Hybrid Convolution Method for Graph Classification Using Hierarchical Topology Feature

Jiangfeng Sun sun2017@bupt.edu.cn
Xinyue Lin bellalin0@foxmail.com
Fangyu Hao haofangyu@bupt.edu.cn
Meina Song mnsong@bupt.edu.cn
School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

Graph classification is a crucial task in the field of graph learning with numerous practical applications. Typically, the first step is to construct vertex features by the statistical information of the graph. Existing graph neural networks often adopt the one-hot degree encoding strategy to construct vertex features. Then, these features are fed into a linear layer, which outputs a low-dimensional real vector serving as the initial vertex representation for the graph model. However, the conventional approach of constructing vertex features may not be optimal. Intuitively, the method of constructing vertex features can have significant impact on the effectiveness of model. Hence, the construction of informative vertex features from the graph and the design of an efficient graph model to process these features pose great challenges. In this paper, we propose a novel method for constructing hierarchical topology vertex features and designing a hybrid convolution method to handle these features. Experimental results on public graph datasets of Social Networks, Small Molecules, and Bioinformatics demonstrate the superior performance of our method compared to baselines.

Keywords: Vertex Feature Construction, Graph Convolution, Hierarchical Topology Feature, Hybrid Convolution Method.

1. Introduction

Graph is consisted of vertices and edges, where vertices represent entities and edges represent relationships between entities. And graph is everywhere. That is why many researchers pay attention to graph learning recently. Up to now, neural networks achieved excellent performance in many tasks of graph learning, such as vertex classification, graph classification, and link prediction. Graph neural networks (GNNs), a kind of well-preforming models, have been widely applied to deal with graph classification. Briefly review it below, GNNs can be divided into Graph Convolutional Neural Networks (GCNs), Graph Auto Encoders (GAEs), and Graph Attention Networks (GATs). In order to study representations of vertices, we can use initial vertex features and aggregate them over the topological structure. In this way, a common pre-step when using these neural networks to process graph data is to construct vertex features (if no pre-prepared vertex features are available). As the first step, this operation can be considered as the basis of graph learning. Therefore, how to construct effective vertex features becomes an important issue in graph learning.

In recent works, one of the most widely used methods is to construct vertex features by degree. Degree-based vertex features are essentially characterized by the connectivity of vertices in the graph. It can intuitively reflect the connectivity property of vertices, which is well reasonable and easy to manage with high efficiency. However, in complex graphs, the degree cannot fully represent the importance of a vertex, and often obeys a power-law distribution, which means using only degree may not be a proper way.

For the learning process, convolution handle the transfer and fusion of graph information, it is an essential and widely used mathematical method. Generally, the graph convolution can be categorized into two types: spectral convolution and spatial convolution. For spectral convolution, we use the Hadamard product and Fourier inverse transform on the input signal in the spectral domain to aggregate vertices' information. And for spatial convolution, it defines convolution operation directly in the spatial domain, this gives it more flexibility, while it also suffers from over-smoothing Li et al. (2018). We have noticed that both convolution methods above only use the 1-hop adjacency relations as the aggregation field of vertex features (see Figure 1(a)), which may not be informative enough intuitively. Obviously, how to construct a more powerful vertex feature and a more effective convolution method deserve further exploration.

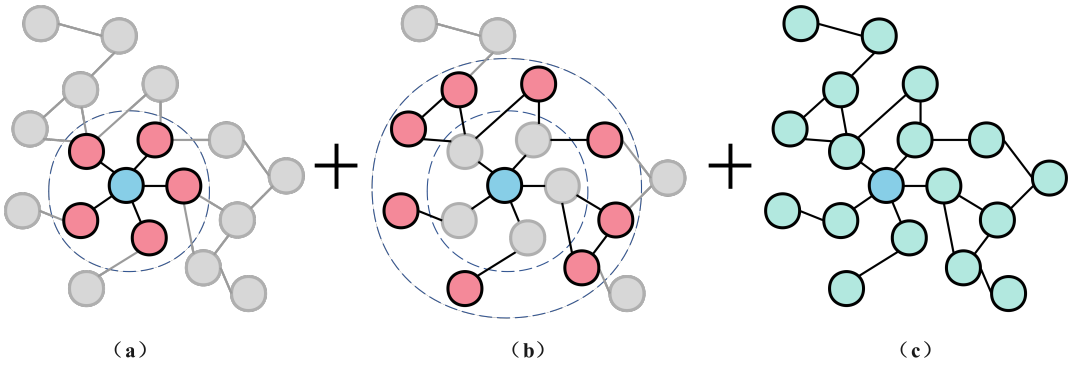

Figure 1: Different Levels of Structural Information (a) Classical Convolution using 1-hop Adjacency; (b) Convolution using 2-hop Adjacency; (c) Global topological structure information with multi-hop neighbors.

Figuratively Speaking, the essence of graph convolution can be regarded as vertex features aggregation by graph structure. In this paper, in order to utilize the rich structural information in graph, we propose a new graph topology feature construction method called Hierarchical Topology Feature (HTF), which constructs topology features fusing different levels of topological relationships based on the adjacency relationship of vertices in the graph (Figure 1 (a)&(b)&(c)) and use the One-Hot degree encoding method as the origin vertex features. Based on the HTF, this paper designs a graph convolution method called Hybrid Convolution Method (HCM), which is specialized in processing these fused features and can operate convolution on different levels of topology at the same time. Our contributions are as follows:

1) Hierarchical Topology Feature (HTF): We introduce a pseudo-node in the global graph and use a fixed step length random walker to obtain the global topological structure of the vertex, while the first-order adjacency matrix is fused to form a structure matrix containing the hierarchical topological structure. After giving the initial vertex features by one-hot degree encoding, the features are multiplied with the structure matrix to obtain the target vertex features. In addition, the use of One-Hot degree encoding also alleviates problems due to uneven degree distribution, because it uses the degree as the index of the feature vector with a value of 1 and all others are 0.

2) Hybrid Convolution Method (HCM): Based on the characteristics of HTF, this paper modifies the traditional graph convolution formula to form a model that can simultaneously perform different levels of graph convolution and fusing them together. In HCM, vertex features can be computed on the first-order neighborhood structure, second-order neighborhood structure, and random neighborhood structure (which represents the global structure information) of the graph in a single convolution operation, greatly improving the efficiency and learning ability of graph convolution models. Specifically, we use additive arithmetic to perform all the fusion operations.

3) Mathematics and Experiments: We give the mathematical form of the HTF and HCM, and explain through derivation that these method can perform convolution on different levels structures. In addition, experimental results on benchmark graph classification datasets from different domains show that our Hybrid Convolution Method is highly competitive with state-of-the-art models, and significantly outperforms many other baseline models for graph classification.

2. Related Work

Graph learning refers to the process of extracting meaningful information and patterns from graph structured data. The typical feature of graph is that data are organized into nodes (vertices) and edges, where nodes represent entities and edges represent relations between nodes. The goal of graph learning is to leverage the inherent structural information of graphs to solve various tasks such as node classification, link prediction, and graph classification. Graph convolution networks (GCNs), are classical and powerful method for learning from graphs. Graph convolution can be divided into spectral approaches (including GCN, AGCN Park et al. (2020), STSGCN Song et al. (2020), etc.) and spatial approaches (including GAT, GraphSAGE, SACNN Li et al. (2020a), etc.). In recent years, GCN research has evolved simultaneously in the optimization of computational resources Gao et al. (2018); Hu et al. (2021), feature enhancement Abu-El-Haija et al. (2019), and node embedding Lee et al. (2019); Mendonça et al. (2020). An end-to-end structure-aware convolutional network (SACN) Shang et al. (2019) combines the benefits of GCN and ConvE together. It has the composition of an encoder that can adapt from the local information quantity and a decoder with identical connection prediction performance to ConvE. Conventional GCNs often suffer from limited expressiveness due to the simplicity of input node features, as well as confinement to local neighborhood structures. Recent advances have attempted to enhance graph learning in different ways. Some work obtain richer node features by

sampling techniques such as random walks, while others explore hierarchical pooling to select and aggregate features of different granularity.

Many regular methods are designed to learn node-level representations based on the adjacent neighbor clustering scheme Tan et al. (2022). Specifically, they update the node embeddings by iteratively clustering the embeddings of adjacent nodes, for example, GCN Kipf and Welling (2016), GAT Veličković et al. (2017), and GraphSAGE Hamilton et al. (2017). These models share the commonality of using information from adjacent neighbors to learn and update node representations. GCN considers first-order neighbors, which may underperform in handling long distance dependencies. GAT incorporates attention mechanisms to dynamically learn the importance of different nodes, but this comes at the cost of higher computational overhead and resource demands. GraphSAGE allows node sampling and is applicable to large-scale graphs, but sampling may lead to information loss, making it unsuitable for all graph structures. In this paper, Hierarchical Topology Feature(HTF) achieves sufficient random walk sampling by introducing pseudo-node. This enables it to directly and effectively integrate global and local multi-granularity structural information, learn node features with richer expressive capabilities, and provide more robust structural support for subsequent graph-learning tasks. The integrated information support in HTF encompasses the global nature of the neighbor structure, the computational efficiency, and the richness of characteristics. Additionally, a network embedding algorithm uses a random walk (similar to the skip gram) method to capture the local structure of a graph Rozemberczki et al. (2021). Furthermore, a GNN architecture based on Generalized PageRank (GPR) optimizes node features and topological information extraction through adaptive learning of GPR weights Chien et al. (2020). Recent advancements in Graph Neural Networks (GNNs) have introduced methods for encoding node connection relationships, effectively encapsulating structural information in a manageable fashion Li et al. (2020b); Dwivedi et al. (2021). Graph convolutional networks (GCNs) has been widely used in computer vision Pang et al. (2022), recommendation systems Zhang et al. (2019), knowledge graph Li et al. (2021) graph classification Nagar et al. (2021), and flow prediction Guo et al. (2019).

Compared to these methods, HTF realizes the multi-granularity of structural information aggregation in a more direct and efficient way, without complex training processes or additional feature extraction modules. By aggregating the first-order adjacency matrix and the global random walk matrix, HTF jointly incorporates the global and local structure at the feature level, providing richer structured feature inputs. Hierarchical pooling can be used to store information from different granularity, supported by relevant research. e.g. ASAP Ranjan et al. (2020), SAGPool Lee et al. (2019), DiffPool Ying et al. (2018), and Graph U-Net Gao and Ji (2019). HTF realizes the joint utilization of multi-granularity structural information in a more concise and efficient way. Geometric graph convolution network (Geom-GCN) Pei et al. (2020) that breaks free from the constraints of long-term dependencies in message passing neural networks (MPNN) and addresses the challenge of losing neighborhood structure. Geo-GCN achieves efficient transformation learning of the graph, encompassing node embedding, structural neighborhood, and dual-level aggregation. While Geom-GCN focuses on overcoming long-term dependency issues and neighborhood structure loss, HCM concentrates on multi-granularity convolution and hierarchical information integration. Compared to Geom-GCN, the advantage of HCM lies in its introduction of multi-granularity convolution, improved convolution efficiency, enhanced learning

capabilities, and support for hierarchical information integration. HCM introduces multi-granularity convolution, bolsters convolution efficiency, enhances learning capacity, and facilitates hierarchical information integration, establishing it as a potent and versatile graph convolution method for various graph learning tasks. In contrast to SACN, HCM is based on HTF, allowing the more comprehensive capture of structural information from the graph and providing a more efficient and flexible solution for the processing of graph data.

3. Preliminary

In this paper, we work on constructing vertex features containing more information about graph structure and more efficient convolution operations. Given a graph G, we define the vertices as V, the edges as E, the first-order adjacency matrix as A, the pseudo-node connected to all nodes as P, the perceptible global topology of vertex which is defined by a fixed step length random walker as matrix T, the degree matrix as D, the initial encoding of the vertex features by one-hot degree as H, and the different levels of convolution use a shared weight matrix W.

4. Methodology

In this section, we will detail the vertex feature construction method and the graph convolution method in this paper. From the general workflow, the HTF is first constructed, then the HTF is entered into a linear layer for pre-processing, after which it is convoluted using the HCM. In HCM, a convolution operation includes two parts, hybrid convolution and linear transformation. After two layers of HCM, the output is obtained by two fully connected layers.

4.1. Hierarchical Topology Feature Construction

For a graph G, we look at the different levels structures in the graph from the vertex perspective, and in this paper, we divide the structural information into three part. First, the first-order nearest neighborhood is the basic structural information in the graph. By analyzing the first-order nearest neighbors, the nature of connections between them can be revealed. This structure can be represented by the first-order adjacency matrix A, in which the existence of edges between vertices is indicated by 0 or 1. The second-order adjacency structure indicates the number of two-step connections between vertices. By analyzing the second-order adjacency matrix, the connectivity between the far distant neighbors can be explored which is important for discovering hidden information at the community level, it can be represented by A^2. The global structure refers to the global topology of the entire graph, that is, all nodes in the graph and the connectivity between them, and the global structure provides a higher-level perspective that can reveal the features and patterns of the entire graph, and it is represented using the matrix T in this paper.

In this paper, these three different levels of graph structures are combined with the construction of vertex features and the convolution method. It is easy to get first-order adjacency structure and the second-order neighbor structure, so how to obtain the global graph structure is a key point here.

We introduce a pseudo-node P in the graph which is the first-order neighbor of all vertices, and then perform a fixed-step random wander from each vertex v_i (except the pseudo-node P), and record the vertices v_j visited by the random wander using a matrix T with all zeroes initially, that is, the position (v_i, v_j) is recorded as 1. In this way, after we have performed the above operations on all vertices, we can obtain a matrix containing the multi-hop neighbor relation information which is the global topological structure information T. The main purpose of introducing pseudo-node is to avoid the problems of random wandering interruption and imbalance sampling caused by isolated vertices. In addition, the step length of random wandering is set to a fixed value of 6 in this paper, referring to the sociological six-degree theory, i.e., the phenomenon that people can be connected with each other through a limited chain of social relationships, and any two people can be connected with each other through no more than six intermediaries at most.

After obtaining A and T, we use the One-Hot degree method to initialize the vertex feature encoding H. Specifically, for a graph G with n vertices, the degree of each vertex can be represented by a vector $d = [d_1, d_2, ..., d_n]$, where d_i is the degree of vertex i, then we can initialize the vertex feature using structure based method, one-hot degree encoding, the feature for vertex i is defined as: $h_i = [0, 0, ..., 1, .., 0]$, in this vector, the 1 is placed at the index corresponding to the degree of vertex i. For example, if vertex i has a degree of d_i, then the 1 is placed at index d_i. Then, we add A and T, and multiply them with H to obtain the HTF.(See Fig.2)

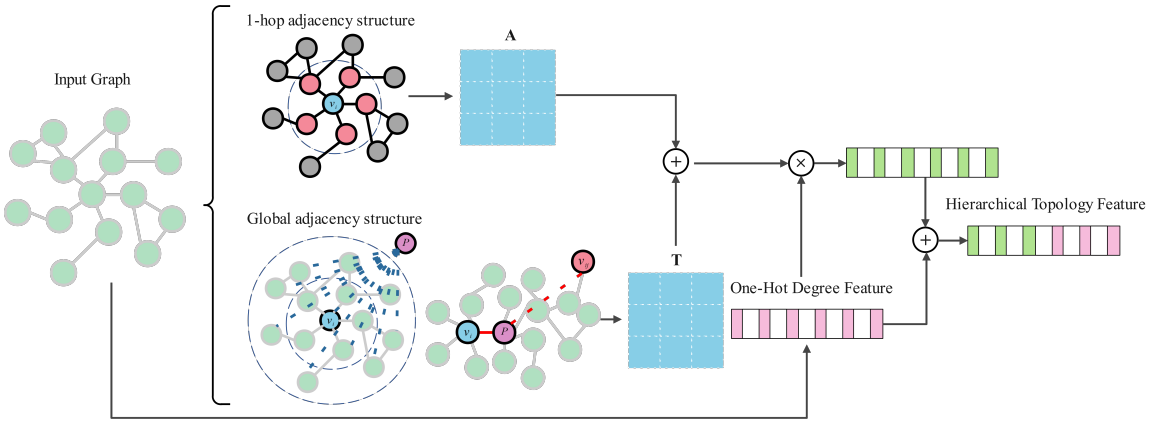

Figure 2: Hierarchical Topology Feature Construction Method

HTF can be represented mathematically as:

$$HTF = ((A + T)H + H) \tag{1}$$

where A represents the first-order adjacency matrix, T represents the global topological structure matrix, H represents the initial vertex features encoded by One-Hot degree, and '+' denotes matrix addition.

4.2. Hybrid Convolution Method

Based on HTF, we propose a new graph convolution called Hybrid Convolution Method (HCM). This method takes HTF as input and enables convolution on first-order structure,

second-order structure, and global structure in the graph, which is greatly enhancing the efficiency and learning capability of graph convolution. And we use H to represent vertex features, then HTF is defined as $((A+T)H+H)$, and the degree matrix is denoted as D. The convolution operation is then defined as follows:

$$D^{-1}A((A+T)H+H) \tag{2}$$

We can expand it as follows:

$$D^{-1}AH + D^{-1}A^2H + D^{-1}ATH \tag{3}$$

Here, AH represents the first-order adjacency convolution, A^2H represents the second-order adjacency convolution, and ATH represents the global structure convolution. This expanded form allows us to perform convolution on the first-order, second-order, and global structures of the graph, enhancing the efficiency and learning capacity of the convolution. We can write the iterative formula for obtaining the final output by adding the results of these convolutions and passing it through a shared weight matrix W.

$$Z^{(l+1)} = D^{-1}A((A+T)Z^{(l)} + Z^{(l)})W \tag{4}$$

where Z is the output at the $(l+1)$ layer, D^{-1} is the inverse degree matrix. This formula shows the propagation of information across the graph through multiple layers of the convolution operation.

We use the negative log loss function as the objective function for training, and the loss function can be expressed as:

$$L = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} log(p_{i,j}) \tag{5}$$

where N represents the number of graphs, C represents the number of classes, $y_{i,j}$ is the true label value of the j^{th} class of the i^{th} sample, and $p_{i,j}$ is the predicted probability value of the j^{th} class of the i^{th} sample. log represents the natural logarithm function.

5. Experiments

5.1. Experimental Setup

To express convenience, this paper uses HCM to represent the whole method proposed in this article. In order to validate the effectiveness of HCM, experiments are conducted on six datasets, including Social Networks, Bioinformatics, and Small Molecules. For each dataset, the vertex features are constructed by HTF. The detail of the datasets are summarized in Table 1.

In our experiments, a 2-layer HCM model is used with a total of 200 iterations. The learning rate is set to 0.001, dropout rate is set to 0.6, and we use a fixed random walk step length which is set to 6. The datasets are splited into training, validation, and testing sets in an 8:1:1 ratio. The proposed methods in this paper require relatively low GPU memory usage, and the experiments are conducted on an NVIDIA GeForce RTX 3090 GPU, using the PyTorch 1.11.0 framework and CUDA 11.3 environment. Furthermore, the experiments are repeated three times to ensure the stability of the results.

Table 1: Summary of the datasets used in our experiments

	SOCIALNETWORKS			BIOINFO	MOLECULES	
	COLLAB	IMDb-B	IMDb-M	PROTEINS	MUTAG	PTC-MR
# Graphs	5000	1000	1500	1113	188	344
# Avg. Nodes	74.49	19.77	13.00	39.06	17.93	14.29
# Avg. Edges	2457.78	96.53	65.94	72.82	19.79	14.69
# Classes	3	2	3	2	2	2
# Training Graphs	80%	80%	80%	80%	80%	80%
# Validation Graphs	10%	10%	10%	10%	10%	10%
# Test Graphs	10%	10%	10%	10%	10%	10%

5.2. Baselines

We compare the HCM with nine baselines for graph classification.

- DropGIN Papp et al. (2021): This model aims to deal with the limitations of standard GNN, and it can also distinguish multiple kinds of graph neighbors that cannot be distinguished by the message-passing GNNs.

- GIN Xu et al. (2018): Graph Isomorphism Network (GIN) proposes theoretical framework to enhance the expressive power of GNNs. The enhancement includes neighborhood aggregation and graph readout functions, showing experimentally that the model is as powerful as WL.

- WEGL Kolouri et al. (2020): The fast framework proposed by this model supports the entire graph vector embedding. Meanwhile, it proposes a low-complexity graph similarity analysis method to measure the differences between node embeddings.

- CT-Layer Arnaiz-Rodríguez et al. (2022): This model adds the weight function for learning the commute times in the traditional message passing framework, and can obtain a better global topology structure.

- PPGN Maron et al. (2019): The first one proposed a model with a 3-WL expressiveness. This model innovatively integrates the multi-layer perceptron (MLP) into the matrix multiplication (second-order operation) and the feature dimension.

- GFN & GFN-light Chen et al. (2019): This paper linearizes the graph filtering and set function parts of the graph classification task respectively, and finds that the linear graph filtering with nonlinear set function can handle the graph classification task efficiently and accurately.

- DGCNN Wu et al. (2018): The model mainly solves the problem that the graph structure information is lost during the transmission process. First, the disordered graph convolutional layer (DGCL) is used in the pretreatment layer. Secondly, in the convolutional layer, DGCNN is further optimized by learning the irregular neighborhood characteristics.

- CapsGNN Xinyi and Chen (2019): This article mainly addresses the graph embedding defects of the existing GNN. It optimizes graph-level expression by a routing mechanism. Secondly, Multiple-Attention are integrated to strengthen the weight of key parts.

5.3. Experiment Results

We compared the graph classification test accuracy of HCM with the baselines, and the results are shown in Table 2. The results indicate that HCM consistently outperforms other methods.(See Table 2) According to our experiments, we have achieved better performance on these datasets for graph classification.

Table 2: Graph Classification Test Accuracy

Model	**COLLAB**	**IMDb-B**	**IMDb-M**	**PROTEINS**	**MUTAG**	**PTC-MR**
CT-Layer	69.87%	69.84%	-	75.38%	87.58%	-
DropGIN	-	75.70%	51.40%	76.30%	90.60%	66.20%
WEGL	79.80%	75.40%	52.00%	76.50%	88.30%	67.50%
CapsGNN	79.62%	73.10%	50.27%	76.28%	86.67%	-
GFN-light	81.34%	73.00%	51.20%	77.44%	89.89%	-
GFN	81.50%	73.00%	51.80%	76.46%	90.84%	-
PPGN	81.38%	72.60%	50.00%	77.20%	90.55%	66.17%
GIN	80.20%	75.10%	52.30%	76.20%	89.40%	64.60%
DGCNN	73.70%	70.00%	47.80%	75.10%	85.80%	65.43%
HCM	**82.80%**	**77.00%**	**55.33%**	**80.36%**	**100%**	**68.57%**

We can see that HCM is better than all the baseline models, and we believe that the main reason is that we use the equation (4) as the computing method which can aggregate first-order adjacency relations, second-order adjacency relations and higher-order information simultaneously. And this may due to the fact that there are many potential relations or structures in the graph which are not connected directly between two vertices, we use the HTF to rebuild these relations and HCM to learn from this kind of features. The ablation study demonstrates it.

In addition, we notice that the model achieves 100% accuracy on MUTAG. We believe it's because that the task of MUTAG is to identify whether the compound would have a mutagenic effect on Salmonella typhimurium and its task is very clear and loose with small amount of graphs. And the Salmonella typhimurium may be associated with some specific molecular structures, so it is only necessary to find the presence or absence of such a molecular structure. We also find that the split rate does affect the results, using more training sets can lead to more competitive results. So we also do a split rate analysis for our model and our model still shows competitive performance, we use */*/* to represent the different training/validation/test split rate. According to our experiments, the split rate does affect the performance within 1.2%. (See Figure 3).

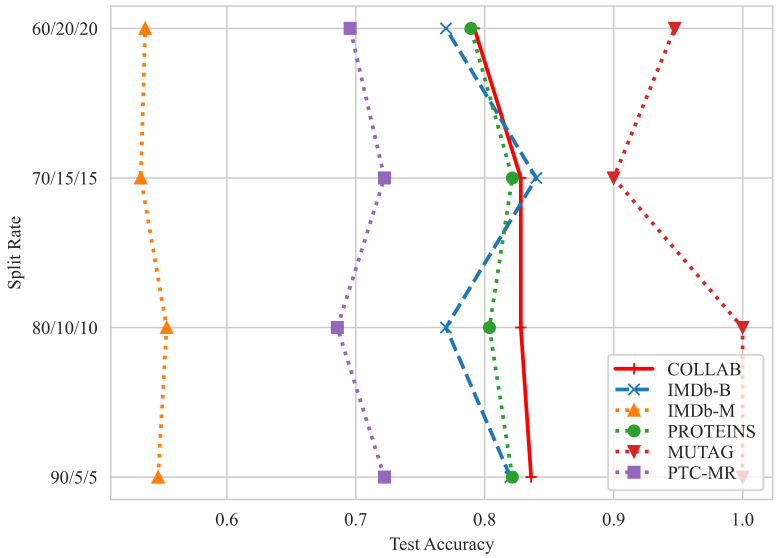

Figure 3: Test accuracy of HCM under different split rate

5.4. Ablation study

We think that HTF is the key factor that affects the model's expression ability. So we just use the initial vertex feature as the model's input. The vertex features are also initialized by one-hot degree method. And we find that the model with HTF outperforms the one without HTF. We use HCN to represent the model without HTF and HCM to represent the model with HTF. In the ablation study we divide the datasets into training/validation/test according to 80%/10%/10%. (See Table 3). The experimental results show that without

Table 3: Ablation Study Results of HCM (Test Accuracy)

Model	COLLAB	IMDb-B	IMDb-M	PROTEINS	MUTAG	PTC-MR
HCN	81.8%	74%	52.67%	74.1%	89.4%	57.14%
HCM	**82.80%**	**77.00%**	**55.33%**	**80.36%**	**100%**	**68.57%**

HTF, the model performance decreased by 1% on the COLLAB, 3% on the IMDb-B, 6.26% on the PROTEINS, 10.6% on the MUTAG, and 11.43% on the PTC-MR, and the models without HTF all show significant decreases, which indicates that HTF indeed significantly affect the model, and aggregating vertex information at higher-order positions in graph convolution can effectively improve the model's performance.

6. Conclusion

This study focuses on the fundamental element of graphs, the vertices. Existing graph neural networks typically rely on vertex features as the basic input for learning. Based on the intuition that vertex features with richer structural information would yield bet-

ter performance, the study proposes a more effective vertex feature construction approach called HTF (Hierarchical Topology Feature). Additionally, leveraging the characteristics of HTF, the study also introduces a HCM (Hybrid Convolution Method) that can perform convolutions across different hierarchical structures simultaneously.

The effectiveness of the proposed methods is demonstrated through experiments conducted on Social Network, Bioinformatics, and Small Molecule graph datasets. This highlights the significance of structural information in graphs for various graph learning tasks. The study primarily focuses on static graph vertex features and convolution methods. However, real-world scenarios often involve dynamic graphs with temporal attributes. Thus, future work will involve designing more effective vertex feature construction methods and graph learning models for dynamic graphs.

Acknowledgments

This work is supported by the National Key Research and Development Program of China under Grant 2020AAA0107500, in part by the National Natural Science Foundation of China under Grant 62076035. Engineering Research Center of In-formation Networks, Ministry of Education.

References

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pages 21–29. PMLR, 2019.

Adrián Arnaiz-Rodríguez, Ahmed Begga, Francisco Escolano, and Nuria Oliver. Diffwire: Inductive graph rewiring via the lov\'asz bound. *arXiv preprint arXiv:2206.07369*, 2022.

Ting Chen, Song Bian, and Yizhou Sun. Are powerful graph neural nets necessary? a dissection on graph classification. *arXiv preprint arXiv:1905.04579*, 2019.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. *arXiv preprint arXiv:2006.07988*, 2020.

Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*, 2021.

Hongyang Gao and Shuiwang Ji. Graph u-nets. In *international conference on machine learning*, pages 2083–2092. PMLR, 2019.

Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1416–1424, 2018.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 922–929, 2019.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Zhiyao Hu, Dongsheng Li, Dongxiang Zhang, Yiming Zhang, and Baoyun Peng. Optimizing resource allocation for data-parallel jobs via gcn-based prediction. *IEEE Transactions on Parallel and Distributed Systems*, 32:2188–2201, 2021.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Soheil Kolouri, Navid Naderializadeh, Gustavo K Rohde, and Heiko Hoffmann. Wasserstein embedding for graph learning. *arXiv preprint arXiv:2006.09430*, 2020.

Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *International conference on machine learning*, pages 3734–3743. PMLR, 2019.

Meng Li, William Hsu, Xiaodong Xie, Jason Cong, and Wen Gao. Sacnn: Self-attention convolutional neural network for low-dose ct denoising with self-supervised perceptual loss network. *IEEE transactions on medical imaging*, 39:2289–2301, 2020a.

Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. Distance encoding: Design provably more powerful neural networks for graph representation learning. *Advances in Neural Information Processing Systems*, 33:4465–4478, 2020b.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. Temporal knowledge graph reasoning based on evolutional representation learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 408–417, 2021.

Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *Advances in neural information processing systems*, 32, 2019.

Matheus RF Mendonça, André MS Barreto, and Artur Ziviani. Approximating network centrality measures using node embedding and machine learning. *IEEE Transactions on Network Science and Engineering*, 8:220–230, 2020.

Omer Nagar, Shoval Frydman, Ori Hochman, and Yoram Louzoun. Quadratic gcn for graph classification. *arXiv preprint arXiv:2104.06750*, 2021.

Shuyang Pang, Xuewen Xiao, Yuao Cui, Shangwei Mao, Xin Cao, Hongsheng Jia, Hao Wang, Fenghua Tong, and Xiaohui Zhang. Gcn-unet: A computer vision method with application to industrial granularity segmentation. *Mobile Information Systems*, 2022, 2022.

Pál András Papp, Karolis Martinkus, Lukas Faber, and Roger Wattenhofer. Dropgnn: Random dropouts increase the expressiveness of graph neural networks. *Advances in Neural Information Processing Systems*, 34:21997–22009, 2021.

Chanhee Park, Jinuk Park, and Sanghyun Park. Agcn: Attention-based graph convolutional networks for drug-drug interaction extraction. *Expert Systems with Applications*, 159: 113538, 2020.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

Ekagra Ranjan, Soumya Sanyal, and Partha Talukdar. Asap: Adaptive structure aware pooling for learning hierarchical graph representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5470–5477, 2020.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3060–3067, 2019.

Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 914–921, 2020.

Yue Tan, Yixin Liu, Guodong Long, Jing Jiang, Qinghua Lu, and Chengqi Zhang. Federated learning on non-iid graphs via structural knowledge sharing. *arXiv preprint arXiv:2211.13009*, 2022.

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

Bo Wu, Yang Liu, Bo Lang, and Lei Huang. Dgcnn: Disordered graph convolutional neural network based on the gaussian mixture model. *Neurocomputing*, 321:346–356, 2018.

Zhang Xinyi and Lihui Chen. Capsule graph neural network. In *International conference on learning representations*, 2019.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.

Jiani Zhang, Xingjian Shi, Shenglin Zhao, and Irwin King. Star-gcn: Stacked and reconstructed graph convolutional networks for recommender systems. *arXiv preprint arXiv:1905.13129*, 2019.