GBK-GNN: Gated Bi-Kernel Graph Neural Networks for Modeling Both Homophily and Heterophily

Lun Du*[†]
lun.du@microsoft.com
Microsoft Research Asia
Beijing, China

Xiaozhou Shi*[‡]
xzh0u.sxz@gmail.com
Beijing University of Technology
Beijing, China

Qiang Fu*
qifu@microsoft.com
Microsoft Research Asia
Beijing, China

Xiaojun Ma[‡]
mxj@pku.edu.cn
Peking University
Beijing, China

Hengyu Liu[‡]
hengyuliu94@gmail.com
Northeastern University
Shenyang, China

Shi Han
Dongmei Zhang
{shihan,dongmeiz}@microsoft.com
Microsoft Research Asia
Beijing, China

ABSTRACT

Graph Neural Networks (GNNs) are widely used on a variety of graph-based machine learning tasks. For node-level tasks, GNNs have strong power to model the homophily property of graphs (i.e., connected nodes are more similar), while their ability to capture heterophily property is often doubtful. This is partially caused by the design of the feature transformation with the same kernel for the nodes in the same hop and the followed aggregation operator. One kernel cannot model the similarity and the dissimilarity (i.e., the positive and negative correlation) between node features simultaneously even though we use attention mechanisms like Graph Attention Network (GAT), since the weight calculated by attention is always a positive value. In this paper, we propose a novel GNN model based on a bi-kernel feature transformation and a selection gate. Two kernels capture homophily and heterophily information respectively, and the gate is introduced to select which kernel we should use for the given node pairs. We conduct extensive experiments on various datasets with different homophily-heterophily properties. The experimental results show consistent and significant improvements against state-of-the-art GNN methods.

CCS CONCEPTS

• Computing methodologies → Neural networks; • Information systems → Social networks.

KEYWORDS

graph neural networks, graph mining, heterophily, homophily

---

*Equal Contribution
[†]Corresponding Author
[‡]Work performed during the internship at MSRA

Figure 1: The left figure illustrates the proportion of number of nodes ordered by node-level homophily ratio range. The right one illustrates node classification accuracy by GCN ordered by node-level homophily ratio range.

1 INTRODUCTION

Graph structured data are ubiquitous due to their vivid and precise depiction of relational objects. As a powerful approach for learning graph representations, Graph Neural Networks (GNNs), such as ChebNet [6], GCN [12], GraphSAGE [9] and GAT [24], are widely used on various applications ranging from social networks to biological networks [7, 29, 31, 33]. The process of node representation learning can be roughly summarized as two steps: (1) feature transformation via one learnable kernel; (2) aggregation of transformed feature from neighbors. The learned node representations can be fed to the downstream neural networks for specific learning tasks such as node classification, link prediction and community detection [2, 14, 23, 26, 30].

Although GNNs have achieved remarkable results on a wide range of scenarios, their ability to model different properties of

graphs has not been analyzed. Many previous studies show that GNNs are more effective in dealing with graphs with good **homophily** property (i.e., connected nodes are more similar), while their ability to capture **heterophily** property (i.e., connected nodes are more opposite) is often doubtful [16, 17, 22, 27, 36, 38]. In addition, even for the graphs with good homophily property, existing works do not model them well enough because they usually treat the homophily as a global property that could be consistently satisfied everywhere in the whole graph. However, they ignore the factor that there may be different levels of homophily among different local regions in a graph. Note that the homophily (and heterophily) property could usually be quantitatively measured with an index Homophily Ratio (HR) [38] that is formally defined as an average of the consistency of labels between each connected node pair. According to our analysis, the Homophily Ratio of different parts of the graph may have a big variance, which indicates the different homophily levels (corresponding to neighborhood patterns related to homophily-heterophily properties) among different sub-graphs.

In more detail, we order the nodes on a graph by Node-level Homophily Ratio (NHR), i.e., Homophily Ratio within a subgraph consisting of a given node and the edges connected the node, to analyze the characteristics of local sub-graphs with the different homophily-heterophily properties. Cora [34] and Texas [22] are often used as representatives of homophily graphs and heterophily graphs in literature, respectively. As shown on the left of Fig. 1, in Texas, the homophily level varies greatly around different nodes, and many nodes are in a mixed state (e.g., NHR in range 20% - 80%) where we cannot easily determine whether a subgraph of a given node is homophily or heterophily. Even in the homophily graph Cora, there are about 30% nodes in the mixed state. On the right of Fig. 1, we illustrate the accuracy of the node classification with the GCN model for nodes with different NHR. GCN obtains impressive performance for high NHR nodes while suffering a rapid decline in effectiveness for nodes in the mixed state on both graphs. The observation hints that we should adaptively model the nodes with different NHR and improve the effectiveness for the nodes in a mixed state.

In this paper, we theoretically analyze the reason for GCN's failure for the graph whose nodes have different homophily levels, and propose a novel GNN model, namely Gated Bi-Kernel Graph Neural Network (GBK-GNN), according to our observations and the identified problem of GCN. When using a GCN model, the representations of nodes in a mixed state will be difficult to distinguish, because only the same kernel is used to transform the features of neighbors with different labels, and the followed aggregation operation will smooth the distinguishable feature of different labels. To avoid smoothing distinguishable features, we propose to use bi-kernels, one for homophily node pairs and the other for heterophily node pairs.

In addition, we introduce a selection gate to determine whether a given node pair is homophily or heterophily, and then select the corresponding kernel to use. This selection gate is learnable with the training objective as minimizing the prediction cross entropy loss between the node pair prediction (i.e., homophily pair or heterophily pair) with the ground-truth, which could be obtained according to the node labels in the training data. Overall, we have

two kernels and a selection gate to learn. They are co-trained together to optimize the total loss, which is the weighted summation of a task loss and a gate prediction loss. Furthermore, we theoretically prove the distinguishability of GBK-GNN is better than GCN.

In summary, the main contributions of our work are as follows:

- We first propose to analyze the homophily-heterophily properties in a fine-grained manner, i.e., from a local perspective instead of a global perspective, to expose the uneven homophily levels among different local sub-graphs. Moreover, we theoretically analyze the reason for GCN's low performance on nodes in the local regions with low homophily levels.
- We propose a model called GBK-GNN to tackle the challenges by introducing the selection gate and the two kernels to model homophily and heterophily, respectively. It can theoretically improve the discriminative ability compared with GCN without introducing many parameters and hyperparameters.
- Extensive experiments are conducted on real-world datasets. Our GBK-GNN outperforms most of the baselines on **seven** graphs significantly with various Homophily Ratios, and the largest error rate deduction is 32.89%. Moreover, the evaluation results verify the model can balance the performance on nodes with various homophily-heterophily properties.

2 PRELIMINARIES AND DEFINITIONS

Let $G = (V, E, X, Y)$ denote a graph with node features and labels, where $V = \{v_i | i = 1, 2, \ldots, n\}$ is the node set with n nodes and $E = \{(v_i, v_j) | v_i, v_j \in V \text{ and } v_i, v_j \text{ is connected}\}$ is the edge set. $X \in \mathbb{R}^{n \times \iota}$ is the node feature matrix where ι is the feature dimension, and x_i is the feature of node v_i. Y is the node label set and the label y_i of v_i satisfies that $\forall v_i \in V, y_i \in \mathbb{Z}_0^+$ and $y_i < k$, and k is the number of categories. $A \in \{0, 1\}^{n \times n}$ denotes the adjacency matrix of the graph G representing the graph connectivity information, and it satisfies if $(v_i, v_j) \in E$ then $A[i, j] = 1$ and otherwise $A[i, j] = 0$. The neighbor set of a center node v_i is denoted as $\mathcal{N}(v_i) = \{v_j | (v_i, v_j) \in E\}$.

In this paper, we mainly focus on the Node Classification task, which is one of the most critical machine learning tasks on a graph and is highly related to the homophily-heterophily properties of the graph. Node classification is a semi-supervised learning problem, and it is formally defined as follows:

DEFINITION 1 (***NODE CLASSIFICATION***). *Node classification is a task to learn a conditional probability $Q(Y|G; \Theta)$ to distinguish the category of each unlabeled node on a single graph $G = (V, E, X, \hat{Y})$, where Θ is the model parameters and $\hat{Y}$ is the partial observed node label set.*

2.1 Graph Neural Networks

Graph Neural Networks (GNNs) can be viewed as an effective node encoder by leveraging the neighbor information. The process of node representation learning can be summarized as two steps: (1) feature transformation via a learnable kernel; (2) aggregation of transformed feature from neighbors. A general GNN architecture

can be formulated as:

$$\mathbf{z}_i^{(l)} = \gamma^{(l)} \left(\mathbf{z}_i^{(l-1)}, \Gamma_{v_j \in \mathcal{N}(v_i)} \phi^{(l)} \left(\mathbf{z}_i^{(l-1)}, \mathbf{z}_j^{(l-1)} \right) \right) \qquad (1)$$

where Γ denotes a differentiable, permutation invariant aggregation operation, e.g. mean, sum, or max. γ and ϕ denote differentiable functions like Multilayer Perceptron (MLP). $\mathbf{z}_i^{(l)}$ is the output of node v_i at the l-th layer. After the GNN encoder, a linear fully connected layer is built above for classification.

Graph Convolutional Network (GCN) [12] is one of the most popular GNN models. In detail, at the l-th layer, GCN use the following formulation to transform the input:

$$\mathbf{Z}^{(l)} = \sigma(\hat{\mathbf{A}} \mathbf{Z}^{(l-1)} \mathbf{W}^l)$$
$$\text{where } \hat{\mathbf{A}} = \mathbf{D}^{-1}(\mathbf{A} + \mathbf{I}). \qquad (2)$$

$\mathbf{D}$ is a diagonal matrix and $D[i,i] = |\mathcal{N}(v_i)| + 1$ is the degree of the node v_i plus one, and $\mathbf{I}$ is an identity matrix.

With the reasonable assumption of the conditional independency, the target conditional distribution is actually equivalent to being decomposed, i.e., $Q(\mathbf{Y}|\mathbf{G};\Theta) = \prod_{v_i \in \mathbf{V}} q(y_i|x_i, \mathcal{N}^{(l)}(v_i);\Theta)$ and $\mathcal{N}^{(l)}(v_i)$ is the neighbor set of the node v_i within l-hops. In other words, only the node feature and the local neighbor information are utilized for classifying a certain node when using GCN to solve the node classification problem.

2.2 Homophily vs. Heterophily in Graphs

Homophily is one of the natural properties of graphs. A graph with good homophily property means that the connected node pairs tend to be more similar. To be specific for the node classification problem, it means the node pairs share the same label with high probability. In contrast, the connected node pairs are more dissimilar on a graph under heterophily. To quantitatively measure the homophily-heterophily properties, Homophily Ratio (HR) [38] is defined as follows:

DEFINITION 2 (*HOMOPHILY RATIO*). *The homophily ratio of a given graph* $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{X}, \mathbf{Y})$ *is defined as*

$$H(\mathbf{G}) = \frac{\sum_{(v_i, v_j) \in \mathbf{E}} \mathbb{1}(y_i = y_j)}{|\mathbf{E}|}, \qquad (3)$$

where $\mathbb{1}(\cdot)$ *is the indicator function (i.e.,* $\mathbb{1}(\cdot) = 1$ *if the condition · holds, otherwise* $\mathbb{1}(\cdot) = 0$*), and* $|\mathbf{E}|$ *is the size of the edge set* $\mathbf{E}$.

HR can measure the global homophily property of the whole graph. In literature, when $H(\mathbf{G}) \to 1$ the graph $\mathbf{G}$ is considered to be strong homophily, while the graph $\mathbf{G}$ is considered to be strong heterophily when $H(\mathbf{G}) \to 0$.

According to our analysis, the Homophily Ratio of different parts of the graph may have a big variance, which indicates the different homophily levels (corresponding to neighborhood patterns related to homophily-heterophily properties) among different sub-graphs. In addition, when using the GNN model to solve a node classification problem, only local neighbor information will be considered for classifying the center node. In order to study the homophily property from a fine-grained local perspective, it is reasonable to define a Node-level Homophily Ratio:

DEFINITION 3 (*NODE-LEVEL HOMOPHILY RATIO*). *Given a center node* v_i *in a given graph* $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{X}, \mathbf{Y})$, *the node-level homophily ratio is defined as the fractions of edges that connect the node* v_i *and another node* v_j *with the same label. It is formally defined as:*

$$h_i = \frac{\sum_{v_j \in \mathcal{N}(v_i)} \mathbb{1}(y_i = y_j)}{|\mathcal{N}(v_i)|}, \qquad (4)$$

where $|\mathcal{N}(v_i)|$ *is the size of the neighbor set* $\mathcal{N}(v_i)$.

3 GENERALIZATION BOUND OF GCN WITH REGARD TO HOMOPHILY RATIO

In this section, we theoretically analyze the generalization ability of Graph Convolutional Networks (GCNs) with regard to different homophily-heterophily levels and further theoretically explain the reason for GCN's failure in classifying the nodes with mixed homophily and heterophily neighbors.

3.1 Complexity Measure Based View for Neural Networks Generalization

In order to theoretically analyze the generalization ability of GCN so as to further explain the reason of GCN's failure for classifying nodes with mixed homophily and heterophily neighbors, the first step is to find a metric to measure the generalization ability of GCN with a given Homophily Ratio. Complexity Measure is the current mainstream method to measure the generalization ability of the model [19]. As we know, a model that can fit real data with lower capacity often has better generalization ability according to Occam's Razor principle. Complexity Measure is defined to measure the capacity of neural networks, so a **lower complexity measure** means a **better generalization ability**. Formally, a Complexity Measure is a measure function $\mathcal{M} : \{\mathcal{H}, \mathcal{S}\} \to \mathbb{R}^+$ where $\mathcal{H}$ is a class of models and $\mathcal{S}$ is a training set. In the following theoretical analysis, we can set $\mathcal{H}$ as GCNs with different parameters and $\mathcal{S}$ as graph data with a certain Homophily Ratio.

There are different forms of Complexity Measure $\mathcal{M}$ specifically designed from different aspects, such as sharpness of a local minimum [8, 11] and the scale of norms of parameters [20, 21]. We select Consistency of Representations [18][1] as our Complexity Measure. This measure is designed based on Davies-Bouldin Index [5]. Mathematically, for a given dataset and a given layer of a model,

$$S_i = \left(\frac{1}{n_i} \sum_{\tau}^{n_i} |O_i^{(\tau)} - \mu_{O_i}|^p \right)^{1/p} \text{ for } i = 1 \cdots k \qquad (5)$$

$$M_{i,j} = ||\mu_{O_i} - \mu_{O_j}||_p \qquad \text{for } i, j = 1 \cdots k, \qquad (6)$$

where i and j are two different classes, $O_i^{(\tau)}$ is the output representation of the τ-th sample belonging to class i for the given layer, μ_{O_i} is the cluster centroid of the representations of class i, S_i is a measure of scatter within representations of class i, and $M_{i,j}$ is a measure of separation between representations of classes i and j. Then, the complexity measure based on the Davies Bouldin Index:

$$C = \frac{1}{k} \sum_{i=0}^{k-1} \max_{i \neq j} \frac{S_i + S_j}{M_{i,j}}. \qquad (7)$$

---

[1]This work won the Winning Solution of the NeurIPS 2020 Competition on Predicting Generalization in Deep Learning.
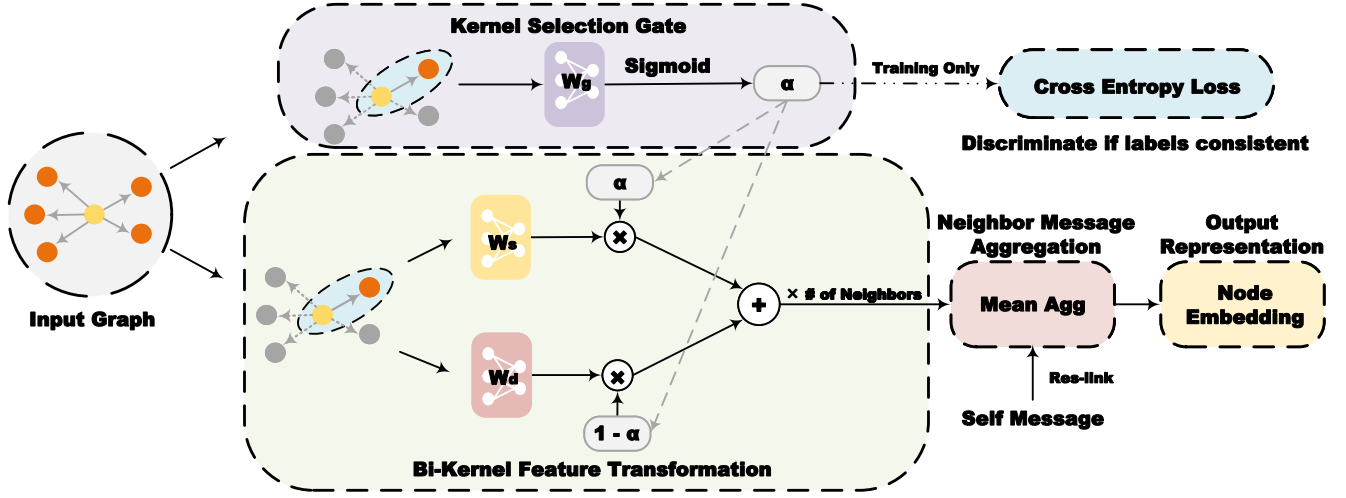
Figure 2: An overview of GBK-GNN. GBK-GNN has two main modules: kernel selection gate learning and bi-kernel feature transformation. The learning process of the kernel selection gate receives input graph information and outputs selective signal α to discriminate if the neighbor node labels are consistent. The bi-kernel feature transformation trains W_s and W_d, namely weights to capture the similarity between nodes and weights to capture the dissimilarity between nodes. It uses the signal α from the former module to combine these two W in the process of message passing, then doing mean aggregation, finally producing node embedding. In the training phase, we have an additional cross-entropy loss to train the selection gate with supervisions.

When p is set as 2. This complexity measure can be roughly interpreted as the ratio of the intra-class variance to the inter-class variance.

3.2 Generalization Bound Analysis and Failure Reason of GCN

We use the complexity measure of Davies-Bouldin Index with $p = 2$ (refer to Eq.(5) to (7)) to analyze the generalization ability of GCNs. We propose several assumptions to simplify the analysis: (1) a binary classification problem is considered (i.e., $k = 2$); (2) given the category y_i of the center node v_i, the probability of its neighbors belonging to the same category is P_{y_i}, e.g., P_0 means for a center node belonging to the 0-th class, the probability of its neighbors belonging to the same category; (3) The number of neighbors is d for all the nodes in the graph; (4) The graph has been added self-loop to facilitate GCN's calculation; (5) We ignore the non-linear activation in GCNs for brevity. Then we have the following theorem:

THEOREM 3.1. *For a binary node classification problem and the input graph satisfies the assumptions above, if $|P_0 + P_1 - 1| \to 0$ then the complexity measure of Consistency of Representations will converge to $+\infty$ for GCNs with an arbitrary kernel (except for a zero matrix). In that case, GCN will lose the generalization ability.*

PROOF. Since $k = 2$ and $p = 2$, Eq.(7) can be simplified as: $C = \frac{S_0 + S_1}{M_{0,1}}$. We first calculate μ_{O_i}:

$$\mu_{O_0} = \mathbb{E}[O_0^{(i)}] = \mathbb{E}[\mathbf{W} \sum_{j \in \mathcal{N}(v_i)} \frac{1}{d} \mathbf{X}^{(j)}]$$
$$= \mathbf{W}(P_0 \cdot \mu_{\mathbf{X}_0} + (1 - P_0) \cdot \mu_{\mathbf{X}_1}), \tag{8}$$

where $\mathbf{X}^{(j)}$ is the j-th node feature and $\mu_{\mathbf{X}_i}$ is the cluster centroid of the node features of class i. Similarly, we have:

$$\mu_{O_1} = \mathbf{W}(P_1 \cdot \mu_{\mathbf{X}_1} + (1 - P_1) \cdot \mu_{\mathbf{X}_0}). \tag{9}$$

Then, $M_{0,1}$ can be obtained by:

$$M_{0,1} = ||\mu_{O_0} - \mu_{O_1}||$$
$$= ||\mathbf{W}(P_0 \cdot \mu_{\mathbf{X}_0} + (1 - P_0) \cdot \mu_{\mathbf{X}_1} - P_1 \cdot \mu_{\mathbf{X}_1} - (1 - P_1) \cdot \mu_{\mathbf{X}_0})||$$
$$= (P_0 + P_1 - 1) \cdot ||\mathbf{W}(\mu_{\mathbf{X}_0} - \mu_{\mathbf{X}_1})||. \tag{10}$$

Then S_0^2 is calculated by:

$$S_0^2 = \mathbb{E}[||O_0^{(i)} - \mu_{O_0}||^2] = \mathbb{E}[< O_0^{(\tau)} - \mu_{O_0}, \, O_0^{(\tau)} - \mu_{O_0} >]$$
$$= P_0^2 \mathbb{E}\left[||\mathbf{W}(\mathbf{X}_0^{(i)} - \mu_{\mathbf{X}_0})||^2\right] + (1 - P_0)^2 \mathbb{E}\left[||\mathbf{W}(\mathbf{X}_1^{(i)} - \mu_{\mathbf{X}_1})||^2\right], \tag{11}$$

where $< \cdot, \cdot >$ is inner production. Let $\sigma_0^2 = \mathbb{E}\left[||\mathbf{W}(\mathbf{X}_0^{(i)} - \mu_{\mathbf{X}_0})||^2\right]$ and $\sigma_1^2 = \mathbb{E}\left[||\mathbf{W}(\mathbf{X}_1^{(i)} - \mu_{\mathbf{X}_1})||^2\right]$, then the above function can be rewritten as:

$$S_0^2 = P_0^2 \sigma_0^2 + (1 - P_0)^2 \sigma_1^2 \geq \frac{\sigma_0^2 \sigma_1^2}{\sigma_0^2 + \sigma_1^2}. \tag{12}$$

Similarly, we have:

$$S_1^2 = P_1^2 \sigma_1^2 + (1 - P_1)^2 \sigma_0^2 \geq \frac{\sigma_0^2 \sigma_1^2}{\sigma_0^2 + \sigma_1^2}. \tag{13}$$

Then we calculate the complexity measure C:

$$
\begin{aligned}
C &= \frac{\sqrt{S_0^2 + S_1^2 + 2S_0 \cdot S_1}}{M_{0,1}} \geq \frac{2\sigma_0\sigma_1}{\sqrt{\sigma_0^2 + \sigma_1^2} \cdot M_{0,1}} \\
&= \frac{2\sigma_0\sigma_1}{|(P_0 + P_1 - 1)| \cdot \sqrt{\sigma_0^2 + \sigma_1^2} \cdot ||\mathbf{W}(\mu_{\mathbf{X}_0} - \mu_{\mathbf{X}_1})||}
\end{aligned}
\tag{14}
$$

Thus, we obtain a lower bound of C. Notice that σ_0 and σ_1 could not be zero, otherwise the classification problem is meaningless. If $|P_0 + P_1 - 1| \to 0$, then the lower bound of $C \to +\infty$, and hence $C \to +\infty$. The model will lose the generalization ability.   □

Intuition of Theorem. 3.1. Theorem. 3.1 tells us that if there are a similar number of homophily neighbors and heterophily neighbors for graph nodes, then GCNs will smooth the outputs from different classes and loss the discrimination ability even though the initial node features are quite distinguishable. Such inter-class smoothness is also a reason why GCNs cannot performance better than a naive Multilayer Perceptron (MLP) for the graph whose nodes have different homophily levels. If we look into the proof, we can find that the distance between representations of 0-th class and 1-th class $M_{i,j}$ tends to be 0, which makes the complexity measure C tend to be infinity. Thus, to avoid $M_{i,j}$ being zero inspires us for our proposed new model Gated Bi-Kernel Graph Neural Networks.

4 GATED BI-KERNEL GRAPH NEURAL NETWORKS

According to the identified problem in GCNs, we propose a new model pertinently, namely Gated Bi-Kernel Graph Neural Networks (GBK-GNN). The overview of GBK-GNN is shown in Fig. 2. There are two main differences compared with vanilla GNNs, i.e., Bi-kernel Feature Transformation and Kernel Selection Gate modules.

4.1 Bi-Kernel Design

As mentioned above, the GCN's problem of lack of distinguishability is mainly caused by a small inter-class variance $M_{i,j}$. A single kernel of GCN cannot adaptively adjust the weights for different types of nodes according to different homophily properties. Mathematically, a single kernel GCN cannot change the proportion of P_0 and P_1 in Eq.(6). Thus, we design a Bi-Kernel feature transformation method to tackle the problem.

In detail, there are two kernels in our model, one for homophily node pairs and the other for heterophily pairs. In an ideal case where we can exactly determine whether a pair is homophily or not, the inter-class variance $M_{0,1}$ will be changed to:

$$
||(P_1\mathbf{W}_s + (P_0 - 1)\mathbf{W}_d)(\mu_{\mathbf{X}_0} - \mu_{\mathbf{X}_1}) + (P_0 - P_1)(\mathbf{W}_s - \mathbf{W}_d)\mu_{\mathbf{X}_0}||, \tag{15}
$$

where $\mathbf{W}_s$ is the kernel for homophily edges and $\mathbf{W}_d$ is the kernel for heterophily edges. The first term in Eq. (15) shows that $\mathbf{W}_s$ and $\mathbf{W}_d$ can adjust the relation of P_1 and $P_0 - 1$ to avoid appearance of $|P_1 + P_0 - 1|$ term. Besides, the second term can support extra distinguishability even if the original features lack discrimination, i.e., $||\mu_{\mathbf{X}_1} - \mu_{\mathbf{X}_0}|| \to 0$.

Compared with Graph Attention Network (GAT), our design can better model the positive and negative correlations by setting one kernel as a positive definite matrix and the other kernel as a negative definite matrix. However, GAT cannot simultaneously model positive and negative correlations because the weights calculated by attention are always positive.

4.2 Selection Gate Mechanism

In reality, we cannot directly determine whether a node pair belongs to the same class or not. Thus, we introduce a learnable kernel selection gate to discriminate node pairs and adaptively select kernels. The formal form of the transformation of the input is listed as follows:

$$
\mathbf{z}_i^{(l)} = \sigma\left(\mathbf{W_f}\mathbf{z}_i^{(l-1)} + \frac{1}{N}\sum_{v_j \in \mathcal{N}(v_i)} \alpha_{ij}\mathbf{W_s}\mathbf{z}_j^{(l-1)} + (1 - \alpha_{ij})\mathbf{W_d}\mathbf{z}_j^{(l-1)}\right)
$$

$$
\alpha_{ij} = \text{Sigmoid}\left(\mathcal{G}_l(\mathbf{z}_i^{(l-1)}, \mathbf{z}_j^{(l-1)}; \mathbf{W_g})\right),
\tag{16}
$$

where $\mathcal{G}_l(\cdot, \cdot)$ is a learnable function ($\mathbb{R}^{n_{l-1}} \times \mathbb{R}^{n_{l-1}} \to \mathbb{R}$) in the l-th layer, $\mathbf{W_s}$, $\mathbf{W_f}$, $\mathbf{W_d}$ and $\mathbf{W_g}$ are learnable parameters, $\mathbf{z}_i^{(l)}$ is the hidden representation of node v_i in the l-th layer, α_{ij} is the gate signal, and n_l is the dimension number of the node hidden representation in l-th layer. $\mathcal{G}_l(\cdot, \cdot)$ can be a multilayer perceptron or a graph neural network (e.g. GraphSage [9]).

4.3 Loss Design and Optimization

Similar to vanilla classification methods, a cross-entropy loss $\mathcal{L}_o$ is used for the node classification problem. The difference is that we utilize an additional cross-entropy loss $\mathcal{L}_g^{(l)}$ for each layer l to guide the training procedure of the selection gate. Thus, the final objective is a combination of two kinds of losses:

$$
\mathcal{L} = \mathcal{L}_o + \lambda \sum_l^L \mathcal{L}_g^{(l)} \tag{17}
$$

where λ is a hyper-parameter to balance two losses. We use AdamW [15] to optimize the loss $\mathcal{L}$.

5 EXPERIMENTS

In this section, we evaluate the empirical performance of GBK-GNN on real-world homophily and heterophily graph datasets on the node classification task and compare with other state-of-the-art models.

5.1 Datasets

We perform our experiments on seven datasets from PyTorch-Geometric, which are commonly used in graph neural network literature. An introduction of the datasets is presented in Table 1. Homophily ratio [38] represents the fraction of edges that connect two nodes of the same label. If the ratio is close to 1, that means the dataset is homophily. And if the ratio is close to 0, that means the dataset is heterophily.

Cora, CiteSeer and PubMed [35] are citation networks based datasets. In these datasets, nodes represent papers, and edges represent citations of one paper by another. Node features are the bag-of-words representation of papers, and node label denotes the academic topic of a paper. Since these citation datasets have a high homophily ratio, they are considered homophily datasets.

Wisconsin, Cornell and Texas are subdatasets of WebKB collected by Carnegie Mellon University. In these datasets, nodes represent web pages and edges represent hyperlinks between them. Node features are the bag-of-words representation of web pages. The task is to classify the nodes into one of the five categories, i.e., student, project, course, staff, and faculty.

Actor [22] is the actor-only induced subgraph of the film-director-actor-writer network. Each node corresponds to an actor, and the edge between two nodes denotes co-occurrence on the same Wikipedia page. Node features correspond to some keywords in the Wikipedia pages. The task is to classify the nodes into five categories in terms of words of actor's Wikipedia.

These two kinds of datasets has a relatively low homophily ratio and are considered as heterophily datasets.

Table 1: Statistics of the node classification datasets.

Dataset	Hom. Ratio	Nodes	Edges	Features	Classes
Cora	0.819	2,708	10,556	1,433	7
CiteSeer	0.703	3,327	9,104	3,703	6
PubMed	0.791	19,717	88,648	500	3
Cornell	0.207	183	298	1,703	5
Texas	0.114	183	325	1,703	5
Wisconsin	0.139	251	515	1,703	5
Actor	0.201	7,600	30,019	932	5

5.2 Baselines

We compare our GBK-GNN with three types of state-of-the-art methods for hetermophily graphs and six graph neural network based methods.

- **CPGNN** [37] incorporates into GNNs a compatibility matrix that captures both heterophily and homophily by modeling the likelihood of connections between nodes in different classes.
- **H2GCN** [38] is a graph neural network combines ego and neighbor-embedding separation, higher-order neighborhoods, and combination of intermediate representations, which performs better on heterophily graph data.
- **GEOM-GCN** [22] is a graph convolutional network with a geometric aggregation scheme which operates in both graph and latent space.
- **DNN** is a basic fully connected neural network model.
- **GCN** [12] is a semi-supervised graph convolutional network model which learns node representations by aggregating information from neighbors.
- **GAT** [24] is a graph neural network model using attention mechanism to aggregate node features.
- **GIN** [32] is a graph neural network which can distinguish graph structures more as powerful as the Weisfeiler-Lehman test.
- **GCNII** [3] is an extension of graph convolutional network with initial residual and identity mapping which can relieves the problem of over-smoothing.

- **GraphSage** [9] is a general inductive framework that leverages node feature information to efficiently generate node embeddings for previously unseen data

5.3 Experimental Setup

We use a public data split provided by [22], the nodes of each class are 60%, 20%, and 20% for training, validation and testing.

We use Adam optimizer and fix the hidden unit to be 16, running 500 epochs for each experiment. For other hyper-parameters, we perform a hyper-parameter search for all models. For fairness, the size of the search space for each model is the same. The searching hyper-parameters include learning rate, weight decay, and λ in our regularization term. The hyper-parameter search process ensures each model has the same search space. Learning rate is searched in $\{1e-3, 1e-4, 1e-5\}$, weight decay is searched in $\{1e-2, 1e-3, 1e-4\}$, and λ is searched in $(0, 64]$. The experimental setup for GBK-GNN and all other baselines are the same except for the layer number of GCNII. We use 64 layers GCNII since it is designed for deep models, and all other models are set to be two layers. We run each experiment on three random seeds and make an average. Model training is done on Nvidia Tesla V100 GPU with 32GB memory. In more detail, our code depends on cudatoolkit 10.2, pytorch 1.7.0 and torch-geometric 1.7.2.

5.4 Results

In this section, we present the results of our experiments and answer four research questions.

5.4.1 RQ1. Does GBK-GNN outperform SOTA methods on both homophily and heterophily datasets? Table 2 shows the comparison of the mean classification accuracy of GBK-GNN with other popular GNN models, including three types of recent state-of-the-art methods and six graph neural network based methods. In general, GBK-GNN achieves state-of-the-art performance on all of the seven datasets. On homophily datasets, the relative error rate reduction compared to the second place models on Cora, CiteSeer, and PubMed are 7.52%, 10.57%, and 8.26%, respectively. This illustrates that we do better on homophily graphs than other state-of-the-art models. On the other hand, we also have a good performance on heterophily datasets. Compared to the second place models, the relative error rate reduction on Wisconsin, Cornell and Texas are significant and can reach 32.89%, 11.37% and 22.20%, respectively. On Actor, we have the relative error rate reduction 5.25%. We notice that DNN and GraphSage have a relatively good performance on heterophily graphs compared with other based GNN models. This phenomenon may be caused by their emphasis on capturing the information of the central node. Since nodes in heterophily graphs have relatively weak connections with neighbors, models like GCN, GAT or GIN which focus more on the neighbors may be more easily confused. CPGNN and H2GCN have good performance on many heterophily datasets because they are specially designed for heterophily graphs. However, we still have better performance on heterophily datasets than them and maintain a much better performance on homophily datasets. Besides, our gate mechanism can also be applied to these methods to pursue a higher accuracy since the main idea of the gate mechanism is generic.

Table 2: Mean classification accuracy of our GBK-GNN and other popular GNN models on both homophily and heterophily graph datasets. The best performing method is highlighted.

Model	Cora	CiteSeer	PubMed	Cornell	Texas	Wisconsin	Actor
DNN	74.64 ± 0.63	71.30 ± 1.94	87.87 ± 0.32	67.74 ± 3.68	74.19 ± 5.22	71.74 ± 6.93	35.41 ± 0.93
GCN	87.22 ± 0.76	74.67 ± 1.33	87.40 ± 0.18	35.48 ± 4.60	48.39 ± 4.15	50.00 ± 5.58	27.67 ± 1.33
GAT	84.31 ± 1.21	71.89 ± 1.40	86.74 ± 0.08	45.16 ± 5.56	41.94 ± 5.33	63.04 ± 5.33	26.56 ± 0.98
GIN	81.93 ± 0.57	68.08 ± 0.57	85.76 ± 0.29	38.71 ± 5.16	29.03 ± 4.42	45.65 ± 4.71	23.61 ± 0.76
GCNII	87.22 ± 0.98	75.11 ± 1.18	86.19 ± 0.57	45.16 ± 4.87	51.61 ± 5.47	58.70 ± 3.25	28.72 ± 0.60
GraphSage	87.77 ± 1.83	76.72 ± 0.64	87.06 ± 0.74	70.97 ± 4.76	70.97 ± 4.46	73.91 ± 6.79	34.10 ± 0.65
CPGNN	79.40 ± 1.39	69.41 ± 0.45	77.40 ± 0.57	70.27 ± 5.12	75.68 ± 5.12	76.47 ± 6.16	35.59 ± 0.86
H2GCN	82.70 ± 0.87	43.70 ± 0.24	78.60 ± 0.62	70.27 ± 4.99	72.97 ± 4.53	70.59 ± 5.76	35.39 ± 1.34
GEOM-GCN	84.10 ± 1.12	76.28 ± 2.06	88.13 ± 0.67	54.05 ± 3.87	67.57 ± 5.35	68.63 ± 4.92	30.00 ± 1.26
GBK-GNN	**88.69 ± 0.42**	**79.18 ± 0.96**	**89.11 ± 0.23**	**74.27 ± 2.18**	**81.08 ± 4.88**	**84.21 ± 4.33**	**38.97 ± 0.97**
Error Reduction	**7.52%**	**10.57%**	**8.26%**	**11.37%**	**22.20%**	**32.89%**	**5.25%**

5.4.2 RQ2. How does GBK-GNN perform on different splits of datasets? We perform experiments on six different splits in which the training set contains 10% to 60% nodes on four datasets. Table 3 and 4 show that our model has the best performance on the most of the splits. On the homophily datasets (i.e., Cora and CiteSeer) GCNII achieves similar performance to our methods. It is partially caused by the depth of GCNII being much greater than ours (64 v.s. 2). On the heterophily datasets (i.e., Cornell and Texas), our method significantly outperforms the baselines on different splits. The results show that we can still have a decent gain when the proportion of the training set is smaller, which means that we take good advantage of labels even though they are few.

5.4.3 RQ3. How does GBK-GNN perform on the homophily nodes and heterophily nodes? Table 5 illustrates the test accuracy of GBK-GNN on the nodes range by homophily ratio on different datasets. We rank the nodes in the whole dataset by their homophily ratio and split them into five separations. Homophily of nodes can be seen as the probability that one node has the same label as its neighbor node. As shown in the table, both the nodes have a low homophily rate and high homophily rate have good performance. We suppose our bi-kernel feature transformation contributes to this phenomenon. If a node has a high homophily rate, W_s will contribute more to the performance. Otherwise, W_d will have more contribution. Since the test dataset may have a distribution shift, there may be no nodes in one proportion of nodes, especially on a small dataset, like Texas which have only 183 and 251 nodes. The test accuracy of GBK-GNN may have some drops on some certain proportion, for example, like 0 ~20 homophily ratio proportion on Cora. That is because the lack of test sample lead to a variance in the results. Cora only has 1.6% nodes in this proportion, so the variance is large.

5.4.4 RQ4. Does GBK-GNN learns the gate well? Table 6 shows the test accuracy of the gate learned by GBK-GNN. The ground truth is whether the neighbor node labels are consistent. It turns out that GBK-GNN has learned the gate well on homophily graphs and some heterophily graphs. Although the gate accuracy in Cornell and Texas datasets is worse than other datasets, the final node classification accuracy of these two datasets is much better than baselines. It suggests that the effectiveness of our method does not

completely depend on the accuracy of the Gate. As long as the accuracy reaches a certain level, the performance of our model can be improved.

6 RELATED WORK

Recently, Graph Neural Networks (GNNs) have been popular for graph-structured data on the node classification problems for it shows the strong ability to mine graph data. Deferred proposed one of the early versions of GNNs by generalizing convolutional neural networks (CNNs) for regular grids (e.g., images) to irregular grids (e.g., graphs) [6]. Kipf et.al proposed Graph Convolution Network (GCN) [13] by simplifying the convolution operation on graphs. After that, other GNN models are proposed to improve the representative power on graph tasks in different application scenarios, such as GAT [25], GIN [32] and GraphSAGE [10]. GAT introduced an attention mechanism to distinguish the importance of each neighbor node. GIN changes the aggregation function to enhance the ability of GNN for graph classification. GraphSAGE was designed for inductive learning for large graphs. These methods are generally proposed for the common graphs which satisfy the homophily assumption while other GNNs attempt to break the limits, including MixHop [1], Geom-GCN [22], and GCNII [4]. MixHop and Geom-GCN designed aggregation schemes that go beyond the immediate neighborhood of each node. The residual layer is introduced to leverage representations from intermediate layers to the previous layers. These models work well when the homophily ratio of the graph is high.

Recently, methods to research on the graph of heterophily arise as the previous GNNs fail to model these graphs. Xiao Wang et.al. proposed AMGCN [28] to introduce new graphs generated by the similarity of features and use three separate GCN to learn the node representations simultaneously. CPGNN [37] incorporates into GNNs a compatibility matrix that captures both heterophily and homophily by modeling the likelihood of connections between nodes in different classes. It overcomes two drawbacks of existing GNNs: it enables GNNs to learn from graphs with either homophily or heterophily appropriately and can achieve satisfactory performance even in the cases of missing and incomplete node features. Jiong Zhu et.al. [38] identified a set of key designs, including ego- and neighbor-embedding separation, higher-order neighborhoods,

Table 3: Evaluation of our GBK-GNN and other popular GNN models on different split of Cora and CiteSeer.

Model	Cora						CiteSeer					
	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
GraphSage	80.55	84.72	84.19	86.18	87.32	87.77	67.60	71.88	70.20	73.89	74.53	76.72
GCNII	83.65	**84.75**	86.24	87.58	87.78	87.22	69.39	71.42	70.29	71.81	72.63	75.11
GIN	61.61	76.30	76.49	79.81	80.84	81.93	54.43	62.61	63.97	64.59	66.23	68.08
GAT	80.39	79.91	80.29	81.61	84.01	84.31	68.84	68.89	67.85	70.52	72.27	71.89
GCN	82.06	83.48	85.93	87.10	87.11	87.22	69.53	71.65	70.54	71.71	72.27	74.67
GBK-GNN	**83.68**	84.68	**86.27**	**87.99**	**88.83**	**88.69**	**69.61**	**72.87**	**72.65**	**74.69**	**75.08**	**79.18**

Table 4: Evaluation of our GBK-GNN and other popular GNN models on different split of Cornell and Texas.

Model	Cornell						Texas					
	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
GraphSage	59.49	65.78	72.13	73.68	75.00	70.97	62.03	69.91	70.03	70.89	70.83	70.97
GCNII	58.23	63.16	62.69	52.63	52.08	45.16	51.90	56.58	62.69	56.14	54.17	51.61
GIN	46.84	56.58	52.24	52.63	43.75	38.71	26.58	26.32	40.30	38.6	43.75	29.03
GAT	46.84	65.79	58.21	52.63	52.08	45.16	18.99	59.21	56.72	52.63	52.08	41.94
GCN	36.71	57.89	61.19	59.65	52.08	35.48	51.90	56.58	58.21	47.37	47.92	48.39
GBK-GNN	**63.95**	**67.90**	**72.41**	**79.31**	**83.33**	**74.27**	**62.79**	**70.49**	**77.64**	**75.51**	**83.33**	**81.08**

Table 5: Mean classification accuracy of nodes range by homophily ratio in GBK-GNN.

Hom. Ratio Prop.(%)	Cora	CiteSeer	PubMed	Cornell	Texas	Wisconsin	Actor
0 ~20	28.91	26.66	61.42	64.28	80.00	79.31	30.97
20 ~40	50.00	42.10	71.34	100.00	N/A	77.77	42.13
40 ~60	75.00	66.66	88.98	100.00	100.00	100.00	47.14
60 ~80	91.66	96.07	92.53	100.00	N/A	66.66	47.91
80 ~100	97.86	92.44	94.46	100.00	N/A	50.00	42.85

Table 6: Mean classification accuracy of the gate in GBK-GNN.

Cora	CiteSeer	PubMed	Cornell	Texas	Wisconsin	Actor
82.39 ± 0.31	90.88 ± 0.22	76.07 ± 0.55	65.10 ± 0.34	66.46 ± 0.25	97.67 ± 0.24	78.50 ± 0.42

and a combination of intermediate representations, to boost the learning from the graph structure under heterophily.

7 CONCLUSION AND FUTURE WORK

In this paper, by analyzing the node-level homophily ratio, we find that the homophily levels may have significant differences in different local regions of a graph. Even for a graph with good overall homophily property, many nodes are still in the mix states, i.e., its neighbors have quite inconsistent labels. Furthermore, by both the empirical experiment results and the theoretical analysis, we demonstrate the fundamental incapability of the traditional GNN models, e.g., GCN, on modeling such mix states where a node has homophily neighbors and heterophily neighbors mixed together. In order to model both homophily and heterophily, we propose the novel gated bi-kernel GNN model, which learns two kernels to model homophily and heterophily, respectively, and learns the selection gate to make the corresponding kernel selection. The significant gain of the extensive evaluation results demonstrates the effectiveness of the approach.

The key idea of our approach is generic, which uses the selection gate and the multiple kernels to model the different kinds of neighbor relationships. In the next step, we plan to use the multi-kernel to model heterophily in a finer-grained manner for multi-class node classification where the total number of node classes is bigger than two. Under such a situation, there are multiple class label combinations, all belonging to heterophily. We could extend the current bi-kernel design to the muti-kernel design to model each kind of heterophily, i.e., each possible pair of class labels, respectively.

8 ACKNOWLEDGEMENTS

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 *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*.

[2] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 2011. Node Classification in Social Networks. *arXiv: Social and Information Networks* (2011), 115–148.

[3] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and Deep Graph Convolutional Networks. *CoRR* abs/2007.02133 (2020). arXiv:2007.02133 https://arxiv.org/abs/2007.02133

[4] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and Deep Graph Convolutional Networks. In *ICML (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 1725–1735.

[5] David L Davies and Donald W Bouldin. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence* 2 (1979), 224–227.

[6] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems* 29 (2016), 3844–3852.

[7] Lun Du, Fei Gao, Xu Chen, Ran Jia, Junshan Wang, Jiang Zhang, Shi Han, and Dongmei Zhang. 2021. TabularNet: A Neural Network Architecture for Understanding Semantic Structures of Tabular Data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 322–331.

[8] Gintare Karolina Dziugaite and Daniel M Roy. 2017. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008* (2017).

[9] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017). arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[10] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*.

[11] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2016. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836* (2016).

[12] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[13] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*.

[14] Qingqing Long, Yiming Wang, Lun Du, Guojie Song, Yilun Jin, and Wei Lin. 2019. Hierarchical community structure preserving network embedding: A subspace approach. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 409–418.

[15] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *ICLR*.

[16] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. 2021. Is Homophily a Necessity for Graph Neural Networks? *arXiv preprint arXiv:2106.06134* (2021).

[17] Sunil Kumar Maurya, Xin Liu, and Tsuyoshi Murata. 2021. Improving Graph Neural Networks with Simple Architecture Design. *arXiv preprint arXiv:2105.07634* (2021).

[18] Parth Natekar and Manik Sharma. 2020. Representation based complexity measures for predicting generalization in deep learning. *arXiv preprint arXiv:2012.02775* (2020).

[19] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. 2017. Exploring generalization in deep learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 5949–5958.

[20] Behnam Neyshabur, Ruslan Salakhutdinov, and Nathan Srebro. 2015. Path-SGD: path-normalized optimization in deep neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*. 2422–2430.

[21] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. 2015. Norm-based capacity control in neural networks. In *Conference on Learning Theory*. PMLR, 1376–1401.

[22] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2019. Geom-GCN: Geometric Graph Convolutional Networks. In *International Conference on Learning Representations*.

[23] Wenbin Tang, Honglei Zhuang, and Jie Tang. 2011. Learning to infer social ties in large networks. In *Joint european conference on machine learning and knowledge discovery in databases*. Springer, 381–397.

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

[25] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*.

[26] Junshan Wang, Zhicong Lu, Guojia Song, Yue Fan, Lun Du, and Wei Lin. 2019. Tag2vec: Learning tag representations in tag networks. In *The World Wide Web Conference*. 3314–3320.

[27] Xiao Wang, Meiqi Zhu, Deyu Bo, Peng Cui, Chuan Shi, and Jian Pei. 2020. Amgcn: Adaptive multi-channel graph convolutional networks. In *Proceedings of the 26th ACM SIGKDD International conference on knowledge discovery & data mining*. 1243–1253.

[28] Xiao Wang, Meiqi Zhu, Deyu Bo, Peng Cui, Chuan Shi, and Jian Pei. 2020. AM-GCN: Adaptive Multi-channel Graph Convolutional Networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1243–1253. https://doi.org/10.1145/3394486.3403177

[29] Yanlin Wang, Lun Du, Ensheng Shi, Yuxuan Hu, Shi Han, and Dongmei Zhang. 2020. *Cocogum: Contextual code summarization with multi-relational gnn on umls*. Technical Report. Microsoft, MSR-TR-2020-16. [Online]. Available: https://www.microsoft.com/en-us/research/publication/cocogum-contextual-code-summarization-with-multi-relational-gnn-on-umls.

[30] Yun Wang, Lun Du, Guojie Song, Xiaojun Ma, Lichen Jin, Wei Lin, and Fei Sun. 2019. Tag2Gauss: Learning Tag Representations via Gaussian Distribution in Tagged Networks.. In *IJCAI*. 3799–3805.

[31] Lanning Wei, Huan Zhao, and Zhiqiang He. 2021. Designing the Topology of Graph Neural Networks: A Novel Feature Fusion Perspective. *arXiv preprint arXiv:2112.14531* (2021).

[32] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

[33] Minghao Xu, Hang Wang, Bingbing Ni, Hongyu Guo, and Jian Tang. 2021. Self-supervised Graph-level Representation Learning with Local and Global Structure. *arXiv preprint arXiv:2106.04113* (2021).

[34] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings, Vol. 48)*, Maria-Florina Balcan and Kilian Q. Weinberger (Eds.). JMLR.org, 40–48. http://proceedings.mlr.press/v48/yanga16.html

[35] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. *CoRR* abs/1603.08861 (2016). arXiv:1603.08861 http://arxiv.org/abs/1603.08861

[36] Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. 2020. Graph neural networks with heterophily. *arXiv preprint arXiv:2009.13566* (2020).

[37] Jiong Zhu, Ryan A. Rossi, Anup B. Rao, Tung Mai, Nedim Lipka, Nesreen K. Ahmed, and Danai Koutra. 2020. Graph Neural Networks with Heterophily. *CoRR* abs/2009.13566 (2020). arXiv:2009.13566 https://arxiv.org/abs/2009.13566

[38] 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. *arXiv preprint arXiv:2006.11468* (2020).