RAW-GNN: RAndom Walk Aggregation based Graph Neural Network

Di Jin[1], **Rui Wang**[1], **Meng Ge**[1*], **Dongxiao He**[1*], **Xiang Li**[2], **Wei Lin**[2] and **Weixiong Zhang**[3]
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Meituan, Beijing, China
[3]Department of Health Technology and Informatics, The Hong Kong Polytechnic University, Kowloon, Hong Kong
{jindi, wr1895, gemeng, hedongxiao}@tju.edu.cn, weixiong.zhang@polyu.edu.hk

Abstract

Graph-Convolution-based methods have been successfully applied to representation learning on homophily graphs where nodes with the same label or similar attributes tend to connect with one another. Due to the homophily assumption of Graph Convolutional Networks (GCNs) that these methods use, they are not suitable for heterophily graphs where nodes with different labels or dissimilar attributes tend to be adjacent. Several methods have attempted to address this heterophily problem, but they do not change the fundamental aggregation mechanism of GCNs because they rely on summation operators to aggregate information from neighboring nodes, which is implicitly subject to the homophily assumption. Here, we introduce a novel aggregation mechanism and develop a RAndom Walk Aggregation-based Graph Neural Network (called RAW-GNN) method. The proposed approach integrates the random walk strategy with graph neural networks. The new method utilizes breadth-first random walk search to capture homophily information and depth-first search to collect heterophily information. It replaces the conventional neighborhoods with path-based neighborhoods and introduces a new path-based aggregator based on Recurrent Neural Networks. These designs make RAW-GNN suitable for both homophily and heterophily graphs. Extensive experimental results showed that the new method achieved state-of-the-art performance on a variety of homophily and heterophily graphs.

1 Introduction

Graphs are ubiquitous in the real world, such as social networks, brain networks, transportation networks and citation networks. Network analysis [Wang *et al.*, 2016; Jin *et al.*, 2021] has been extensively studied and broadly applied in many fields, ranging from computer science to social sciences, biology, physics, and many more. Recently, message-passing neural networks (MPNNs) [Kipf and Welling, 2017] have been successfully adopted for various problems on graphs, e.g., node classification, graph classification, link prediction, and anomaly detection [Wang *et al.*, 2022; Yu *et al.*, 2021].

An MPNN runs an iterative process and in each iteration, every node sends its features as a message to its neighbors and then aggregates messages from other nodes to update its representation. GCN [Kipf and Welling, 2017], as a typical MPNN, works under the homophily assumption – i.e., most connected nodes are from the same class and have similar attributes. GCN and its variants, like GraphSAGE [Hamilton *et al.*, 2017], possess great power for learning on graphs and have shown excellent performance on many downstream tasks on networks with homophily.

However, many real-world networks do not satisfy the homophily assumption. But rather, there exist many heterophily or low homophily networks where most adjacent nodes may belong to different classes and have dissimilar attributes. For example, in protein structures, an amino acid type is more likely to connect to different amino acid types rather than the same amino acid type; In email networks, spam users often contact normal users; In e-commerce networks, fraudsters are more likely to connect to accomplices than to other fraudsters. Conventional GCNs are not designed for heterophily networks and they use message propagation mechanisms based on the homophily assumption, as a result, information from different classes will get mixed during propagation on heterophily networks using such message propagation methods.

To deal with the homogeneity restriction in GCNs, several methods have already been proposed. Based on the aggregation mechanisms that they use, these methods can be divided into two categories: (1) Methods adjusting attention weights between neighbors of different labels in aggregation, which include GGCN [Yan *et al.*, 2021], CPGNN [Zhu *et al.*, 2021a], HOG-GCN [Wang *et al.*, 2022], GNN-LF/HF[Zhu *et al.*, 2021b] and BM-GCN [He *et al.*, 2022]. In essence, these methods assign a weight to each connected node pair with the guidance of heterophily information. In specific, label-guided methods [Zhu *et al.*, 2021a; Wang *et al.*, 2022; He *et al.*, 2022] integrate label information into their framework and aggregate nodes with the same label in the neighborhood and reduce the degree of aggregation of neighbors with different labels; Signed-message-guided methods [Yan *et al.*, 2021; Zhu *et al.*, 2021b] extend attention weight from

[*]Corresponding authors

$[0, 1]$ to $[-1, 1]$. As a result, messages between nodes of the same type of label are assigned positive weights, which boost message passing in the same class, and messages between nodes of different labels are given negative values, which prevent dissimilar neighbors from passing harmful and irrelevant information to one another. (2) Methods directly aggregating messages among higher-order neighbor nodes, including Geom-GCN [Pei *et al.*, 2020], H2GCN [Zhu *et al.*, 2020], GPR-GNN [Chien *et al.*, 2021] and LINKX [Lim *et al.*, 2021]. These methods assume that directly adjacent neighborhoods may be heterophily-dominant for heterophily network, but the higher-order neighborhoods are homophily-dominant which can provide more useful information for the target node. By explicitly aggregating information from higher-order neighborhoods, these methods alleviate the heterophily problem to certain extent.

However, these methods do not change the fundamental aggregation mechanism of GCNs because of the summation operator in the aggregation process, which is implicitly subject to the homophily assumption. To be specific, label-guided methods try to learn the weights between nodes of different labels to 0. In fact, this is equivalent to omitting the information of heterophily nodes in the aggregation, which is useful for network representation learning and downstream tasks. In signed-message-guided methods, weight -1 is introduced based on an analysis of graphs with two kinds of labels, so that this design is only effective on networks with a small class number close to 2 and do not work well for networks with large number of classes. In short, these designs alleviate the homogeneity restriction problem to some extent, but they are still constraint by the summation aggregator. In addition, the existing methods that directly aggregate higher-order neighbor use the classical summation aggregator to aggregate higher-order neighbors based on their distances to the target node and concatenate aggregated results with different distances. These methods also suffer from the constraints of the summation aggregator.

To address these problems, we introduce a new aggregation mechanism and propose a RAndom Walk aggregation-based Graph Neural Network, short-handed as RAW-GNN. We integrate random walk sampling into graph neural networks and extend the conventional neighborhoods to k-hop path-based neighborhoods. A k-hop path formed by random walks preserves the original attributes on this k nodes and the original structural connections of these nodes in the random walk sequence. In this way, the path-based neighborhoods can represent the neighborhood distribution of the target node better than the conventional neighborhoods. Furthermore, we utilize breadth-first search random walk (BFS) to capture homophily information and depth-first search (DFS) to collect heterophily information. To go beyond the exiting aggregation mechanism of GCNs and to take full advantage of the path-based neighborhoods, we adopt a sequential Recurrent Neural Networks (RNN) based aggregator, which can take into consideration the order information of neighbor nodes preserved by random walks. The RNNs have the advantage that they can handle the diverse attributes of adjacent nodes, which accommodates the need for heterophily networks. Finally, we use the attention mechanism to learn the importance of different paths from DFS channel (and BFS channel respectively), which can better extract heterophily (and homophily) neighborhood distribution with minimal mixing of information and can enable the model to automatically make a trade-off between homophily and heterophily according to different network characteristics.

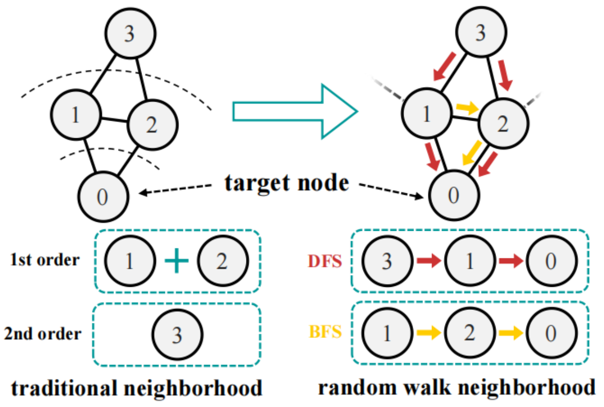


Figure 1: Existing methods omit some of the direct message channels between nodes of different distances to the target node, like edge $e_{1,3}$ and $e_{2,3}$. RAW-GNN employs the path-based neighborhoods detected by random walks to tackel this problem.

2 Preliminaries

We now present notations and related definitions, including path-based neighborhood and generalized homophily ratio, which are important for our work.

2.1 Basic Notations

Let $G = (V, E, X)$ denote an undirected, unweighted and attributed network, where $V = \{v_1, v_2, ..., v_n\}$ is a set of n nodes and $X \in \mathbb{R}^{n\times f}$ is a set of node attributes. Every node i is associated with f attributes x_i, which form the i-th row of X. $E = \{e_{ij}\} \subseteq V \times V$ is a set of edges represented by an adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{n\times n}$, where $a_{ij} = 1$ if nodes v_i and v_j are connected, or $a_{ij} = 0$ otherwise.

In this paper, we focus on semi-supervised node classification task. In a semi-supervised node classification task, every node belongs to a class $c \in C$ and $|C|$ is the number of classes. Here, the labels of nodes are given in set V_L. Every node $v_i \in V_L$ is assigned with a label $y_i \in L = \{1, 2, ...|C|\}$. The objective of node classification task is to predict the labels for all the unlabeled nodes in node set $V \backslash V_L$.

2.2 Message-Passing Neural Network

In a message-passing neural network, neighborhood is defined as neighbor nodes that are one or more hops away. Messages from nearby nodes are aggregated to the target node and updated iteratively. The l-th layer of a MPNN can be defined as follows:

$$\begin{aligned} n_i^{(l)} &= \text{aggregate}^{(l)}\left(\left\{h_i^{(l-1)} : i \in N_i\right\}\right) \\ h_i^{(l)} &= \text{combine}^{(l)}\left(h_i^{(l-1)}, n_i^{(l)}\right) \end{aligned} \tag{1}$$

where $h_i^{(l)}$ is the feature vector of node i in the l-th layer. The beginning vector $h_i^{(0)}$ is x_i, and N_i is a set of neighbor nodes of node i. Different choices of N_i, $\mathrm{aggregate}^{(l)}$ and $\mathrm{combine}^{(l)}$ result in different models. Usually, the neighbor nodes of node i are either the adjacent nodes of i or the k-hop neighbor nodes of i. In the following subsection, we will define another kind of neighborhood, which is based on paths.

2.3 Path-based Neighborhoods

The path-based neighborhood has already been applied to heterogeneous graphs [Fu *et al.*, 2020] and knowledge graphs [Du *et al.*, 2021]. Here we modify their definition to make it suitable for heterophily graphs. Formally, a path P is defined in the form of an ordered list $P = \{v_{p_1}, v_{p_2}, ..., v_{p_K}\}$, where $v_{p_k} \in V, k = 1, 2, ..., K$ and $e_{p_k p_{k+1}} \in E, k = 1, 2, ..., K-1$, and K is the length of the path. A path-based neighbor of node i is denoted as P_i, where the ending node in the list v_{p_K} is v_i. All path-based neighbor P_i of node i collected by strategy $s \in S$ forms the path-based neighborhood N_i^s, where S is the set of strategies. In other words, all $P_i \in N_i^s$.

2.4 Generalized Edge Homophily Ratio

The edge homophily ratio [Zhu *et al.*, 2020] measures the overall homophily in a graph. Specifically, it measures the fraction of edges that connect nodes that have the same label and it is defined as $H_E^{label}(G) = |\{(u, v) : e_{u,v} \in E \wedge y_u = y_v\}|/|E|$. This metric is based on node labels. Here we generalize edge homophily ratio to node features. First, we define the similarity function between two nodes as $\mathrm{sim} : V \times V \to [0, 1]$. This function should have such property that, when node i and node j are similar, $\mathrm{sim}(i, j)$ is close to 1 and when node i and node j are dissimilar, $\mathrm{sim}(i, j)$ is close to 0. Then the generalized edge homophily ratio $H_E : G \to [0, 1]$ is defined as:

$$H_E(G) = \frac{\sum_{(i,j)\in E} \mathrm{sim}(i, j)}{|E|} \tag{2}$$

when the similarity function $\mathrm{sim}(i, j)$ is Eq. (3), Eq. (2) becomes the label-based edge homophily ratio mentioned above.

$$\mathrm{sim}(i, j) = \begin{cases} 1, y_i = y_j \\ 0, y_i \neq y_j \end{cases} \tag{3}$$

We can also use node features to define node similarity, e.g., cosine similarity in Eq. (4), then we can generalize the concept of homophily ratio to features.

$$\mathrm{sim}(i, j) = \cos(x_i, x_j) \tag{4}$$

3 The Architecture

In this section, we describe the proposed RAW-GNN for homophily and heterophily graph embedding. We start with a brief overview and then introduce the details of components.

3.1 Overview

To let the aggregation mechanism truly go beyond the homophily assumption, here we propose a novel approach that consists of four components: random walk generator, RNN-based path aggregator, attention-based intra-strategy combinator and inter-strategy combinator. The whole framework of our approach RAW-GNN is shown in Fig. 2. Different from the frameworks of other GCNs, we encode the homophily and heterophily information into two different channels. In specific, we use two random walk generators to sample k-hop paths, i.e., using Breadth-First Search random walk (BFS) generator to sample homophily neighborhood distribution and Depth-First Search (DFS) generator to sample heterophily neighborhood distribution. Each sampling strategy (BFS or DFS) will sample multiple paths so as to capture the neighborhood distribution more accurately. After getting BFS neighborhood paths and DFS neighborhood paths, two RNN-based path aggregators are used to aggregate the ordered attributes of nodes in a path to form a path embedding. Then, we use the attention mechanism to learn the importance of different paths from DFS strategy and BFS strategy respectively, and form two strategy-specific embeddings. At last, we concatenate the two embeddings from BFS channel and DFS channel to form the whole embedding which can represent homophily and heterophily information together in the same framework.

3.2 Random Walk Generator

The key role of neighbors is to provide useful information for the target node, so it doesn't have to be like neighborhoods in classical GCNs. Here, we use paths to define neighborhood. However, given a receptive field, there are more paths than nodes so that it is not computationally feasible to use all the paths especially when the receptive field is large. To collect more information with fewer paths in the search space, we adopt a 2-nd order random walk with two parameters p and q from Node2Vec [Grover and Leskovec, 2016] to simulate Breadth-first Search and Depth-first Search.

Consider a random walk that has traversed edge $e_{t,s}$, is now on node s and is going to visit the next node r. We set the unnormalized transition probability as follows:

$$P_{pq}\left(v_i = r | v_{i-1} = s,\ v_{i-2} = t\right) = \begin{cases} 1/p \text{ if } d_{tr} = 0 \\ 1 \text{ if } d_{tr} = 1 \\ 1/q \text{ if } d_{tr} = 2 \\ 0\ otherwise \end{cases} \tag{5}$$

where d_{tr} is the length of the shortest path between nodes s and r. When we set $p < 1$ and $q > 1$, the random walk tends to behave like BFS; and when $p > 1$ and $q < 1$, the random walk tends to behave like DFS.

BFS collects information from the immediate neighbors of the target node and can extract homophily information (Fig. 2). DFS tends to sequentially reach nodes at increasing distances from the source node, which can extract heterophily information. Since real networks exhibit both homophily and heterophily, so both search strategies are applied.

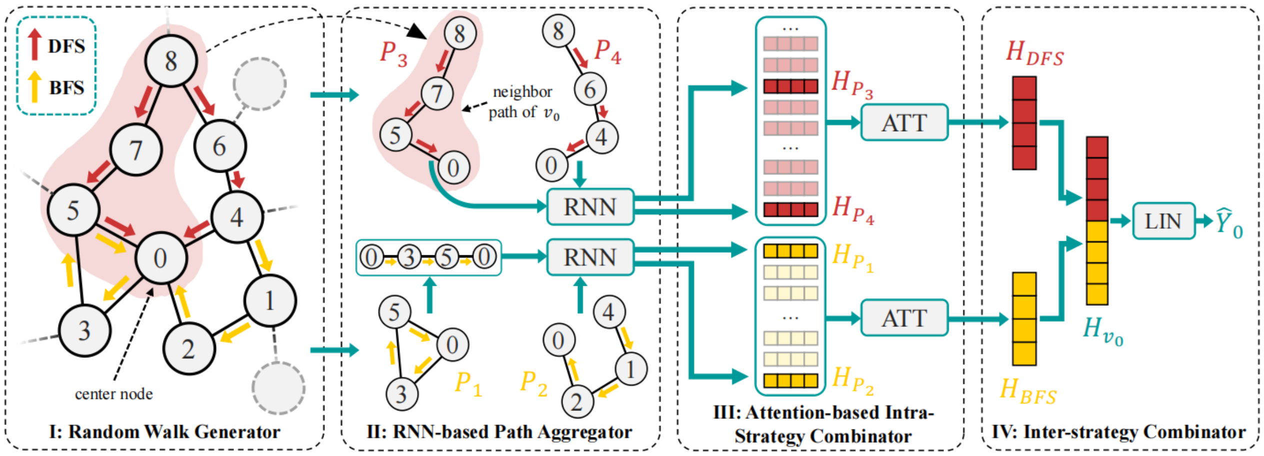


Figure 2: The framework of the proposed RAW-GNN. We extend the neighborhood of traditional GCNs to the ordered path-based neighborhood sampled by the random walk generator. Then the sequential RNN-based aggregator is applied. This combination can alleviate the problem brought by the difference of node attributes of adjacent nodes under heterophily. Next, the attention-based intra-strategy combinator receives path embeddings sampled by the same strategy and combines them into a strategy-specific embedding. Last, the inter-strategy combinator concatenates embeddings from different strategies to get the final embedding with the minimal information mixing.

3.3 RNN-based Path Aggregator

Given a path P_i, a path aggregator should learn the structural and semantic information of all nodes on P, not just the starting node v_i and the ending target node, but also all the context nodes in between. Note that the existing methods that treat all neighboring nodes as an unordered set. Nevertheless, a path naturally comes with an order, which also preserves the ordered connections among the nodes on the path. The objective of the path aggregator is to encode all the nodes on the path and consider the order of the nodes on the path to preserve more relational information among the connected nodes. The path aggregator $\mathrm{f_{path}}^{(l)} : \mathbb{R}^{K\times d_l} \to \mathbb{R}^{d_l}$ for the l-th layer is defined as:

$$h_P^{(l)} = \mathrm{f_{path}}^{(l)}(P) = \mathrm{f}_\theta(\left\{h_{n_1}^{(l)}, h_{n_2}^{(l)}, ..., h_{n_K}^{(l)}\right\}) \quad (6)$$

where $h_P^{(l)} \in \mathbb{R}^{d_l}$ is the l-th layer embedding vector of path $P = \{v_{n_1}, v_{n_2}, ..., v_{n_K}\}$, $h_{p_k}^{(l)} \in \mathbb{R}^{d_l}$ is the feature of node v_{n_k} in layer l, $k = 1, 2, ..., K$ and θ represents all the learnable parameters of the aggregator. To encode a path, we can use any encoder that takes the order of elements into consideration. Here we choose a simple sequence encoders GRU (Gate Recurrent Unit) [Cho *et al.*, 2014], a variant of RNN with a gating mechanism as the message function.

For simplicity, we set the layer to l and omit the superscript indicating the layer of an embedding in the rest of the paper unless otherwise specified.

3.4 Attention-based Intra-Strategy Combinator

After computing the path embeddings $h_P^s \in N_i^S$ for every node $v_i \in V$ for strategy s, we need to combine them into strategy-specific node embeddings. We assume that given a certain sampling strategy, the generated paths obey a corresponding distribution . Since all paths are sampled from the same distribution, it is reasonable to sum these paths to approximate the corresponding distribution. It is worth noting that this summation is different from combining node embeddings in GCNs because the pattern exhibited in a path cannot be well captured in a single node and beginning with summing node embeddings like typical GCNs weakens such path pattern. Furthermore, different paths may contribute differently to the target node embedding, so that we adopt an attention mechanism to learn the different weight of the path embeddings in N_i^S, which is defined as:

$$\begin{aligned} \mathrm{e}_P &= \mathrm{LeakyReLU}(a_P^T \cdot h_P^s) \\ \alpha_P &= \frac{\exp(\mathrm{e}_P)}{\sum_{Q\in N_i^S}\exp(\mathrm{e}_Q)} \\ h_i^s &= \sigma(\sum\nolimits_{P\in N_i^S} \alpha_P \cdot h_P^s) \end{aligned} \quad (7)$$

where $a_P \in \mathbb{R}^{d_l}$ is the learnable attention parameter, a_P^T denotes the transpose of a_P, e_P is the unnormalized importance weight of path P, and α_P is the path weight normalized across all the paths in N_i^S using softmax. It is worth noting that we do not consider the embedding of the target node h_i separately again during the combination procedure like Eq. (1), since every path in N_i^S already contains h_i as its ending node embedding. To further stabilize the learning process, we adopt the standard multi-head attention. Specifically, H attention heads in Eq. (7) are used and then their embeddings are concatenated, which is formalized as:

$$h_i^s = \mathop{\Big\Vert}_{h=1}^{H} \sigma(\sum\nolimits_{P\in N_i^S} \alpha_P^h \cdot h_P^s) \quad (8)$$

3.5 Inter-Strategy Combinator

After aggregating the path information within every strategy, we need to combine them using an inter-strategy combination layer. Since different strategies gather paths of different distribution, in order to combine embeddings from different strategies without mixing them, we use concatenation to combine embeddings from different strategies, rather than summing them as done in the GCN [Kipf and Welling, 2017] model, which is given by:

$$h_i = \overset{S}{\underset{s=1}{||}} h_i^s \tag{9}$$

where $h_i \in \mathbb{R}^{d_{final}}$ is the final embedding of node v_i, $d_{final} = H \times d_L \times |S|$, and $|S|$ is the number of strategies.

3.6 Classifier

In this work, we focus on the semi-supervised node classification task. We use a linear layer followed by a softmax to compute the predicted label probabilities, and employ the standard cross-entropy as the loss:

$$\hat{y}_i = \text{softmax}(h_i \cdot W)$$
$$L = -\frac{1}{|V_L|} \sum_{v_i \in V_L} \sum_{c=1}^{|C|} Y_{ic} \log(\hat{y}_{ic}) \tag{10}$$

where $W \in \mathbb{R}^{d_{final} \times |C|}$ is the learnable weight matrix and $Y_i, \hat{y}_i \in \mathbb{R}^{|C|}$ is the one-hot embedding of label y_i and the predicted label of i respectively, $Y_{ic} = 1$ when $c = y_i$. The other elements in Y_i are all set to zero.

4 Experiments

We now compare our RAW-GNN with the state-of-the-art models on the problems of node classification and visualization using seven real-world datasets varying from strong heterophily to strong homophily.

4.1 Datasets

To demonstrate that RAW-GNN can adaptively learn path propagation mechanism under both homophily and heterophily, we evaluate the performance of RAW-GNN and the existing state-of-the-art methods on seven real-world datasets, including three homophilic networks and four heterophilic networks. The features of these datasets are summarized in Table **??**. $L.H.R.$ represents the label-defined edge homophily ratio of the network. $F.H.R$ represents the cosine feature edge homophily ratio of the network.

Cora, Citeseer and Pubmed are homophilic citation network benchmark datasets [Sen *et al.*, 2008; Namata *et al.*, 2012], where nodes represent papers, and edges represent citations between papers. Node features are the bag-of-words representation of papers, and node labels are academic topics.

Cornell, Texas and Wisconsin are webpage datasets collected from computer science departments of corresponding universities [Pei *et al.*, 2020], where nodes represent web pages, edges are hyperlinks, node features are the bag-of-words representation of webpages, and node labels are pages categories (student, project, course, staff, and faculty). Actor is a heterophilic actor co-occurrence network [Tang *et al.*, 2009], in which nodes correspond 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. Node labels are categories in term of words of actor's Wikipedia.

Datasets	**Texa.**	**Wisc.**	**Acto.**	**Corn.**	**Cite.**	**Pubm.**	**Cora**
Nodes	183	251	7600	183	3327	19717	2708
Edges	309	499	33544	295	4732	44338	5429
Features	1703	1703	931	1703	3703	500	1433
Classes	5	5	5	5	6	3	7
$L.H.R$	0.06	0.18	0.22	0.30	0.74	0.80	0.81
$F.H.R$	0.35	0.34	0.16	0.31	0.19	0.27	0.17

Table 1: The Statistics of Datasets

4.2 Baselines

We compare our proposed approach RAW-GNN with the following baseline methods: 1) MLP (Multi-Layer Perceptron), which only uses node attributes; 2) Node2Vec [Grover and Leskovec, 2016], which only uses graph structures. Since our work adopt a similar random walk sampling strategy of Node2Vec, we add it for comparison; 3) Traditional GNN models : GCN [Kipf and Welling, 2017] and GraphSAGE [Hamilton *et al.*, 2017], which work under homophily assumption; 4) Frameworks designed for heterophily: H2GCN [Zhu *et al.*, 2020], CPGNN [Zhu *et al.*, 2021a], GPR-GNN [Chien *et al.*, 2021], BM-GCN [He *et al.*, 2022] and HOG-GCN [Wang *et al.*, 2022]. In this paper, we choose the best results of each method for comparison, since some methods have more than one variants.

4.3 Parameter Setup

Following [Pei *et al.*, 2020] and [Wang *et al.*, 2022], we generate 10 random splits for all datasets. In each dataset, 48% of the nodes are used as the training set, 32% of the nodes are used as the validation set, and the rest as the test set. For a fair comparison, all methods use the same 10 random splits. All the parameters of the baseline methods were set as what were used by their authors. For the random walk sampling in RAW-GNN, we use DFS strategy with $p = 10, q = 0.1$ and BFS strategy with $p = 0.1, q = 10$. We choose different path lengths from $\{3, 4, 5, 6, 7\}$ for different datasets. With every strategy, we sample 6 paths for each node in one epoch. For the RNN-based aggregator, we use GRU with 32 hidden units and attention head number is set to 2. The learning rate is set to 0.05. We adopt the Adam optimizer and the default initialization in Pytorch.

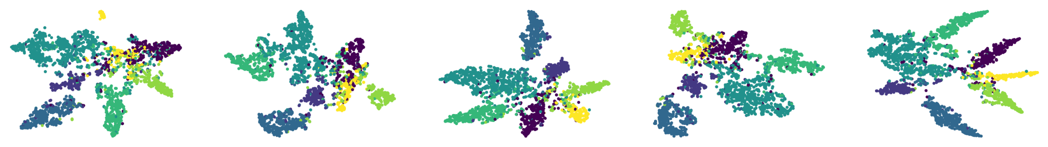
4.4 Node Classification

We conduct experiments on seven real-world datasets to compare the performance of different models for node classification (Table **??**). We use the mean accuracy and standard deviation as the evaluation metric. As shown, our RAW-GNN performs best on 5 out of 7 networks. Below are the detailed observations.

- Our RAW-GNN approach performs the best on all of the four heterophilic networks, i.e., Texas, Wisconsin, Actor, and Cornell, which empirically proves the effectiveness of RAW-GNN. To be specific, RAW-GNN outperforms heterophily-agnostic models, i.e., Node2Vec, GCN and GraphSAGE, by 34.20%, 25.33% and 2.81% respectively. This is largely because the rival methods cannot generalize to heterophilic scenarios. GraphSAGE performs relatively well and we assume the reason is the use of neighbor sampling like our RAW-GNN,

Dataset	Texas	Wisconsin	Actor	Cornell	Citeseer	Pubmed	Cora	Avg Acc	Avg Rank
MLP	83.24±5.77	86.47±3.33	36.49±1.16	84.32±6.14	69.10±2.69	86.37±0.64	72.98±2.31	74.14	6.00
Node2Vec(DFS)	46.76±4.84	41.96±4.90	23.40±1.17	47.57±5.43	52.00±2.40	76.20±0.46	77.91±2.32	52.26	9.86
Node2Vec(BFS)	44.05±6.29	40.39±5.28	23.33±0.96	45.95±8.29	45.80±2.62	65.76±0.44	72.03±1.86	48.19	11.00
GCN	55.68±9.61	53.73±7.65	30.64±1.49	55.14±7.57	74.81±1.87	87.25±0.56	86.60±1.44	63.41	8.29
GraphSAGE	85.41±5.16	85.49±3.53	35.99±1.52	78.38±6.84	74.29±1.67	89.30±0.57	86.42±1.55	76.47	5.57
H2GCN	82.16±8.21	82.57±3.21	36.48± 1.16	78.92±5.24	75.95±2.18	88.78±0.53	87.69±1.37	76.08	5.00
CPGNN	74.32±7.38	81.76±6.74	35.51±1.85	63.51±5.83	75.52±1.84	89.08±0.67	87.18±1.13	72.41	6.29
GPR-GNN	84.59±4.37	83.92±3.14	36.47±1.38	82.97±5.68	75.12±1.98	87.38±0.63	86.70±1.03	76.74	5.43
BM-GCN	85.13±4.64	82.82±8.89	36.32±1.35	79.14±8.44	75.94±2.36	**90.25±0.71**	87.71±1.11	76.76	3.86
HOG-GCN	85.17±4.40	86.67±3.36	36.82±0.84	84.32±4.32	**76.15±1.88**	88.79±0.40	87.04±1.10	77.85	2.86
RAW-GNN	**85.95±4.15(4)**	**88.24±3.72(4)**	**37.45±1.21(5)**	**84.86±5.43(4)**	75.38±1.68(5)	89.34±0.66(4)	**87.89±1.52(7)**	**78.44**	**1.71**

Table 2: Classification Results with mean value and standard deviation. The best result is bold and the second best is underlined. For our RAW-GNN, the number after deviation denotes the random walk path length chosen for the corresponding dataset.



(a) GCN (b) CPGNN (c) H2GCN (d) GPRGNN (e) RAWGNN

Figure 3: Visualization results on Cora dataset

which demonstrates the effectiveness of sampling in heterophily graphs. Compared with the methods designed for heterophily, such as H2GCN, CPGNN and GPR-GNN, BM-GCN, and HOG-GCN, RAW-GNN also achieves an improvement between 0.88% and 10.35% by mean accuracy. These results demonstrate the effectiveness of our new method on heterophily networks.

- On homophilic networks, i.e., Cora, Citeseer, and Pubmed, RAW-GNN also has competitive performance compared with the baselines. Specifically, RAW-GNN performs the best on Cora and the second best on Pubmed. We assume that since every dataset is a mixture of homophily and heterophily, the heterophily information from the DFS channel also helps model performance on these homophily datasets. Note that RAW-GNN outperforms GCN and GraphSAGE on all these datasets by 1.32% and 0.87% by mean accuracy. These results demonstrate that RAW-GNN still maintains comparable performance on homophilic datasets. Furthermore, for datasets with lower cosine feature homophily ratios ($F.H.R < 0.2$), the best path-length is longer, which indicates that heterophily networks need large receptive field to extract the hidden neighborhood distribution in the network. These results show the effectiveness and robustness of the proposed framework empirically.

4.5 Visualization

In addition to the quantitative node classification, we also visualize node embeddings on Cora dataset to assess the embedding results qualitatively. We project the node embeddings into a 2-dimensional space using t-SNE [LJPvd and Hinton, 2008]. Here we illustrate the visualization results of GCN, H2GCN, CPGNN, GPR-GNN and our RAW-GNN in Fig. 3, where points with different color indicate different classes. As shown, the visualization results of GCN and CPGNN are less satisfactory in this case, since points with the same class are dispersed and some points with different classes are mixed. As shown, the visualization of GPR-GNN and H2GCN are better but still blurred along the border of different classes. The result of our RAW-GNN is the best, where the border between different classes is the most discernible. This result is consistent with classification result.

5 Conclusion

In this paper, we propose a random walk aggregation based graph neural network that can process homophily and heterophily graphs at the same time. The proposed framework extends the neighborhoods in traditional GCNs to k-hop path-based neighborhoods generated by two random walk sampling strategies (i.e., breadth-first search and depth-first search). Then, the proposed framework uses the sequential RNN-based aggregator to encode the ordered attribute information of neighbor nodes. Then, the path embeddings for each strategy are gathered to the target node with an attention mechanism to form strategy-specific embedding. At last, node embeddings from different strategy channels are concatenated to prevent information of different characteristics from mixing and enable the model to automatically tradeoff between homophily and heterophily according to different network characteristics. Experiments on seven real-world datasets demonstrate that the proposed approach outperforms existing methods under heterophily, and also performs competitively under homophily.

Acknowledgments

This research was partly supported by the Natural Science Foundation of China under grants 61876128, the Tianjin Municipal Science and Technology Project (Grant No. 19ZXZNGX00030), and Meituan Project.

References

[Chien *et al.*, 2021] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *Proceeding of the 9th International Conference on Learning Representations*, 2021.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734, 2014.

[Du *et al.*, 2021] Zhengxiao Du, Chang Zhou, Jiangchao Yao, Teng Tu, Letian Cheng, Hongxia Yang, Jingren Zhou, and Jie Tang. Cogkr: Cognitive graph for multi-hop knowledge reasoning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[Fu *et al.*, 2020] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. MAGNN: metapath aggregated graph neural network for heterogeneous graph embedding. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2331–2341, 2020.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. Node2vec: scalable feature learning for networks. In *Proceedings of the 22th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 855–864. ACM, 2016.

[Hamilton *et al.*, 2017] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30*, pages 1024–1034, 2017.

[He *et al.*, 2022] Dongxiao He, Chundong Liang, Huixin Liu, Mingxiang Wen, Pengfei Jiao, and Zhiyong Feng. Block modeling-guided graph convolutional neural networks. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 2022.

[Jin *et al.*, 2021] Di Jin, Cuiying Huo, Chundong Liang, and Liang Yang. Heterogeneous graph neural network via attribute completion. In *Proceedings of the Web Conference 2021*, pages 391–400, 2021.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceeding of the 5th International Conference on Learning Representations*, 2017.

[Lim *et al.*, 2021] Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *Advances in Neural Information Processing Systems*, volume 34, pages 20887–20902, 2021.

[LJPvd and Hinton, 2008] Maaten LJPvd and GE Hinton. Visualizing high-dimensional data using t-sne. *J Mach Learn Res*, 9(2579-2605):9, 2008.

[Namata *et al.*, 2012] Galileo Mark Namata, Ben London, Lise Getoor, and Bert Huang. Query-driven active surveying for collective classification. In *Workshop on Mining and Learning with Graphs (MLG)*, 2012.

[Pei *et al.*, 2020] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *Proceeding of the 8th International Conference on Learning Representations*, 2020.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine.*, 29(3):93–106, 2008.

[Tang *et al.*, 2009] Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 807–816, 2009.

[Wang *et al.*, 2016] Zheng Wang, Chaokun Wang, Jisheng Pei, Xiaojun Ye, and Philip S. Yu. Causality based propagation history ranking in social networks. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 3917–3923, 2016.

[Wang *et al.*, 2022] Tao Wang, Di Jin, Rui Wang, Dongxiao He, and Yuxiao Huang. Powerful graph convolutioal networks with adaptive propagation mechanism for homophily and heterophily. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 2022.

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

[Yu *et al.*, 2021] Zhizhi Yu, Di Jin, Ziyang Liu, Dongxiao He, Xiao Wang, Hanghang Tong, and Jiawei Han. Asgcn: Adaptive semantic architecture of graph convolutional networks for text-rich networks. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 837–846. IEEE, 2021.

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

[Zhu *et al.*, 2021a] Jiong Zhu, Ryan A. Rossi, Anup B. Rao, Tung Mai, Nedim Lipka, Nesreen K. Ahmed, and Danai Koutra. Graph neural networks with heterophily. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021.

[Zhu *et al.*, 2021b] Meiqi Zhu, Xiao Wang, Chuan Shi, Houye Ji, and Peng Cui. Interpreting and unifying graph neural networks with an optimization framework. In *WWW '21: The Web Conference 2021*, pages 1215–1226, 2021.