Graph Neural Network and its applications

Jiawei Gao\textsuperscript{1, a, *}, \textsuperscript{†}, Lexin Hao\textsuperscript{2, b, *, †}

\textsuperscript{1} College of Science, Purdue University, West Lafayette, IN, 47906, US;
\textsuperscript{2} School of Engineering, IT and Physical Sciences, Hebei University of science and technology, Shijiazhuang, Hebei 050000, China;
\textsuperscript{*}Corresponding author e-mail: \textsuperscript{a}gao504@purdue.edu, \textsuperscript{b}30393024@students.federation.edu.au
\textsuperscript{†}These authors contributed equally.

Abstract. In recent years, people have become more interested in expanding deep learning methods on graphs, and a lot of progress has been made in the field. Although traditional deep learning methods have been applied to extract the features of Euclidean spatial data with great success, the data in many practical application scenarios are generated from non-Euclidean spaces. The performance of traditional deep learning methods in processing non-Euclidean spatial data is still not optimistic. Driven by the success of many factors, the researchers used the ideas of convolutional networks, recurrent networks, and deep autoencoders to define and design the neural network structure for processing graph data. This is a new research hotspot—Graph Neural Networks (GNNs). It is necessary to summarize the latest GNNs-related studies and propose improved algorithms to further promote breakthroughs in more applications of GNNs. This paper mainly provides an overview of the current research status of graph neural networks and proposes future work from three aspects.

1. Introduction

In the past few years, the rise and application of neural networks have successfully promoted pattern recognition and data mining research. People have gradually made great breakthroughs in processing speech, images, and natural language as deep learning is skilled in processing structured data. However, not everything in the real world can be represented by structured data such as a sequence or a grid. Many things are unstructured, like social networks, knowledge graphs, complex file systems, etc. The standard neural networks like CNNs and RNNs cannot handle the unstructured graph input properly in that they stack the feature of nodes by specific order. The existence of Graphical Neural Networks (GNNs) provides a method to extract irregular data resources.

Graph neural network is proposed to collect and summarize the information from the graph structure [1]. GNNs primarily provide Graph Embedding, a technology used to perform graphical representation training, introduces traditional graph analysis, and extends deep learning processing capabilities for non-Euclidean data.

GNNs gradually become a widely concerned machine learning method. It covers more and more application scenarios such as social media, recommendation systems, financial risk control, physical systems, molecular chemistry, biological sciences, traffic prediction, and other fields [1]. For instance, significant advances have been made in discovering drug repositioning candidates by utilizing graph neural networks for multi-viewers [2].
This paper provides a thorough review of different graph neural network models and their applications. Although there are many related reviews, the development of the GNNs network is rapid, new algorithms and applications are emerging one after another. It is necessary to collectively aggregate the latest GNNs related research and put forward which algorithm to improve, thus further promote GNNs to make breakthroughs in more applications.

To summarize, the main contributions of this work can be summarized as follows: GNNs methods, comprehensive review, and future work.

This paper aims to introduce the extends of GNNs. The remainder of this paper is organized as follows: In section 2 of this review, the sufficient details of fundamentals of GNNs with basic functions and model techniques used are presented. Section 3 introduces the representative algorithm. The performances of different GNNs methods are examined in section 4. Finally, section 5 discusses the current challenges and suggests future directions of GNNs.

2. Graph Neural Network (GNN)

To understand the principle and algorithm of GNN, it is helpful to begin with one of its applications—note classification. Define a graph $G$ as a pair $(V, N)$, where $V$ and $N$ is the set of vertices and edges that $G$ contains, respectively [1]. In a graph, each note is associated with nature and a label naturally. The goal of GNN is to learn a state embedding that contains the labeled notes and the information of the neighborhood of each node to predict the distribution of the unlabeled nodes. The state embedding $h_v \in \mathbb{R}^d$ is defined by

$$h_v = f(X_v, X_{\partial[v]}, h_{n[e[v]], X_{n[e[v]]}})$$

(1)

where $X_v, X_{\partial[v]}, h_{n[e[v]], X_{n[e[v]]}}$ denote the features of $v$, the features of the edges connecting with $v$, the embedding state of the neighboring nodes of $v$, and the features of the neighboring nodes of $v$, respectively [3]. The $f$ function here is the local transition function. GNN utilizes Banach's Fixed Point Theorem and assumes that the transition function is a contraction map, thus ensuring that the node's state vector $X$ finally converges to a contraction Point. So, it is also known as the convergence-based method. A compact equation is

$$H = F(H, X),$$

(2)

let $H$ and $X$ denote the matrix containing all the states and all the features, respectively, and $F$ be the global transition function. After applying Banach's Fixed Point Theorem, the equation can be rewritten as

$$H^{t+1} = F(H^t, X).$$

(3)

Let $g$ be the local output function, then the output $o_v$ is defined as

$$o_v = g(h_v, X_v),$$

(4)

where $h_v$ is the state embedding and $X_v$ are the features of $v$.

3. Review on Graph Neural Networks

The current network representation learning work is aimed at characteristic tasks, and models need to be trained on different data sets. It is impossible to realize the transferring work on different data sets. Qiu et al. proposed the GCC algorithm, which uses contrastive graph learning to perform pre-training of graph neural networks [4]. Li believes that due to complex data representation and nonlinear transformation, it is challenging to explain the decisions made by GNNs, such as the classification basis in node classification tasks. The article extends GNN and proposes a method of interpreting graph features to identify the information components and important node features in the graph. In addition, methods for identifying important factors for node classification are also proposed. Specifically, the interpretation method defined in the article can reversely explain the importance of a certain part based on the experimental results [5]. Kim et al. proposed a self-supervised attention network SuperGAT algorithm, which can effectively deal with the noise that appears in the network. The SuperGAT aims to study the interpretability of attention and use a large number of experiments to reversely verify the theoretical results. Two forms of attention are applied to self-supervised tasks of predicting edges. The presence or absence of edges contains inherent information about the
importance of the relationship between nodes. In the experimental part, four attention design schemes are constructed based on basic GAT and SuperGAT, and the difference between them through experiments is found. In addition, it also discovered two characteristics of the network, namely homogeneity and average degree, and combined experiments to give guidance on what kind of attention design should be used when these two network characteristics are known [6]. From acoustics, images to natural language processing, deep learning has been successful in many fields. However, it is not easy to apply deep learning to ubiquitous graph data because the graph is a non-European data structure with unique characteristics. Zhang et al. comprehensively reviewed various methods applied to graph deep learning, summarized the existing graph network types, and divided the existing methods into five categories: graph recurrent neural networks, graph convolutional networks, graph autoencoders, graph enhancement learning, and graph confrontation methods [7]. Pareja et al. use GCN to learn node representation and use GRU or LSTM to learn the parameters in GCN, that is, the weight matrix. In this way, the weight matrix can continue to evolve over time, and the final parameters that need to be learned are only the parameters used to evolve the weight matrix in GRU or LSTM. For the weight matrix evolution, the article proposes two options. One is based on GRU. The weight matrix and node embedding at the previous moment are put into the GRU to calculate the weight matrix at this moment. The other is based on LSTM, directly to the previous moment. The weight matrix is retained and forgotten, and the weight matrix of the moment is obtained [8]. Zhou et al. divided the node classification task into multiple subtasks and uses the characteristics of continuous learning to improve performance for specific tasks. An experience buffer pool is set up to store the experience nodes obtained from the previous subtasks. For a single subtask, the training set corresponding to the subtask and the experience nodes extracted from the buffer pool are trained simultaneously so that the experience of the past tasks can be absorbed. To avoid the catastrophic forgetting problem that often occurs in continuous learning, the article proposes three strategies for selecting experience nodes after the subtask. Finally, it proposes the ER-GNN algorithm [9]. Huang et al. divided the node classification task into multiple subtasks and uses the characteristics of continuous learning to improve performance for specific tasks. An experience buffer pool is set up to store the experience nodes obtained from the previous subtasks. For a single subtask, the training set corresponding to the subtask and the experience nodes extracted from the buffer pool are trained simultaneously so that the experience of the past tasks can be absorbed. To avoid the catastrophic forgetting problem that often occurs in continuous learning, the article proposes three strategies for selecting experience nodes after the subtask. Finally, it proposes the ER-GNN algorithm [9]. Huang et al. believed that the current GNN algorithm is often too complex, and it is difficult to identify which sectors are working. Therefore, a relatively simple structure is proposed, combining the shallow model ignoring the graph structure with two simple post-processing steps using the correlation in the label structure, called C&S, proving its effectiveness through experiments, and deriving the true. What is effective is the learning of node labels. These two steps are (1) Error correlation, which is used to disperse the residual errors in the training data to correct the errors in the test data; (2) Prediction correlation, which predicts the test data smoother [10].

Baek et al. believed that the current pooling function has more or fewer drawbacks and proposes the GMT algorithm, which treats the problem of full graph representation as a multi-set encoding problem. Specifically, the algorithm uses an improved multi-head attention mechanism to complete the pooling operation and uses the self-attention mechanism to capture the relationship between nodes [10]. Wu et al. reviewed the related works of GNNs in the field of text mining and machine learning and divide GNN into four categories: recurrent graph neural network, convolution graph neural network, graph auto-encoding, and Spatio-temporal graph neural network. In addition, they discuss the application of graph neural networks across various fields, summarizes open-source code, data set, and graph neural network evaluation indicators. Finally, possible research directions are given [12]. The GNNs are used to capture the implicit relationship between semantic objects or important image regions. Himanshu et al. use the visual context-aware attention model to select significant visual information for answer prediction. The experimental results of the VQA 1.0 and VQA 2.0 data sets show that the performance of the model is much better than the SOTA model [13]. Predicting future pedestrian trajectories is a basic research topic in many practical applications, such as video surveillance, self-driving cars, and robotic systems. This task has two main challenges: modeling complex interactions between pedestrians and the extraction of each pedestrian's unique movement pattern. In response to these two challenges, an attention-based interactive perception spatiotemporal graph neural network is proposed to predict the trajectory of pedestrians [14]. The proposed method has two components: a spatial graph neural network for interactive modeling and a temporal graph.
neural network for motion feature extraction. The spatial graph neural network uses the attention mechanism to capture the spatial interaction between all pedestrians at each time step. At the same time, the temporal graph neural network uses an attention mechanism to capture the temporal movement pattern of each pedestrian. Finally, the time extrapolator convolutional neural network is used in the time dimension of the ensemble graph feature to predict the future trajectory. Experimental results on two benchmark pedestrian trajectory prediction datasets prove that this method has a competitive advantage in terms of final displacement error and average displacement error measurement compared with the latest trajectory prediction methods. Few text classifications aim to learn a classifier from a few labeled text instances in each class. A small amount of previous research work in NLP is mainly based on the prototype network, which encodes the support set samples of each class into a prototype representation. It calculates the distance between the query and the prototype of each class. In the prototype aggregation process, a lot of useful support set information and the differences between samples of different categories were ignored. Instead, Xu et al. focused on all query support pairs without losing information. They proposed a graph neural network (Frog-GNN) based on multi-view aggregation, which can observe through the eyes (support and query examples) and speak through the mouth (a pair) to take fewer shots of text classification. They construct graphs through pre-trained pair representations and use instance-level representations to gather information from the neighborhood for message passing. After iterative interaction between instances, the final relationship feature of the pair means intra-class similarity and inter-class similarity. In addition, Frog-GNN with a meta-learning strategy can be well generalized to invisible classes. The experimental results show that the proposed GNN model is better than the existing short-shot methods in the short-short text classification and relational classification of the three benchmark data sets [15].

4. Discussion
Graph neural network algorithm is one of the hotspots of current deep learning research. In the past two years of artificial intelligence and data mining top conferences (ICLR, WWW, etc.), a lot of research progress has been made on the combination of deep learning and graphs. Graph neural networks can be viewed as an extension of deep learning in non-Euclidean spaces. Convolutional networks, attention mechanisms, autoencoders, and other ideas in deep learning have been applied to graph data to form corresponding graph algorithms, such as convolutional networks (GCN) and graph attention networks (GAT). This article conducts qualitative and quantitative analyses on the four algorithms of GNN.

| algorithm | advantages | disadvantages | improvement |
|-----------|------------|---------------|-------------|
| GCN       | The space-based method is more flexible, and sampling technology can be introduced to improve efficiency | difficult to handle large-scale graphs, directed graphs and dynamic graphs | A variant of the traditional convolution algorithm on graph structure data, which can be directly used to process graph structure data.  
1. Use the self-attention mechanism in the Transformer to assign different weights according to the characteristics of neighbor nodes.  
2. Training GCN does not need to understand the entire graph structure. Just know the neighbor nodes of each node.  
3. Introduce the multi-head self-attention mechanism to... |
| GAT       | The attention mechanism can amplify the impact of the most important part of the data. The attention function can be used to adaptively control the contribution of neighboring nodes j to node i, or to integrate multiple models, or to guide node sampling. | The computational cost and memory consumption increase rapidly with the calculation of the attention weight between each pair of neighbors. |...|
GAE: This algorithm can re-weight the items in the adjacency matrix to improve the fitting ability of the model. 1. Cannot completely overcome the sparsity of the adjacency matrix A. 2. High computational cost and memory consumption.

Migrate the Variational Auto-Encoders to the field of graphs, use a known graph to encode (graph convolution) to learn the distribution represented by the node vector, sample the vector representation of the node from the distribution, and then decode it (link prediction) to rebuild the diagram.

Graph Pooling: Easy to operate, suitable for simple max pooling and mean pooling. Inefficient and ignores the order information of nodes.

DIFFPOOL is proposed, a graph pooling module that can learn the hierarchical expression of graphs and combine various end-to-end GNNs with differentiable structures. DIFFPOOL learns differentiable soft cluster assignment for each layer of nodes and maps the nodes to a set of clusters, and this set of clusters will be used as the input (coarsened input) of the next layer of the GNN Layer.

Aiming at the problem of semi-supervised node classification, this article mainly conducts two experiments: one is the experiment on the literature citation network. The other is the experiment on the knowledge graph (NELL). In the literature citation network, edges are constructed using citation chains, and nodes represent corresponding documents. This article uses three reference network data sets: Citeseer, Cora, and Pubmed. The results of the data statistics are shown in the following table (Label rate represents the proportion of labeled nodes):

| Dataset   | Type          | Nodes | Edges | Classes | Features | Label rate |
|-----------|---------------|-------|-------|---------|----------|------------|
| Citeseer  | Citation network | 3327  | 4732  | 6       | 3703     | 0.036      |
| Cora      | Citation network | 2708  | 5429  | 7       | 1433     | 0.052      |
| Pubmed    | Citation network | 19717 | 44338 | 3       | 500      | 0.003      |
| NELL      | Knowledge graph | 65755 | 266144 | 210     | 5414     | 0.001      |

The experimental results on these data sets are shown in the following table:

| Method   | Citeseer | Cora | Pubmed | NELL |
|----------|----------|------|--------|------|
| MainReg  | 60.1     | 59.5 | 70.7   | 21.8 |
| SemiEmb  | 59.6     | 59   | 71.1   | 26.7 |
It can be seen that compared with other models, the effect of the semi-supervised node classification model based on GCN has been greatly improved.

5. Future work

In this section, we propose future work from three aspects:

**Scalability**: In the real world, the graph may contain numerous nodes and edges. To achieve the scalability of GNNs, the completeness of the graph is sacrificed. The sampling method may make a node lose its influential neighbors. In comparison, the clustering method may make the graph lose the structural pattern. Thus, finding a way to get the best of both worlds, preserving both scalability and graph integrity, will be the future work target [16].

**Robustness**: GNNs are vulnerable to adversarial attacks. The attacks focus not only on the features of the graph, but also on the structural information. The performance of the GNNs will decline dramatically if adversaries attack the features and/or the structure in the input graph. Thus, constructing a robust GNN architecture could effectively reinforce them against adversarial attacks and could be a future direction.

**Interpretability**: Since the major obstacle to applying GNNs to real-world issues is interpretability. It is crucial to apply GNN models to real-world applications with trusted explanations. Finding a general interpretable framework for GNNs could be a valuable future direction to investigate.

6. Conclusion

In this paper, an overview of graph neural networks is well conducted. It introduces the algorithm of recurrent neural networks and then reviews the related work on GNN. This paper conducts qualitative and quantitative analysis on the four algorithms of GNNs, which are GCN, GTA, GAE, and Graph Pooling. The advantages, disadvantages, and improvements of the four algorithms of GNNs are also examined. Aiming at the problem of semi-supervised node classification, Finally, three future research directions of graph neural networks are suggested, including scalability, robustness, and interpretability.

References

[1] Zhou J, Cui G, Zhang Z, et al. Graph neural networks: A review of methods and applications[J]. arXiv preprint arXiv:1812.08434, 2018.

[2] Yella, Jaswanth Kumar, and Anil G. Jegga. "MGATRx: Discovering Drug Repositioning Candidates Using Multi-view Graph Attention." bioRxiv, p3, 2020.

[3] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner and G. Monfardini, "The Graph Neural Network Model," in IEEE Transactions on Neural Networks, vol. 20, no. 1, pp. 61-80, Jan. 2009, doi: 10.1109/TNN.2008.2005605.

[4] Qiu J, Chen Q, Dong Y, et al. Gcc: Graph contrastive coding for graph neural network pre-training[C]. Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2020: 1150-1160.

[5] Li X. Explain Graph Neural Networks to Understand Weighted Graph Features in Node Classification[C]. International Cross-Domain Conference for Machine Learning and Knowledge Extraction. Springer, Cham, 2020: 57-76.

[6] Kim D, Oh A. How to find your friendly neighborhood: Graph attention design with self-supervision[C]. International Conference on Learning Representations. 2021.
[7] Zhang Z., Cui P., Zhu W., Deep learning on graphs: A survey [J]. IEEE Transactions on Knowledge and Data Engineering, 2020.

[8] Pareja A, Domeniconi G, Chen J, et al. Evolvegcnn: Evolving graph convolutional networks for dynamic graphs [C]. Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(04): 5363-5370.

[9] Zhou F, Cao C. Overcoming Catastrophic Forgetting in Graph Neural Networks with Experience Replay[J]. arXiv preprint arXiv:2003.09908, 2021.

[10] Huang Q, He H, Singh A, et al. Combining Label Propagation and Simple Models Out-performs Graph Neural Networks[J]. arXiv preprint arXiv:2010.13993, 2020.

[11] Back J, Kang M, Hwang S J. Accurate Learning of Graph Representations with Graph Multiset Pooling[J]. arXiv preprint arXiv:2102.11533, 2021.

[12] Wu Zonghan, Pan Shirui, Chen Fengwen, Long Guodong, Zhang Chengqi, Yu Philip S. A Comprehensive Survey on Graph Neural Networks.[J]. IEEE transactions on neural networks and learning systems, 2020.

[13] Sharma Himanshu, Jalal Anand Singh. Visual question answering model based on graph neural network and contextual attention[J]. Image and Vision Computing, 2021, 110(prepublish).

[14] Zhou Hao, Ren Dongchun, Xia Huaxia, Fan Mingyu, Yang Xu, Huang Hai. AST-GNN: An attention-based spatio-temporal graph neural network for Interaction-aware pedestrian trajectory prediction[J]. Neurocomputing, 2021, 445.

[15] Xu Shiyao, Xiang Yang. Frog-GNN: Multi-perspective aggregation based graph neural network for few-shot text classification[J]. Expert Systems With Applications, 2021, 176.

[16] Zhou, Y., Zheng, H., & Huang, X. (2020). Graph Neural Networks: Taxonomy, Advances and Trends. arXiv preprint arXiv:2012.087