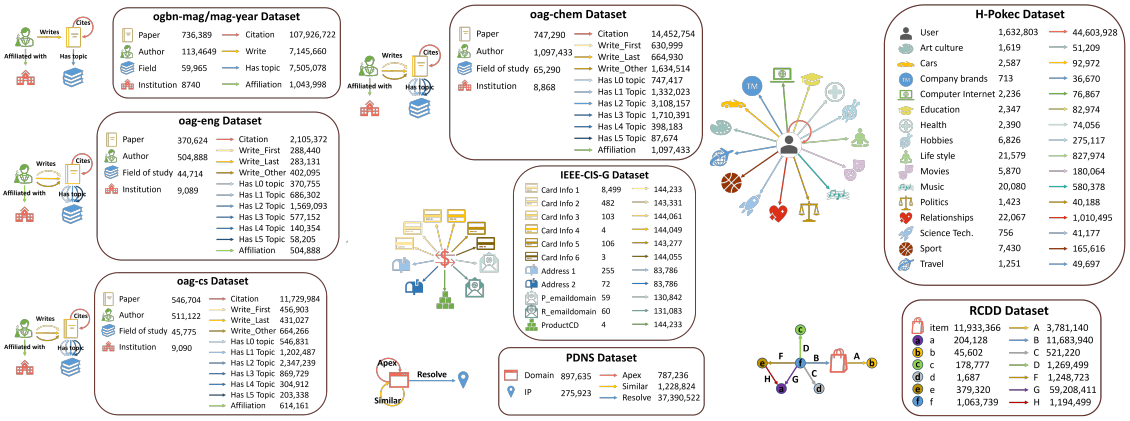


Figure 6: The schema and node/edge information of each dataset in $\mathcal{H}^2$GB.

(2) ***Structure-only.*** **Label propagation** [42, 62]: Spreads labels based on graph connectivity. **SGC** [53]: Linearizes GCN by collapsing weight matrices and removing nonlinearities.

(3) ***Homogeneous Homophilic GNNs.*** **GCN** [23]: A GNN that uses a localized first-order approximation of spectral graph convolutions. **GraphSAGE** [15]: A GNN that employs a sampling and aggregation framework to efficiently generate node embeddings. It concatenates the self-node features with neighbors' features and has been shown to perform well when the graph exhibits some heterophily [64]. **GAT** [49]: A GNN that employs the attention mechanism to weight the significance of neighbors. **GIN** [55]: A GNN designed to capture the power of the Weisfeiler-Lehman graph isomorphism test by using a sum aggregator to update the node representations. **APPNP** [10]: A GNN that combines the propagation of labels throughout a graph with a personalized PageRank scheme for effective learning. **NAGphormer** [5]: A transformer-based GNN that integrates node features and graph topology through attention mechanisms.

(4) ***Homogeneous Heterophilic GNNs.*** **MixHop** [1]: A heterophilic GNN that aggregates features from a node's neighbors at various distances, allowing the model to learn more complex patterns of heterophily. **FAGCN** [4]: A heterophilic GNN with improved aggregation mechanisms considering the influence of neighboring nodes based on their label discrepancy. **ACM-GCN** [35]: A heterophilic GNN designed to discriminate between different types of node relationships. **LINKX** [30]: A heterophilic GNN that decouples structure and feature transformation, making it simple and scalable. **LSGNN** [6]: A heterophilic GNN that models heterophily using local similarity and has been shown to outperform powerful heterophilic GNNs, such as GloGNN [29].

(5) ***Heterogeneous Homophilic GNNs.*** **RGCN** [46]: A heterogeneous GNN that introduces relation-specific transformations to separately aggregate neighbors based on relations. **RGraphSAGE**: GraphSAGE extended to handle heterogeneous graphs by incorporating edge-type information into the aggregation process. **RGAT**: GAT extended to heterogeneous graphs by integrating relational attention into its computation. **HAN** [51]: A GNN that applies both node-level and semantic-level attention, focusing on information aggregation along different metapaths. **HGT** [21]: A heterogeneous GNN that introduces a type-aware attention mechanism to learn node and edge type-dependent representations. **SHGN** [36]: A heterogeneous GNN that improves node representation learning by leveraging type-specific embeddings, incorporating attention mechanisms and residual connections, and applying an ℓ_2-norm to the output for regularization and stability. **HINormer** [38]: A heterogeneous GNN that uses a long-range aggregation mechanism for node representation learning by using a local structure encoder and a heterogeneous relation encoder.

C.2 Implementation Details

(1) ***Experiment Configurations.*** Hyperparameters are initialized based on official settings and tuned for each dataset. All configurations are available at https://github.com/junhongmit/H2GB/.

(2) ***Minibatching.*** Many heterophilic GNNs are not scalable to large graphs. We apply minibatching using fixed sampling parameters across models to avoid OOM errors and ensure fair comparisons.

(3) ***Graph Encoding.*** **Featureless Nodes:** Learnable embeddings are assigned to node types lacking input features, such as in `H-Pokec` and `IEEE-CIS`. **Feature Projection:** All features are projected into a shared embedding space.

(4) ***Model Adaptation.*** **Relational Extensions:** We adapt GraphSAGE and GAT to heterogeneous graphs via PyG's relational wrappers, creating R-GraphSAGE and R-GAT. **Optimized Attention:** We provide an efficient cross-type heterogeneous attention implementation using sparse operations to handle fragmented edge representations in PyG/DGL.

(5) ***Optimization.*** We use the AdamW optimizer with cosine annealing and warmup [33], with weight decay set to 10^{-5}.