A Survey on Using GNN for Graph Node Classification

Moustafa Ismail
e246581@metu.edu.tr
Middle East Technical University
Ankara, Turkey

Abstract

Node Classification basically means, given a graph, predict a class for each node of that graph. This problem arises in many fields, such as classifying citation graphs' nodes or social network nodes. It plays a crucial role in applications like recommendation systems, fraud detection, and biological network analysis, where understanding the roles or categories of nodes helps in making informed decisions. The ability to accurately classify nodes can significantly enhance the performance of downstream tasks in graph-based systems. Graph Node Classification is a challenging problem, since it requires considering both node features and connectivity information when predicting a specific node's class. In order to conduct classification, node embedding should be generated. Existing node embedding techniques lack the ability to efficiently encode the connectivity information. Recently, Graph Neural Networks has demonstrated themselves as a powerful technique at generating node embeddings that capture both node features and connectivity information. In this survey paper, we explore how GNN based approach perform at node classification problem with focus on semi supervised node classification problem. We review the architecture given by [1] and how it addresses the short comings of other techniques in the task of graph node classification. The architecture is implemented using PyTorch on cora graph dataset. The generated model showed 80.2 percent accuracy with a limited training time.

ACM Reference Format:
Moustafa Ismail. 2024. A Survey on Using GNN for Graph Node Classification. In *Proceedings of (Conference acronym 'XX).* ACM, New York, NY, USA, 3 pages. https://doi.org/XXXXXXX.XXXXXXX

Conference acronym 'XX, June 14–14, 2024, Ankara, Turkey

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Graphs are crucial data structures. Many modern problems are best represented in terms of graphs. For example, social networks, molecular chemistry, recommendation systems, and citation networks are best expressed as graphs. The development of efficient machine learning methods that operate on graphs is needed. These methods are required to make predictions at multiple levels about the graph. These levels include node level, edge level, or graph level. The difficulty with operating on graphs is finding a suitable way to encode all the data in the graph to a neural network or other machine learning methods. Specifically how to encode the connectivity information of the graph.

Graph neural networks (GNNs) solve the problems that graphs face with other machine learning methods. They can utilize all the data stored in the nodes as well as the connectivity of the graph to make predictions at various levels about the graph. They are inspired by CNN. GNNS utilizes message passing between the adjacent nodes to encode the connectivity data about the graph. To some extect they are a more genrealized version of CNNs. GNNs have shown better performance in many real-world applications which motives studying.

The focus will be on the use of GCNs for semi supervised node-level classification take. Node level classification is one of the most fundamental problems in graph learning, where the goal is to predict the label of each node using both node features and the structure of the graph. Graph Convolutional Networks (GCNs) are chosen for implementation because of their simple and effective model that extends the idea of convolutions to graphs. They are suitable for semi supervised node classification task tasks, which are common in real-world scenarios where only a small portion of the graph is labeled. The GCN model introduced by Kipf and Welling is widely adopted will be surveyed in this paper.

Cora dataset is chosen to test the usage of GCNs for classification. Cora is a citation network where each node represents a research paper, and an edge between two nodes means one paper cites the other. Each one of the papers is represented by a vector that represents is features. Each paper also belongs to one of several classes. The task is to predict the class of the paper using the citation structure and the feature vectors. This dataset was used by many other researches to evaluate

semi-supervised node classification methods like GCN, because it reflects real-world graph structure and label sparsity.

The rest of the paper is organized as follows. Section 2 provides background on graph data and GNN fundamentals, including how message passing works. Section 3 surveys major GNN models, with emphasis on the GCN model and its variants. Section 4 explains the experimental setup, describes the implementation of GCN on the Cora dataset, and presents results. Section 5 discusses challenges faced and limitations observed during the study. Section 6 concludes the paper and highlights future directions for research in GNNs and node classification.

2 Background and Problem Definition

Graphs are a cricuial data structure that is used to represent wide array of problems. Formally, a graph G = (V, E) consists of a set of nodes V and a set of edges E. Each node can have features associated with it. Edges in the graph can be directed or undirected. Traditional machine leanring methods nad techniques such as CNN are desiged to operate on vector or grid structured data. They have limited ability to operate on graph structured data beacause they can not make use of the information stored in the structure of the graph.

Graph Neural Networks (GNNs) are designed to operate on graph utilizing information stored in nodes and edges. It also make use of the strucure of the graph during operation. The main idea that allows GNNs to make use of all these kinds of intormation is message passing framework where each node collect information from its neighbors (edges and nodes) to update its representation. Performaing several rounds of message passing allow each node to have a representation that encodes information from its local subgraph. GNNs come in various architectures such as Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), and GraphSAGE.

This paper focues on semi-supervised node-level classification on graph data. Given a graph with feature vectors for each node and labels for only a small subset of nodes, the goal is to predict the labels of the remaining nodes. This problem is important in many practical applications where labeling is expensive or time-consuming, but graph structure and node features are available. These applications include predicting user interests in social networks, identifying fraudulent accounts in transaction graphs, or classifying scientific papers in citation networks.

This paper focuses on the class of GNNs used for semi supervies node classification, particularly Graph Convolutional Networks (GCNs). GCNs use graph convolution operation that is efficient for semi-supervised learning. They work by stacking layers of graph convolution. Each layer updates node embeddings by mixing information from the node and its neighbors. The Kipf and Welling model [1] is widely used due to its simplicity and strong performance.

3 Preliminary Literature Review

Graph Convolutional Networks (GCNs) were introduced by Kipf and Welling in their 2016 paper "Semi-Supervised Classification with Graph Convolutional Networks" [1]. They introduced the first model of Graph Convolutional Networks (GCNs) that was the basis for other models. They showed how convolutional neural network operations can be extended to graph-structured data using a an approximation. In , the authors proposed a layer-wise propagation rule that combines information from a node's immediate neighbors, weighted by the graph structure, to update node representations. The proposed approach enables learning node embeddings that incorporate both feature information and graph connectivity, which is crucial for tasks like semi-supervised node classification. Their proposed model assumes that labels are only available for a subset of nodes and learns to predict the labels of the rest by infromation gained from the labeld part of the graph. The method involves stacking a small number of GCN layers, followed by a softmax classifier to produce predictions. The approach is simple, efficient, and performs competitively on benchmark datasets like Cora.

Kipf and Welling's GCN model is straightforward and has become a widely used baseline in the development of other GNN achitecuters. One of its strengths is its efficiency and ease of implementation. The model faces some problems such as its dependence of fixed graph structure which limits is prediction ability.

This project directly implements the GCN model described in Kipf and Welling (2016) using the Cora citation network dataset. This model is particularly suitable because it has a well-defined architecture, is reproducible, and demonstrates the effectiveness of message-passing-based learning for node classification.

4 Methodology and Implementation

The method implemented in this project is the Graph Convolutional Network (GCN) introduced by Kipf and Welling in []. The model uses a layer-wise propagation rule based on a first-order approximation of spectral graph convolutions. Specifically, each GCN layer applies the transformation:

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

where:

- $\tilde{A}$ is the adjacency matrix with added self-loops,
- $\tilde{D}$ is the degree matrix of $\tilde{A}$,
- $H^{(l)}$ is the input to layer l,

- $W^{(l)}$ is the trainable weight matrix,
- σ is an activation function (e.g., ReLU).

This operation allows each node to update its embedding based on its neighboring nodes. The final layer of the GCN is a softmax classifier that produces probability distributions over classes for each node.

The dataset used is Cora, a citation network where each node represents a scientific publication and edges denote citation relationships. Each node is described by a sparse bag-of-words feature vector, and the task is to predict the category of each paper. Cora contains 2,708 nodes, 5,429 edges, 1,433 features per node, and 7 classes. Only a small subset of the nodes are labeled for training, with the rest split into validation and test sets.

The implementation is carried out in PyTorch using the PyTorch Geometric library, which simplifies graph operations and model design. The model consists of two Graph Convolutional Network (GCN) layers: the first layer transforms input features into a hidden representation using ReLU activation, and the second layer maps these representations to class probabilities.

- **Framework:** PyTorch + PyTorch Geometric
- **Model architecture:** 2-layer GCN
- **Hidden dimension:** 16
- **Activation:** ReLU
- **Dropout:** 0.5
- **Loss function:** Cross-entropy (computed only over labeled nodes)
- **Optimizer:** Adam with a learning rate of 0.01
- **Evaluation metric:** Accuracy on the test set

5 Preliminary Results and Discussion

The GCN was trained on cora dataset for semi-supervised node classification. The training loss and validation loss is shown in figure 1. The accuracy results is shown in figure 2

The results of the training showed that a very simple GCN with minimal training was able to reach 80 percent accuracy which demonstrate the efficiency of GCNs.

6 Challenges and Open Problems

The model architecture provided by [1] has some limitations such as not being able to deal with the changing graph structure. For it to be possible, it is necessary that the graphs have fixed structures.

There are several challenges that need to be addressed:

- How to aggregate information from neighboring nodes.
- How to choose the activation function.
- How to determine the number of layers.

Figure 1. training loss vs verification loss

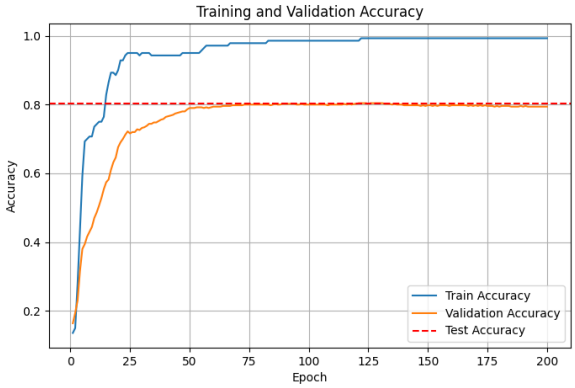

Figure 2. training accuracy vs verification accuracy

7 Conclusion

In this paper, the utilization of graph convolutional networks (GCNS) for semi-supervised node classification was investigated. GCNs addressed the limitations of machine learning models when operating on graphs. GCNS can utilize the features of the nodes as well as the structure of the graph to make predictions.

The implementation of a simple two-layer GCNs on the Core dataset showed the efficiency of using GCNs for semi-supervised node level classification. This opens the door for more research about various CGNS architectures to improve its effiecicy and accuracy.

[1]

References

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