Graph Convolution Network Based on Graphlets Structural Information

Guo Zhimeng, Liu Qinbo, Qi Yuhao

December 27, 2019

Abstract

Graph is a kind of data structure that can show node information and node connection at the same time, which is very useful in organizing data. Graph Convolutional Network(GCN) introduces machine learning method into graph research and achieves good effect on the semi-supervised classification of nodes. However, at present, GCN only uses the attribute information of graph without considering the structure information of graphs (Kipf and Welling, 2016). We get inspiration from the research method of graphlets for complex network and aim to use both the structure information of graph and attribute information of node to classify nodes (Yaveroğlu et al., 2014). Our model calculates the index of structural information for each node in the graph and adds it to the attribute information of nodes for the later convolution operations. In a number of experiments, our approach has improved the accuracy about 0.5% of Graph Convolutional Networks.

1 Introduction

We are surrounded by a variety of graphs, such as social networks, brain connectome, or world trade networks. A typical graph consists of many edges between nodes, where nodes represent different individuals in a real system and edges represent relationships between individuals. For example, in WeChat Friends Network(Li et al., 2016), each person is a node, and if two people are friends, there will be a link between the two nodes. Graphs allow us to model complex systems, which enables us to deal with complex interaction systems and to view various physical phenomena from a more macro perspective.

The graph structure is so complicated and irregular that it has always been difficult to solve the graph-based tasks in a efficient way. A fast and accurate way to accomplish graph-based semi-supervised learning is urgently needed to be found. There are different types of nodes(such as document) in a graph(such as a citation network). The labels are only available for a small subset of nodes. Our goal is to classify the unlabeled nodes with the given information, which is called the graph-based semi-supervised learning.(Kipf and Welling, 2016)

Many researchers thought the rising trend of Neural Networks may solve the problem. So there have been increasing attempts in the study to extend Convolutional Neural Networks (CNNs) to deal with graphs. These approaches are usually categorized as spectral approaches and non-spectral approaches. The non-spectral approaches define convolutions directly on the graph, and the challenge is to define an operator that works with neighbors of different sizes and maintain the weight sharing characteristic of CNNs (Duvenaud et al., 2015). In spectral approaches, the convolution operation is defined in the Fourier domain by computing the eigendecomposition of the graph Laplacian, resulting in spatially localized filters (Bruna et al., 2013).

In 2017, Kipf and Welling (2016) introduced a novel approach, which used an efficient layer-wise propagation rule and improved the accrucy of semi-supervised classification on graph-structure by a large margin. Their work has two main contributions. First, they used a simple but effective first-order approximation of spectral graph convolutions to operate directly on graphs. Then they applied the method to solve scalable semi-supervised classification of nodes and refreshed the record of present state-of-the-art methods by a large margin both in classification accuracy and efficiency. Many researchers followed their idea and came up with different improvements, such as the GraphSAGE (Hamilton et al., 2017) method and the Graph Attention Networks (Veličković et al., 2017).

However, the GCN model just uses the information of neighbour nodes and it neglects the structural information, which is potential to be more important than any other information in graphs.

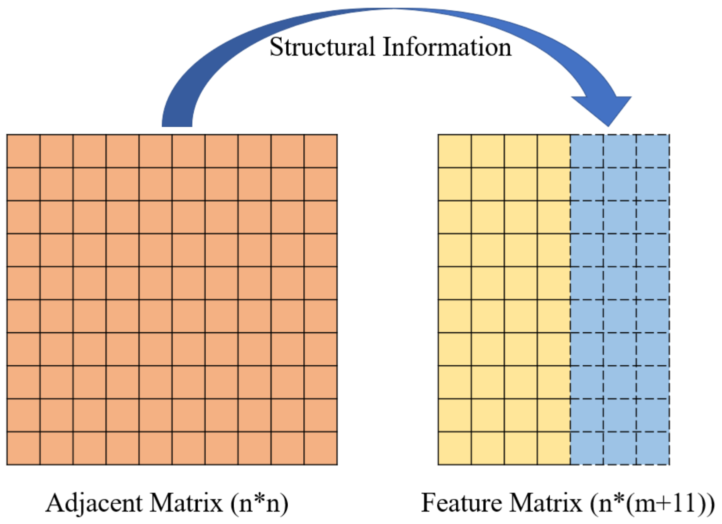

Figure 1: The implement of our method: We add the structural information collected from the adjacent matrix to the feature matrix.

On the basis of former works, We came up with the idea of structural GCN, which combines the feature information and structural information for each node. As Figure 1, the input of GCN is an adjacent matrix and a feature matrix. The structural information is stored in the adjacent matrix. We get inspiration from researches on complex networks (Yaveroğlu et al., 2014) and we count the frequency for each node to touch the orbits in graphlets. The frequency is considered to be internal structural information and we add it to the formal feature matrix so that our model makes full use of the two kinds of information.

We evaluate our method on three node-classification benchmarks and found that our approach is able to improve the accuracy about 0.5% than GCN proposed by Kipf et al.(2017) .

2 Graph Convolution Networks

In this section, we provide theoretical motivation for a specific graph-based neural network model f(X, A) that we will use as a base of our structural graph neural network in the rest of this paper. We consider a multi-layer Graph Convolution Network (GCN) with the following layer-wise propagation rule:

$$H^{l+1} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{1}$$

Here, $\tilde{A} = A + I_N$ is the adjacency matrix of the undirected graph $\mathcal{G}$ with added self-connections. I_N is the identity matrix, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ and $W^{(l)}$ is layer-specific trainable weight matrix. $\sigma(\cdot)$ denotes an activation function, such as the $ReLU(\cdot) = max(0, \cdot)$. $H^{(l)} \in \mathbb{R}^{N\times D}$ is the matrix of activations in the l^{th} layer; $H^{(0)} = X$, X is matrix of feature information. In the following, we show that the form of this propagation rule can be motivated via a first-order approximation of localized spectral filters on graphs.

Formally, a Graph Convolutional Neural Network (GCN) is a neural network that operates on graphs. Given a graph $G = (V, E)$, a GCN takes as input

- an input feature matrix $N \times F^0$ feature matrix, X, where N is the number of nodes and F^0 is the number of input features for each node, and
- an $N \times N$ matrix representation of the graph structure such as the adjacency matrix A of G.

A hidden layer in the GCN can thus be written as $H^i = f(H^{i-1}, A)$ where $H^0 = X$ and f is a propagation. Each layer corresponds to an $N \times F^i$ feature matrix where each row is a feature representation of a node. At each layer, these features are aggregated to form the next layer's features using the propagation rule f. In this way, features become increasingly more abstract at each consecutive layer. In this framework, variants of GCN differ only in the choice of propagation rule f.

We need the aggregated representation of a node including its own features, so we added the identity matrix I_N to the adjacency matrix A before applying the propagation rule.

What's more, nodes with large degrees will have large values in their feature representation while nodes with small degrees will have small values. This can cause vanishing or exploding gradients [1, 2], but is also problematic for stochastic gradient descent algorithms which are typically used to train such networks and are sensitive to the scale (or range of values) of each of the input features. To normalize the feature representation, there have been two successful method. The first is the mean rule which transformed the adjacency matrix A by multiplying it with the inverse degree matrix D. And another method is the spectral rule proposed by Kipf and Welling (2016), which raised the degree matrix of -0.5 and multiply it on each side of A. The spectral rule weighs neighbor in the weighted sum higher if they have a low-degree and lower if they have a high-degree. This may be useful when low-degree neighbors provide more useful information than high-degree neighbors. Due to the advantage of spectral method, we used the spectral method in our method. To sum up, we have the spectral propagation rule:

$$f(X,A) = \sigma(D^{-0.5}\hat{A}D^{-0.5}XW) \tag{2}$$

3 GCN based on Graphlets Structural information

3.1 Graphlets Structural Information

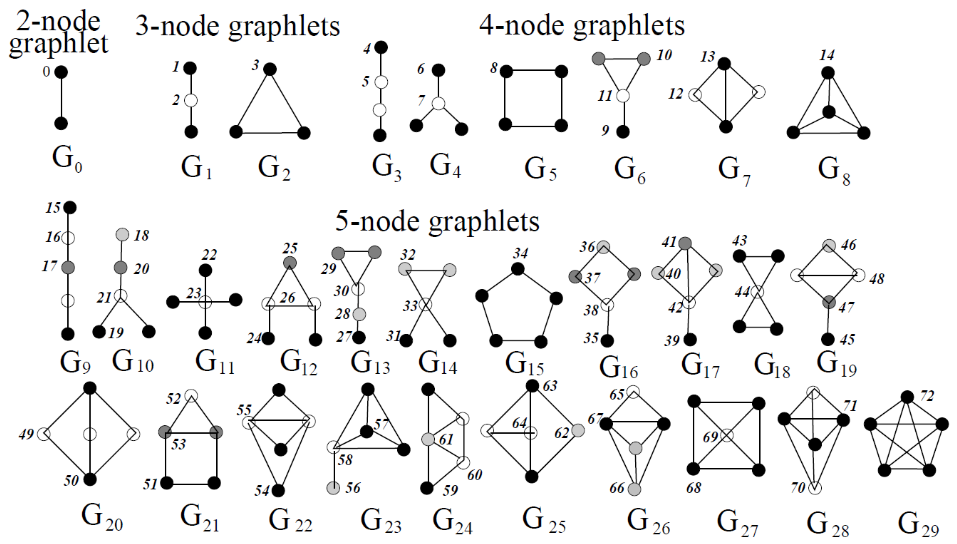

Figure 2: Automorphism orbits 0,1,2,...,72 for the thirty 2, 3, 4, and 5-node graphlets $G_0, G_1, ..., G_{29}$. In a graphlet G_i, $i \in 0, 1, ..., 29$, nodes belonging to the same orbit are of the same shade.(Yaveroğlu et al., 2014)

Automorphism orbits of graphlets illustrated as 0 to 72 have been used to generalize the degree distribution into the spectrum of 73 Graphlet Degree Distributions that correspond to the 73 orbits from 2-node to 5-node graphlets: the first of these distributions is the familiar degree distribution, the second gives the number of nodes in the network that touch k orbits 1 of graphlet G_1 for all values of k, etc. for all 73 orbits.

If we denote by C_i the i^{th} graphlet degree of a node, which is the number of times the node is touched by orbit i, some mathematical correlation among different C_i may appear. For example, $\binom{C_0}{2} = C_2 + C_3$ (Yaveroğlu et al., 2014). So that Yaveroğlu et al. (2014) identified and eliminated redundancies which is caused by the correlations.

All in all, they found the 11 orbits with 2, 3, and 4-node graphlets was the best choice to represent the structural information. Concretely, for each node in the graph, it appears as different orbits of different graphlets. As 2, they count the frequency for each node appears as the orbits 0, 1, 2, 4, 5, 6, 7, 8, 9, 10, 11 and got a 10-dimension vector for each node. And this is the structural information that we want.

3.2 Structural Graph Convolution Network

In this part, we introduce our method to apply graphlets structural information to graph convolution network. Imagine we have got the matrix of structure information $S \in \mathbb{R}^{N \times G}$, which represents the coordinates corresponding to 11 non-redundant orbits with the method introduced in the last part.

Then we concatenate matrix X and matrix S to get the $\tilde{X} = [X, S]$ combining feature information and graphlets structural information. After that we replace $H^0 = X$ in propagation of graph convolution network with $H^0 = \tilde{X}$. Therefore, we get our propagation rule of structural convolution neural network as follows:

$$H^{(l+1)} = \begin{cases} f(H^l, A) & l > 0 \\ f([X, S], A) & l = 0 \end{cases}$$

Here,

$$f(H^l, A) = \sigma(D^{-0.5}\tilde{A}D^{-0.5}H^l W) \tag{3}$$

$$f(\tilde{X}, A) = \sigma(D^{-0.5}\tilde{A}D^{-0.5}\tilde{X}W) \tag{4}$$

As we can see, a Structural Graph Convolutional Neural Network(Structural GCN) is a neural network that operates on graphs and structure information of the network. Given a graph $G = (V, E)$, a Structural GCN takes as input

- an input feature matrix $N \times F^0$ feature matrix, X, where N is the number of nodes and F^0 is the number of input features for each node, and
- an $N \times 11$ matrix, S, containing the graphlets structural information, and
- an $N \times N$ matrix representation of the graph structure such as the adjacency matrix A of G.

It is very similar to GCN excluding that it uses the structural graphlets information. Features become increasingly more abstract at each consecutive layer. The adjacency matrix is added the iddentity matrix I_N to include its own features. And the spectral rule of normalization is used to transform $\hat{A}$ to be $D^{-0.5}\hat{A}D^{-0.5}$ in our propagation.

4 Related Works

A kinds of methods to solve the semi-supervised learning on graphs based on Graph Neural Networks have been came up with. And these approaches are usually categorized as spectral approaches and non-spectral approaches. The non-spectral approaches define convolutions directly on the graph while the spectral approaches do the convolution operation in the Fourier domain by computing the eigendecomposition of the graph Laplacian. (Bruna et al., 2013).

The non-spectral method uses multiple cycles directly, while the spectral method uses the characteristics of matrix to calculate. The non-spectral method requires defining a convolution operation that is always valid for different numbers of neighbors and can realize weight sharing of CNN. There are several approaches trying to perfect the method. Duvenaud first came up with the idea to introduce a convolutional neural network that operates directly on graphs, which generalized standard molecular feature extraction methods and got a good performance(Duvenaud et al., 2015). Niepert avoided the neighborhood problem with normalizing neighborhoods containing a fixed number of nodes(Niepert et al., 2016). To make the CNN structure more general, Monti proposed a unified framework allowing to generalize CNN architecures to non-Euclidean domains and learn more specific features(Monti et al., 2017). Hamilton came up with GraphSAGE(Hamilton et al., 2017), a general framework for inductive task, which used neighborhood to collect local information, which has yielded excellent accuracy in many tasks.

The spectral method is more mathematical. Bruna considered taking advantage of the spectrum of the graph Laplacian to generalize CNNs on more general domains, which were addressed by many subsequent works(Bruna et al., 2013). Henaff extented the spectral networks with a Graph Estimation procedure and made spectral filters spatially localized(Henaff et al., 2015). Defferrard proposed to approximate the filters by means of a Chebyshev expansion of the graph Laplacian, removing the need to compute the eigenvectors of the Laplacian and yielding spatially localized filters(Defferrard et al., 2016). However, the sophisticated mathematical methods resulted in a unbearable burden of calculation for large graphs. Also, the explicit graph-based regularization in the loss function made its propagation for multilayer very difficult.

Convolutional neural networks are derived from regular data structures and often ignore the extraction of relationships. The research on complex networks is very enlightening. Former research did network statistics to mine the intrinsic attributes of the network. The simplest concept in networks is degree, which is the number of edges of a node. Some researcher think a higher degree indicates the node is more important (Opsahl et al., 2010). However, some evidence shows two networks with the same degree can be extremely different, which inspire us to compare networks from single nodes to the interaction between a small number of nodes in a network. Pržulj came up with small subnetworks called graphlets to keep the interaction (Pržulj, 2007). Begin with the work, some algorithm were designed and optimized once and once (Marcus and Shavitt, 2012). With the graphlet-base edge clustering, researcher found pathogen-interacting proteins (Hayes et al., 2013). Now the graphlet has been accepted as a important indicator in network analysis and comparison.(Yaveroğlu et al., 2014)

5 Experiments

We test the performance of our model on an established graph-based benchmark task: semi-supervised document classification in citation networks. This section summarizes our experimental datasets, set-up, and baselines.

5.1 Datasets

We utilize three standard citation network benchmark datasets: Citeseer, Cora and Pubmed (Sen et al., 2008) and closely follow the experimental setup of Yang et al. (2016). The datasets, whose statistics

Table 1: Statistics of the datasets used in our experiments.

Dataset	Type	Nodes	Edges	Classes	Features	Label rate
Citeseer	Citation network	3,327	4,732	6	3,703	0.036
Cora	Citation network	2,708	5,429	7	1,433	0.052
Pubmed	Citation network	19,717	44,338	3	500	0.003

are summarized in Table 1, contain sparse bag-of-words feature vectors for each document and a list of citation links between documents. In these citation network datasets, nodes correspond to documents, edges correspond to citation links, and node features correspond to elements of a bag-of-words representation of a document. Each node has a class label. Label rate denotes the number of labeled nodes that are used for training divided by the total number of nodes in each dataset. We treat the citation links as (undirected) edges and construct a binary, symmetric adjacency matrix A. For training, we only use 20 labels per class, but all feature vectors.

5.2 Experiment set-up

In our experiment, a two-layer Structural GCN is trained with the prediction accuracy on a test set of 1000 labeled examples predicted. We used the dataset of Citeseer, Cora and Pubmed, with additional validation set of 500 labeled examples for hyperparameter optimization (0.5 dropout rate for all layers, L2 regularization factor for the first Structural GCN layer and 16 number of hidden units). It's worth noting that, the validation set labels aren't used for training.

On the dataset of citation network, we only optimize hyperparameters on Cora. Furthermore, the same set of parameters are used for Citeseer and Pubmed. Our models are trained for a maximum of 200 epochs using Adam , setting learning rate to be 0.01 and window size of early stopping to be 10, i.e. the training process will end if the validation loss is not decreasing for consecutive epochs up to 10. Our weights initialization is the same as the method in Kipf and Welling (2016), hence normalizing the input feature vectors by row at the same time.

5.3 Baselines

We compare against the same strong baseline methods and state-of-the-art approaches as specified in Kipf and Welling (2016), where we always choose their bestperforming model variant as a baseline. This includes label propagation (LP) (Zhu et al., 2003), semi-supervised embedding (SemiEmb) (Weston et al., 2012), manifold regularization (ManiReg)(Belkin et al., 2006) skip-gram based graph embeddings (Deep-Walk) (Perozzi et al., 2014), the iterative classification algorithm (ICA) (Lu and Getoor, 2003) and Planetoid (Yang et al., 2016). We also directly compare our model against GCN (Kipf and Welling, 2016).

6 Results

6.1 Classification under different methods

The results of our experiments are summarized in Tables 2. The numbers shown in the table are the classification accuracy expressed by percentage. In addition, we reuse the results of other baseline methods already shown in Kipf and Welling (2016) for comparison. For our model, specifically, we set up 11 orbits in graphlets, the normalization of the matrix is linear, and the obtained structural feature matrix is discretized.

Table 2: Summary of results in terms of classification accuracy.

Method	Citeseer	Cora	Pubmed
LP	45.3	68.0	63.0
SemiEmb	59.6	59.0	71.1
ManiReg	60.1	59.5	70.7
DeepWalk	43.2	67.2	65.3
ICA	69.1	75.1	73.9
Planetoid	64.7	75.7	77.2
GCN	70.3(3.1s)	81.5(4.8s)	79.0(16.5s)
Structural GCN (ours)	**71.6(1.6s)**	**82.1(1.3s)**	**79.8(13.3s)**

We further report wall-clock training time in seconds until convergence (in brackets) for our model and for GCN (Kipf and Welling, 2016). For both models, we ran the experiment in the same hardware environment. We used the following sets of hyperparameters for the datasets: 0.5 (dropout rate), $5{\cdot}10^{-4}$ (L2 regularization) and 16 (number of hidden units).

Table 3: Summary of results in terms of classification accuracy.

Structural GCN in different settings	Citeseer	Cora	Pubmed
11 orbits, linear normalization, discretization	71.6	**82.1**	**79.8**
58 orbits, linear normalization, discretization	71.6	**82.1**	79.5
11 orbits, nonlinear normalization, discretization	**72.1**	82.0	77.2
58 orbits, nonlinear normalization, discretization	71.3	**82.1**	48.0
11 orbits, nonlinear normalization, serialization	71.6	**82.1**	79.4
58 orbits, nonlinear normalization, serialization	71.5	81.4	79.1
11 orbits, linear normalization, serialization	71.6	**82.1**	79.4
58 orbits, linear normalization, serialization	71.6	81.6	79.5

6.2 Classification of the Structural GCN in different settings

When structural information is propagated from neighboring nodes in every layer, considering that the orbits in graphlets can be set to different numbers(Yaveroğlu et al., 2014), the normalization of structural feature matrix can be done in both linear and nonlinear ways, discretization and serialization ways, we further compared the effects of classification under different settings with our Structural GCN for experiments. The experimental results are shown in table 3.

6.3 Training time per epoch

At the last part, we show our runtime on GPU with Tensorflow. We compare our method with GCN in Figure 3 [1]

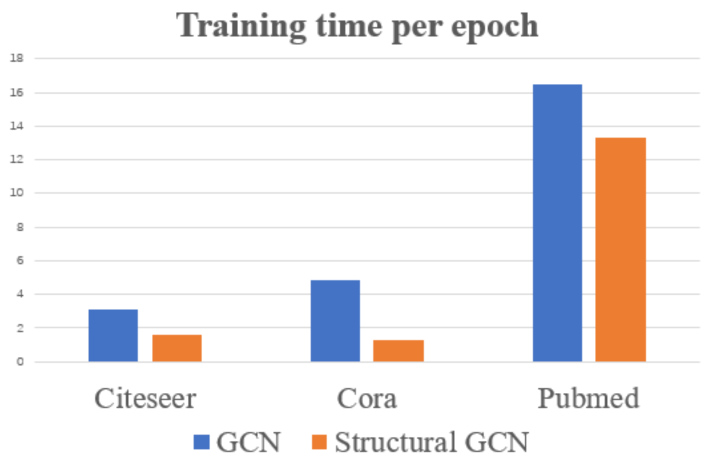

Figure 3: Runtime for GCN and Structural GCN

7 Discussion

7.1 Structural GCN

Through the above experiments and results, we have shown that by utilizing the correlations between the characteristics of nodes, we can sensitively and stably discover network types and track network dynamics. In terms of the accuracy and running time of the classification task, our method that utilize both the structure information of graphs and attribute information of nodes works better than most recent work that only use attribute information.

7.2 Limitations and Future Work

There are several limitations of our work and we need to overcome them in the future.

[1]Hardware used: 16-core Intel R Xeon R CPU E5-2640 v3 @ 2.60GHz, GeForce R GTX TITAN Xp

Hyperparameters: There are several hyperparameters in our method which can be seen at table 3. The method is not stable. Some change of them can result in a fluctuation of accuracy and we haven't found the reason. Nor have we found the best set of hyperparameters and we need to finish it in the future.

Convolution: We know that the label of nodes in a graph is set by human, which has a very high randomness. And the hypothesis that neighbouring nodes are more probably to be same is unreliable in some circumstance. Our method haven't gotten rid of the assumption. In the future maybe we need to change the convolution structure to make our method more accurate.

The success of our method inspire us that the structural information is very useful in improving the efficient of Graph Neural Networks.Also, we have set up a bridge between the two research areas, which is potentially to be a new growth direction.

8 Conclusion

Combining the study of complex network and Graph Convolutional Networks, we introduce the method of using the structure information of graphs in the traditional semi-supervised classification. We use the Graphlets structural information and add it to the feature matrix. Also, we have tried different settings of hyperparameters and achieved a good accuracy, which suggests that the structural information is a useful way to improve the expression of GCN. And we think this is a direction worth studying in depth.

References

Belkin, Mikhail, Partha Niyogi, and Vikas Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *Journal of machine learning research*, 2006, *7* (Nov), 2399–2434.

Bruna, Joan, Wojciech Zaremba, Arthur Szlam, and Yann LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

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

Duvenaud, David K, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams, "Convolutional networks on graphs for learning molecular fingerprints," in "Advances in neural information processing systems" 2015, pp. 2224–2232.

Hamilton, Will, Zhitao Ying, and Jure Leskovec, "Inductive representation learning on large graphs," in "Advances in Neural Information Processing Systems" 2017, pp. 1024–1034.

Hayes, Wayne, Kai Sun, and Nataša Pržulj, "Graphlet-based measures are suitable for biological network comparison," *Bioinformatics*, 2013, *29* (4), 483–491.

Henaff, Mikael, Joan Bruna, and Yann LeCun, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*, 2015.

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

Li, Zhuqi, Lin Chen, Yichong Bai, Kaigui Bian, and Pan Zhou, "On diffusion-restricted social network: A measurement study of wechat moments," in "2016 IEEE International Conference on Communications (ICC)" IEEE 2016, pp. 1–6.

Lu, Qing and Lise Getoor, "Link-based classification," in "Proceedings of the 20th International Conference on Machine Learning (ICML-03)" 2003, pp. 496–503.

Marcus, Dror and Yuval Shavitt, "Rage–a rapid graphlet enumerator for large networks," *Computer Networks*, 2012, *56* (2), 810–819.

Monti, Federico, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition" 2017, pp. 5115–5124.

Niepert, Mathias, Mohamed Ahmed, and Konstantin Kutzkov, "Learning convolutional neural networks for graphs," in "International conference on machine learning" 2016, pp. 2014–2023.

Opsahl, Tore, Filip Agneessens, and John Skvoretz, "Node centrality in weighted networks: Generalizing degree and shortest paths," *Social networks*, 2010, *32* (3), 245–251.

Perozzi, Bryan, 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.

Pržulj, Nataša, "Biological network comparison using graphlet degree distribution," *Bioinformatics*, 2007, *23* (2), e177–e183.

Sen, Prithviraj, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad, "Collective classification in network data," *AI magazine*, 2008, *29* (3), 93–93.

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

Weston, Jason, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert, "Deep learning via semi-supervised embedding," in "Neural Networks: Tricks of the Trade," Springer, 2012, pp. 639–655.

Yang, Zhilin, William W Cohen, and Ruslan Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," *arXiv preprint arXiv:1603.08861*, 2016.

Yaveroğlu, Ömer Nebil, Noël Malod-Dognin, Darren Davis, Zoran Levnajic, Vuk Janjic, Rasa Karapandza, Aleksandar Stojmirovic, and Nataša Pržulj, "Revealing the hidden language of complex networks," *Scientific reports*, 2014, *4*, 4547.

Zhu, Xiaojin, John Lafferty, and Zoubin Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions," in "ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining," Vol. 3 2003.