Block Modeling-Guided Graph Convolutional Neural Networks

Dongxiao He, Chundong Liang, Huixin Liu, Mingxiang Wen, Pengfei Jiao, Zhiyong Feng

College of Intelligence and Computing, Tianjin University
{hedongxiao, liangchundong, liuhuixin, pjiao, zyfeng}@tju.edu.cn, ynwmx@qq.com

Abstract

Graph Convolutional Network (GCN) has shown remarkable potential of exploring graph representation. However, the GCN aggregating mechanism fails to generalize to networks with heterophily where most nodes have neighbors from different classes, which commonly exists in real-world networks. In order to make the propagation and aggregation mechanism of GCN suitable for both homophily and heterophily (or even their mixture), we introduce block modeling into the framework of GCN so that it can realize *"block-guided classified aggregation"*, and automatically learn the corresponding aggregation rules for neighbors of different classes. By incorporating block modeling into the aggregation process, GCN is able to aggregate information from homophilic and heterophilic neighbors discriminately according to their homophily degree. We compared our algorithm with state-of-art methods which deal with the heterophily problem. Empirical results demonstrate the superiority of our new approach over existing methods in heterophilic datasets while maintaining a competitive performance in homophilic datasets.

Introduction

In recent years, graph-based information exploration (Namata et al. 2012b) has been extensively studied in deep learning (Wu et al. 2021; Scarselli et al. 2009; Zhou et al. 2018; Cai, Zheng, and Chang 2018; Cui et al. 2019; Xu et al. 2019). Graph Convolutional Network (GCN) (Kipf and Welling 2017), which is famous for its effectiveness in graph representation learning, follows the principal learning mechanism where adjacency nodes attain similar representations through aggregating their neighboring information. The representations support subsequent downstream tasks such as node classification (Duvenaud et al. 2015; Lee, Lee, and Kang 2019; Ying et al. 2018; Hu et al. 2019) and link prediction (Schlichtkrull et al. 2018; Zhang and Chen 2018; Kipf and Welling 2016; Gao, Denoyer, and Gallinari 2011).

Despite its wide application, GCN and its variants (Wu et al. 2019; Chen, Ma, and Xiao 2018) are typically limited by the implicit homophily assumption where nodes within the proximity have similar representations, that is, birds of a feather flock together (Miller and Cook 2001). But this is not satisfied in many real-world heterophilic datasets, e.g., fraud detection networks or the protein structure graphs, where nodes from different classes tend to make connections due to opposites attract. Some recent studies show that the performance of GCN can be severely restricted in this type of datasets, due to the fact that GCN's aggregating mechanism is not designed for heterophily settings.

Recently some methods aiming to solve the GCN homophily problems have been proposed. Based on the designing methodologies, these algorithms can be mainly divided into two types. 1) Aggregating higher-order neighborhoods, such as H2GCN (Zhu et al. 2020) and MixHop (Abu-El-Haija et al. 2019). These algorithms are based on the idea that direct neighborhoods may be heterophily-dominant, but the higher-order neighborhoods are homophily-dominant and thereby provide more valuable information. Therefore, by explicitly aggregating information from higher-order neighborhood, the heterophily problem of GCN brought by aggregating information from immediate neighborhoods can be alleviated. 2) Passing signed messages between heterophilic neighbors, such as GGCN (Yan et al. 2021) and GPR-GNN (Chien et al. 2021). These algorithms normally assign a weight to every connected node based on the similarity between them. As the nodes aggregate information from neighbors, they get positive messages from neighbors with the same class while negative messages from neighbors with different classes. In this way, positive messages allow neighbors of similar class to intensify their impact, while negative messages prevent dissimilar neighbors from bringing irrelevant information which may harm performance.

However, existing algorithms for heterophily have two main drawbacks. The first is the damage of network topology. Existing methods typically expand high-order nodes as neighbors and try to find more homophilic information, which will change the original topology of networks. But in network science, network topology, even having heterophilic connecting pattern, possess vital information which can be preserved and fully utilized. The second is the limitation of aggregating mechanism. Methods like MixHop and GPR-GNN extend the same treatment to all neighbors in different classes which is not satisfactory, while methods like GGCN is an opposite extreme that gives every neighbor a weight parameter which is highly computational expensive.

Then, an intuitive solution may be that we allow neigh-

bors of the same class aggregated in the same way while neighbors of different classes aggregated in different ways. Block modeling which is to describe structural regularities (including homophily, heterophily, or their mixture) of networks should have a big potential to address this problem. But unfortunately, while block modeling depicts regular relationships between classes via a so called block matrix (which describes the possibility of nodes in two blocks connected by an edge), it still cannot be used for guiding a classified aggregation in the graph convolutional framework.

1. The first challenge is how to derive the block matrix in GCN since it is only a statistical modeling for network structural regularities. For getting block matrix, the block class labels of all nodes must be known. GCN is a semi-supervised learning model and only part of labels are available. So in order to solve this problem, we introduce a multilayer perception (MLP) into the whole learning framework and use it to learn soft labels for all nodes using attribute information. And then we use the soft labels to derive the block matrix.
2. The second and more important challenge is how to derive the classified aggregation mechanism based on the derived block matrix since this matrix (which depicts the probability distribution for generating an edge between nodes in any two blocks) cannot be used for guiding classified aggregation directly. For this problem, we propose to create a new block similarity matrix based on the derived block matrix, which can characterize the similarity degree between different blocks in the connecting pattern of blocks. By doing so, we can use this new matrix as an aggregation indicator to construct the graph convolutional operation and finally realize the classified aggregation for both homophilic and heterophilic graphs.

Then, based on the above idea, we present a new **G**raph **C**onvolutional **N**etwork with **B**lock **M**odelling-guided-classified aggregation, namely BM-GCN, which is suitable for both homophilic and heterophilic situations. In the proposed BM-GCN, the MLP is applied to learn unknown labels and then derive the block similarity matrix, and the derived block similarity matrix as well as the learned labels can co-guide attribute information propagating and aggregating on network topology. The process of learning block similarity matrix and block similarity-guided graph convolutional operation are integrated into a unified framework. In this way, the learning of unknown class labels can help realize classified aggregation, and block-guided graph convolutional operation can further help MLP improve its performance on learning the unknown labels.

Preliminaries

Problem Setup. A attributed graph can be formulated as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ the set of edges and X the node attributes. Each row in X indicates an attribute vector of a node. The edge set can also be represented by an adjacency matrix $A \in \{0,1\}^{n\times n}$, where $n = |\mathcal{V}|$. For semi-supervised tasks, nodes in training set ($\mathcal{T}_{\mathcal{V}}$) have ground truth labels. The labels are formulated as a label matrix $Y \in \mathbb{R}^{n\times c}$ in which each row is a one-hot label vector, where c is the number of classes.

Notations	Explanations
$\mathcal{G}$	A graph
$\mathcal{V}$	The set of nodes in graph $\mathcal{G}$
$\mathcal{E}$	The set of edges in graph $\mathcal{G}$
$\mathcal{T}_{\mathcal{V}}$	Training node set
$\mathcal{N}_i$	The set of neighbor of node v_i
A	Adjacency matrix
X, X_i	Attribute matrix, attribute vector of node v_i
Y, Y_i	Label matrix, one-hot label vector of node v_i
B, B_i	Soft label matrix, soft label vector of node v_i
H, $h_{i,j}$	Block matrix, an element in H
Q, $q_{i,j}$	Block similarity matrix, an element in Q
Z	Node representations

Table 1: Notations and Explanations.

Homophily Ratio. The homophily ratio (Pei et al. 2020) can measure the overall homophily level in a graph. It counts the ratio of same-class neighbor nodes to the total neighbor nodes in a graph, defined as

$$h = \frac{1}{|\mathcal{V}|}\sum\nolimits_{v_i\in\mathcal{V}} \frac{|\{v_j | v_j \in \mathcal{N}_i, Y_j = Y_i\}|}{|\mathcal{N}_i|} \tag{1}$$

where $\mathcal{N}_i$ is the neighbor set of node v_i. In this work, we use homophily ratio h to determine whether a graph is homophilic or heterophilic.

Block Matrix. Given the labels $Y \in \mathbb{R}^{n\times c}$ for all nodes and the adjacency matrix $A \in \{0,1\}^{n\times n}$, the block matrix is defined as

$$H = \left(Y^T AY\right) \oslash \left(Y^T AE\right) \tag{2}$$

where E an all-ones matrix with the same size as Y, and $\oslash$ the Hadamard (element-wise) division operation. Block matrix models the linked possibility of nodes in any two blocks. In this work, blocks represent the classes of labels in a graph. From the node-wise level, $H_{i,j}$ is the probability that a node in the i-th class connects with a node in the j-th class.

The notations are summarized in Table 1.

Methods

Here we show the proposed method, starting with an overview, following the detail designs, and finally give illustrative examples to show the effectiveness of the core design.

Overview

To solve the problem that GCN is constrained by homophily assumption, we introduce block modeling into GCN and developed a new graph convolutional network that is suitable for both homophilic and heterophilic situations. This new framework contains two parts: one is to learn the block matrix and the other is to derive a new mechanism based on block matrix and use it to conduct new propagation and aggregation (which are key operations in GCN). For learning block matrix, we need labels of all nodes, while GCN is typically a semi-supervised method (Li, Han, and Wu 2018),

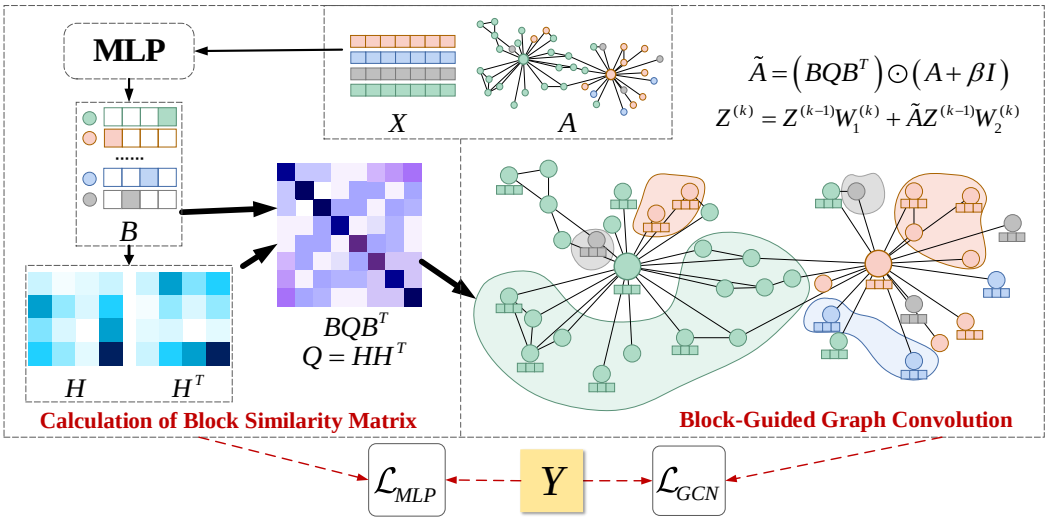


Figure 1: The structure of BM-GCN. It consists of two parts, the calculation of block similarity matrix and the block-guided graph convolution. In the block similarity matrix calculating part, a MLP layer is first applied to generate soft labels B, then block matrix H and block similarity matrix Q are computed based on soft labels B. In block-guided graph convolution part, the graph convolutional operation is conducted under the guidance of block similarity matrix Q and soft labels B. Particularly, BM-GCN achieves a *classified aggregation* mechanism in graph convolutional operation via BQB^T optimization (different color means different aggregation for different classes). Finally, two semi-supervised loss are combined to optimize BM-GCN model in an end-to-end manner.

i.e., only part of labels is available. For solving this problem, we use a multi-layer perception (MLP) to learn soft labels for nodes without class labels. After getting labels of all nodes, we use a combination of the given and learned labels as well as the topological structure of networks to compute the block matrix. Block matrix denotes the probability for generating an edge between nodes in any two blocks (classes). Block matrix reflects homophily and heterophily among parts of networks while it cannot be used directly to conduct propagation and aggregation. So in this case, we developed a new aggregation mechanism, i.e., block-guided classified aggregation, by a smartly using of the block matrix. In specific, we use the block matrix to define a block similarity matrix which measures the similarity degree of any two blocks based on connecting patterns among blocks, and contains homophilic and heterophilic information among blocks. Block similarity matrix provides potentially valuable rules for information propagation in graph convolutional layers, which can ignore the homophily or heterophily of a graph and finally achieve a classified aggregation. Then, we use this block similarity matrix and the learned soft labels from MLP to redefine a new aggregation operation in which class labels of nodes and the similarities among blocks can co-guide the attribute information propagating and aggregating on network topology. At last, these two parts of BM-GCN, the learning of block matrix and the block-guided propagation and aggregation, work together to form a unified graph convolutional framework which is optimized jointly by back propagation.

Learning of Block Matrix

We first need to learn the block matrix. Indicated by Eq. (2), computing block matrix requires complete labels. However, graph convolutional networks are semi-supervised models, in which a large number of labels are unknown. In order to fill this gap, BM-GCN adopts the way of learning the unknown labels (Nguyen, Valizadegan, and Hauskrecht 2014) from the given data (the known labels and the attribute network) to compute the block matrix. Considering that the soft labels should come from the original data while the topology may be not trustworthy in heterophilic graphs, here BM-GCN uses node attributes alone to generate soft labels. Specifically, BM-GCN adopts a multilayer perception (MLP) to transform node attributes into soft labels

$$\bar{B} = \sigma\left(\mathrm{MLP}\left(X\right)\right) \tag{3}$$

where X is node attributes, $\bar{B}$ the output of MLP, and $\sigma\left(\cdot\right)$ an activation function. Then a softmax operation are applied to $\bar{B}$ for generating soft labels

$$B = \mathrm{softmax}\left(\bar{B}\right) \tag{4}$$

In order to ensure the reliability of the soft label B, BM-GCN first pre-trains the MLP layer with the training ground-truth labels for several iterations. Specifically, the pre-training process aims to minimize the loss function

$$\mathcal{L}_{MLP} = \sum\nolimits_{v_i \in \mathcal{T}_\mathcal{V}} f\left(B_i, Y_i\right) \tag{5}$$

where B_i is the soft label of node v_i, Y_i is the ground-truth label of v_i, $\mathcal{T}_\mathcal{V}$ is the nodes in training set, and $f\left(\cdot\right)$ is the cross entropy.

After the pre-training process, there are ground-truth labels and soft labels available for nodes in training set, and only soft labels available for the remaining nodes. BM-GCN maximizes the use of available ground-truth labels by assembling the two kinds of labels

$$Y_s = \{Y_i, B_j | \forall v_i \in \mathcal{T}_\mathcal{V}, \forall v_j \notin \mathcal{T}_\mathcal{V}\} \tag{6}$$

where Y_s is the assembled label matrix with each row being an label vector of each node. Then the block matrix can be computed via Eq. (2), which can be rewritten as

$$H = \left(Y_s^T A Y_s\right) \oslash \left(Y_s^T A E\right) \tag{7}$$

where A is the adjacency matrix, E is an all-ones matrix with the same size as Y_s, and $\oslash$ is the Hadamard (element-wise) division operation. The block matrix H can represent the connecting pattern between blocks (classes), which can well reflect the homophily ratio of the graph. In particular, the more frequently the nodes with the same soft label are connected, the higher homophily ratio the graph has.

Block Similarity Matrix

The block matrix H describes the possibility distribution of two nodes in any two blocks (classes) to be connected by an edge. However, this matrix cannot directly guide GCN to achieve a classified aggregation process. This is because in a heterophilic graph where edges tend to connect nodes in different classes, the possibility values between different classes may be larger than that within the same class. So, in order to realize the block-guided classified aggregation, it is necessary to modify the element values in block matrix H so that these elements can reflect potentially valuable information on propagating rules between various classes of nodes in the graph convolutional operation. With this purpose, we innovatively propose a new block similarity matrix based on block matrix, which is defined as

$$Q = HH^T \tag{8}$$

The similarity matrix Q measures the similarity degree of different blocks in H, indicating that blocks (classes) with similar structural connecting pattern will have more information propagation with each other. Furthermore, since nodes within the same class should have more information exchange, BM-GCN enhance the information propagating ratio within the same class, i.e.,

$$\mathrm{Diag}\left(Q\right) \leftarrow \alpha \cdot \mathrm{Diag}\left(Q\right) \tag{9}$$

where $\mathrm{Diag}\left(\cdot\right)$ means the diagonal elements of a matrix, and α the enhancement factor.

Block-Guided Graph Convolution

Based on the new created block similarity matrix Q, BM-GCN can assign different information propagating rules for different class-combinations. Furthermore, soft labels can indicate which class-combination the two nodes belong to. In this way, the information propagating process can be jointly guided by soft label B and block similarity matrix Q. Particularly, consider two nodes v_i and v_j with their soft labels $B_i = \{b_i^1,\ b_i^2,\ ...,\ b_i^c\}$ and $B_j = \{b_j^1,\ b_j^2,\ ...,\ b_j^c\}$ respectively, where c is the number of classes. There are c^2 candidate class-combinations for node pairs $\langle v_i, v_j \rangle$, each of which can be probabilized as

$$p\left(\varphi(v_i) = Y_r,\ \varphi(v_j) = Y_t\right) = b_i^r b_j^t \tag{10}$$

where $\varphi\left(\cdot\right)$ is a function that maps a node to its class, and $r, t \in \{1, 2, ..., c\}$. Meanwhile, the block similarity matrix Q indicates information propagating probability between any two classes, i.e., the more similar two classes, the more information should be propagated. Therefore, propagating probability between nodes v_i and v_j can be seen as the expectation of elements in Q, i.e.,

$$\omega_{ij} = \sum_{r=1}^{c} \sum_{t=1}^{c} q_{r,t} b_i^r b_j^t \tag{11}$$

where $q_{r,t}$ is an element in Q, indicating information propagating probabicity between the r-th class and the t-th class. Eq. (11) demonstrates that the propagation probability between nodes v_i and v_j are guided by their soft labels and block similarity matrix Q simultaneously. Then, for all node pairs in a graph, propagating probabilities along with these pairs can be formulated as a weight matrix

$$\Omega = BQB^T \tag{12}$$

Eq. (12) is the matrix-level expression of Eq. (11). Then, we use weight matrix Ω to refine the topology

$$A' = \Omega \odot (A + \beta I) \tag{13}$$

where I is the identity matrix, β a hyper-parameter, and $\odot$ the Hadamard (element-wise) multiplication operation. Then, BM-GCN normalizes the weights on edges by a softmax operation

$$\tilde{a}_{i,j} = \frac{\exp\left(a'_{i,j}\right)}{\sum_{v_s \in \mathcal{N}} \exp\left(a'_{i,s}\right)} \tag{14}$$

where $a'_{i,j}$ is an element in A'. Here we name the normalized topological matrix as $\tilde{A}$, and replace the normalized graph laplacian used in GCN with our new $\tilde{A}$. In this way, graph convolutional operation in BM-GCN can finally realize a classified aggregation mechanism because the information propagation process is under the guidance of soft labels and block similarity matrix Q. Node pairs belonging to different soft label combinations will have different information exchange, and the information exchange ratio is determined by Q. Then, the new graph convolutional layer can be written as

$$Z^{(k)} = Z^{(k-1)} W_1^{(k)} + \tilde{A} Z^{(k-1)} W_2^{(k)} \tag{15}$$

where $Z^{(k)}$ denotes node representations in the k-th layer, $W_1^{(k)}$ and $W_2^{(k)}$ are learnable parameters specific to the k-th layer, and $Z^{(0)} = X$.

Model Optimization

Similar to the MLP layer (Eq. 5), BM-GCN adopts a semi-supervised loss as

$$\mathcal{L}_{GCN} = \sum\nolimits_{v_i \in \mathcal{T}_\mathcal{V}} f\left(Z_i, Y_i\right) \tag{16}$$

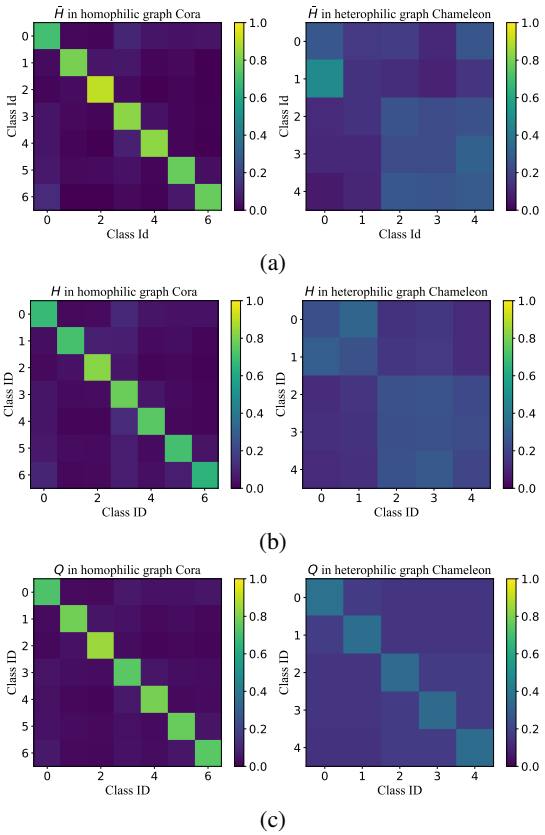


Figure 2: Visualization of (a) block matrix $\bar{H}$ calculated based on ground truth, (b) block matrix H learned by our approach, and (c) the new block similarity matrix Q created based on H. The left column is the result on homophilic graph Cora and the right is that on heterophilic graph Chameleon. x-axis (and y-axis) denotes the id of the block (class). The lager the brighter for elements in each matrix.

where Z is the output of the last graph convolutional layer. Here, BM-GCN also optimizes pre-trained MLP layer in a fine-tune manner (Lu et al. 2021). Specifically, BM-GCN integrates the process of block similarity learning and the process of block guided graph convolutional operation into a unified framework. Incorporating the loss function in MLP layer and in graph convolutional operation, the final loss function can be written as

$$\mathcal{L}_{final} = \lambda \mathcal{L}_{GCN} + (1 - \lambda)\, \mathcal{L}_{MLP} \tag{17}$$

where λ is the balance parameter (with the default value 0.5). By minimize $\mathcal{L}_{final}$, BM-GCN trains all modules in an end-to-end manner.

Illustration on Why Block Modeling Effective

The proposed BM-GCN aims to realize a block-guided classified aggregation so that nodes sharing the same or similar classes will have more information exchange. In order to achieve this goal, block matrix is introduced into BM-GCN to model the relationships between classes. The elements in the block matrix H indicates the connecting possibility between various classes of nodes. Here we take a homophilic graph Cora (Bojchevski and Günnemann 2017) and a heterophilic graph Chameleon (Rozemberczki, Allen, and Sarkar 2021) as examples to show how BM-GCN works.

Datasets	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	c	d	h
texas	183	295	5	1703	0.11
squirrel	5,201	198,493	5	2,089	0.22
chameleon	2,277	31,421	5	2.325	0.23
cora	3,327	4,676	7	3,703	0.74
pubmed	19,717	44,327	6	500	0.80
citeseer	2,708	5,278	3	1,433	0.81

Table 2: Statistics of datasets. $|\mathcal{V}|$, $|\mathcal{E}|$, c, d, h are the number of nodes, edges, classes, and features, and homophily ratio, respectively.

Fig. 2(a) show the block matrix $\bar{H}$ of Cora (or Chameleon) calculated based on the ground truth. As shown, the distribution of these two matrices have different patterns. The connecting possibility of nodes within the same class is larger in the homophilic graph Cora, while the connecting possibility of nodes between different classes is larger in the heterophilic graph Chameleon. Fig. 2(b) shows the block matrix H of Cora (or Chameleon) learned based on the soft labels calculated via Eq. (4). As shown, our H is always close to $\bar{H}$ on either Cora or Chameleon, which partly shows the strong learning ability of our new approach.

However, in this case the learned block matrix H can only help aggregate more same-class information on homophilic graphs rather than heterophilic graphs. That means, the off-diagonal elements with big values in H of the heterophilic graph (right in Fig. 2b) will still cause nodes receiving too many noises during graph convolution and thus lead to performance degradation. But fortunately, we create a new block similarity matrix Q based on H using Eq. (8), which can well measure the relationship between two classes from a new perspective. That is, *if two classes have the similar connected value distributions with all kinds of classes in a graph, the two classes should be more similar.* The visualizations of Q are shown in Fig. 2(c). As shown, the values of diagonal elements in this new matrix Q are then always bigger than off-diagonal elements, on both homophilic and heterophilic graphs. It successfully achieved the role of block-guided classified aggregation on heterophilic graphs (right in Fig. 2c), and meanwhile, preserved the original distributions of block matrix on homophilic graphs (left in Fig. 2c), ensuring the stable performance on both homophilic and heterophilic graphs.

Experiments

We first discuss the experiment setup, including datasets, baselines, and parameter settings. We then evaluate the proposed methods on node classification and visualization tasks, and finally give the parameter analysis.

Method/ Accuracy (%)	heterophilic graphs			homophilic graphs		
	Texas $h = 0.09$	**Squirrel** $h = 0.23$	**Chameleon** $h = 0.22$	**Cora** $h = 0.81$	**Citeseer** $h = 0.74$	**Pubmed** $h = 0.8$
MLP	82.70±6.19	33.35±1.24	48.20±2.63	74.14±1.40	69.58±2.31	86.38±0.61
GCN	55.41±3.47	44.07±1.95	67.04±2.23	86.48±1.12	72.67±1.99	87.39±0.68
H2GCN	82.16±8.21	28.91±1.78	51.58±1.51	87.69±1.37	75.95±2.18	88.78±0.53
GPR-GNN	84.59±4.37	29.45±1.27	**69.78±1.97**	86.70±1.03	75.12±1.98	87.38±0.63
CPGNN-MLP	77.09±4.22	28.65±1.50	52.63±1.79	85.23±1.71	74.29±2.41	86.83±0.78
CPGNN-Cheby	77.03±5.83	30.95±1.24	54.05±4.67	86.82±1.11	75.42±1.85	89.08±0.67
BM-GCN(Ours)	**85.13±4.64**	**51.41±1.10**	69.58±2.90	**87.99±1.29**	**76.13±1.92**	**90.25±0.75**

Table 3: Node classification accuracy on six datasets. The best results are in bold and the second best results are underlined.

Experiment Setup

Datasets. We conduct experiments on six real-world datasets with different homophily ratio. Among them, Cora, Citeseer and Pubmed (Bojchevski and Günnemann 2017; Sen et al. 2008; Namata et al. 2012a) are three citation networks with high homophily ratios. Texas, Chameleon and Squirrel (Rozemberczki, Allen, and Sarkar 2021) are three webpage datasets collected from university websites or Wikipedia, having low homophily ratios. The number of nodes, edges, classes, and features, as well as homophily ratios of the above datasets, are summarized in Table 2.

Baselines. We compare our model with six baselines including a classic GCN (Kipf and Welling 2017), an attribute-only based MLP, and four GNN-based SOTA methods aiming to analyze heterophilic graphs (i.e., H2GCN (Zhu et al. 2020), GPR-GNN (Chien et al. 2021), CPGNN-MLP, and CPGNN-Cheby (Zhu et al. 2019)). Among them, H2GCN aggregated more homophilic information by considering higher order neighborhoods. GPR-GNN tried to distill more valuable homophilic information by assigning signed weights to connected nodes. CPGNN propagated soft labels under the guidance of a compatibility matrix. CPGNN-MLP and CPGNN-Cheby are two variants of CPGNN with different soft label learning methods, i.e., MLP and GCN-Cheby (Defferrard, Bresson, and Vandergheynst 2016).

Parameters. For the baselines, we use their default parameter settings as they often lead to the best results. For our proposed method, we set the number of GCN layers k to 2 for Texas and 3 for the other five datasets. We set the balance parameter of loss λ to 0.5, dropout ratio to 0.5, learning rate to 0.001, and weight decay to 0.0005. We search on the enhancement factor α and self-loop coefficient β from 0 to 4 for datasets. For all datasets, we use the same splits with Geom-GCN (Pei et al. 2020) and measure the performance of all models on the test sets over 10 random splits.

Node Classification

The results are shown in Table 3 where the best results are in bold fonts and the second-best results are underlined. The newly proposed method achieved the best performance on five of the six datasets while the second best performance on the remaining dataset. Considering the six datasets, our BM-GCN is on average 11.03%, 7.91%, 7.58%, 4.58%, 9.30%, and 7.86% more accurate than MLP, GCN, H2GCN, GPR-GNN, CPGNN-MLP, and CPGNN-Cheby respectively. To be specific, for the three datasets with heterophily (left in Table 3), it is obvious that BM-GCN, H2GNN, CPGNN and GPR-GNN have higher mean accuracy than GCN and MLP. This indicates that special designs for heterophily are necessary when analyzing heterophilic graphs. Among these methods, our BM-GCN have the best performance, which demonstrates that our new idea of block-guided classified aggregation is sound and more useful for heterophilic graphs. For three datasets with homophily (right in Table 3), the classic GCN has a good performance since it is designed under a strict homophily assumption and can be naturally applied to homophilic graphs. Other baselines can also achieve competitive performance. Despite these, our BM-GCN steadily outperforms all baselines, which demonstrates its generality.

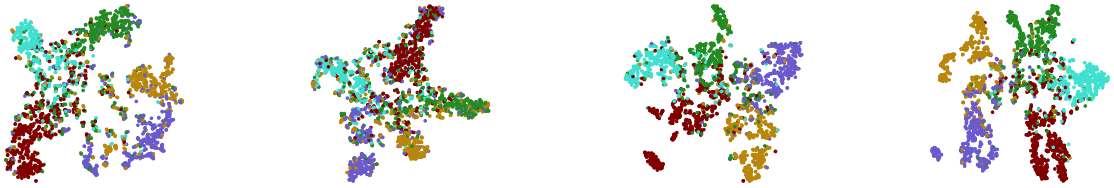
Visualization

In order to show the effectiveness of our proposed model in a more intuitive way, we further conduct visualization task on a heterophilic network Chameleon as an example. We extract the output embedding in the final layer of our BM-GCN as well as three SOTA baselines (CPGNN, GPR-GNN, and H2GCN) and present the learned embedding using t-SNE (van der Maaten and Hinton 2008). The result is shown in Fig. 3, in which nodes are colored by ground-truth labels.

From Fig. 3 we can find that, the results of CPGNN, GPR-GNN and H2GCN are not so satisfactory as many nodes with different labels are mixed together. The performance of our proposed model is apparently the best, because the visualization of learned embeddings has a more compact structure, the highest intra-class tightness and the clearest boundaries among different classes. This further validate the effectiveness of our new idea of block-guided classified aggregation.

Parameter Analysis

We choose two datasets, i.e., a homophilic graph Cora and a heterophilic graph Chameleon to analyze some important hyper-parameters in BM-GCN, including the number of GCN layers k, the enhancement factor α in aggregating weight matrix Q, and the loss balancing parameter λ.



(a) H2GCN (b) CPGNN (c) GPR-GNN (d) BM-GCN

Figure 3: Visualization results on Chameleon dataset. Different colors correspond to different ground truth classes.

Datasets/ Accuracy (%)	The number of graph convolutional layers k					
	1	2	3	4	5	6
Cora	81.27	86.92	87.99	87.30	60.10	40.16
Chameleon	58.71	67.74	69.58	65.00	49.28	34.69

Table 4: Node classification accuracy of BM-GCN with graph convolutional layers k varying from 1 to 6.

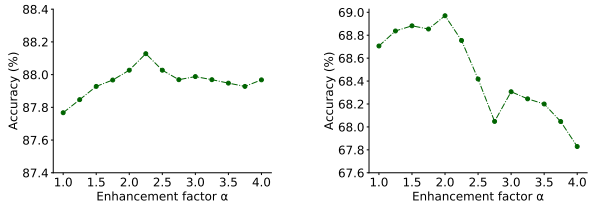
Number of GCN layers k. We test the node classification accuracy of BM-GCN with GCN layers k varying from 1 to 6. The result are reported in Table 4. Results on the two datasets have the same trend. BM-GCN has a good and stable performance when $k = 2, 3, 4$ and will have a performance degradation when k is too large. This is a manifestation of the over-smoothing problem of GCN-based models. Even though, our BM-GCN can achieve competitive performance when $k = 4$ while the classic GCN typically limits $k = 2$. This is because BM-GCN uses aggregating weight matrix Q to further optimized the topology so that noise information can be filtered to alleviate the over-smoothing problem.

Enhancement factor α in Q. We test the node classification accuracy of BM-GCN with enhancement factor α varying from 1 to 4. The results are reported in Fig. 4. BM-GCN achieves the best performance when α is around to 2, which means that properly enhancing the diagonal elements of aggregating weight matrix Q is effective since it can make nodes of the same class more closely connected. Even the performance will decrease when α is too large or too small, the performance changes is within 1%. Empirically, it is feasible to set α around to 2, which demonstrates the easy selectivity of the enhancement factor α.

Loss balance parameter λ. We test the node classification accuracy of BM-GCN with loss balancing parameter λ varying from 0 to 1, and the result are reported in Fig. 5. When $\lambda = 0$, BM-GCN has a poor performance. This is because the parameters in GCN layers will not be optimized. When $\lambda > 0$, BM-GCN has a stable and competitive performance. This demonstrates the robustness of loss balancing parameter in our proposed model.

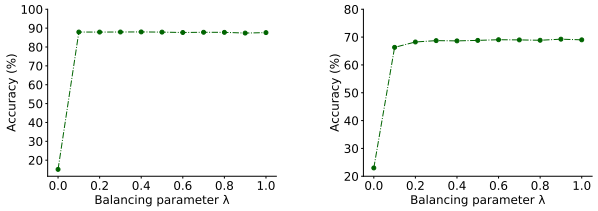
Conclusion

In this paper, in order to develop a better graph convolutional operation universally suitable for both heterophily and homophily we propose a novel framework BM-GCN. The proposed BM-GCN introduces block modeling into graph convolutional operation in order to realize the principle of *block guided classified aggregation*, which aggregates more information from neighbors of the same class while less from different classes. Specifically, BM-GCN consists of two main parts: one learns the similarity between classes according to the learned block matrix, and the other conducts graph convolutional operation with the guidance of block (class) similarity. Based on these two parts, we obtain a unified framework that achieves mutual optimization. We evaluate BM-GCN on both homogeneous and heterogeneous datasets. Experimental results on six real-world datasets verify the superiority of BM-GCN over the-state-of-art methods including those methods designed for heterophily.



(a) Cora (b) Chameleon

Figure 4: Parameter analysis of enhancement factor α in Q on Cora and Chameleon datasets. We report the average node classification accuracy over 10 random splits.



(a) Cora (b) Chameleon

Figure 5: Parameter analysis of loss balancing parameter λ on Cora and Chameleon datasets. We report the average node classification accuracy over 10 random splits.

References

Abu-El-Haija, S.; Perozzi, B.; Kapoor, A.; Alipourfard, N.; Lerman, K.; Harutyunyan, H.; Ver Steeg, G.; and Galstyan, A. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *ICML*, 21–29.

Bojchevski, A.; and Günnemann, S. 2017. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*.

Cai, H.; Zheng, V. W.; and Chang, K. C. 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Trans. Knowl. Data Eng.*, 30(9): 1616–1637.

Chen, J.; Ma, T.; and Xiao, C. 2018. FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling. In *ICLR*.

Chien, E.; Peng, J.; Li, P.; and Milenkovic, O. 2021. Adaptive universal generalized pagerank graph neural network. In *ICLR*.

Cui, P.; Wang, X.; Pei, J.; and Zhu, W. 2019. A Survey on Network Embedding. *IEEE Trans. Knowl. Data Eng.*, 31(5): 833–852.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29: 3844–3852.

Duvenaud, D.; Maclaurin, D.; Aguilera-Iparraguirre, J.; Gómez-Bombarelli, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *NeurIPS*, 2224–2232.

Gao, S.; Denoyer, L.; and Gallinari, P. 2011. Temporal link prediction by integrating content and structure information. In *International Conference on Information and Knowledge Management*, 1169–1174.

Hu, F.; Zhu, Y.; Wu, S.; Wang, L.; and Tan, T. 2019. Hierarchical graph convolutional networks for semi-supervised node classification. In *IJCAI*, 4532–4539.

Kipf, T. N.; and Welling, M. 2016. Variational Graph Auto-Encoders. *NeurIPS Workshop on Bayesian Deep Learning*.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

Lee, J.; Lee, I.; and Kang, J. 2019. Self-Attention Graph Pooling. In *ICML*, 3734–3743.

Li, Q.; Han, Z.; and Wu, X. 2018. Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, 3538–3545.

Lu, Y.; Jiang, X.; Fang, Y.; and Shi, C. 2021. Learning to pre-train graph neural networks. In *AAAI*, 4276–4284.

Miller, M.; and Cook, S. L. M. 2001. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*, 27: 415–444.

Namata, G.; London, B.; Getoor, L.; Huang, B.; and EDU, U. 2012a. Query-driven active surveying for collective classification. In *10th International Workshop on Mining and Learning with Graphs*, volume 8, 1.

Namata, G. M.; London, B.; Getoor, L.; and Huang, B. 2012b. Query-driven active surveying for collective classification. In *Workshop on Mining and Learning with Graphs (MLG)*.

Nguyen, Q.; Valizadegan, H.; and Hauskrecht, M. 2014. Learning classification models with soft-label information. *Journal of the American Medical Informatics Association*, 21(3): 501–508.

Pei, H.; Wei, B.; Chang, K. C.; Lei, Y.; and Yang, B. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *ICLR*.

Rozemberczki, B.; Allen, C.; and Sarkar, R. 2021. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2): cnab014.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2009. The Graph Neural Network Model. *IEEE Trans. Neural Networks*, 20(1): 61–80.

Schlichtkrull, M. S.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling Relational Data with Graph Convolutional Networks. In *ESWC*, 593–607.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine*, 29(3): 93–93.

van der Maaten, L.; and Hinton, G. 2008. Visualizing Data using t-SNE. *J. Mach. Learn. Res.*, 9(2605): 2579–2605.

Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. 2019. Simplifying Graph Convolutional Networks. In *ICML*, 6861–6871.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. S. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.*, 32(1): 4–24.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How powerful are graph neural networks? In *Proceeding of the 7th International Conference on Learning Representations*.

Yan, Y.; Hashemi, M.; Swersky, K.; Yang, Y.; and Koutra, D. 2021. Two Sides of the Same Coin: Heterophily and Oversmoothing in Graph Convolutional Neural Networks. *arXiv preprint arXiv:2102.06462*.

Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W. L.; and Leskovec, J. 2018. Hierarchical Graph Representation Learning with Differentiable Pooling. In *NeurIPS*, 4805–4815.

Zhang, M.; and Chen, Y. 2018. Link Prediction Based on Graph Neural Networks. In *NeurIPS*, 5171–5181.

Zhou, J.; Cui, G.; Zhang, Z.; Yang, C.; Liu, Z.; and Sun, M. 2018. Graph Neural Networks: A Review of Methods and Applications. *arXiv preprint arXiv:1812.08434*.

Zhu, J.; Rossi, R. A.; Rao, A.; Mai, T.; Lipka, N.; Ahmed, N. K.; and Koutra, D. 2019. Graph neural networks with heterophily. In *AAAI*.

Zhu, J.; Yan, Y.; Zhao, L.; Heimann, M.; Akoglu, L.; and Koutra, D. 2020. Generalizing graph neural networks beyond homophily. In *NeurIPS*.