Integrating Co-Training with Edge Discrimination to Enhance Graph Neural Networks under Heterophily

Siqi Liu[1], Dongxiao He[1], Zhizhi Yu[1], Di Jin[1*], Zhiyong Feng[1], Weixiong Zhang[2]

[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Department of Computing, Department of Health Technology and Informatics, The Hong Kong Polytechnic University, Kowloon, Hong Kong
{siqiliu, hedongxiao, yuzhizhi, jindi, zyfeng}@tju.edu.cn, weixiong.zhang@polyu.edu.hk

Abstract

Graph Neural Networks (GNNs) have recently achieved significant success in several graph-related tasks. However, traditional GNNs and their variants are constantly limited by the implicit homophily, assuming neighboring nodes belong to the same class. This results in weak performance on heterophilic graphs where most nodes are linked to neighbors of different classes. Despite the numerous attempts to adequately deal with heterophily, most methods still use the uniform propagation aggregation mechanism. In this paper, we argue that identifying neighbors with different class labels and exploiting them individually is crucial for heterophilic GNNs. We then propose a simple and efficient novel co-training approach, EG-GCN, which uses group aggregation to handle homophilic and heterophilic neighbors separately. In EG-GCN, we first use an edge discriminator to classify edges and split the neighborhood of every node into two parts. We then apply group graph convolution to the divided neighborhoods to obtain node representations. During training, we continuously optimize the edge discriminator to improve neighborhood partition and use the node classification results to identify highly confident unlabeled nodes to expand the edge training set. This co-training strategy enables both components to enhance each other mutually. Extensive experiments demonstrate that EG-GCN significantly outperforms the state-of-the-art approaches.

Introduction

Graphs or networks widely exist in the real world, such as social graphs (Tang et al. 2009), biology graphs (Mason and Verwoerd 2007), and molecular graphs (Ashkenasy et al. 2004). Recently, Graph Neural Networks (GNNs) have become the dominant methods for learning graph representations and have shown significant potential in handling graph-related problems like node classification (Dai et al. 2024), link prediction (Zhang and Chen 2018), and graph classification (Zhang et al. 2018).

Traditional GNNs and their variants, e.g., Graph Convolution Network (GCN) (Kipf and Welling 2017), typically rely on the message-passing mechanism (Gilmer et al. 2017), where the representation of a node is updated by aggregating the representations of its neighbors. Such message-passing-based GNNs are often constrained by the implicit homophily assumption that connected nodes are likely to have similar features or the same labels (Zhu et al. 2020a). However, many real-world heterophilic graphs violate the homophily assumption, where linked nodes tend to have dissimilar features or different labels. For instance, amino acids of different types are more likely to aggregate to form protein structures (Zhu et al. 2020a); most people prefer to connect with others of the opposite sex in dating networks (Zhu et al. 2020b). Most existing GNN approaches perform poorly on heterophilic graphs.

A series of heterophilic GNNs has been proposed. Some of them use graph structure learning strategies to modify the origin graph structure to enhance its homophily, allowing aggregating information in a homophilic way on heterophilic graphs. For example, the method in (Qiu et al. 2024) iteratively refines a latent homophilic structure by considering interaction among multiple nodes; the approach in (Zou et al. 2023) exploits a structural entropy strategy to enhance the robustness of heterophilic graphs. Another line of research focuses on designing personalized aggregation operations to collect more homophilic information. For example, (He et al. 2022) introduce block modeling to automatically learn distinct aggregation rules for neighbors of different classes based on their homophily degree; (Zhu et al. 2020a; Jin et al. 2021) aggregate potential homophilic neighbors in 2-hop, or kNN graphs rather than merely distinct 1-hop neighbors. Other works attempt to change the model architectures and integrate information across layers by various skip-connections. For example, (He et al. 2024) adopt the initial compensation based on the local distinguishability of class, making the model sensitive to node classes, and (Zhu et al. 2020a) utilize an inter-layer combination scheme to acquire a more comprehensive node representation.

However, the existing heterophilic graph methods have a serious drawback. They usually adopt a uniform message-passing mechanism, which gives equal weight (i.e., using a uniform weight matrix) to all neighbors when aggregating their features during the message-passing process. In a heterophilic graph, a node's neighbors consist of nodes of various classes with distinct structural patterns and feature distributions. Simply mixing all neighbors and using a unified weight matrix for information aggregation will fail

*Corresponding Author.
Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

to acknowledge the differences between different classes of neighbors and reduce the model's sensitivity to local structures. Although some specialized aggregation methods apply different strategies to different classes of neighbors, they use uniform weight matrices and share the same parameters for all nodes. As a result, they build models capturing and propagating features across whole graphs, exacerbating the homophily bias inherent in GNNs.

An intuitive solution is to use different weight matrices for homophilic and heterophilic neighbors separately. This allows the model to distinguish and learn the unique information of the two types of neighbors. However, this approach is not as straightforward as it may seem. First, differentiating between homophilic and heterophilic neighbors in graph structures may be challenging since the homophily or heterophily of neighbors is relative and context-dependent. Inaccurate neighbor classification results in erroneous feature learning and information extraction. Additionally, using separate weight matrices for homophilic and heterophilic neighbors might overlook the potential interactions between them. Effectively integrating information from both is another critical aspect to consider.

We thus propose an **E**dge discrimination-guided **G**roup **G**raph **C**onvolution **N**etwork, dubbed **EG-GCN**. In EG-GCN, an edge discriminator is introduced to precisely classify edges by node features and network topology and to divide the neighborhoods of nodes to guide node aggregation. During the message-passing process, we employ distinct aggregation functions to independently handle homophilic and heterophilic neighbors and combine them with a mixer operation to learn informative node representations. Considering the limited number of labeled edges in semi-supervised learning, we increase the edge discriminator's supervision signals by adding confidently predicted unlabeled nodes to the edge training set. In this way, edge discrimination and group graph convolution learning can reciprocally enhance each other in a closed loop.

Our main contributions are summarized below.

- We propose the EG-GCN algorithm for addressing challenges in heterophilic graphs. EG-GCN contains an edge discriminator to guide the learning of neighbor partition and a group graph convolution module to learn node representations by aggregating homophilic and heterophilic neighbors separately.
- We design a co-training approach, where the edge discriminator's neighborhood partition directly influences grouping results while the node predictions further augment its edge training set. These two components are alternately trained, enhancing their performance.
- We carry out extensive experimental analysis on eight benchmark datasets, including three general homophilic datasets, three general heterophilic datasets, and two large-scale heterophilic datasets, to demonstrate that EG-GCN achieves state-of-the-art performance compared with a range of baselines.

Preliminaries

We first present the notations, followed by an introduction to graph neural networks and the concept of homophily ratios.

Notations. Consider an undirected and unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$, with a set $\mathcal{V}$ of n nodes, a set $\mathcal{E}$ of m edges, and a set $X \in R^{n\times d}$ of d-dimensional features. For simplicity, we use the notation $u, v \in \mathcal{V}$ to index nodes. The graph's adjacency matrix is represented by $A \in R^{n\times n}$, where $A_{u,v} = 1$ if edge $e_{u,v} \in \mathcal{E}$, otherwise $A_{u,v} = 0$. The graph's diagonal degree matrix is denoted as $D \in R^{n\times n}$, where $D_{u,u} = d_u$ is the degree of node u. We use $\mathcal{N}(u)$ to represent the 1-hop neighbor set of node u. Let $Y \in R^{n\times|C|}$ be the one-hot label matrix, where C is the set of classes, and $y_{ui} = 1$ if u belongs to class C_i, or $y_{ui} = 0$ otherwise. Given some labeled nodes Y_L of $\mathcal{V}_L \subset \mathcal{V}$, our objective is to predict the labels Y_U for the remaining unlabeled objects $\mathcal{V}_U = \mathcal{V} \setminus \mathcal{V}_L$.

Graph Neural Networks. Training Graph Neural Networks (GNNs) often adopt a message-passing mechanism to acquire node representations. This mechanism consists of two steps: neighborhood aggregation and representation combination. Specifically, each node first aggregates its neighbors' representations and then combines them with its own representation to update itself. That is,

$$\mathbf{m}_u^{(k)} = \mathrm{AGGREGATE}^{(k)}\left(\left\{\mathbf{h}_v^{(k-1)} : v \in \mathcal{N}(u)\right\}\right), \tag{1}$$

$$\mathbf{h}_u^{(k)} = \mathrm{COMBINE}^{(k)}\left(\mathbf{h}_u^{(k-1)}, \mathbf{m}_u^{(k)}\right), \tag{2}$$

where $\mathbf{h}_u^{(k)}$ denotes the representation of the node u in the k-layer, $\mathcal{N}(u)$ is the set of the neighbors of the node u, and $\mathbf{h}_u^{(0)} = \mathbf{x}_u$. AGGREGATE and COMBINE are the aggregation and combination functions, respectively. The former collects information from neighbors, mostly through addition, mean pooling, max pooling, or weighted average. The latter merges the ego-information and the aggregated message, typically using addition or concatenation operations.

Homophily Ratios. Node homophily ratio (Pei et al. 2020) and edge homophily ratio (Zhu et al. 2020a) are the two most popular measures of homophily. Specifically, the node homophily ratio $h_{\text{node}}(\mathcal{G})$ measures the average proportion of a node's neighbors that belong to the same class, and the edge homophily ratio $h_{\text{edge}}(\mathcal{G})$ measures the average proportion of edges that connect two nodes with the same class, which are defined as:

$$h_{\text{node}}(\mathcal{G}) = \frac{1}{n}\sum_{u\in\mathcal{V}}\frac{|\{v : v \in \mathcal{N}(u) \wedge y_u = y_v\}|}{|\mathcal{N}(u)|}, \tag{3}$$

$$h_{\text{edge}}(\mathcal{G}) = \frac{1}{m}|\{e_{u,v} : e_{u,v} \in \mathcal{E} \wedge y_u = y_v\}|, \tag{4}$$

where the range of both $h_{\text{node}}(\mathcal{G})$ and $h_{\text{edge}}(\mathcal{G})$ is [0,1]. A graph with strong homophily (low heterophily) has ratios near 1. Conversely, a graph with high heterophily (low homophily) has ratios near 0.

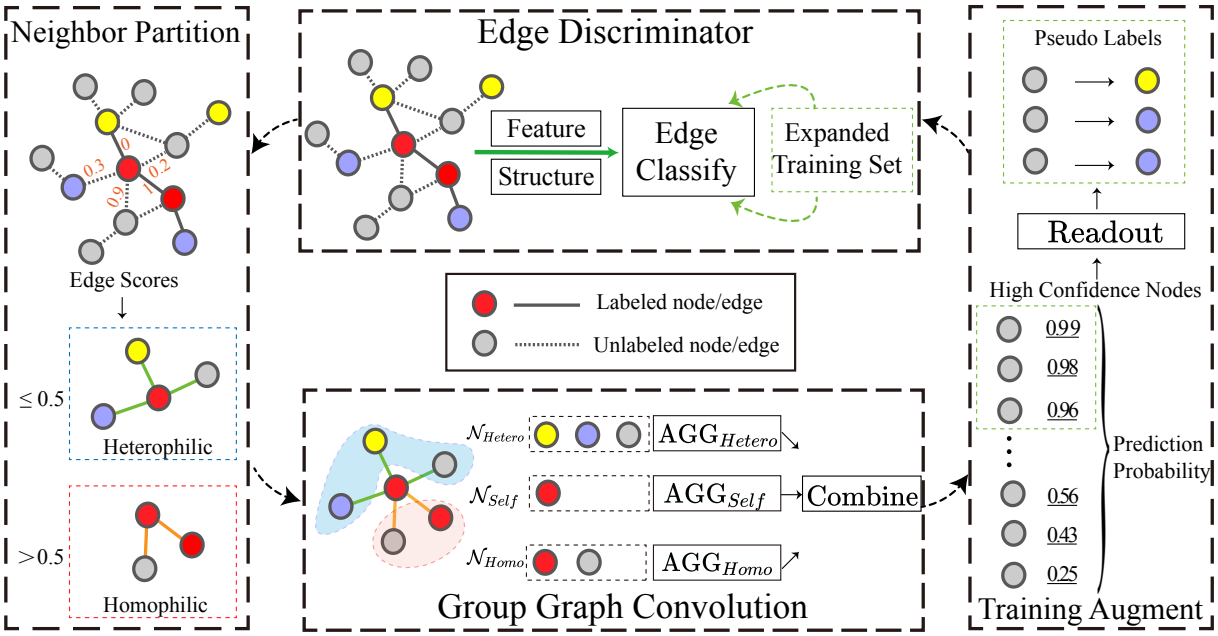


Figure 1: A sketch of EG-GCN, which trains an edge discriminator by node features and structural encoding for dividing the neighborhood and a group graph convolution module together for learning node representations. During training, the edge discriminator constantly refines the neighborhood partition, while the group graph convolution augments its training set with confidently predicted unlabeled nodes.

Methodology

We first briefly overview our proposed EG-GCN, followed by a detailed description of each component.

Overview

To handle homophilic and heterophilic neighbors separately, we propose a simple but effective co-training approach named EG-GCN, which first discriminates edges and then performs group aggregation based on the partitioned node neighborhoods. These two components iteratively optimize each other during the training process. The key idea of EG-GCN is to utilize edge information to segment the message-passing process, allowing the model to classify neighbors by learning different weights for similar and dissimilar neighbors during message passing. This strategy improves the model's capacity to adjust the impact of connected nodes from different classes adaptively. EG-GCN consists of two primary components (Figure 1): an edge discriminator that discriminates the homophilic and heterophilic edges, and a group graph convolution module that separately aggregates homophilic and heterophilic neighbor information. During training, the edge discriminator improves effectiveness of the group graph convolution by providing an accurate neighborhood partition. Meanwhile, the group graph convolution module expands its edge training set with high confidence predicted unlabeled nodes. This mutually boosting optimization mechanism enables EG-GCN to continuously improve classification accuracy, where a focal loss is used to train the edge discriminator to alleviate edge class imbalance and a cross-entropy loss to learn node representations.

Edge Discriminator

To distinguish homophilic and heterophilic edges, we develop an edge discriminator to predict the edge type of each edge, which can be regarded as a semi-supervised binary classification problem.

First, based on the node training set $\mathcal{V}_L$, we construct an edge training set $\mathcal{E}_L$ by utilizing the labels of source and target nodes, defined as:

$$y_{e_{u,v}} = \begin{cases} 1 & y_u = y_v \\ 0 & y_u \neq y_v \end{cases}, u, v \in \mathcal{V}_L \,, \tag{5}$$

where homophilic and heterophilic edges are labeled 1 and 0, respectively. To predict edge types, we utilize the feature and structure information of the connected nodes to learn informative edge representations. Specifically, we employ raw feature X and structural encoding P based on random walk diffusion processes (Dwivedi et al. 2022; Liu et al. 2023) as input, where the structural encoding captures complex relationships in graph structures. For node u, its k-th structural encoding is defined with k-steps of random walk as:

$$\mathbf{p}_u = \left[\mathrm{RW}_{uu}, \mathrm{RW}^2_{uu}, \cdots, \mathrm{RW}^k_{uu}\right] \in R^k, \tag{6}$$

where $\mathrm{RW} = AD^{-1}$ is the random walk operator. Then we construct the input and feed it into the edge discriminator to estimate an edge score. The edge discriminator is built with a multi-layer perceptron (MLP). Specifically, for each edge $e_{u,v}$, its predicted edge score is defined as:

$$\mathbf{h}_u = \mathbf{W_1}\left(\mathbf{x}_u \| \mathbf{p}_u\right), \mathbf{h}_v = \mathbf{W_1}\left(\mathbf{x}_v \| \mathbf{p}_v\right), \tag{7}$$

$$s_{e_{u,v}} = \mathrm{Sigmoid}\left(\mathbf{W_2}\left[\mathbf{h}_u \| \mathbf{h}_v \| \left(\mathbf{h}_u - \mathbf{h}_v\right)\right]\right), \tag{8}$$

where $\mathbf{W_1}$ and $\mathbf{W_2}$ are the learnable weight matrices, $\|$ is the concatenation operation, and the $\mathrm{Sigmoid}$ function is used to normalize the output of MLP into the range of $[0, 1]$.

Neighbor Partition

After predicting scores for all edges, we create pseudo-labels of the unlabeled edges, where edges with scores above 0.5 are classified as homophilic or as heterophilic otherwise. For $e_{u,v} \in \mathcal{E}_U$:

$$y_{e_{u,v}} = \begin{cases} 1 & s_{e_{u,v}} > 0.5 \\ 0 & s_{e_{u,v}} \leq 0.5 \end{cases}, e_{u,v} \in \mathcal{E}_U \ . \tag{9}$$

Depending on the type of edges, we then partition the neighborhood of each node into two distinct groups: homophilic neighborhood $\mathcal{N}_{Homo}$ and heterophilic neighborhood $\mathcal{N}_{Hetero}$. For node u:

$$\mathcal{N}_{Homo}(u) = \{v : v \in \mathcal{N}(u) \wedge y_{e_{u,v}} = 1\}, \tag{10}$$

$$\mathcal{N}_{Hetero}(u) = \{v : v \in \mathcal{N}(u) \wedge y_{e_{u,v}} = 0\}. \tag{11}$$

Group Graph Convolution

After partitioning the original neighborhoods, the homophilic neighborhood $\mathcal{N}_{Homo}$ contains similar neighbors that belong to the same class, and the heterophilic neighborhood $\mathcal{N}_{Hetero}$ includes dissimilar neighbors from different classes. Unlike the traditional unified message-passing mechanism, our approach captures the complex relationships between nodes by separately handling homophilic and heterophilic neighbors. Specifically, we employ two independent aggregation functions for homophilic and heterophilic neighbors:

$$\mathbf{m}_{u_{Homo}}^{(k)} = \mathrm{AGG}_{Homo}^{(k)}\left(\left\{\mathbf{h}_v^{(k-1)} : v \in \mathcal{N}_{Homo}(u)\right\}\right), \tag{12}$$

$$\mathbf{m}_{u_{Hetero}}^{(k)} = \mathrm{AGG}_{Hetero}^{(k)}\left(\left\{\mathbf{h}_v^{(k-1)} : v \in \mathcal{N}_{Hetero}(u)\right\}\right), \tag{13}$$

where AGG_{Homo} and AGG_{Hetero} are related to the contribution of the homophilic neighbors and heterophilic neighbors, which are parameterized by the weight matrices $\mathbf{W}_{Homo}$ and $\mathbf{W}_{Hetero}$, respectively. AGG_{Homo} aggregates features from homophilic neighbors, enabling the model to integrate shared information among similar nodes effectively. In contrast, AGG_{Hetero} captures features from heterophilic neighbors, allowing the model to leverage the distinct information these neighbors carry to obtain informative node representations. These two aggregations may use different functions. In our implementation of the method, we adopt graph convolution operators as neighborhood aggregation functions:

$$\mathrm{AGG}^{(k)}(u) = \sigma\left(\sum_{v \in \mathcal{N}(u)} \frac{1}{\sqrt{d_u' d_v'}} \mathbf{W}^{(k)} \mathbf{h}_u^{(k-1)}\right), \tag{14}$$

where d' is the new degree of node after neighborhood partition. Additionally, we handle the ego-representation and the surroundings independently to retain more particular information. Then, we combine the neighborhood and self-representations to update the node representations:

$$\mathbf{m}_{u_{Self}}^{(k)} = \mathrm{AGG}_{Self}^{(k)}(\mathbf{h}_u^{(k-1)}) = \mathbf{W}_{Self}^{(k)} \mathbf{h}_u^{(k-1)}, \tag{15}$$

$$\begin{aligned} \mathbf{h}_u^{(k)} &= \mathrm{COMBINE}^{(k)}\left(\mathbf{m}_{u_{Self}}^{(k)}, \mathbf{m}_{u_{Homo}}^{(k)}, \mathbf{m}_{u_{Hetero}}^{(k)}\right) \\ &= \sigma(\mathbf{m}_{u_{Self}}^{(k)} + \beta \mathbf{m}_{u_{Homo}}^{(k)} + (1-\beta)\mathbf{m}_{u_{Hetero}}^{(k)}), \end{aligned} \tag{16}$$

where $\mathbf{h}_u^{(0)} = \mathbf{x}_u$, σ is the activation function, and β is a balance parameter of mixer operation. For node u, we denote its output in the final layer as $\mathbf{z}_u$.

Training Augment

We can now predict the node label distribution $\hat{\mathbf{y}}$ of node u as follows:

$$\hat{\mathbf{y}}_u = \mathrm{Softmax}(\mathbf{z}_u). \tag{17}$$

During training, we aim to improve the edge discriminator's performance by augmenting the edge training set. Initially, the edge training set $\mathcal{E}_L$ has a limited number of edges that provide an inadequate supervision signal, which might lead to poor performance. Therefore, we select unlabeled nodes with high confidence after node classification to expand the edge training set. Specifically, we rank the unlabeled nodes by their highest class prediction probability and take the nodes with the highest prediction probability:

$$\mathcal{V}_P = \{u : u \in \mathcal{V}_U \wedge \max(\hat{\mathbf{y}}_u) > \tau\}, \tag{18}$$

where τ is the confidence threshold to form a high-confidence node set $\mathcal{V}_P$. We then use the predicted classes as pseudo labels for the nodes in $\mathcal{V}_P$ and utilize Eq. (5) to create an expanded edge training set. The high-confidence edge set serves as an additional supervised signal to the edge discriminator in the next iteration.

Using the augmented edge training sets and supervision signals, the edge discriminator can rapidly acquire adequate edge types. This results in a more precise differentiation between homophilic and heterophilic neighbors while further enhancing the node representation learning of group graph convolution. This mutually reinforcing process is repeated for a total of T iterations.

Optimization Objective

Due to the typical imbalance of homophilic and heterophilic edges in graphs, we adopt the focal loss in the edge discriminator inspired by (Lin et al. 2020):

$$\mathcal{L}_{\text{edge}} = (y_e - 1)(1-\delta)s_e^{\gamma} \log(1 - s_e) - y_e \delta (1 - s_e)^{\gamma} \log s_e, \tag{19}$$

where δ is a weighting factor addressing the class imbalance, which we set to the inverse class frequency, γ is a tunable focusing parameter, s_e is the predicted edge score obtained by Eq. (8), and y_e is the true edge label.

For the group graph convolution module, we define the node loss function by a cross entropy as:

$$\mathcal{L}_{\text{node}} = -\sum_{u \in \mathcal{V}_L} \sum_{c=1}^{C} y_{uc} \log \hat{y}_{uc}, \tag{20}$$

where $\hat{\mathbf{y}}$ denotes the predicted node label and $\mathbf{y}$ represents the true node label.

Experiments

We first discuss the experimental setup, evaluate our EG-GCN method on a node classification task, assess its competitiveness on large datasets, and analyze its robustness.

Experiment Setup

Datasets. We compared the performance of our proposed EG-GCN method with the existing state-of-the-art methods on both homophilic and heterophilic graphs. For the homophilic graphs, we chose three classical citation datasets: Cora, Citeseer, and PubMed (Sen et al. 2008; Namata et al. 2012). For the heterophilic graphs, we selected three datasets, Texas, Wisconsin (Pei et al. 2020) from WebKB, and an actor co-occurrence graph Actor (Tang et al. 2009). In addition, to evaluate the model's performance on large datasets, we selected two recently constructed large datasets: Penn94 and Genius (Lim et al. 2021) from LINKX datasets. Table 1 lists the characteristics of these datasets.

Datasets	#Nodes	#Edges	#Features	#Classes	h_{edge}
Cora	2,708	5,429	1,433	7	0.81
Citeseer	3,327	4,732	3,703	6	0.74
PubMed	19,717	44,338	500	3	0.80
Texas	183	309	1,703	5	0.11
Wisconsin	251	499	1,703	5	0.21
Actor	7,600	33,544	931	5	0.22
Penn94	41,554	1,362,229	5	2	0.47
Genius	421,961	984,979	12	2	0.62

Table 1: Dataset statistics.

Baselines. We compared EG-GCN with three groups of baseline methods: (1) Non-graph models: MLP, (2) General GNNs: GCN (Kipf and Welling 2017) and GAT (Velickovic et al. 2018), and (3) Heterophilic GNNs: H2GCN (Zhu et al. 2020a), BM-GCN (He et al. 2022), GREET (Liu et al. 2023), LSGNN (Chen et al. 2023), UniFilter (Huang et al. 2024) and PCNet (Li, Pan, and Kang 2024).

Implementation Details. For small datasets, we employed the same split as detailed in (Pei et al. 2020). For large datasets, we utilized ten random splits to partition the Penn94 and Genius datasets. Each dataset was divided into training/validation/test sets at a ratio of 48%/32%/20%. The reported Accuracy and Macro-F1 are averages with their standard deviation over the results of the 10 splits of each tested dataset. In our experiments, we adopted the suggested or default parameters for the corresponding codes for all baselines. For EG-GCN, we used Optuna (Akiba et al. 2019) to tune the hyperparameters 200 times.

Performance on Node Classification

Tables 2 and 3 list, respectively, the Accuracy and Macro-F1 scores associated with standard deviations for EG-GCN and baseline models for node classification on homophilic and heterophilic datasets.

- As shown in Table 2, EG-GCN outperforms the SOTA method on all three homophilic datasets, where EG-GCN scores an average of 0.80% better in Accuracy and 1.02% in Macro-F1 than the best baseline model. This performance improvement can be attributed to separately aggregating homophilic and heterophilic neighbors during the message-passing processes. For information propagation in homophilic graphs, the group graph convolution module of EG-GCN enables the center node to capture more relevant information from homophilic neighbors, thereby enhancing the overall quality of node representations. These results demonstrate that although our model was originally designed for heterophilic graphs, it also performs better on homophilic graphs.
- As listed in Table 3, EG-GCN achieves the best performance on all three heterophilic datasets, where it scores an average of 2.51% higher in Accuracy and 5.53% in Macro-F1 than the second-best model. Especially in the Wisconsin dataset, EG-GCN outperforms the best baseline by 4.31% (11.31%) in Accuracy (Macro-F1). This illustrates the importance of distinguishing homophilic and heterophilic neighbors with distinct labels and exploiting them separately in heterophilic graphs. For information propagation in heterophilic graphs, employing a uniform message-passing mechanism might lead to the aggregation of a large amount of information from heterophilic neighbors, thereby causing the model parameters to be dominated by the majority of heterophilic nodes. The precise partition of neighborhoods and distinct aggregation of information in our model effectively mitigate this issue, contributing to its superior performance. These results highlight the EG-GCN's efficacy in the heterophily scenario.

Comparison on Large Datasets

For a comprehensive evaluation, we conducted experiments on two large heterophilic datasets, Penn94 and Genius. Since some models require a large number of parameters and computational resources, we only compared runnable baselines.

As listed in Table 4, EG-GCN achieves the best Accuracy and Macro-F1 scores on the Penn94 dataset. Additionally, it earns the highest Accuracy and virtually the best Macro-F1 score on the Genius dataset. EG-GCN's outstanding performance validates its remarkable efficacy and universality, exhibiting its competitive advantage on large datasets.

Robustness Analysis

To further verify the robustness of EG-GCN, we evaluated it under the attack of topological perturbations. We adopted one representative topology attack, metattack (Zügner et al. 2020), and adjusted the perturbation rate by altering the proportion of edges from 0 to 25% with an increment of 5%. We evaluated the robustness of EG-GCN against GCN, GAT, H2GCN, LSGNN, and UniFilter on the homophilic dataset Citeseer and the heterophilic dataset Wisconsin. Following (Jin et al. 2020), each of these datasets was divided into training/validation/test sets at a ratio of 10%/10%/80%.

Methods	Cora		Citeseer		PubMed		Avg	Avg
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Acc	F1
MLP	72.25±2.58	69.79±3.12	70.46±2.06	66.32±1.64	87.48±0.23	87.36±0.26	76.73	74.49
GCN	86.70±0.71	85.31±1.40	76.00±1.23	69.96±2.31	87.64±0.57	85.68±0.80	83.45	80.32
GAT	86.50±1.20	85.41±1.06	75.50±1.32	70.66±2.51	86.39±0.67	87.04±0.75	82.80	81.04
H2GCN	87.73±0.94	86.61±1.60	75.65±1.52	69.10±2.59	86.08±0.62	85.55±0.75	83.15	80.42
BM-GCN	87.55±1.31	87.62±1.60	76.13±1.98	71.43±1.84	89.30±0.66	88.99±0.73	84.33	82.38
GREET	87.10±1.25	85.80±1.63	75.25±1.87	68.66±1.09	88.20±0.32	88.08±0.37	83.52	80.85
LSGNN	87.38±0.89	85.65±1.19	75.47±0.99	70.19±2.12	88.22±0.53	87.97±0.61	83.69	81.25
UniFilter	87.34±1.07	86.09±1.24	75.49±1.41	71.48±1.81	88.26±0.45	87.80±0.58	83.70	81.79
PCNet	87.81±0.73	86.80±1.62	76.22±1.33	69.66±1.65	89.37±0.35	88.97±0.45	84.47	81.81
EG-GCN	**88.07±1.16**	**86.87±0.86**	**78.09±1.20**	**74.00±1.71**	**89.64±0.38**	**89.33±0.47**	**85.27**	**83.40**

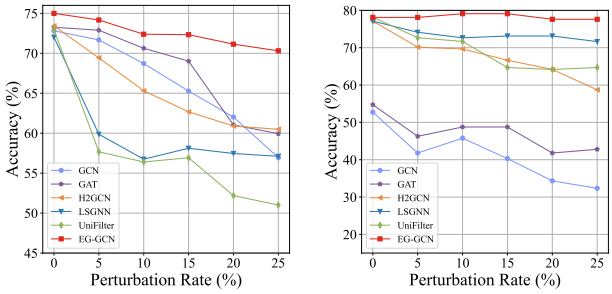
Table 2: Node classification performance on homophilic datasets with mean value and standard deviation in terms of Accuracy (%) and Macro-F1 (%). The best result is bold and the second best is underlined.

Methods	Texas		Wisconsin		Actor		Avg	Avg
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Acc	F1
MLP	80.81±3.91	63.93±9.95	84.90±2.92	58.99±5.07	34.52±1.27	32.81±1.22	66.74	51.91
GCN	59.46±4.84	22.50±5.16	53.73±6.09	23.82±4.42	28.52±1.44	23.53±1.47	47.24	23.29
GAT	59.19±4.43	24.73±6.19	54.12±6.51	21.66±3.03	28.85±1.16	23.30±1.69	47.39	23.23
H2GCN	81.89±7.26	67.55±14.66	81.17±4.31	53.56±8.47	35.09±1.08	29.07±0.80	66.05	50.06
BM-GCN	83.24±5.90	70.77±12.72	81.57±4.74	61.45±12.05	34.01±0.96	29.53±1.14	66.27	53.92
GREET	79.19±4.84	67.55±10.21	80.78±5.17	56.41±6.78	35.34±1.37	32.00±1.35	65.10	51.99
LSGNN	86.76±4.90	75.89±10.14	86.08±4.34	67.18±10.79	34.28±1.04	30.68±0.97	69.04	57.92
UniFilter	81.89±4.20	68.39±10.63	80.98±4.21	62.58±10.44	35.11±0.91	31.16±2.80	65.99	54.04
PCNet	86.49±5.80	70.17±10.54	86.08±4.25	66.74±9.22	37.02±0.87	32.35±1.31	69.86	56.42
EG-GCN	**88.92±3.30**	**78.88±10.07**	**90.39±3.56**	**78.49±10.07**	**37.80±0.64**	**32.98±1.26**	**72.37**	**63.45**

Table 3: Node classification performance on heterophilic datasets with mean value and standard deviation in terms of Accuracy (%) and Macro-F1 (%). The best result is bold and the second best is underlined.

Methods	Penn94		Genius	
	Accuracy	Macro-F1	Accuracy	Macro-F1
MLP	73.07±0.37	72.94±0.36	79.96±0.21	44.44±0.05
GCN	82.13±0.28	81.94±0.17	75.61±0.53	44.45±0.04
GAT	81.38±0.16	73.65±1.00	75.59±8.68	49.76±5.61
LSGNN	81.11±1.09	81.04±1.08	85.75±0.23	72.68±7.31
UniFilter	83.88±0.34	83.84±0.35	90.47±0.33	**79.63±0.52**
PCNet	84.78±0.32	84.73±0.32	78.19±3.53	66.66±9.70
EG-GCN	**84.86±0.33**	**84.85±0.39**	**90.80±0.31**	79.49±0.88

Table 4: Node classification performance on large-scale datasets with mean value and standard deviation in terms of Accuracy (%) and Macro-F1 (%). The best result is bold and the second best is underlined.



(a) Citeseer (b) Wisconsin

Figure 2: Robustness Analysis on Citeseer and Wisconsin datasets.

EG-GCN provides consistently better results than the other methods under all perturbation rates on the two datasets tested (Figure 2). As the rate of perturbation increases, EG-GCN's accuracy fluctuates slightly but remains relatively stable, whereas the performance of other baselines declines to varying degrees. Since metattack destroys the original topology structures by adding or removing edges, those models that directly aggregate information based on graph structures are highly sensitive to the changes, resulting in performance degradation after the attack. In contrast, EG-GCN utilizes both node features and structural encoding in the edge discriminator to provide precise and context-aware edge classification. Additionally, the high-confidence unla-

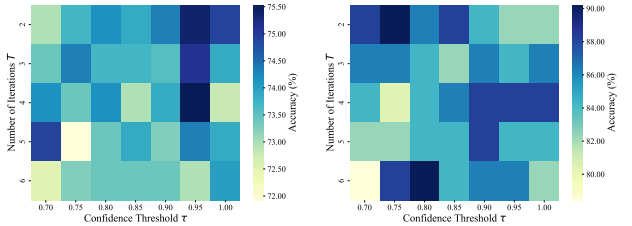
beled nodes identified during training further strengthen EG-GCN's resistance to perturbations. The strong performance of EG-GCN demonstrates its outstanding flexibility.

Parameter Analysis

The key parameters in our EG-GCN method are the confidence threshold τ and the total number of iterations T. We chose the homophilic dataset Citeseer and the heterophilic dataset Wisconsin to analyze the impact of these parameters on model performance. The results are shown in Figure 3.

Confidence Threshold τ. We tested the node classification accuracy of EG-GCN with the confidence threshold τ varying from 0.7 to 1 with an increment of 0.05. The accuracy increases initially but then declines subsequently as τ increases. This is primarily because τ determines the number of high-confidence unlabeled nodes. If the threshold is too low, the pseudo-labels may be noisy, affecting the performance of the edge discriminator. Conversely, if the threshold is too high, very few nodes will meet this criterion, resulting in insufficient extra supervisory signals and failing to achieve the goal of iterative optimization between both edge discriminator and group graph convolution.

Number of Iterations T. We tested the node classification accuracy of EG-GCN with the total number of iterations T varying from 2 to 6 with an increment of 1. When T is too small, the model lacks adequate opportunity for both edge discriminator and group graph convolution to optimize each other mutually. As the number of iterations increases, the model's accuracy stabilizes once it has reached a sufficient level of training. After reaching a specific accuracy, additional iterations yield diminishing returns.



(a) Citeseer (b) Wisconsin

Figure 3: Parameter Analysis on Citeseer and Wisconsin datasets.

Related Work

Several published works, including classic graph neural networks and heterophilic graph neural networks, are related to our research.

Classic Graph Neural Networks. Graph Neural Networks (GNNs) have emerged as a powerful tool for modeling and processing graph-structured data (Wang et al. 2023; Yu et al. 2023; Wang et al. 2022). GNNs can be broadly categorized into two main approaches: spectral-based GNNs and spatial-based GNNs. Spectral GNNs introduce spectral graph filters based on graph signal theory. For instance, ChebNet (Defferrard, Bresson, and Vandergheynst 2016) applies Chebyshev polynomials to approximate a spectral filter function; GCNII (Chen et al. 2020) adopts a K-order polynomial filter with arbitrary coefficients; JacobiConv (Wang and Zhang 2022) uses Jacobi basis because of its orthogonality and adaptability to several weight functions. Spatial GNNs iteratively aggregate and update node representation through topology structures. For example, GCN (Kipf and Welling 2017) updates the node representations by aggregating features from one-hop neighbors; SGC (Wu et al. 2019) removes nonlinearities and collapses weight matrices between layers to simplify GCN; GAT (Velickovic et al. 2018) introduces learnable attention weights to adaptively aggregate information from neighbors.

Heterophilic Graph Neural Networks. Heterophilic Graph Neural Networks have garnered widespread attention, and various methods have been proposed to tackle the problem of heterophilous graphs. For example, H2GCN (Zhu et al. 2020a) enhances GNN performance by designing some elements like ego-neighbor separation, high-order neighbor aggregation, and the merging of intermediate layers; Geom-GCN (Pei et al. 2020) leverages node embeddings in latent geometric spaces to effectively capture structural information; BM-GCN (He et al. 2022) introduces block modeling into GCN to improve aggregation from the same class neighbors while reducing it from different classes; GloGNN (Li et al. 2022) enhances node embeddings by aggregating information from all nodes in a graph; LSGNN (Chen et al. 2023) uses a local similarity to learn node-level weights for multi-hop fusion; PCNet (Li, Pan, and Kang 2024) employs a dual filtering technique and Poisson-Charlier polynomials to efficiently carry out the aggregation of global information from a significant distance through heterophilic interactions. However, all these methods just rely on a uniform message-passing mechanism, potentially limiting their ability to capture the diverse relationships in heterophilic graphs fully.

Conclusion

We proposed a simple and efficient novel co-training method, EG-GCN, that can be applied to both homophilic and heterophilic graphs. The proposed approach deviates significantly from the conventional unified message-passing mechanism and applies distinct aggregate functions to deal with homophilic and heterophilic neighbors separately. EG-GCN consists of an edge discriminator to partition the neighborhoods of the nodes in a graph and a group graph convolution for neighborhood aggregation from homophilic and heterophilic neighbors. By exploiting the highly confident unlabeled nodes obtained from a predictor to augment the edge training set, the edge discriminator can improve its ability to distinguish different types of edges due to the extra supervised signal. This reciprocal optimization mechanism enables EG-GCN to improve classification accuracy. Extensive experiments on large benchmark datasets confirm that EG-GCN achieves superior performance compared to the existing state-of-the-art methods, and the final EG-GCN models are effective and robust on diverse large real-world datasets.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (No. 2023YFC3304503), the National Natural Science Foundation of China (No. 62422210, No. 62276187, No. 92370111, No. 62402337 and No. 62372323), the Postdoctoral Fellowship Program of CPSF under Grant Number GZC20241207, the Hong Kong RGC theme-based Strategic Target Grant (RGC grant STG1/M-501/23-N), the Hong Kong Health and Medical Research Fund (HMRF grant 10211696), the Hong Kong Global STEM Professor Scheme, and the Hong Kong Jockey Club Charity Trust.

References

Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; and Koyama, M. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2623–2631.

Ashkenasy, G.; Jagasia, R.; Yadav, M.; and Ghadiri, M. R. 2004. Design of a directed molecular network. *National Academy of Sciences*, 101(30): 10872–10877.

Chen, M.; Wei, Z.; Huang, Z.; Ding, B.; and Li, Y. 2020. Simple and Deep Graph Convolutional Networks. In *Proceedings of the 37th International Conference on Machine Learning*, 1725–1735.

Chen, Y.; Luo, Y.; Tang, J.; Yang, L.; Qiu, S.; Wang, C.; and Cao, X. 2023. LSGNN: Towards General Graph Neural Network in Node Classification by Local Similarity. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence*, 3550–3558.

Dai, E.; Zhao, T.; Zhu, H.; Xu, J.; Guo, Z.; Liu, H.; Tang, J.; and Wang, S. 2024. A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability. *Machine Intelligence Research*, 21(6): 1–51.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Proceedings of the 30th Neural Information Processing Systems*, 3837–3845.

Dwivedi, V. P.; Luu, A. T.; Laurent, T.; Bengio, Y.; and Bresson, X. 2022. Graph Neural Networks with Learnable Structural and Positional Representations. In *Proceedings of the 10th International Conference on Learning Representations*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, 1263–1272.

He, D.; Liang, C.; Liu, H.; Wen, M.; Jiao, P.; and Feng, Z. 2022. Block Modeling-Guided Graph Convolutional Neural Networks. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 4022–4029.

He, D.; Liu, S.; Ge, M.; Yu, Z.; Xu, G.; and Feng, Z. 2024. Improving Distinguishability of Class for Graph Neural Networks. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 12349–12357.

Huang, K.; Wang, Y. G.; Li, M.; and Lio, P. 2024. How Universal Polynomial Bases Enhance Spectral Graph Neural Networks: Heterophily, Over-smoothing, and Over-squashing. In *Proceedings of the 41st International Conference on Machine Learning*.

Jin, D.; Yu, Z.; Huo, C.; Wang, R.; Wang, X.; He, D.; and Han, J. 2021. Universal Graph Convolutional Networks. In *Proceedings of the 35th Neural Information Processing Systems*, 10654–10664.

Jin, W.; Ma, Y.; Liu, X.; Tang, X.; Wang, S.; and Tang, J. 2020. Graph Structure Learning for Robust Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 66–74.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations*.

Li, B.; Pan, E.; and Kang, Z. 2024. PC-Conv: Unifying Homophily and Heterophily with Two-Fold Filtering. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 13437–13445.

Li, X.; Zhu, R.; Cheng, Y.; Shan, C.; Luo, S.; Li, D.; and Qian, W. 2022. Finding Global Homophily in Graph Neural Networks When Meeting Heterophily. In *Proceedings of the 39th International Conference on Machine Learning*, 13242–13256.

Lim, D.; Hohne, F.; Li, X.; Huang, S. L.; Gupta, V.; Bhalerao, O.; and Lim, S. 2021. Large Scale Learning on Non-Homophilous Graphs: New Benchmarks and Strong Simple Methods. In *Proceedings of the 35th Neural Information Processing Systems*, 20887–20902.

Lin, T.; Goyal, P.; Girshick, R. B.; He, K.; and Dollár, P. 2020. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2): 318–327.

Liu, Y.; Zheng, Y.; Zhang, D.; Lee, V. C. S.; and Pan, S. 2023. Beyond Smoothing: Unsupervised Graph Representation Learning with Edge Heterophily Discriminating. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, 4516–4524.

Mason, O.; and Verwoerd, M. 2007. Graph theory and networks in biology. *IET Systems Biology*, 1(2): 89–119.

Namata, G.; London, B.; Getoor, L.; Huang, B.; and Edu, U. 2012. Query-driven active surveying for collective classification. In *Proceedings of the 10th international workshop on mining and learning with graphs*, 1.

Pei, H.; Wei, B.; Chang, K. C.; Lei, Y.; and Yang, B. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *Proceedings of the 8th International Conference on Learning Representations*.

Qiu, C.; Nan, G.; Xiong, T.; Deng, W.; Wang, D.; Teng, Z.; Sun, L.; Cui, Q.; and Tao, X. 2024. Refining Latent Homophilic Structures over Heterophilic Graphs for Robust Graph Convolution Networks. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 8930–8938.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Gallagher, B.; and Eliassi-Rad, T. 2008. Collective Classification in Network Data. *AI Mag.*, 29(3): 93–106.

Tang, J.; Sun, J.; Wang, C.; and Yang, Z. 2009. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 807–816.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks.

Wang, X.; Dong, Y.; Jin, D.; Li, Y.; Wang, L.; and Dang, J. 2023. Augmenting affective dependency graph via iterative incongruity graph learning for sarcasm detection. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, 4702–4710.

Wang, X.; and Zhang, M. 2022. How Powerful are Spectral Graph Neural Networks. In *Proceedings of the 39th International Conference on Machine Learning*, 23341–23362.

Wang, Z.; Mu, C.; Hu, S.; Chu, C.; and Li, X. 2022. Modelling the Dynamics of Regret Minimization in Large Agent Populations: a Master Equation Approach. In *Proceedings of the 31th International Joint Conference on Artificial Intelligence*, 534–540.

Wu, F.; Jr., A. H. S.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. Q. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning*, 6861–6871.

Yu, Z.; Jin, D.; Wang, X.; Li, Y.; Wang, L.; and Dang, J. 2023. Commonsense knowledge enhanced sentiment dependency graph for sarcasm detection. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence*, 2423–2431.

Zhang, M.; and Chen, Y. 2018. Link Prediction Based on Graph Neural Networks. In *Proceedings of the 32nd Neural Information Processing Systems*, 5171–5181.

Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018. An End-to-End Deep Learning Architecture for Graph Classification. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 4438–4445.

Zhu, J.; Yan, Y.; Zhao, L.; Heimann, M.; Akoglu, L.; and Koutra, D. 2020a. Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. In *Proceedings of the 34th Neural Information Processing Systems*.

Zhu, S.; Pan, S.; Zhou, C.; Wu, J.; Cao, Y.; and Wang, B. 2020b. Graph Geometry Interaction Learning. In *Proceedings of the 34th Neural Information Processing Systems*, 7548–7558.

Zou, D.; Peng, H.; Huang, X.; Yang, R.; Li, J.; Wu, J.; Liu, C.; and Yu, P. S. 2023. SE-GSL: A General and Effective Graph Structure Learning Framework through Structural Entropy Optimization. In *Proceedings of the ACM Web Conference*, 499–510.

Zügner, D.; Borchert, O.; Akbarnejad, A.; and Günnemann, S. 2020. Adversarial Attacks on Graph Neural Networks: Perturbations and their Patterns. *ACM Transactions on Knowledge Discovery from Data*, 14(5): 57:1–57:31.