DUAL PATH GRAPH CONVOLUTIONAL NETWORKS

Yunhe Li

University of Montreal
Montreal, Quebec, Canada

Yaochen Hu, Yingxue Zhang

Huawei Noah's Ark Lab
Huawei Technologies Canada
Montreal, Quebec, Canada

ABSTRACT

Graph Convolutional Networks (GCNs) are a powerful approach for learning graph representations and show promising results in various applications. Despite their success, they are usually limited to shallow architectures due to the vanishing gradients, over-smoothing, and over-squashing problems. As Convolutional Neural Networks benefit tremendously from stacking very deep layers, recently techniques such as various types of residual connections and dense connections are proposed to tackle these problems and make GCNs go deeper. In this work, we further study the problem of designing deep architectures for GCNs. Firstly, we introduce the Higher Order Graph Recurrent Networks (HOGRNs), which can unify most existing architectures of GCNs. Then we show that ResGCN and DenseGCN are special cases of HOGRNs. To enjoy the benefits from both residual connections and dense connections and compensate for the drawbacks from each other, we propose Dual Path Graph Convolutional Networks (DPGCNs), which exploit a new topology of connection paths internally. In DPGCNs, we maintain both a residual path and a densely connected path while learning the graph representations. Extensive experiments on OGB datasets demonstrate superior performances of the proposed DPGCNs over competitive baseline methods on the large-scale graph learning tasks of node property prediction and graph property prediction.

Index Terms— Graph Convolutional Networks, Graph Representation Learning

1. INTRODUCTION

Graph Convolutional Networks (GCNs) are a powerful deep learning tool for graph-structured data and have been gaining a lot of attention in recent years. GCNs and their variants [1, 2, 3] have demonstrated to be valuable in a wide range of applications, including modelling proteins for drug discovery [4], social network analysis [5], recommendation systems [6], point clouds classification and segmentation [7], and computer vision [8].

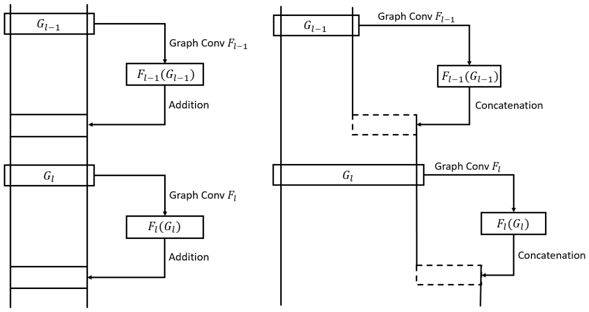
Despite the huge success of GCNs, most of them are usually limited to shallow architectures. For examples, GCN [1] and GAT [3] achieve their best performance at 2-layer on the datasets of citation networks. These shallow architectures lack the ability to extract long-range information from the high-order neighbours. Some applications such as molecular property prediction usually require long-range information, because chemical properties of a molecule may depend on the combination of atoms at its opposite sides [9]. Therefore, deeper architectures of GCNs are needed to capture such long-range information in these applications. However, stacking more layers into a GCN may lead to several problems, including the vanishing gradient, over-smoothing [10] and over-squashing [11] problems. Over-smoothing [10] is a problem that as the number of layers in a GCN increases, the output node features may tend to converge to the same vector and become indistinguishable. The phenomenon of over-squashing [11] is caused by a information bottleneck in which as the number of layers increases, information from the exponentially-growing neighbours is compressed into fixed-size vectors.

Recently, several works of various residual connections are proposed to tackle the above problems. Borrowing ideas from Convolutional Neural Networks (CNNs), ResGCN [12] leverages residual connections and dilated convolutions to train very deep GCN architectures. DeeperGCN [13] proposes novel generalized aggregation functions and a pre-activation version of residual connections to reliably train very deep GCNs. GCNII [14] is proposed to leverage initial residual connections and identity mapping to train deep GCNs. These works empirically demonstrate that residual connections can relieve the over-smoothing and vanishing gradient problems. Adding residual connections can ensure that there exists a deep GCN achieving at least the same performance as its shallow counterpart does: the shallow layers of the deep GCN are copied from the shallow GCN, and the other layers of the deep GCN are identity mappings [15, 14]. In addition, residual connections encourage the feature re-usage and thus reduce the feature redundancy [16]. Adding an identity mapping to the propagation function can also improve the expressive power of GCNs. However, adding residual connections may still suffer from the over-squashing problem as the number of layers in the GCN increases.

In addition to residual connections, various dense connections are recently proposed to tackle the over-smoothing and over-squashing problems. JKNet [17] leverages dense skip connections to combine all node feature vectors from the previous layers to preserve the locality of the final node representations. DenseGCN [12] exploits dense connectivity among different GCN layers, which can improve information flows in the GCN. There are several advantages of dense connections as well. Dense connections can flexibly leverage different neighborhood ranges to enable better structure-aware representations and preserve the locality of the node representations. By combining all node feature vectors from the previous layers, dense connections relieve the problem of over-smoothing, because only the parts from the deep layers tend to converge to the same vector. Dense connections can also alleviate the over-squashing problem. As the number of layers increases, the length of the output node feature vectors increases as well. This can reduce the information bottleneck caused by compressing information from the exponentially-growing neighbours into fixed-size vectors. Moreover, dense connections encourage to explore new features from previous layers' outputs but may suffer from high feature redundancy [16].

In this work, inspired by Dual Path Networks [16] from com-

978-1-6654-0540-9/22/$31.00 ©2022 IEEE



(a) ResGCN (b) DenseGCN

Fig. 1. The illustrations of the network architectures of ResGCN and DenseGCN.

puter vision, we further study the problem of designing deep architectures for GCNs. Firstly, we introduce the Higher Order Graph Recurrent Networks (HOGRNs) which can unify most existing architectures of GCNs. Then we show that ResGCN and DenseGCN are special cases of HOGRNs when the aggregation functions across layers of HOGRNs are a summation operator and a concatenation operator, respectively. To enjoy the benefits from both residual connections and dense connections and relieve the drawbacks from each other, we propose Dual Path Graph Convolutional Networks (DPGCNs) which present a new topology of connection paths internally. In DPGCNs, we use two kinds of aggregation functions across layers so that we maintain both a residual path and a densely connected path. DPGCNs are the first GCN architecture leveraging both residual connections and dense connections at the same time. Extensive experiments on Open Graph Benchmark (OGB) datasets clearly demonstrate superior performances of our proposed DPGCNs over all the competitive baseline methods on the large-scale graph learning tasks of node property prediction and graph property prediction.

2. PRELIMINARIES

2.1. Notations

A graph G is defined as a tuple of two sets $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \cdots, v_N\}$ is the set of nodes and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the set of edges. For an undirected graph, $e_{ij} = (v_i, v_j) \in \mathcal{E}$ is an undirected edge between nodes v_i and v_j; for a directed graph, e_{ij} is a directed edge pointing from the node v_i to the node v_j. We associate each node $v \in \mathcal{V}$ with a node feature vector $\boldsymbol{h}_v \in R^D$ where D is the number of dimensions of node feature vectors. Hence the graph G is associated with a set of node feature vectors $\mathcal{H} = \{\boldsymbol{h}_v | v \in \mathcal{V}\}$.

We define a general graph mapping F as a mapping which maps a graph $G = (\mathcal{V}, \mathcal{E})$ associated with a set of node feature vectors $\mathcal{H}$ to a graph $G' = (\mathcal{V}, \mathcal{E}')$ associated with a set of node feature vectors $\mathcal{H}'$, i.e. $G' = F(G)$. Graph convolutional networks such as GCN [1], MPNN [2], GAT [3], GIN [18], GCNII [14] as well as their corresponding graph convolutional operators are kinds of graph mappings.

2.2. Residual Connections in GCNs

Designing deep architectures of GCNs still remains an open problem. Borrowing the ideas from the huge success of ResNet [15], ResGCN [12] is proposed to make GCNs go deeper. Specifically, the graph mapping at the l-th layer in ResGCN [12] is formulated as the following:

$$\begin{aligned} G_{l+1} &= F_l(G_l) + G_l \\ &= \sum_{k=0}^{l} F_k(G_k) + G_0 \end{aligned} \tag{1}$$

where $F_k(\cdot)$ is a graph convolutional operator at k-th layer. After G_l is transformed by $F_l(\cdot)$, a node-wise addition operating on the sets of node feature vectors is performed to obtain G_{l+1}. Figure 1(a) graphically shows the network architecture of ResGCN.

2.3. Dense Connections in GCNs

Inspired by DenseNet [19], DenseGCN [12] adapts a similar idea to GCNs in order to exploit information flows from different layers. Specifically, the graph mapping at the l-th layer in DenseGCN [12] is formulated as the following:

$$\begin{aligned} G_{l+1} &= T(G_l, F_l(G_l)) \\ &= T(G_0, F_0(G_0), F_1(G_1), \cdots, F_l(G_l)) \end{aligned} \tag{2}$$

where $F_k(\cdot)$ is a graph convolutional operator at k-th layer, $k = 0, 1, \cdots, l$, and T is a node-wise concatenation operator operating on the sets of node feature vectors that densely fuses the input graph G_0 with the outputs of all intermediate layers of the GCN. Figure 1(b) graphically shows the network architecture of DenseGCN.

3. HIGHER ORDER GRAPH RECURRENT NETWORKS

We first introduce the Higher Order Graph Recurrent Networks (HOGRNs). The update rule of HOGRNs is as the following:

$$G_k = J^k\left(\xi^k(G_0, F_0^k(G_0), F_1^k(G_1), \cdots, F_{k-1}^k(G_{k-1}))\right) \tag{3}$$

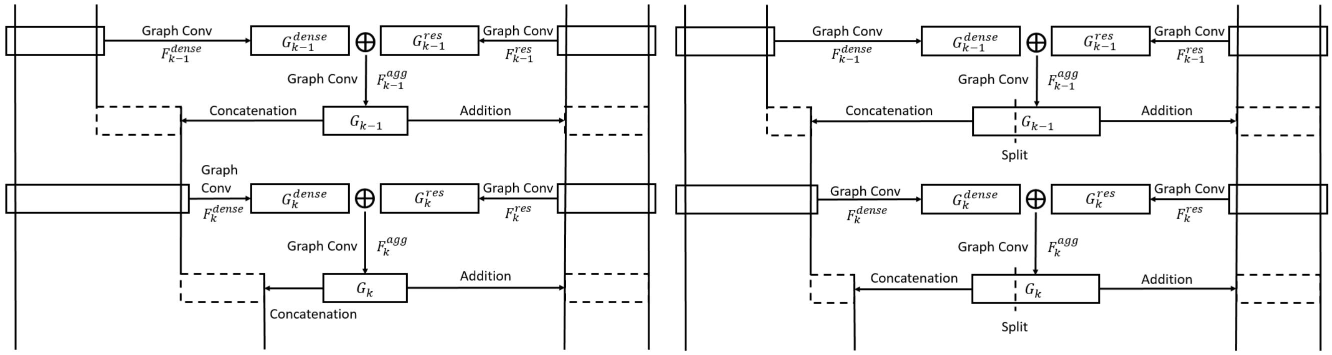
where k is the index of the current step, G_t denotes the graph produced by the HOGRNs at the t-th step, and G_0 is the input graph. For $t = 0, 1, \cdots, k-1$, $F_t^k(\cdot)$ is a differentiable graph mapping which takes the graph produced at the t-th step as input and outputs a transformed graph at the current step k. ξ^k is the node-wise aggregation function across layers at the k-th step operating on the sets of node feature vectors. ξ^k can be a summation operator, a concatenation operator, a weighted summation operator, an attention operator, a pooling operator, etc. $J^k(\cdot)$ is a differentiable graph mapping which transforms the aggregated graph to the output graph at the current step k.

Then we show that ResGCN is a special case of HOGRNs. If taking $F_t^k(\cdot) = F_t(\cdot)$ for $t = 0, 1, \cdots, k-1$, the aggregation function across layers ξ^k as the summation operator, and $J^k(\cdot)$ as an identity graph mapping, Eqn.(3) will become the same as Eqn.(1). Therefore, ResGCN is a special case of HOGRNs.

Finally, we show that DenseGCN is a special case of HOGRNs. If taking $F_t^k(\cdot) = F_t(\cdot)$ for $t = 0, 1, \cdots, k-1$, the aggregation function across layers ξ^k as the concatenation operator T, and $J^k(\cdot)$ as an identity graph mapping, Eqn.(3) will become the same as Eqn.(2). Therefore, DenseGCN is a special case of HOGRNs.

4. DUAL PATH GRAPH CONVOLUTIONAL NETWORKS

Residual connections in GCNs can relieve the over-smoothing and vanishing gradient problems, improve the expressive power of GCNs, and encourage the feature re-usage and thus reduce the feature redundancy, but may suffer from the over-squashing problem.



(a) A plain DPGCN.

(b) A more efficient version of DPGCN.

Fig. 2. The illustrations of the network architectures of a plain DPGCN and a carefully designed more efficient DPGCN. $\bigoplus$ denotes the aggregation function across dual paths which can be a node-wise concatenation operator or a node-wise summation operator operating on the sets of node feature vectors.

On the other hand, dense connections can flexibly leverage different neighborhood ranges to enable better structure-aware representations, preserve the locality of the node representations, relieves the over-smoothing and over-squashing problems, and encourage to explore new features from previous layers' outputs, but may suffer from high feature redundancy.

Based on the above analysis, to enjoy the benefits from both residual connections and dense connections and relieve the drawbacks from each other, we propose Dual Path Graph Convolutional Networks (DPGCNs) which present a new topology of connection paths internally. In DPGCNs, we use two kinds of aggregation functions across layers so that we maintain both a residual path and a densely connected path. We formulate DPGCNs as follows:

$$G_k^{dense} = F_k^{dense}\left(\xi_k^{dense}(G_0, G_1, \cdots, G_{k-1})\right), \tag{4}$$

$$G_k^{res} = F_k^{res}\left(\xi_k^{res}(G_0, G_1, \cdots, G_{k-1})\right), \tag{5}$$

$$G_k^{agg} = \xi_k^{agg}(G_k^{dense}, G_k^{res}), \tag{6}$$

$$G_k = F_k^{agg}(G_k^{agg}), \tag{7}$$

where ξ_k^{dense} and ξ_k^{res} are the aggregation functions across layers of the densely connected path and the residual path at k-th layer respectively, $F_k^{dense}(\cdot)$ and $F_k^{res}(\cdot)$ are differentiable graph mappings of the densely connected path and the residual path at k-th layer respectively, ξ_k^{agg} is the aggregation function across dual paths at k-th layer, and $F_k^{agg}(\cdot)$ is a differentiable graph mapping which takes the aggregated graph G_k^{agg} as input and outputs the transformed graph G_k at k-th layer. ξ_k^{dense} can be a node-wise concatenation operator, while ξ_k^{res} can be a node-wise summation or weighted summation operator. ξ_k^{agg} can be a node-wise concatenation or summation operator.

Figure 2 graphically shows two versions of DPGCNs. In the more efficient version of DPGCN in Figure 2(b), the two aggregation functions across layers ξ_k^{dense} and ξ_k^{res} include split operations. Thus the growth rate of the densely connected path in Figure 2(b) is smaller than that in Figure 2(a). As a result, the number of parameters in Figure 2(b) is smaller than that in Figure 2(a) when they have the same number of layers.

5. EXPERIMENTS

To evaluate the effectiveness of our proposed carefully designed deep architectures for GCNs, we conduct extensive experiments on the Open Graph Benchmark (OGB) [20]. Firstly, we do ablation study on the ogbn-proteins dataset. Then we evaluate our proposed DPGCNs on all the 4 datasets and compare the performances with competitive baseline methods.

5.1. Datasets and Experimental Setup

Datasets. We conduct our experiments on the recently published datasets of Open Graph Benchmark (OGB) [20], which is a diverse set of challenging and realistic benchmark datasets to facilitate scalable, robust, and reproducible graph machine learning research. In this work, our experiments are conducted on two OGB datasets (ogbn-proteins and ogbn-arxiv) for node property prediction and the other two OGB datasets (ogbg-molhiv and ogbg-ppa) for graph property prediction.

Node Property Prediction. For the task of node property prediction, the two chosen datasets are protein-protein association networks (ogbn-proteins) and paper citation networks (ogbn-arxiv). ogbn-proteins is an undirected, weighted, and typed graph containing $132,534$ nodes and $39,561,252$ edges. For ogbn-proteins, the task is to predict the presence of protein functions in a multi-label binary classification setup and evaluated by the average of ROC-AUC scores. ogbn-arxiv consists of $169,343$ nodes and $1,166,243$ directed edges. For ogbn-arxiv, the task is to predict the primary categories of the arxiv papers and evaluated using accuracy.

Graph Property Prediction. For the task of graph property prediction, the two chosen datasets are molecular graphs (ogbg-molhiv) and protein-protein association networks (ogbg-ppa). ogbg-molhiv has $41,127$ graphs. For ogbg-molhiv, the task is to predict the target molecular properties and evaluated by ROC-AUC scores. ogbg-ppa consists of $158,100$ protein association graphs. For ogbg-ppa, the task is to predict what taxonomic group the graph originates from and evaluated by accuracy.

Implementation Details. We follow the same experimental setup used in [13]. The size of hidden channel is set to 64. The growth rate of the densely connected path is set to 16. An Adam optimizer with a learning rate of 0.01 is used to train models for 500 epochs. For the graph convolution operators $F_k^{dense}(\cdot)$ and $F_k^{res}(\cdot)$ in Eqn.(4) and Eqn.(5), we use the GENeralized Graph

Table 1. Comparisons with competitive baseline methods. The notation * denotes that virtual nodes are used. The notation - denotes that the results of the baseline methods are not reported on the official OGB leaderboards for the corresponding datasets.

	UniMP	DeepGCN	DeeperGCN	GCNII	GIN	GIN*	GSN	**Ours**
ogbn-proteins	0.8642	0.8496	0.8580	-	-	-	-	**0.8649**
ogbn-arxiv	0.7311	-	0.7192	0.7274	-	-	-	**0.7354**
ogbg-ppa	-	-	0.7712	-	0.6892	0.7037	-	**0.7727**
ogbg-molhiv	-	-	0.7858	-	0.7558	0.7707	0.7799	**0.7871**

Table 2. The AUC scores for ablation study on the dual paths on the ogbn-proteins dataset.

# of layers	DPGCN-Res	DPGCN-Dense	DPGCN	DPGCN*
5	0.8391	0.8150	**0.8411**	0.8402
10	0.8450	0.8384	**0.8507**	0.8471
20	0.8489	0.8513	**0.8571**	0.8558
40	0.8523	0.8596	0.8626	**0.8628**
60	0.8556	0.8598	**0.8642**	0.8640
80	0.8575	0.8597	**0.8649**	0.8631

Convolution (GENConv) operator [13]. For the graph convolution operator $F_k^{agg}(\cdot)$ in Eqn.(7), we use the ClusterGCN graph convolutional operator [21]. We use the more efficient version of DPGCN in Figure 2(b) as the default DPGCN in the experiments. We implement our models based on PyTorch Geometric [22] and run our experiments on NVIDIA V100 32GB. More detailed information about implementations can be found in the code repository.

5.2. Results

Effect of Dual Paths. Our proposed DPGCNs are composed of both a residual path and a densely connected path. To show the effect of dual paths, we construct some ablated models. DPGCN-Res is the same as DPGCN except that it only maintains a residual path. DPGCN-Dense is the same as DPGCN except that it only maintains a densely connected path. DPGCN* is the same as DPGCN except that DPGCN* use the summation operator as the aggregation function across dual paths, while DPGCN use the concatenation operator as the aggregation function across dual paths. We conduct ablation study on the ogbn-proteins dataset.

Experimental results in Table 2 shows that DPGCN outperforms other models on average. The models with dual paths are better than the models only maintaining a single path. This means that DPGCNs can really exploit the benefits from both residual connections and dense connections, and further improve the overall performance. We also observe that as the depth of the models increasing, the performances become better. This means that protein property prediction can benefit from the deeper architectures of GCNs. Because this kind of applications usually require long-range information and thus deeper architectures of GCNs are needed to take advantage of these long-range interactions.

Comparison with Competitive Baseline Methods. We apply our proposed DPGCNs to 4 OGB datasets and compare the results with competitive baseline methods posted on the OGB learderboards. These competitive baseline methods include UniMP [23], DeepGCN [12], DeeperGCN [13], GCNII [14], GIN [18], GIN with virtual nodes and GSN [24]. The provided experimental results on each dataset are obtained by averaging the results from 10 independent runs. Experimental results in Table 1 shows that our proposed DPGCNs outperform all the competitive baseline methods in all four datasets. The improvements on the four datasets are substantial and thus demonstrate the effectiveness and advantages of our proposed DPGCNs.

6. CONCLUSION

In this work, we study the problem of designing deep architectures for GCNs. Firstly, we introduce the Higher Order Graph Recurrent Networks (HOGRNs) which can unify most existing architectures of GCNs. Then we show that ResGCN and DenseGCN are special cases of HOGRNs. To exploit the benefits from both residual connections and dense connections and compensate for the drawbacks from each other, we propose Dual Path Graph Convolutional Networks (DPGCNs), which are composed of both a residual path and a densely connected path to better encode the long-range structural information. DPGCNs are the first GCN architecture leveraging both residual connections and dense connections at the same time. Extensive experiments on Open Graph Benchmark (OGB) datasets clearly demonstrate superior performances of our proposed DPGCNs over competitive baseline methods on the large-scale graph learning tasks of node property prediction and graph property prediction.

7. REFERENCES

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

[2] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl, "Neural message passing for quantum chemistry," in *ICML*, 2017.

[3] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018.

[4] Marinka Zitnik and Jure Leskovec, "Predicting multicellular function through multi-layer tissue networks," *Bioinformatics*, vol. 33, no. 14, pp. i190–i198, 2017.

[5] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang, "Deepinf: Social influence prediction with deep learning," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2110–2119.

[6] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 974–983.

[7] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon, "Dynamic graph cnn for learning on point clouds," *Acm Transactions On Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[8] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N Metaxas, "Semantic graph convolutional networks for 3d human pose regression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3425–3435.

[9] Matthew K. Matlock, A. Datta, N. L. Dang, Kevin Jiang, and S. Joshua Joshua Swamidass, "Deep learning long-range information in undirected graphs with wave networks," *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2019.

[10] Qimai Li, Zhichao Han, and Xiao-Ming Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," *arXiv preprint arXiv:1801.07606*, 2018.

[11] Uri Alon and Eran Yahav, "On the bottleneck of graph neural networks and its practical implications," *arXiv preprint arXiv:2006.05205*, 2020.

[12] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem, "Deepgcns: Can gcns go as deep as cnns?," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9267–9276.

[13] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem, "Deepergcn: All you need to train deeper gcns," *arXiv preprint arXiv:2006.07739*, 2020.

[14] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li, "Simple and deep graph convolutional networks," *arXiv preprint arXiv:2007.02133*, 2020.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[16] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng, "Dual path networks," in *Advances in neural information processing systems*, 2017, pp. 4467–4475.

[17] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka, "Representation learning on graphs with jumping knowledge networks," in *International Conference on Machine Learning*, 2018, pp. 5453–5462.

[18] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka, "How powerful are graph neural networks?," in *International Conference on Learning Representations*, 2018.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[20] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," *arXiv preprint arXiv:2005.00687*, 2020.

[21] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 257–266.

[22] Matthias Fey and Jan Eric Lenssen, "Fast graph representation learning with pytorch geometric," *arXiv preprint arXiv:1903.02428*, 2019.

[23] Yunsheng Shi, Zhengjie Huang, Shikun Feng, and Yu Sun, "Masked label prediction: Unified massage passing model for semi-supervised classification," *arXiv preprint arXiv:2009.03509*, 2020.

[24] Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M Bronstein, "Improving graph neural network expressivity via subgraph isomorphism counting," *arXiv preprint arXiv:2006.09252*, 2020.

[25] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun, "Graph neural networks: A review of methods and applications," *arXiv preprint arXiv:1812.08434*, 2018.

[26] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[27] Will Hamilton, Zhitao Ying, and Jure Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[28] 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*, 2018, pp. 1416–1424.

[29] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu, "Heterogeneous graph attention network," in *The World Wide Web Conference*, 2019, pp. 2022–2032.

[30] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel, "Gated graph sequence neural networks," *arXiv preprint arXiv:1511.05493*, 2015.

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

[32] Yunhe Li, Yaochen Hu, and Yingxue Zhang, "Graph representation learning via adversarial variational bayes," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 3237–3241.

[33] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-yan Liu, and Liwei Wang, "Graphnorm: A principled approach to accelerating graph neural network training," *arXiv preprint arXiv:2009.03294*, 2020.

[34] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*. ACM, 2015, pp. 1067–1077.

[35] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[36] Aditya Grover and Jure Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 855–864.