A Graph Neural Network Recommendation Model with Knowledge Graph and Attention Mechanism

Mabeline Hill (MabelineHill1993.port@hotmail.com)
University of Portsmouth
Horatio Finch
University of Portsmouth
Priya Phillips
University of Portsmouth
Aisha Martin
University of Portsmouth

Research Article

Keywords: Recommendation Systems, Graph Neural Networks, Collaborative Filtering, Information, Knowledge Graph

Posted Date: April 17th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2815491/v1

A Graph Neural Network Recommendation Model with Knowledge Graph and Attention Mechanism

Mabeline Hill, Horatio Finch, Priya Phillips, Aisha Martin,

Abstract—With the growth of e-commerce and social media platforms, recommendation systems have become crucial for businesses to increase user satisfaction and retention. Collaborative filtering has been the primary recommendation algorithm, but it faces challenges such as the cold start problem and data sparsity. Researchers have tried to introduce supplementary information to complement the data, and Graph Neural Networks (GNNs) have been useful in capturing the relevance of graphs in recommender systems. In addition, traditional GNNs only fuse the first-order neighborhood features of nodes and cannot capture the deeper structural relationships of nodes in the network. Therefore, when the data set is sparse and each node has only a very small number of neighbors, the recommendation quality of traditional GNN-based recommendation algorithms decreases significantly. In this paper, we propose a GNN recommendation model based on user-side information, which can effectively capture the potential interest of users through user relationship graph for recommendation, and fully exploit the large amount of association information among users. The traditional collaborative filtering idea is incorporated with the knowledge graph as the auxiliary information to fully utilize the potential information from both ends to improve the accuracy and diversity of the recommendation model. In addition, the interaction behaviors and opinions in the user-item graph are jointly captured. Different strengths of user-item interactions are considered, which indicate that various interactions contribute differently to user preferences (or item characteristics), and thus attention mechanisms are used to achieve this goal. The algorithm has been verified by public dataset, and it has better recommendation effect than several other classical recommendation algorithms.

Index Terms—Recommendation Systems, Graph Neural Networks, Collaborative Filtering, Information, Knowledge Graph.

1 Introduction

With the rapid growth of e-commerce and social media platforms, recommendation systems have become an essential tool for many businesses. An effective recommendation system can increase user satisfaction and retention by accurately capturing user preferences and discovering items that may be of interest to users. A recommendation system evaluates user preferences based on user interests and item attributes [1, 2]. Since both user interests and item features are represented as compressed vectors, learning user/item interactions and other auxiliary information, such as social relationships and knowledge graph embedding representations, are key problems to be solved in this neighborhood. The traditional recommendation system based on collaborative filtering [3] has been the main direction of recommendation algorithm research because it can take advantage of big data to find users with the most similar interests and recommend items that they are interested in. However, there are two problems that are difficult to solve in collaborative filtering, namely, the cold start problem caused by too little initial data of new users and the sparse interaction data problem caused by too many items and too few items interacting with users. To solve these two problems, researchers have tried to introduce various kinds of auxiliary information to complement the data, such as point-of-interest information [4], comment information [5], social network [6, 7], and contextual information [8], which have been useful in specific scenarios. The introduction of these auxiliary information plays a role in specific scenarios. Traditional collaborative filtering algorithms are based on the user. Item rating matrix, but since this matrix is very sparse in real life, the collaborative filtering algorithm faces the challenge of data sparsity [9]. Later on, machine learning and deep learning are gradually combined with traditional recommendation algorithms [10]. In recommender systems, most of the information has a graph structure, and Graph Neural Network (GNN) techniques can capture the relevance of graphs through message passing between graph nodes, so GNNs are often used to generate embedded representations of users/items. Knowledge graphs have been widely used in several fields because of their ability to fuse data from multiple sources, expand the semantic information of the data, and infer more potential information about the data [11]. Recently, researchers have started to try to introduce knowledge graphs as auxiliary information to improve the performance of recommendation systems. In addition to alleviating the problems of cold start [12] and data sparsity [13] in traditional collaborative filtering methods, the introduction of knowledge graphs as auxiliary information for recommendation has the following advantages.The introduction of knowledge graphs can be combined with collaborative filtering methods to bring more accurate recommendations to users. The inferred

*Mabeline Hill is the corresponding author.

- *Mabeline Hill, Horatio Finch, Priya Phillips, and Aisha Martin are with University of Portsmouth, UK. (e-mail: MabelineHill1993.port@hotmail.com).*

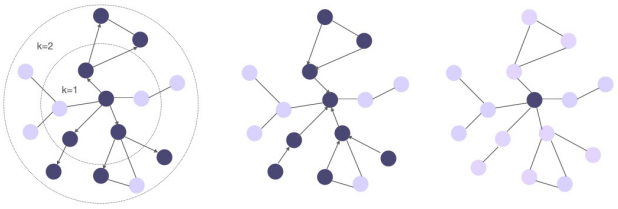

Fig. 1: Neighboring node sampling example.

implicit information contained in the knowledge graph enhances the ability of the knowledge graph-based recommendation system to explore users' latent interests and provide users with richer recommendation results.

However, traditional GNN-based recommendation algorithms are only able to handle regular topological graphs composed of a single type of nodes, while the data in the current network is not composed of only a single type of nodes [14]. In addition, traditional GNNs only fuse the first-order neighborhood features of nodes and cannot capture the deeper structural relationships of nodes in the network. Therefore, when the data set is sparse and each node has only a very small number of neighbors, the recommendation quality of traditional GNN-based recommendation algorithms decreases significantly. In this paper, we propose a GNN recommendation model based on user-side information, which can effectively capture the potential interest of users through user relationship graph for recommendation, and fully exploit the large amount of association information among users. The traditional collaborative filtering idea is incorporated with the knowledge graph as the auxiliary information to fully utilize the potential information from both ends to improve the accuracy and diversity of the recommendation model. In addition, the interaction behaviors and opinions in the user-item graph are jointly captured. Different strengths of user-item interactions are considered, which indicate that various interactions contribute differently to user preferences (or item characteristics), and thus attention mechanisms are used to achieve this goal. The algorithm has been verified by public dataset, and it has better recommendation effect than several other classical recommendation algorithms.

2 Related Work

2.1 Traditional Recommendation Algorithms

Collaborative filtering-based recommendation algorithms can be further divided into neighborhood-based collaborative filtering and model-based collaborative filtering. Neighborhood-based collaborative filtering can be further subcategorized into user-based collaborative filtering and item-based collaborative filtering [15]. In user-based collaborative filtering, the similarity between users is calculated based on their past behaviors, and personalized recommendations are generated based on this similarity metric. On the other hand, item-based collaborative filtering calculates the similarity between items based on a user's historical behavior and generates recommendations accordingly. Model-based collaborative filtering, on the other hand, uses the user-item rating matrix to train a prediction model that generates recommendations. This prediction model is trained by the item rating matrix, and it predicts the missing items in the rating matrix. Some of the classical models used for this purpose are the clustering model [16], the graph model [17], and the matrix decomposition model [18].

The advanced feature representation capabilities of deep learning have significantly reduced the need for human feature selection in recommendation systems while improving their accuracy. Researchers have explored various neural network structures for feature learning tasks to help models better learn implicit feature vectors from user-item history behaviors. Some experts argue that user-item interactions are complex and require the capturing of non-linear interactions. In recent years, deep neural network models have shown great potential for effective feature representation learning in fields such as speech recognition [19], computer vision [20], and natural language processing [15]. Several studies have successfully applied deep neural networks to recommendation tasks [21]. In 2016, Cheng et al. [16] proposed the $Wide\&Deep$ model, which uses a logistic regression model as the Wide component to mine correlations in historical interaction behaviors, enabling better memory capability and improving accuracy. The Deep component of the model is a deep neural network that enhances generalization capability. However, this model is dependent on appropriate feature selection. To address this issue, Guo et al. [17] proposed the DeepFM model in 2017. It replaces the Wide component with a factorization machine model that uses additive and inner product operations to obtain linear and first-order interactions, while a multilayer perceptron is used to obtain higher-order linear features. This approach reduces the reliance on manual feature selection.

2.2 Graph Neural Network based Recommendation Algorithms

Graph neural networks were first proposed by Gori et al. [22] in 2005 and then further elucidated by Scarselli et al. [23] in 2009. In early graph data-based studies, for the target node representation, researchers learned by iteratively propagating neighbor information through recurrent neural networks. In recent years, there has been an increasing focus on the study of graph representations [24]. The computational effort in this process is very large, and many researchers hope to alleviate this problem, which has become a hot topic of research in recent years. The relationship between graph neural networks and network embeddings (also known as graph embeddings) is close and is briefly described here. Network embedding is the process of projecting the nodes in a graph into a low-dimensional vector space while preserving the network topology and node information, thus producing a low-dimensional vector representation of the nodes, which can then be used for graph data analysis (e.g., classification, recommendation, etc.) [21, 25]. General network embedding algorithms can be broadly classified into: matrix decomposition [26], random walk [27], and deep learning methods. Among the deep learning methods for network embedding, which also belong to graph neural networks, include algorithms based on graph self-encoders (e.g., SDNE [28]) and graph

convolutional neural networks with unsupervised learning (GraphSage [29]). Using graph neural networks to make recommendations is actually using graph neural networks to learn the embedded representations of users and items, which can also be understood as using graph neural networks to learn the hidden features of users and items. The user-item interaction matrix is modeled as the user-item interaction matrix. Item interaction matrix is modeled as a user-item graph. Project graph, where users and projects correspond to nodes in the graph one by one. Graph neural networks have been shown to be good at learning embedding representations of nodes on graphs, so using graph neural networks here can be good at learning embedding representations of users and items, thus further improving recommendation accuracy.

2.3 Attention mechanisms

In recent years, attention models have been widely used in various areas of deep learning, whether in image processing, recommender systems, or various tasks in natural language processing. The main purpose of attention in deep learning is to select the information that is more useful for the current task from a large amount of information, which is similar in nature to human visual attention. The attention model introduced by Bahdanau et al. [30] is also called soft attention. The soft attention model uses the weighted average of all hidden states in the input sequence to construct the content vector. Xu et al. [31] proposed the hardattention model (hardattention), and the content vector is computed from a random sample of hidden states in the input sequence. Luong et al. [32] proposed two attention models in the context of machine translation, namely the local attention model (Local) and Global attention models. Among them, the local attention model is between soft attention and hard attention, and the main idea is to first detect the attention point in the input sequence, and then select a window around that location to create a local attention model. The global attention model is similar to the soft attention model. Attention models are also widely used in recommender systems, mainly to assign attention weights to that user's interaction items, which are then used for user analysis, thus allowing to capture more effectively the user's long-term and short-term interests, user. This allows us to capture more effectively the user's long-term and short-term interests, the interaction behaviors of different intensities between items. The reason why the attention model can be used in these areas is that users' interests are variable and the different ratings of items by users reflect the different interests behind these interactions.

3 METHODOLOGY

3.1 Problem Definition

In the recommendation system, let $U = \{u_1, u_2, \ldots, u_m\}$ and $I = \{i_1, i_2, \ldots, i_n\}$ denote the set of users and the set of items, respectively, where m denotes the number of users and n denotes the number of items. Assume that the user-item rating matrix is $\forall R \in R_{m \times n}$ also called the user-item rating graph. If user u has a rating for item i, then r_{ab} is the true value of the rating, and conversely if no rating is observed for user u for item i, then it is denoted by 0, $r_{ab} = 0$. Let $U = \{< u_a, i_b > | r_{ab} \neq 0\}$ denote the set of observed ratings and $T = \{< u_a, i_b > | r_{ab} = 0\}$ denote the set of observed ratings. The observed ratings r_{ab} can be viewed as the viewpoint of user u with respect to item i. Denote by $B(b)$ the set of users who have interactions with item i_b and by $C(a)$ the set of items that have direct connections with user u_a. Given the user-item rating matrix R, the purpose of this chapter is to predict the missing ratings in R. In this chapter, we use the embedding vector $e_u \in R^d$ to denote a user $u_a, e_i \in R^d$ to denote an item i_b, where d is the size of the embedding vector dimension.

3.2 Construction of user relationship graph

First, the user-item cross-rating data itself is a directed heterogeneous graph, with users and items as separate entities and ratings as relationships, which, of course, poses great difficulties for the model and is not conducive to the GNN method. The purpose of this paper is to obtain interest relationships between users, so we directly connect users who have common ratings on the same items to obtain a multidimensional homogeneous graph, but there is still a problem because using items as relationships, i.e., two similar users may theoretically have an increasing number of relationships as the total number of items increases. Considering that different items as relationship types do not have substantial differentiation and do not need to be treated as features, this paper further simplifies the graph to obtain a homogeneous graph, and the number of common rating items between users is used as a relationship type to connect user entities, which is in line with the expected goal of this paper, because the relationship can well reflect the similarity of interests between different users. Finally, we use the processed user relationship graph as the input of the model, and directly connect users as entity nodes in the graph, and the number of common interaction records between users as the connection relationship, and the number of common interaction records as the weight to connect users, forming an entitled undirected homogeneous graph.

3.3 Sampling method

GNN-based recommendation models usually use random fixed-size neighbor sampling, which can easily lose important neighbor node information. In this paper, we consider a more targeted selection of relatively important neighborhood nodes in the sampling process to make the model learn valuable potential attributes of items more easily. Specifically, the importance of different nodes is measured by calculating the relationship density between all nodes in the network and the central node, and the higher the relationship density, the more important the node is to the central node, and the corresponding node is the target sampling node. Without obtaining the specific characteristics of nodes, the commonly used metrics for calculating the relationship tightness are common neighbors(CN) [33], Adamic/Adar coefficient(AA) [34], resource allocation coefficient(RA) [35], etc.

The formula for the CN is as follows:

$$S_{xy} = |\Gamma(x) \cap \Gamma(y)| \tag{1}$$

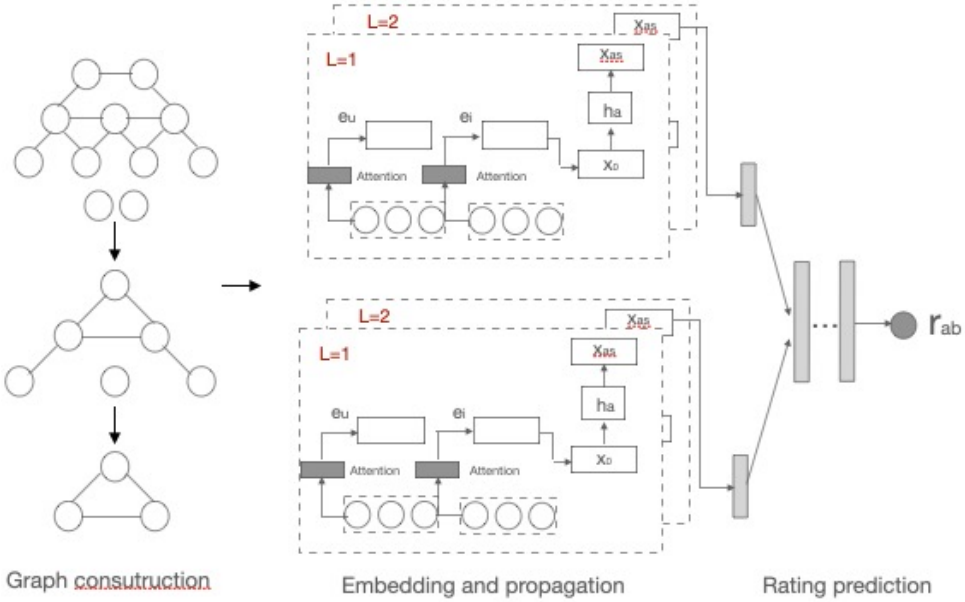

Fig. 2: Schematic diagram of our model.

The formula for the AA is as follows:

$$S_{xy} = \sum \frac{1}{\log k(i)}, k \in \Gamma(x) \cap \Gamma(y) \tag{2}$$

The formula for the RA is as follows:

$$S_{xy} = \sum \frac{1}{k(i)}, k \in \Gamma(x) \cap \Gamma(y) \tag{3}$$

where S_{xy} represents the relational tightness of the neighborhood node y and the central node x, $\Gamma(x)$ denotes the set of 1st order neighbor nodes of node x, and $k(i)$ is the node degree value. The CN coefficient directly compares the number of common neighbors, and the number of common neighbors is used to consider the tightness of the relationship. The more the number of common neighbors, the closer the relationship between nodes. The RA coefficient introduces the consideration of degree value based on the number of common neighbors, and differentiates the influence of common neighbors, considering that the higher the degree value of common neighbors, the more valuable it is. The AA coefficient is logarithmic to the degree calculation in the RA coefficient, which takes into account the possibility that some nodes may have a high degree that interferes with the calculation of the relationship tightness. For example, if a node is connected to most of the nodes, it will become a common neighbor of many nodes, and its degree value is large, which makes the role of other common neighbors become very small, which makes the degree value size is not conducive to the calculation of relationship tightness. The effect of the degree factor can be effectively considered without making it too large. Considering that the number of common neighbors of nodes in the knowledge graph in the recommendation scenario is not large, the node with a large degree value only indicates that the entity features are more average, and the calculation of the degree value will bring exponential computational cost, so the CN coefficient is used as the criterion for calculating the relationship tightness in this paper.

3.4 Embedding layer

Modeling user-item interactions based on user embedding and item embedding. We describe a user u_a(item i_d) with an embedding vector $e_u \in R^d$ ($e_i \in R^d$), where d denotes the size of the embedding.

$$E = [e_{u1}, \ldots, e_{um}, e_{i1}, \ldots, e_{in}, e_1, \ldots, e_5] \tag{4}$$

where $e_{u1}, \ldots, e_{um}$ is the user embedding, $e_{i1}, \ldots, e_{in}$ is the item embedding, and $e_1, \ldots, e_5$ represents the rating level, of the viewpoint embedding. We jointly consider the user embedding, item embedding, and viewpoint embedding, feed the end-user embedding and item embedding into a multilayer perceptron, and then output the predicted ratings. In the model, this paper optimizes the user and item embeddings by propagating the embeddings on the user-item interaction graph.

3.5 Propogation layer

Capturing collaborative filtering signals along the graph network structure and fine-tuning the embedding requires the construction of an information dissemination architecture for GNNs. Items that interact with users can provide direct evidence of user preferences. In the same way, users who adopt an item can provide evidence of the item's characteristics, and this can be used to measure the similarity of the two items. Users. The item graph contains the users, the interactions between items and the users' views (or rating) of them. In this paper, we propose a method for jointly capturing views and interactions in the user. Item graphs to jointly capture views and interactions in order to learn users' potential factor e_u. For a connected user-item pair, we define the message from i_s to u_a as follows.

$$h_a = \sigma(wAgg_i(\{x_{sa}, \forall s \in C(a)\}) + b) \tag{5}$$

where $C(a)$ denotes the set of items with which user u_a has interaction, x_{sa} denotes the viewpoint-based embedding representation of the interaction between user u and item

i, and Agg_i is the item aggregation function. In addition σ denotes the nonlinear activation function, w,b are the weights and biases of the neural network, respectively.

Clearly, users can express their opinions by rating interactions with items and noting them as r. These opinions about items capture how users prefer items. To model views, for each rating level, a viewpoint embedding representation is defined. For the interaction between a user u_a with rating level r and an item i_s, a two-layer neural network is used to combine the item embedding e_{ib} with the viewpoint embedding e_r, thus modeling the viewpoint-based representation of the interaction x_{sa}. This can be represented as a fusion network in which g_v takes as input a cascade of item embeddings and viewpoint embeddings, and the output is an interaction-based viewpoint representation.

$$x_{sa} = g_v([e_{is} \oplus e_r]) \tag{6}$$

where $\oplus$ denotes the concatenation operation of two vectors. The classical aggregation function Agg_i is based on the mean operator, which takes the mean value of the elements of the vector in $x_{as}, \forall s \in C(a)$.

$$h_a = \sigma(w \cdot (\{ \sum_{s \in C(a) \beta_a x_{sa}} \}) + b) \tag{7}$$

The approach assumes that all interactions contribute equally to understanding the user. In fact, not every interaction has the same impact on the user's viewpoint-based embedding, so we allow various interactions to contribute differently to the user's latent factor. Therefore, in this paper, we use heterogeneous weights to represent unequal contributions. Naturally, this is achieved through an attention mechanism as a way to improve the β_{as} of u_a.

$$h_a = \sigma(w \cdot (\{ \sum_{s \in C(a) \beta_{as} x_{sa}} \}) + b) \tag{8}$$

When understanding user preferences from historical interactions $C(a)$, β_{as} denotes the attentional weight of the contribution of interactions with is to the potential factor of user u_a when understanding the potential factor of user u_a. The attentional network is defined as follows.

$$\beta_{as}* = w_2^T \cdot \sigma([x_{as} \oplus e_{ua}] + b_1) + b_2 \tag{9}$$

The above attention scores are normalized using the Softmax function to obtain the target attention weights such that the contribution to the user potential factor of user u_a is as follows.

$$\beta_{as} = \frac{exp(\beta_{as}*)}{\sum_{s \in C(a)} exp(\beta_{as}*)} \tag{10}$$

In this paper, we improve the representation of u_a by aggregating the information coming from the neighbors of the user u_a.

$$D_1 = e_{ua} \oplus h_a \tag{11}$$

$$D_2 = \sigma(w_1 \cdot D_1 + w_2) \tag{12}$$

$$e_{ua}^{(l)} = \sigma(w_3 \cdot D_2 + w_4) \tag{13}$$

where $e_{ua}^{(l)}$ denotes the user u_a embedding representation obtained after the first embedding propagation layer. We consider an initialization of the user u_a embedding that preserves the information of the original features. First, e_{ua} and h_a are concatenated, and the results are fed into a two-layer neural network, and the first embedding is obtained through the attention mechanism and the embedding propagation layer to optimize the initialize the initial embedding.

$$e_{ua}^{(l)} = leakyReLU(m_{u \to u} + \sum_{s \in C(a)} m_{u \to i}) \tag{14}$$

The activation function of LeakyReLU allows messages to encode both positive and negative signals.

$$e_{ib}^{(l)} = leakyReLU(m_{u \to u} + \sum_{t \in B(b)} m_{u \to i}) \tag{15}$$

Now that the modeling of single-layer connectivity has been completed, more embedded layers can be stacked to explore multi-layer connectivity information. Since multilayer connectivity is essential to encode cooperative signals between user and item pairs. By stacking embedded propagation layers, users (and items) are able to receive information propagated from their hopping neighbors. In the l-th hop step, the user u_a is represented as follows.

$$e_{ua}^{(l)} = leakyReLU(m_{u \to u}^{(l)} + \sum_{s \in C(a)} h_a^{(l)}) \tag{16}$$

3.6 Rating prediction

After propagating layer l, multiple representations of user u_a are obtained. Since the representations obtained in the different layers reflect the information transmitted through different connections, they have different contributions in representing the user preferences. Therefore, we concatenate them to form the final embedding of the user. The same operation is then performed on the project.

$$e_{ua}* = e_{ua}^{(0)} || \dots || e_{ua}^{(l)} \tag{17}$$

$$e_{ib}* = e_{ib}^{(0)} || \dots || e_{ib}^{(l)} \tag{18}$$

where $||$ denotes the connection operation. We use the embedding propagation layer to optimize the initial embedding representation and control the information propagation range by adjusting the size of l. We first connect the obtained potential factors of users and items, and then feed them into a multilayer perceptron for scoring prediction.

$$G_1 = e_{ua} \oplus e_{ib} \tag{19}$$

$$G_2 = \sigma(w_1 \cdot G_1 + b_2) \tag{20}$$

$$G_{l-1} = \sigma(w_l \cdot G_l + b_l) \tag{21}$$

$$r_{ab} = w^T \cdot G_{l-1} \tag{22}$$

where l is the number of layers in the hidden layer and r_{ab} is the prediction score of user u_a for item i_b.

TABLE 1: Model Performance Comparison on Movielens-1M.

Model	RMSE	MAE
GCN	1.1524	0.9638
NeuMF	1.0658	0.9412
GCMC	0.9827	0.9124
GraphRec	0.9352	0.8956
Ours	0.8932	0.8556

TABLE 2: Model Performance Comparison on Last.FM.

Model	RMSE	MAE
GCN	1.2633	1.0846
NeuMF	1.2016	1.0425
GCMC	1.1546	0.9875
GraphRec	1.0943	0.9667
Ours	1.0723	0.9451

4 EXPERIMENTS

4.1 Datasets

The MovieLens-1M dataset is a widely used benchmark dataset for movie recommendations that contains approximately 6036 users with a total of 2445 entries and 1 million rating records (from 1 to 5).

Last.FM2015 is a dataset of music by artists, albums, tracks and users, as well as information about individual listening events. The dataset contains a total of 15,471 projects, 12,134 subscribers and 3 million rating records.

4.2 Metrics

For rating prediction scenarios, user preferences are usually modeled in a point.wise manner, i.e., considering the degree of preference of a single user for a single item. The modeling approach uses the minimization of squared loss, so the evaluation metrics for rating prediction scenarios are root mean square error (RMSE) and mean absolute error (MAE). For a pair of user u and item i in the test set, let the actual rating of user u for item i be r_{ui} and the predicted rating calculated by the recommendation algorithm be $\hat{r}_{ui}$.

$$RMSE = \sqrt{\frac{\sum_{u,i \in R}(r_{ui} - \hat{r}_{ui})^2}{|R|}} \tag{23}$$

MAE uses absolute values to calculate the prediction error.

$$MAE = \frac{\sum_{u,i \in R}|r_{ui} - \hat{r}_{ui}|}{|R|} \tag{24}$$

The higher the evaluation index score, the better the performance of the recommendation.

4.3 Baselines

We compare our model with some relevant or current state-of-the-art methods.

- GCN. it is a scalable semi-supervised learning method based on graph structured data. The method is a classical improvement of convolutional methods on graph networks. The model varies the choice of convolution kernel by local first-order approximation of the spectral graph convolution.
- NeuMF. This is a neural collaborative filtering model that uses multiple hidden layers and concatenated user and item embeddings to capture their nonlinear features.
- GCMC. The model uses a GCN encoder to generate a representation of users and items, considering only first-order nearest neighbors. Therefore, a graph convolution layer is used, as suggested by GCMC, where the hidden dimension is set to the embedding size.
- GraphRec. This model uses GNNs to integrate node information and topology to model user-project graphs, user-user social graphs, and different intensities in a consistent manner.

4.4 Result Analysis

First, we compare the recommended performance of the various methods. Table 1 shows the overall scoring prediction error:RMSE and MAE of the data set recommendation methods. the main findings are as follows. NeuMF obtains better performance than GCN. Both methods utilize only rating information. However, NeuMF is based on a neural network architecture, which shows the power of neural network models in recommender systems. graphRe outperforms GCMC. both methods utilize only graph neural networks. However, GraphRec is based on the embedding propagation architecture, which illustrates the power of the embedding propagation model. The method in this paper outperforms all baseline methods in most cases. In contrast to GraphRec, the model in this paper utilizes an attention mechanism in the user item graph to simultaneously fuse the interaction behavior of user items and the user's perspective. This illustrates the importance of simultaneously fusing the interaction behavior of user items and user perspectives.

Table 3 gives the performance comparison of the embedding length of the model proposed in this paper on Movielens-1M and Last.FM. In Table 3, the performance first increases and then decreases as the embedding dimension increases. When the embedding size increases from 16 to 64, the performance can be improved. However, when the embedding size is 256, our model decreases the performance. It indicates that using a large number of embedding dimensions has a strong representation capability. However, the complexity of the model increases significantly when the embedding dimension is large. Therefore, it is necessary to find the appropriate embedding dimension.

5 CONCLUSION AND FUTURE WORK

An effective recommendation system can increase user satisfaction and retention by accurately capturing user preferences and discovering items that may be of interest to users. A recommendation system evaluates user preferences based on user interests and item attributes. Since both user interests and item features are represented as compressed vectors, learning user/item interactions and other auxiliary information, such as social relationships and knowledge graph embedding representations, are key problems to be solved in this neighborhood. The traditional recommendation system based on collaborative filtering has been the main direction of recommendation algorithm research because it can take advantage of big data to find users with the most similar interests and recommend

TABLE 3: Effect of Embedding Size Comparison.

Data set	Metrics	16	32	64	128	256
Movielens-1M	**RMSE**	0.9735	1.0523	0.9675	0.9728	0.9812
Last.FM	**RMSE**	1.0615	1.0237	1.0165	1.0687	1.0722
Movielens-1M	**MAE**	0.8653	0.9017	0.8235	0.8452	0.8467
Last.FM	**MAE**	0.8426	0.8217	0.8095	0.8473	0.8676

items that they are interested in. However, there are two problems that are difficult to solve in collaborative filtering, namely, the cold start problem caused by too little initial data of new users and the sparse interaction data problem caused by too many items and too few items interacting with users. Recently, researchers have started to try to introduce knowledge graphs as auxiliary information to improve the performance of recommendation systems. In addition to alleviating the problems of cold start and data sparsity in traditional collaborative filtering methods, the introduction of knowledge graphs as auxiliary information for recommendation has the following advantages.The introduction of knowledge graphs can be combined with collaborative filtering methods to bring more accurate recommendations to users. The inferred implicit information contained in the knowledge graph enhances the ability of the knowledge graph-based recommendation system to explore users' latent interests and provide users with richer recommendation results.

However, traditional GNN-based recommendation algorithms are only able to handle regular topological graphs composed of a single type of nodes, while the data in the current network is not composed of only a single type of nodes. In addition, traditional GNNs only fuse the first-order neighborhood features of nodes and cannot capture the deeper structural relationships of nodes in the network. Therefore, when the data set is sparse and each node has only a very small number of neighbors, the recommendation quality of traditional GNN-based recommendation algorithms decreases significantly. In this paper, we propose a GNN recommendation model based on user-side information, which can effectively capture the potential interest of users through user relationship graph for recommendation, and fully exploit the large amount of association information among users. The traditional collaborative filtering idea is incorporated with the knowledge graph as the auxiliary information to fully utilize the potential information from both ends to improve the accuracy and diversity of the recommendation model. In addition, the interaction behaviors and opinions in the user-item graph are jointly captured. Different strengths of user-item interactions are considered, which indicate that various interactions contribute differently to user preferences (or item characteristics), and thus attention mechanisms are used to achieve this goal. The algorithm has been verified by public dataset, and it has better recommendation effect than several other classical recommendation algorithms.

6 Conflict of Interest Statement

All authors have no conflict and declare that: (i) no support, financial or otherwise, has been received from any organization that may have an interest in the submitted work ; and (ii) there are no other relationships or activities that could appear to have influenced the submitted work.

References

[1] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in *Proceedings of the ninth ACM international conference on web search and data mining*, 2016, pp. 153–162.

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

[3] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 549–558.

[4] X. Cao, G. Cong, and C. S. Jensen, "Mining significant semantic locations from gps data," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 1009–1020, 2010.

[5] W. Zhang, G. Ding, L. Chen, C. Li, and C. Zhang, "Generating virtual ratings from chinese reviews to augment online recommendations," *ACM Transactions on intelligent systems and technology (TIST)*, vol. 4, no. 1, pp. 1–17, 2013.

[6] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proceedings of the fourth ACM conference on Recommender systems*, 2010, pp. 135–142.

[7] X. Wang, W. Lu, M. Ester, C. Wang, and C. Chen, "Social recommendation with strong and weak ties," in *Proceedings of the 25th ACM international on conference on information and knowledge management*, 2016, pp. 5–14.

[8] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender systems handbook*. Springer, 2010, pp. 217–253.

[9] Y. Wei, X. Wang, Q. Li, L. Nie, Y. Li, X. Li, and T.-S. Chua, "Contrastive learning for cold-start recommendation," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 5382–5390.

[10] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, 2009.

[11] Y. Deldjoo, M. Schedl, B. Hidasi, Y. Wei, and X. He, "Multimedia recommender systems: Algorithms and challenges," in *Recommender systems handbook*. Springer, 2022, pp. 973–1014.

[12] W.-P. Lee and C.-Y. Ma, "Enhancing collaborative recommendation performance by combining user preference and trust-distrust propagation in social networks," *Knowledge-Based Systems*, vol. 106, pp. 125–134, 2016.

[13] K. Zhou, S.-H. Yang, and H. Zha, "Functional matrix factorizations for cold-start recommendation," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 2011, pp. 315–324.

[14] Y. Wei, X. Wang, L. Nie, X. He, and T.-S. Chua, "Graph-refined convolutional network for multimedia recommendation with implicit feedback," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 3541–3549.

[15] H. Li, "Deep learning for natural language processing: advantages and challenges," *National Science Review*, 2017.

[16] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir *et al.*, "Wide & deep learning for recommender systems," in *Proceedings of the 1st workshop on deep learning for recommender systems*, 2016, pp. 7–10.

[17] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[19] D. Yu, M. L. Seltzer, J. Li, J.-T. Huang, and F. Seide, "Feature learning in deep neural networks-studies on speech recognition tasks," *arXiv preprint arXiv:1301.3605*, 2013.
[20] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3642–3649.
[21] Y. Tan, Y. Hao, X. He, and X. Yang, "Selective dependency aggregation for action classification," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 592–601.
[22] F. Scarselli, S. L. Yong, M. Gori, M. Hagenbuchner, A. C. Tsoi, and M. Maggini, "Graph neural networks for ranking web pages," in *The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*. IEEE, 2005, pp. 666–672.
[23] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.
[24] Y. Wei, X. Wang, L. Nie, X. He, R. Hong, and T.-S. Chua, "Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 1437–1445.
[25] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," *Advances in neural information processing systems*, vol. 20, 2007.
[26] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems." in *IJCAI*, vol. 17. Melbourne, Australia, 2017, pp. 3203–3209.
[27] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.
[28] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 1225–1234.
[29] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.
[30] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," *arXiv preprint arXiv:1412.1602*, 2014.
[31] C. Cao, X. Liu, Y. Yang, Y. Yu, J. Wang, Z. Wang, Y. Huang, L. Wang, C. Huang, W. Xu *et al.*, "Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2956–2964.
[32] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.
[33] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A: statistical mechanics and its applications*, vol. 390, no. 6, pp. 1150–1170, 2011.
[34] L. Adamic and E. Adar, "How to search a social network," *Social networks*, vol. 27, no. 3, pp. 187–203, 2005.
[35] T. Zhou, L. Lü, and Y.-C. Zhang, "Predicting missing links via local information," *The European Physical Journal B*, vol. 71, pp. 623–630, 2009.