Methods and Optimizations for Node Classification

Yifan Ren
ivan-ren@sjtu.edu.cn

Abstract

Recent years, some graph neural network has shown their outstanding abilities in unsupervised learning and semi-supervised learning like node classification, link prediction, graph classification and so on. In this paper, based on node prediction task, I will implement some of the famous graph neural networks and analyze their advantages and disadvantages. Later, I will aggregate those networks and construct a new model to better solve the task.

1 Introduction and Background

Graph is a structure made of vertices and edges, graph node classification task is to predict unknown node labels by node features and the known labels of some specific nodes. We can learn the social networks, knowledge mapping, citation relationships and protein-protein interaction.

We know the Convolutional Neural Networks and Recurrent Neural Networks can be applied to solving the almost classification task and get a well result. But when it comes to graph structure, we must change our model for better node embedding and forward function to describe the node characteristic accurately.

So I choose three famous models and one optimization. First, the Graph Convolutional Networks [2] which was invented to solve semi-supervised learning, so it perfectly matches this task. And we also have Graph Attention Networks [3] and GraphSAGE(Graph SAmple and aggreGatE) [1] . Finally we've got the Jump Knowledge Networks [4] which try to use the information in residual network. I will compare their differences and use different poolings to construct my model according to these models' results.

1.1 Related work

Graph Convolutional Networks Latest five years, several graph convolutional neural network models for learning over graphs have been proposed. The original GCN algorithm is designed for semi-supervised learning in a transductive setting, and the exact algorithm requires that the full graph structure is known during training.

Graph Attention Networks Instead of using the Laplacian matrix like GCN, GAT uses attention coefficient, which can better integrate the correlation between node features into the model. GAT also use multi-head attention to concatenate those activation function.

GraphSAGE GraphSAGE also can implement on inductive graph. Instead of train a concrete node embedding like GCN, GraphSAGE train a representation method for all node which can work on any graph. For every node, GraphSAGE aggregate the message from all his neighbors.

Jump Knowledge Networks Experiments show that it is not feasible to increase the layer number on the basic graph neural network to optimize the effect. This is mainly because more layers can also spread noise information from the multiplied extended neighborhood members. To increase depth, a natural idea is to use residual connection. With JKN, all layers can jump to the last layer and aggregate, so that nodes can adaptively choose the size of receptive domain.

Preprint. Under review.

1.2 Motivation

There are several motivations for me to complete this project. First and foremost. I am very interested in the working principle and construction of different kinds of neural networks. Moreover. Beautiful graph structure, whether it is a specific graph or an abstract graph of some kind, has deeply attracted me. That why for me to study machine learning and graph theory and combination at the same time this semester. In addition, I am also deeply aware of the importance of graph node classification in practical application. The social networks, knowledge mapping, citation relationships and protein-protein interaction, These applications cover business, academia, and the frontier of science. So I hope we can make a breakthrough on the semi-supervised learning on graph as soon as possible to promote the progress of other fields.

2 Semi-Supervised Node Classification

In my really work, I will analyze the advantages and disadvantages of different models. And later I will exhibit my model of this work.

2.1 Graph Convolutional Networks

The multi-layer Graph Convolutional Network (GCN) usually has the following layer-wise propagation rule:

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

This propagation rule simply uses adjacency matrix after regularization and train itself with the cross-entropy loss function. And surely it summarizes the characteristics of the graph effectively.

But because it uses the Laplacian matrix, so it can not imply on inductive graph and directed graph, has a poor scalability. And the normal layers will spread noise so we cannot simply increase the layers number.

2.2 Graph Attention Networks

We conclude the multi-layer Graph Attention Network (GCN) usually has the following layer-wise propagation rule:

$$\vec{h'_i} = \sigma(\frac{1}{K}\Sigma_{k=1}^{K}\Sigma_{j\in N(i)}\alpha_{ij}^{k}W^{k}\vec{h_j})$$

The innovation of GAT is to add attention mechanism to give importance to the edges between nodes and help the model learn structural information. The relative disadvantage is the lack of training methods, sampling nodes also have room to improve.

2.3 GraphSAGE

GraphSAGE has different type of aggregator. The mean I use has the following layer-wise propagation rule:

$$h_v^k = \sigma(W \cdot MEAN(\{h_v^{k-1}\} \cup \{h_u^{k-1}, \forall u \in N(v)\}))$$

GraphSAGE aims to learn an aggregator instead of a representation for each node, which can improve the flexibility and generalization of the model. In addition, because of the flexibility, it can train in batches to improve the convergence speed. However, the problem is that the number of sampling nodes increases exponentially with the number of layers, resulting in the poor performance of the model on time per batch.

2.4 My Model

I designed a unique convolution structure and super parameter values for each model. And try different aggregator in every convolution layer. Adjust the dropout rate in order to get a best grade from trade off. After all models has been trained. I will implement a average pooling or max pooling on the four prediction and use a softmax function to get the final answer for each test node.

3 Experiments

3.1 Dataset

I used three citation network datasets: **Cora**, **Citeseer** and **Pubmed** [1]. The datasets contain sparse bag-of-words feature vectors for each document and a list of citation links between documents. Each document has a class label.

3.2 Experimental Set-Up

Finally, I use a 3-layer GCN, a 3-layer GAT, a 3-layer GraphSAGE, and a 6-layer GCN with JKN optimization. I set 128 to the hidden features of all models. Both GCN model have the both aggregator, GraphSAGE model uses the mean aggregator. 0.5 dropout rate for models without JKN, and 0.4 dropout rate for GCN with JKN. Each basic model train for 256 times.

3.3 Baseline

I use these four basic models as four baselines. Just use their original prediction for the test mask. And compare those with my poolings.

3.4 Result

Table 1: Summary of results in terms of classification accuracy (in percent).

Method	**Cora**	**Citeseer**	**Pubmed**
GCN	**77.3**	**63.1**	**75.4**
GAT	76.0	61.9	73.7
GraphSAGE	75.0	61.6	73.0
GCN with JKN	75.0	57.8	74.8
Average Pool	**80.3**	**65.4**	**76.3**
Max Pool	77.8	61.6	74.9

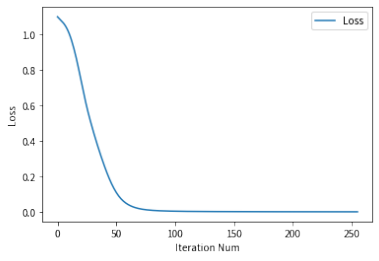

Figure 1: GCN

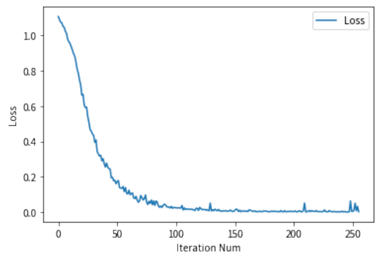

Figure 2: GAT

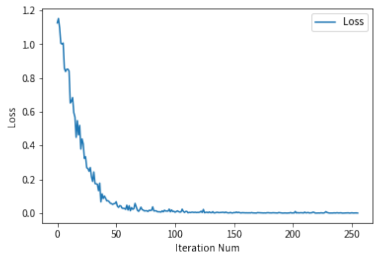

Figure 3: GraphSAGE

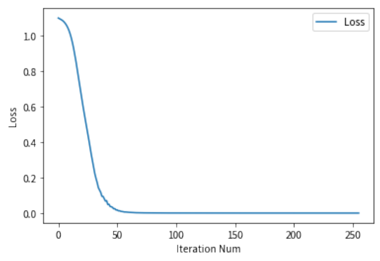

Figure 4: GCN_JKN

[1] https://linqs.soe.ucsc.edu/data

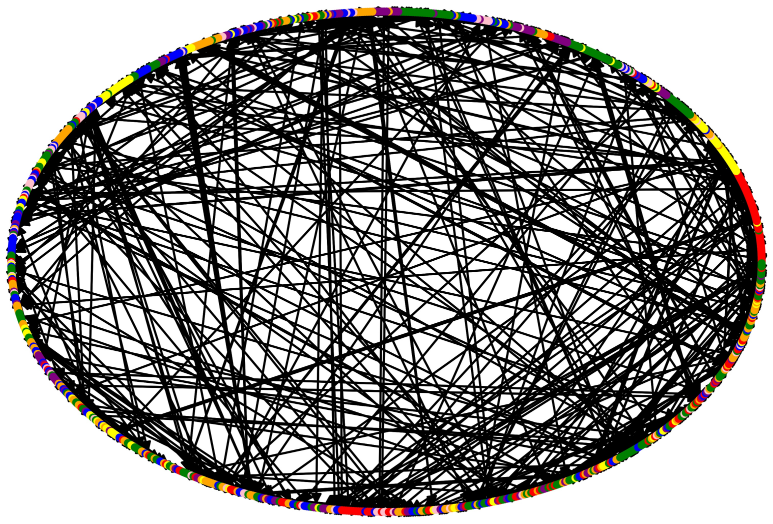

Figure 5: Cora Test Mask

Results are summarized in Table 1. We can find that the average pool wins max pool on all datasets. And GCN ranked first among the four basic models, which surprised me at that time. And the loss history of the four basic models are shown in Figure 1-4. We can clearly see that GCN loss history curve is much smoother than the other three basic models. This also confirms our discussion about the poor flexibility and scalability of GCN. Figure 5 shows the prediction of Cora test mask. Although the composition is unsatisfactory, we can still see the aggregation behavior of some of the same color point(same label node).

4 Conclusion

We find that the average pool performs better than max pool and all basic models, so we successfully verify the effectiveness of average pool. Though, the max pool is inferior to GCN in some cases, so max pool is not always effective.

And I think the transductive graph is the reason why GCN ranked first among the four basic models. Surely in this situation GCN can get some good score, but because of its poor flexibility and scalability, it will be replaced by GAT and GraphSAGE in other task. But there is another problem that JKN optimization seems to have no effect. I think the reason is the hardware limitations.

References

[1] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. *arXiv e-prints*, page arXiv:1706.02216, June 2017.

[2] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv e-prints*, page arXiv:1609.02907, September 2016.

[3] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *arXiv e-prints*, page arXiv:1710.10903, October 2017.

[4] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation Learning on Graphs with Jumping Knowledge Networks. *arXiv e-prints*, page arXiv:1806.03536, June 2018.