Proceedings of the 4th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/79/20241646

# Graph neural networks in recommender systems

**Xingyang He**

YK Pao School, Shanghai, 201620, China

chrishe123321@gmail.com

**Abstract.** As a way to alleviate the information overload problem arisen with the development of the internet, recommender systems receive a lot of attention from academia and industry. Due to its superiority in graph data, graph neural networks are widely adopted in recommender systems. This survey offers a comprehensive review of the latest research and innovative approaches in GNN-based recommender systems. This survey introduces a new taxonomy by the construction of GNN models and explores the challenges these models face. This paper also discusses new approaches, i.e., using social graphs and knowledge graphs as side information, and evaluates their strengths and limitations. Finally, this paper suggests some potential directions for future research in this field.

**Keywords:** Recommender Systems, Graph Neural Networks, Social Recommendation, Knowledge Graph.

## 1. Introduction

The rapid growth of the internet has led to a proliferation of online information. Users must traverse through hundreds of redundant information before finding what they want, greatly denting their experience and satisfaction. To alleviate the information overload problem and enhance user experience, recommender systems have been developed to model user preferences and filter out redundant information [1, 2]. Recommender systems can be utilized on numerous occasions, such as product recommendations on e-commerce (e.g., Amazon, Taobao) and video recommendations on online video platforms (e.g., YouTube, Bilibili). Due to its commercial values (i.e., increased user engagement and revenue [3]), recommender systems have been a popular research topic for decades, sprouting numerous designs and architectures.

Among the various designs, Graph Neural Network (GNN) proves to be the new state-of-the-art approach in recommender systems [4]. GNN is a kind of neural network that operates on graph data. A graph consists of two elements: nodes and edges. A node is an entity and an edge connecting two nodes describes the relationship between the two entities. In recommender systems, nodes are typically users and items (e.g., products, videos, etc.) and edges are interactions between users and items (e.g., buying a product, viewing a video, etc.) The main reason for GNN's high performance in recommendation tasks is that GNN can access information from high-order neighbors (i.e., other users with similar preferences and the items they interact with). In contrast, traditional approaches using matrix factorization [5] and deep neural networks [6-8] can only utilize information from the user's first-order neighbors (i.e., items the user interacts with).

DOI: 10.54254/2755-2721/79/20241646

The remainder of the article is organized as the following: Section 2 introduces the typical GNN model, outlines representative approaches and their limitations to each of the model's steps, and summarizes new designs that tackle the problems. Section 3 introduces enhancements to the basic model (i.e., social graphs and knowledge graphs). Section 4 discusses some current issues in the field and future directions. Section 5 concludes the survey.

## 2. GNN: Basic Model

### *2.1. Overview*

GNN's goal is to learn nodes' vector embeddings that incorporate both their feature information and the graph structure. The user's preference for an item is then computed based on the similarity between the user and the item's learned embeddings. In essence, GNN takes in the graph's structure and feature information and iteratively updates each node's representation by aggregating feature information from its neighbors and integrating it with the node's current representation [9]. Thus, the process of GNN can be broken down into five steps: graph construction, aggregation, update, iteration, and final node representation. This section will explain each of the steps and outline the current methods used in each step. New and novel designs in each step will be brought up and discussed.

### *2.2. Graph construction*

This process involves creating graphs that serve as input to the GNN. Most GNN models use the original user-item interaction graph as input, such as [10] and [11]. However, for large-scale graphs, aggregating from all the neighbors might generate high computation costs. Thus, approaches have been proposed to decrease neighborhood size by sampling. PinSage [12] utilizes a random walk-based sampling method to construct the neighborhood of a node by choosing the nodes with the highest visit counts. [13] proposes a method to sample the neighborhood by choosing the users that have the closest interaction probability distribution with that of the original user. However, essential information may be lost during the sampling process, leading to lower effectiveness of the model. Thus, more focus should be put on exploring more efficient sampling methods and finding the balance between sampling and keeping the original information.

Another important issue is that as layers are stacked, high-order neighbors with dissimilar interests may also contribute to the user's representation, causing over-smoothing problems. To alleviate this problem, IMP-GCN [14] performs graph convolution operations on sub-graphs generated from the original graph that groups users with similar interests. This way, only those with similar interests will contribute to one's embedding.

### *2.3. Aggregation*

This process involves pooling information from neighbors. A simple aggregation operation is mean pooling, calculating the mean of neighbor embeddings. However, this method views every neighbor equally and does not consider the different influences neighbors may have. A video that aligns with a user's preference should have a larger influence on the user's representation than a video that merely catches attention. Thus, methods based on graph attention networks (GAT) [15] and self-attention mechanisms have been proposed to learn different weights for the neighbors, such as [16] and [17]. Another approach is graph convolutional networks (GCN), which are based on graph spectral theory and run convolutional operations on the input graph. Some of the methods using GCN are [18] and [19].

### *2.4. Update*

This process involves integrating the information aggregated from the neighbors with the original node embedding to update the node's representation. A simple approach is to linearly combine the representations through sum-pooling or mean-pooling methods. However, information may be lost. Thus, to preserve more information, some methods concatenate the two embeddings and perform a linear transformation with a non-linear activation function, such as [20]. Inspired by [21] and seeing high

DOI: 10.54254/2755-2721/79/20241646

computational costs for non-linearities, some methods such as LR-GCCF [22] remove non-linear activation functions, seeing an increase in computational efficiency and performance.

### *2.5. Iterations*

Aggregate and update operations are performed at each layer of the model. The vector representations learned by the previous layer through the operations are used as the input of the next layer. As more layers are stacked, more information from the neighbors can be collected and used. However, irrelevant information from distant neighbors may be incorporated into nodes' representations as layers are stacked. Thus, how many layers should be stacked is still an area of discussion.

### *2.6. Final node representation*

This process involves determining the final vector representations for both users and items, which are the outputs of the model. Some methods use the node representations for the last layer as the final representation, such as [11]. However, some argue that different layers can carry different information and should not be ignored in the final node representation [23]. Specifically, lower layers carry more individual information and higher layers carry more neighbor information. Thus, NGCF [23], MBGCN [24], and others concatenate the representations of different layers to form the final node representation. Another approach to retaining information is to aggregate and calculate the weighted average of every layer, similar to the approach in [25].

## 3. Enhancements to the Basic Model

Apart from the tools mentioned above, social and knowledge graphs that can enhance user and item representations are receiving more and more attention.

### *3.1. Social graph*

Social graphs have been leveraged to enhance user representations. Social graphs represent the relationship between users. A node represents a user and an edge linking two nodes indicates that the two users are friends. The advancement of online social platforms (e.g., Facebook, Twitter, Instagram, etc.) enables researchers to leverage this user-user relationship during the recommendation process. Models can use the representations of the user's friends to augment the user's representation. The effectiveness of this approach is supported by two concepts: social homophily and social influence. Social homophily refers to users' tendencies to connect with other users who have similar preferences and attributes. Social influence indicates that users with direct or indirect relations tend to influence each other to make themselves more similar. Thus, neighboring nodes tend to have similar preferences and aggregating information from neighboring users will enhance the users' representations.

Social graphs are incorporated into the graph construction process. The user-item interaction graph and the social graph can be integrated into a unified graph and aggregation is performed on the entire graph. This method allows information on social relationships to diffuse in the graph and affect the item representations. DiffNet++ [26] leverages GAT mechanisms to aggregate the interest information from user-item interactions and influence information from user-user relationships. In the update process, instead of mean or max pooling, it utilizes a second GAT to determine the contributions of the node's previous embedding and the aggregated information to the representation of the node. This second GAT considers the different influence levels their interest and friends have on their preferences: some users will be affected more by their friends than others.

Another approach to incorporating the social graphs is viewing the user-item interaction graph and social graph as separate graphs. It applies aggregation on both graphs and learns two separate user embeddings. Then, it integrates the learned representations to form the final user representation. Different methods have been proposed for integrating the representations. Some leverage weighted-sum pooling [27, 28] and some use multi-layer perceptron over the concatenated vectors [29, 30].

DOI: 10.54254/2755-2721/79/20241646

### *3.2. User-user and item-item graph*

Hand-coded user-user and item-item graphs have been used to enhance both user and item representation. Multi-GCCF [31] states that the original user-item interaction graph cannot fully exploit the proximity information among user pairs and item pairs. Thus, it constructs user-user and item-item graphs by identifying pairs of users and items that have a similar click/rating vector and adding an edge between them. It then performs sum aggregation on the graphs and fuses the learned embeddings with the learned embeddings from the user-item interaction graph.

### *3.3. Knowledge graph*

Knowledge graphs have been leveraged to enhance both user and item representation. Two types of knowledge graphs are used in the recommendation process: item knowledge graphs and user-item knowledge graphs [1]. In item knowledge graphs, nodes are item attributes and edges are relationships between items, including information from the users like "co-viewed" or "co-buy". In user-item knowledge graphs, both users and items are nodes and their relationships are edges.

Aggregation is, directly or indirectly, performed on knowledge graphs, replacing the user-item interaction graph. AKUPM [32] extracts multi-hop triplet sets that have the user's interacted items as their head entities from the item knowledge graph. It then calculates the user's representation by aggregating the information in each layer of the triplet set using self-attention mechanisms. This approach can provide explanations of its recommendation result since the high-weight edges linking the user and the recommended item can be selected and outputted. However, only the user representation is updated in this process. To solve this, KGAT [33] proposes to perform aggregation on the user-item knowledge graph. The high-order connectivity pattern can be exploited, but it will also introduce irrelevant neighbors into the aggregation process.

## 4. Problems Recommendation System Face and Future Research Directions

### *4.1. Scalability and Sparsity*

In industry-level recommendations, datasets may contain billions of nodes and edges. For example, YouTube has roughly 2.5 billion users and 800 million videos. Applying traditional GNNs on the dataset requires extremely long training time and expensive computational costs. Moreover, a typical YouTube user might have watched only a small proportion of the 800 million videos. This means that the adjacency matrix will be extremely sparse, influencing the performance of GNN models. Future GNN models should consider the sparse nature of the dataset in its design and provide a scalable approach to recommendations.

### *4.2. Dynamic graph*

Most methods discussed before operate on static graphs. However, in the real world, users interact with new items all the time, and their relationship changes. Retraining the model every time a new edge is added is expensive. Moreover, the changes in preference over time may carry information. Thus, it is important to find ways to utilize and incorporate this continuous change of input.

### *4.3. Reliability of input data*

In the original user-item interaction graph, user-item pairs that have an interaction are considered positive samples and the else are considered negative samples. However, some interactions may not be relevant to recommendations since users might accidentally click an item. Also, items that a user hasn't interacted with shouldn't immediately be considered as not interesting to the user since the user simply might not see the item when browsing. Furthermore, there might be pseudo or missing relationships in social graphs. Some neighbors may have no influence on the user and some that have strong influences may not be represented. These are all sources of uncertainty in the input data, and these uncertainties are inevitable. Methods have been developed to cope with this uncertainty, such as ESRF [34] which

DOI: 10.54254/2755-2721/79/20241646

leverages the autoencoder mechanism to augment the social graph by filtering out irrelevant edges. However, more research should be done on how to deal with this uncertainty.

### *4.4. Diverse interests*

A user may have multiple and diverse interests. An embedding might not be sufficient to represent one user. Additionally, most models use the cosine similarity between the user and item embeddings to determine the likeliness of the user to interact with the item. This approach may result in the learned user embedding trying to fit into the different interests and result in irrelevant recommendations.

### *4.5. Atypical interactions*

A user may interact with an item that is very different from his past preferences, such as watching a basketball video when before he only watches football. This atypical interaction can be interpreted as a signal for the user's change in preference, or it may just be an accidental click. How to process this atypical interaction can be an interesting area of research.

### *4.6. Beyond-accuracy recommendation*

The goal of recommender systems is to recommend things that users like to them. However, this may cause ethical issues such as information cocoons and discriminatory recommendations. Users will be recommended items that align with their values and thoughts, trapping them in a cocoon where they cannot see contrasting perspectives. Moreover, users might become bored with the similar content they have been recommended. Thus, further research should focus on recommending a diverse range of items to users and make sure there are no discriminatory practices.

## 5. Conclusion

Graph neural network proves to be a promising tool for recommender systems due to its effectiveness in processing graph data. This survey provides a comprehensive review of recent works on GNN-based recommender systems. This survey is divided based on the construction steps of a typical GNN model. For each step, it briefly states the goal of the step, details some of the representative approaches, discusses their limitations, and offers insight into new approaches to solve the issue. Enhancements to the typical GNN model, such as social graphs and knowledge graphs, are explained and representative approaches are discussed. Furthermore, this survey outlines some directions for future research in this field. I hope this survey will provide readers with knowledge of recent developments in this field and shed light on future development.

## References

[1] Q. Guo et al., "A Survey on Knowledge Graph-Based Recommender Systems," IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 8, pp. 3549-3568, 2022, doi: 10.1109/tkde.2020.3028705.

[2] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou, "Recommender systems," Physics reports, vol. 519, no. 1, pp. 1-49, 2012.

[3] D. Jannach and M. Jugovac, "Measuring the Business Value of Recommender Systems," ACM Transactions on Management Information Systems, vol. 10, no. 4, pp. 1-23, 2019, doi: 10.1145/3370082.

[4] C. Gao et al., "A Survey of Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions," ACM Transactions on Recommender Systems, vol. 1, no. 1, pp. 1-51, 2023, doi: 10.1145/3568022.

[5] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," Computer, vol. 42, no. 8, pp. 30-37, 2009.

[6] H.-T. Cheng et al., "Wide & deep learning for recommender systems," in Proceedings of the 1st workshop on deep learning for recommender systems, 2016, pp. 7-10.

DOI: 10.54254/2755-2721/79/20241646

[7] K. R, P. Kumar, and B. Bhasker, "DNNRec: A novel deep learning based hybrid recommender system," Expert Systems with Applications, vol. 144, 2020, doi: 10.1016/j.eswa.2019.113054.
[8] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural Collaborative Filtering," presented at the Proceedings of the 26th International Conference on World Wide Web, 2017.
[9] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: a survey," ACM Computing Surveys, vol. 55, no. 5, pp. 1-37, 2022.
[10] L. Zheng, C.-T. Lu, F. Jiang, J. Zhang, and P. S. Yu, "Spectral collaborative filtering," in Proceedings of the 12th ACM conference on recommender systems, 2018, pp. 311-319.
[11] C. Li, K. Jia, D. Shen, C.-J. R. Shi, and H. Yang, "Hierarchical Representation Learning for Bipartite Graphs," in IJCAI, 2019, vol. 19, pp. 2873-2879.
[12] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. 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.
[13] W. Shafqat and Y.-C. Byun, "Incorporating Similarity Measures to Optimize Graph Convolutional Neural Networks for Product Recommendation," Applied Sciences, vol. 11, no. 4, p. 1366, 2021, doi: 10.3390/app11041366.
[14] F. Liu, Z. Cheng, L. Zhu, Z. Gao, and L. Nie, "Interest-aware message-passing gcn for recommendation," in Proceedings of the Web Conference 2021, 2021, pp. 1296-1305.
[15] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," arXiv preprint arXiv:1710.10903, 2017.
[16] C. Feng, Z. Liu, S. Lin, and T. Q. Quek, "Attention-based graph convolutional network for recommendation system," in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019: IEEE, pp. 7560-7564.
[17] Z. Xu, H. Liu, J. Li, Q. Zhang, and Y. Tang, "CKGAT: Collaborative Knowledge-Aware Graph Attention Network for Top-N Recommendation," Applied Sciences, vol. 12, no. 3, p. 1669, 2022.
[18] J. Zhang, X. Shi, S. Zhao, and I. King, "Star-gcn: Stacked and reconstructed graph convolutional networks for recommender systems," arXiv preprint arXiv:1905.13129, 2019.
[19] S. Zhang, H. Yin, T. Chen, Q. V. N. Hung, Z. Huang, and L. Cui, "Gcn-based user representation learning for unifying robust recommendation and fraudster detection," in Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, 2020, pp. 689-698.
[20] Y. Xu, Y. Zhang, W. Guo, H. Guo, R. Tang, and M. Coates, "Graphsail: Graph structure aware incremental learning for recommender systems," in Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 2861-2868.
[21] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in International conference on machine learning, 2019: PMLR, pp. 6861-6871.
[22] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach," in Proceedings of the AAAI conference on artificial intelligence, 2020, vol. 34, no. 01, pp. 27-34.
[23] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval, 2019, pp. 165-174.
[24] B. Jin, C. Gao, X. He, D. Jin, and Y. Li, "Multi-behavior recommendation with graph convolutional networks," in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 659-668.
[25] J. Wu et al., "Graph convolution machine for context-aware recommender system," Frontiers of Computer Science, vol. 16, no. 6, p. 166614, 2022.

DOI: 10.54254/2755-2721/79/20241646

[26] L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang, "Diffnet++: A neural influence and interest diffusion network for social recommendation," IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 10, pp. 4753-4766, 2020.
[27] Q. Wu et al., "Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems," in The world wide web conference, 2019, pp. 2091-2102.
[28] Y. Jiang, H. Ma, Y. Liu, Z. Li, and L. Chang, "Enhancing social recommendation via two-level graph attentional networks," Neurocomputing, vol. 449, pp. 71-84, 2021.
[29] X. Wang, Z. Liu, N. Wang, and W. Fan, "Relational metric learning with dual graph attention networks for social recommendation," in Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, Proceedings, Part I 24, 2020: Springer, pp. 104-117.
[30] W. Fan et al., "Graph neural networks for social recommendation," in The world wide web conference, 2019, pp. 417-426.
[31] J. Sun et al., "Multi-graph convolution collaborative filtering," in 2019 IEEE international conference on data mining (ICDM), 2019: IEEE, pp. 1306-1311.
[32] X. Tang, T. Wang, H. Yang, and H. Song, "AKUPM: Attention-enhanced knowledge-aware user preference model for recommendation," in Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 1891-1899.
[33] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," in Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 950-958.
[34] J. Yu, H. Yin, J. Li, M. Gao, Z. Huang, and L. Cui, "Enhancing social recommendation with adversarial graph convolutional networks," IEEE Transactions on knowledge and data engineering, vol. 34, no. 8, pp. 3727-3739, 2020.