Graph Attention Networks for Neural Social Recommendation

Nan Mu[*†], Daren Zha[*], Yuanye He[*], Zhihao Tang[*]
[*]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*
[†]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China*
Email: {*munan,zhadaren,heyuanye,tangzhihao*}*@iie.ac.cn*

***Abstract*—In recent years, social recommendation is a research hotspot because it contains social network information which can effectively solve the problem of data sparsity and cold start. But the social recommendation task faces two problems: one is that how to accurately learn user latent vector and item latent vector from user-item interaction graph and social graph, the other is that how to depict the intrinsic and complex interaction between users and items. With the development of graph neural networks, node embedding is becoming more and more accurate on the graph. Besides neural collaborative filtering explores the interaction of users and items deeply. So in this paper, we propose a novel model: graph attention networks for neural social recommendation (GAT-NSR). This model adopts multi-head attention mechanism for message passing on the two graphs, which get user&item latent vector from different perspectives. And also we design a neural collaborative recommendation module to capture the inherent characteristics of user–item interaction behavior for recommendation. Finally, detailed experimental results on two real-world datasets clearly prove the effectiveness of our proposed model.**

***Index Terms*—social recommendation, collaborative filtering, multi-head attention, graph neural networks**

I. Introduction

With the rapid development of information technology and Internet, people have entered the age of information overload. Recommendation system is a very effective technique to help users find valuable information for themselves, and also it makes the information present to interested users. The most important method of recommendation system is Collaborative Filtering(CF), which makes recommendation based on the collaborative behaviors of users. Collaborative filtering has achieved remarkable results in many application scenarios, but the sparsity of user-item interaction is the key to limit its ability to improve performance. Fortunately, the rise of social networks has brought a new idea which is called social recommendation.

In the social recommendation, users' preferences are easily influenced by their social friends and social network data can effectively solve the problem of sparsity to improve recommendation performance. For this reason, social recommendation is a research hotspot all the world. The traditional methods SoRec [8], SoReg [9], SocialMF [10], TrustMF [12] and TrustSVD [11] are all based matrix factorization along with a series of supplementary information. Among the above methods, TrustSVD performs best because it fuses implicit feedback and explicit feedback information on the social recommendation. But with the development of graph neural networks, socialGCN [22] and GraphRec [26] achieve better model effect than the traditional methods. GCMC [25] is also an excellent recommendation system algorithm based Graph Convolutional Networks(GCN) [20] and GraphRec combines GCMC and social networks, which obtains good experimental results. However, most of the above methods do not take into account internal interaction behaviors between users and items, just as mentioned in the Neural Collaborative Filtering [5] method. So for the social recommendation task, in order to improve the effectiveness of the model, the following two problems need to be enhanced: (1) How to accurately depict the user latent vector and item latent vector from user-item interaction graph and social graph is a big challenge, and also we need to merge the two graphs which provide information from different perspectives. (2) How to obtain the intrinsic and deep interactive characteristics of users and items behaviors is also a problem.

Deep neural network has achieved remarkable preformance in many fields in the present era, which provide us broad ideas to solve the above problems. Graph neural networks aims to learn node embedding on the graph and achieves excellent results in most scenarios. Except for the well-know GCN method mentioned above, GraphSAGE [21] extends the graph convolution network to the inductive learing task. Graph attention networsk(GAT) [28] applies the attention mechanism to graph representation learning and performs well on node embedding. So we can incorporate the graph neural networks into the social recommendation system to get a more accurate representation of users and items. Besides, several researchers actively explored the approach [5], [29], [30] to combining deep neural network with collaborative filtering models. Neural Collaborative Filtering [5] provides a good idea to learn complex interactions of users with items deeply and inherently.

In this paper, we propose a novel model: graph attention networks for neural social recommendation (GAT-NSR), which bulid on the user-item interaction graph and social graph. Each user is affected not only by the connected items with the rating scores but also by the associated users, so we adopts the multi-head attention mechanism [27], [28] for message passing on each graph and merge the information of two graphs to generate the user latent vector. For the item latent vector, we also use the multi-head attention method to aggregate the

2375-0197/19/$31.00 ©2019 IEEE
DOI 10.1109/ICTAI.2019.00183

relevant users with rating sorces for each item. And then, the user latent vector and item latent vector are fed into a unique neural collaborative recommendation module to depict the deep complex relationships between users and items. And the goal of our model GAT-NSR is to predict the each user's rating to the unrated items. So the main contributions of this paper are as follows:

1) We propose a social recommendation model GAT-NSR to accurately learn the user&item latent vector by multi-head attention mechanism and deeply merge the information of user-item graph and social graph.
2) GAT-NSR adopts a neural collaborative recommendation module to depict the intrinsic and complex interactive characteristics of users and items behaviors.
3) We compare GAT-NSR with many state-of-the-art baselines on two real-world datasets. The experimental results clearly show the superiority of our proposed model.

II. Related Work

A. Collaborative Filtering

Collaborative filtering makes recommendations or predictions of the unknown preferences based on the collaborative behaviors. Usually, CF maps users and items to a same low-dimensional space and then the unknown ratings from users to items can be calculated by the similarity of user latent vector and item latent vector in this space [1], [2]. Although CF has a good effect on the recommendation system, data sparsity is a big problem to face in the real world applications. In order to solve this issue, SVD++ combines the users' explicit and implicit feedbacks to improve the performance of latent vector [3]. There is other research work [4] to address the sparsity problem. In recent years, with the development of deep learning, NeuMF [5] presents a new framework named Neural Collaborative Filtering to explore the deep and internal interactions between users and items.

B. Social Recommendation

In social networks, a user's behavioral preferences are influenced by the users associated with him and some classic research results [6], [7] have proved this view. Social recommendation then applies this theory to enhance the performance of recommendation system. SoRec [8] combines the user-item interaction matrix with the social network by probability matrix factorization. SoReg [9] proposes a matrix factorization framework with social regularization and solves the problem of trust-aware recommendation. SocialMF [10] applies the trust information and propagation of trust information into the matrix factorization for the social recommendation. TrustSVD [11] extends the method SVD++ [3] by further incorporating both the explicit and implicit influence of trusted users on the prediction of items for an active user. TrustMF [12] generate truster model and trustee model to map users into the same latent feature spaces but with different implications that can explicitly describe the feedback how users affect or follow the opinions of others.

C. Graph Representation Learning

Grpah representation learning is a recent research hotspot and its main goal is to learn node embedding accurately on graph structure data. Some classical graph repersentation methods [13]–[16] have achieved good results, but the new method, graph neural networks [17]–[19], achieves better performance for the graph data. Graph convolution neural network introduces the convolution operation into the graph structure and Kipf proposes a simple and efficient convolution kernel to aggregate information from the neighbors [20]. GraphSAGE [21] extends graph convolution network to the inductive learning task and generalizes the unknown nodes. Graph attention networks (GAT) [28] incorporates the attention mechanism into the message passing step and calculates the representation of each node following a self-attention strategy. More and more researchers are using the advantages of grpah representation learning to improve the performance of recommendation system. GCMC [25] views matrix completion as a link prediction problem on graphs and uses graph auto-encoder combining interaction data with side information. GraphRec [26] provides a principled approach to jointly capture interactions with opinions in the user-item graph and introduces the attention mechanism into the model.

III. The Proposed Model

In a social based recommender system, we have a userset $U = \{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \ldots, \mathbf{u}_N\}$ with N users and an itemset $V = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots, \mathbf{v}_M\}$ with M items. The ratings expressed by users on items are given in a user-item interaction matrix $\mathbf{R} \in \mathbb{R}^{N\times M}$. In the rating matrix, r_{ij} denotes that user $\mathbf{u}_i$ gives a rating to the item $\mathbf{v}_j$, but if $\mathbf{u}_i$ does not rate $\mathbf{v}_j$, $r_{ij} = 0$. We also define a set $C(i)$ representing a collection of items that have been rated by the user $\mathbf{u}_i$. Similarly, $B(j)$ represents the set of users which have interacted with the item $\mathbf{v}_j$. At the same time, we have a matrix $\mathbf{S} \in \mathbb{R}^{N\times N}$, which refers to the social network relationship between users. In the social graph, if $\mathbf{u}_k$ has a relation to $\mathbf{u}_i$, $s_{ki} = 1$, otherwise $s_{ki} = 0$. We define $N(i)$ to represent the set of users connected to the user $\mathbf{u}_i$. Given the user-item interaction matrix $\mathbf{R}$ and social matrix $\mathbf{S}$, our goal of the social recommendation is to predict each user's rating to the unrated items.

A. Overall Structure of the Proposed Framework

In this subsection, we introduce the overall architecture of the model GAT-NSR. Our model consists of three modules: initial embedding, user&item latent vector, neural collaborative recommendation.

The first part initial embedding is shown in Fig. 1. Following [5], the input layer is one hot embedding user i, item j and rating r, and through three different full connection layer, we get the initial embedding $\mathbf{u}_i \in \mathbb{R}^d$, $\mathbf{v}_j \in \mathbb{R}^d$ and $\mathbf{e}_r \in \mathbb{R}^d$, where d is the length of embedding vector.

The second part user&item latent vector is shown in Fig. 2. We have two different graphs: user-item graph and social graph, which represent different perspectives in the social recommendation. For user latent vector, through these two

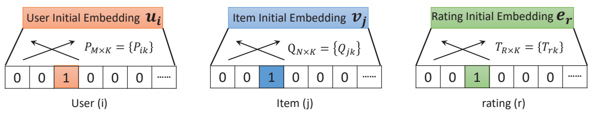
User Initial Embedding u_i
$P_{M\times K}=\{P_{ik}\}$
0 0 1 0 0 0 ......
User (i)
Item Initial Embedding v_j
$Q_{N\times K}=\{Q_{jk}\}$
0 0 1 0 0 0 ......
Item (j)
Rating Initial Embedding e_r
$T_{R\times K}=\{T_{rk}\}$
0 0 1 0 0 0 ......
rating (r)

Fig. 1. Initial embedding.

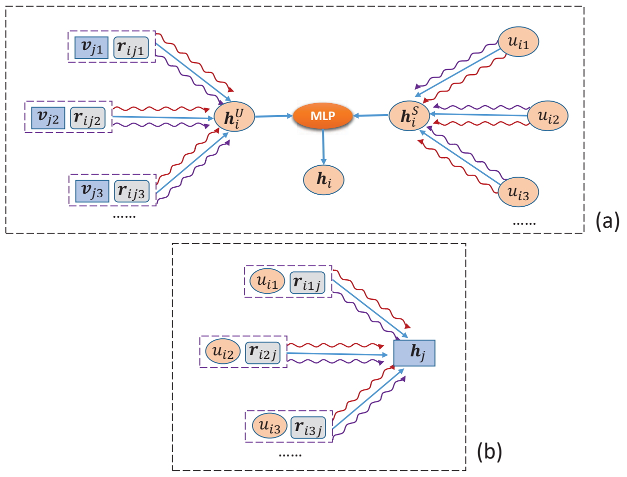

Fig. 2. User&item latent vector. (a) User latent vector $\mathbf{h}_i$ is from two parts: $\mathbf{h}_i^U$ is generated by the aggregation of connected items and corresponding rating scores for $\mathbf{u}_i$ on the user-item graph and $\mathbf{h}_i^S$ is generated by the aggregation of associated users on the social graph. (b) Item latent vector $\mathbf{h}_j$ is from the aggregation of connected users with rating scores for item $\mathbf{v}_i$ on the user-item graph. It's worth noting that $\mathbf{r}_{ij}$ is the rating initial embedding $\mathbf{e}_r$ in Fig. 1.

graphs we can enhance the accuracy of node representation and effectively solve the cold start problem. Graph representation learning can be viewed as a type of message passing on the graph. In the user-item interaction graph, each user is affected by the items connected with it, and the rating of this user on each item is also a very impotant factor, so item and rating jointly determine the representation of the user. similarly, the item latent vetor is affected by the directly connected users and ratings. In the social graph, each user is influenced by his associated users. In this part, we use multi-head attention [27], [28] for message passing, to learn features in different representation spaces.

Finally, the third module neural collaborative recommendation is shown in Fig. 3. This module takes user&item latent vector as input, and then has collaboration Layer, neural collaborative filtering layers and output layer. Next, we will introduce the second and third module in detail.

B. User&Item Latent Vector

This module aims to learn user latent vector $\mathbf{h}_i$ from user-item interaction graph and social graph, and also item latent vector $\mathbf{h}_j$ from user-item interaction graph. The initial embedding $\mathbf{u}_i$, $\mathbf{v}_j$ and $\mathbf{e}_r$ are from the first module. Then we will introduce the representation learning method of user latent vector.

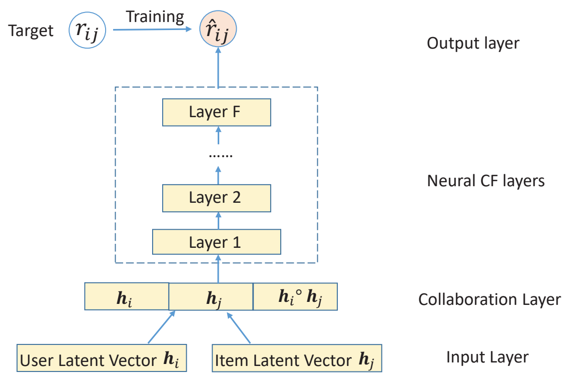

Fig. 3. Neural collaborative recommendation.

1) User Latent Vector: As mentioned above, how to inherently combine the information of the two graphs is a big challenge in the social recommendation. In our model, for each user, we firstly aggregate all the features of directly connected items and corresponding rating scores in the user-item graph, and then we get the embedding vector $\mathbf{h}_i^U$. The second aggregation is the message passing of the associated users in the social graph and we can get another embedding vector $\mathbf{h}_i^S$. Finally, we combine these two vetcors to form a new embedding vector, that is user latent vector $\mathbf{h}_i$.

In the user-item graph, rating score represents the user's preference for the item, so in order to accurately depict the user, we use a Multi-Layer Perceptron (MLP) to combine the initial embedding item $\mathbf{v}_j$ and rating $\mathbf{e}_r$, as shown in Eq.1:

$$\mathbf{q}_{jr} = g_u(\mathbf{v}_j \oplus \mathbf{e}_r) \tag{1}$$

$\mathbf{v}_j \oplus \mathbf{e}_r$ is the concatenation of v_j and e_r and also g_u stands for a type of MLP to integrate the user and rating information. $\mathbf{q}_{jr} \in \mathbb{R}^d$ is the final combination embedding and d is the length of the embedding vector.

Now it's the message passing process as shown in Fig. 2. Firstly we use the self-attention method:

$$e_{ij} = f(\mathbf{W}_u\mathbf{u}_i, \mathbf{W}_u\mathbf{q}_{jr}) \tag{2}$$

e_{ij} is the attention coefficients which indicate the importance of $\mathbf{q}_{jr}$ to user $\mathbf{u}_i$ and $\mathbf{W}_u \in \mathbb{R}^{d'\times d}$ represents a linear transformation, in which d' is the length of output vector. In addition, f is a function mapping from $\mathbb{R}^{d'} \times \mathbb{R}^{d'}$ to $\mathbb{R}$. It's worth noting that the matrix $\mathbf{W}_u$ and function f are shared by every node. In the user-item interaction graph, we only need to focus on the set of items that are directly connected to the user. Just like we defined above, the set $C(i)$ represents a collection of items that have been rated by the user $\mathbf{u}_i$ and we pass the attention coefficients through a softmax function to get the final weight of each node:

$$\alpha_{ij} = softmax(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k\in C(i)} exp(e_{ik})} \tag{3}$$

In our experiments, we use a two-layer neural network to implement the attention coefficients e_{ij}:

$$e_{ij} = \mathbf{W}_2^T \cdot \sigma(\mathbf{W}_1 \cdot (\mathbf{W}_u \mathbf{u}_i \oplus \mathbf{W}_u \mathbf{q}_{jr}) + \mathbf{b}_1) + \mathbf{b}_2 \qquad (4)$$

So combine the Eq.3 and Eq.4, we can obtain the attention weights α_{ij}, then we can get the user embedding vector $\mathbf{h}_i^U$ from the aggregation of items' feature and rating scores:

$$\mathbf{h}_i^U = \sigma(\sum_{j \in C(i)} \alpha_{ij} \mathbf{W}_u \mathbf{q}_{jr}) \qquad (5)$$

To make the model work better, we use multi-head attention [27], [28], which can get more information from different perspectives:

$$\mathbf{h}_i^U = \mathop{\Big\|}_{k=1}^{K} \sigma(\sum_{j \in C(i)} \alpha_{ij}^k \mathbf{W}_u^k \mathbf{q}_{jr}) \qquad (6)$$

where $\|$ represents concatenation of the vectors, and α_{ij}^k, $\mathbf{W}_u^k$ denote the k-th attention weights and linear transformation's weight matrix.

Next, we will discuss how to generate the social embedding vector $\mathbf{h}_i^S$ from the social graph. There are no ratings between users, just relationships in this graph. So the social embedding is shown in the Fig. 3, and we also use the multi-head attention for message passing. We use a two-layer neural network to calculate attention coefficients:

$$e_{ij} = \mathbf{W}_4^T \cdot \sigma(\mathbf{W}_3 \cdot (\mathbf{W}_s \mathbf{u}_i \oplus \mathbf{W}_s \mathbf{u}_j) + \mathbf{b}_3) + \mathbf{b}_4 \qquad (7)$$

$\mathbf{u}_i$ and $\mathbf{u}_k$ are connected users and $\mathbf{W}_s \in \mathbb{R}^{d' \times d}$ represents a shared linear transformation in the social graph. $\oplus$ also denotes the concatenation of two vectors. The final attention weights are shown in Eq. 8:

$$\alpha_{ij} = softmax(e_{ij}) = \frac{exp(e_{ij})}{\sum_{o \in N(i)} exp(e_{io})} \qquad (8)$$

as stated earlier, $N(i)$ is the collection of users which have relationship with $\mathbf{u}_i$, Then we use the multi-head attention to implement the message passing process:

$$\mathbf{h}_i^S = \mathop{\Big\|}_{k=1}^{K} \sigma(\sum_{o \in N(i)} \alpha_{io}^k \mathbf{W}_s^k \mathbf{u}_o) \qquad (9)$$

And now we've generated the two parts of the user latent vector: user embedding vector $\mathbf{h}_i^U$ in the user-item graph and social embedding vector $\mathbf{h}_i^S$ in the social graph. These two vectors represent different perspectives on learning the representation of user $\mathbf{u}_i$,so in order to merge the information of two vectors we use a Multi-Layer Perceptron (MLP):

$$\mathbf{h}_i = g_{us}(\mathbf{h}_i^U \oplus \mathbf{h}_i^S) \qquad (10)$$

g_{us} is the MLP module and $\mathbf{h}_i$ is the final user latent vector which deeply integrates user-item graph and social graph.

2) Item Latent Vector: In our model, we learn the item latent vector $\mathbf{h}_j$ from the user-item interaction graph as shown in Fig. 3. For each item $\mathbf{v}_j$, we aggregate the users who have rated this item and corresponding rating scores. So this process is very similar to the user embedding vector method as described in Fig. 3. Firstly, we integrate the features of user and rating:

$$\mathbf{p}_{ir} = g_v(\mathbf{u}_i \oplus \mathbf{e}_r) \qquad (11)$$

$\mathbf{u}_i$ and $\mathbf{e}_r$ are initial embedding described in the first module and g_v is also a Multi-Layer Perceptron (MLP). The attention coefficients can be calculated as:

$$e_{ij} = \mathbf{W}_6^T \cdot \sigma(\mathbf{W}_5 \cdot (\mathbf{W}_v \mathbf{v}_j \oplus \mathbf{W}_v \mathbf{p}_{ir}) + \mathbf{b}_5) + \mathbf{b}_6 \qquad (12)$$

$\mathbf{W}_v$ is the shared linear transformation in this process and $\mathbf{W}_v \mathbf{v}_j \oplus \mathbf{W}_v \mathbf{p}_{ir}$ is the fuse of item $\mathbf{v}_j$ and user $\mathbf{u}_i$ with rating $\mathbf{e}_r$. And then it's natural to get the attention weights:

$$\alpha_{ij} = softmax(e_{ij}) = \frac{exp(e_{ij})}{\sum_{t \in B(j)} exp(e_{tj})} \qquad (13)$$

$B(j)$ is the collection users directly connected with the item $\mathbf{v}_j$. In the end through the multi-head attention we can get the item latent vector $\mathbf{h}_j$:

$$\mathbf{h}_j = \mathop{\Big\|}_{k=1}^{K} \sigma(\sum_{i \in B(j)} \alpha_{ij}^k \mathbf{W}_v^k \mathbf{p}_{ir}) \qquad (14)$$

Through the above statement, we finally get the user latent vector $\mathbf{h}_i$ and item latent vector $\mathbf{h}_j$, so next we will introduce the third module of our framewok: neural collaborative recommendation.

C. Neural Collaborative Recommendation

When we generate the user latent vector $\mathbf{h}_i$ and item latent vector $\mathbf{h}_j$, how to design an effective structure to capture the internal relationship of user–item interaction behavior for recommendation is a big challenge. Xiangnan He proposed a method neural collaborative filtering [5] to solve this problem. Based on his thought, we design a neural collaborative recommendation module shown in Fig. 3. This module contains four layers: input layer, collaboration layer, neural collaborative filtering layers and output layer, and then we will describe these four layers in detail.

1) Input Layer: The input layer contains user latent vector $\mathbf{h}_i$ and item latent vector $\mathbf{h}_j$, which are described detailedly in the second module.

2) Collaboration Layer: In this layer, we explicitly fuse $\mathbf{h}_i$ and $\mathbf{h}_j$ to capture the shallow linear user–item interaction before the neural collaborative filtering layers:

$$\mathbf{X}_{ij} = [\mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_i \odot \mathbf{h}_j] \qquad (15)$$

$\mathbf{h}_i \odot \mathbf{h}_j$ is the element-wise of two vectors, which is often used in shallow models. $[\mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_i \odot \mathbf{h}_j]$ is the concatenation of these three vectors. So this layer carries more information about the shallow collaborative user–item interaction behavior and we put $\mathbf{X}_{ij}$ into the neural collaborative filtering layers to make the learning process more accurate and rapid.

3) Neural Collaborative Filtering Layers & Output Layer: Neural collaborative filtering layers are the core of the model to learn deep and internal characteristic of user–item interaction behavior. We use a standard Multi-Layer Perceptron (MLP) to endow the model a large level of flexibility and non-linearity to learn the relationships between users and items, because These multilayered feedforward networks are useful as they are demonstrated to be able to approximate any measurable function. We define the neural CF layers as following:

$$\begin{aligned} \mathbf{X}_{ij}^1 &= \sigma(\mathbf{W}_n^1 \mathbf{X}_{ij} + \mathbf{b}_n^1) \\ \mathbf{X}_{ij}^2 &= \sigma(\mathbf{W}_n^2 \mathbf{X}_{ij}^1 + \mathbf{b}_n^2) \\ &\cdots \\ \mathbf{X}_{ij}^l &= \sigma(\mathbf{W}_n^l \mathbf{X}_{ij}^{l-1} + \mathbf{b}_n^l) \end{aligned} \tag{16}$$

$\mathbf{X}_{ij}^l$ is the output of the lth-layer and $\mathbf{W}_n^l, \mathbf{b}_n^l$ is the weight matrix and bias of the lth-layer.

So finally we can get the predicted rating $\hat{r}_{ij}$ from user $\mathbf{u}_i$ to item $\mathbf{v}_j$:

$$\hat{r}_{ij} = \mathbf{W}_r^T \mathbf{X}_{ij}^l \tag{17}$$

D. Model Training

During the model training, since we focus on the rating prediction task in the recommendation system, we use a common objective function to estimate the model parameters of GAT-NSR:

$$\arg\min_{\Theta} \sum_{(u,i)\in D} (r_{ij} - \hat{r}_{ij})^2 \tag{18}$$

Θ is the parameters set of the model GATRec and D represents the collections that users have rating scores on items in the training data. r_{ij} is the real rating in the dataset and $\hat{r}_{ij}$ is the predicted rating generated from our model.

All the parameters in the above loss function are differentiable. In practice, we implement our proposed method with pytorch[1] to train model parameters. We also use mini-batch gradient decent, which divides training data into several batches and updates parameters by each batch. The mini-batch is an effective regularization method and also reduces the memory requirement to train the model.

IV. Experiment

A. Experimental Settings

1) Datasets: In our experiments, we choose two real-world public datasets: FilmTrust[2] and Epinions[3]. All the two datasets contain user-item interaction information and social relationships by active users. The ratings in FilmTrust are from 0.5 to 4.0 with step 0.5, so FilmTrust has eight different ratings: {0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0}. The ratings in Epinions are integers from 1 to 5: {1, 2, 3, 4, 5}. The statistics of datasets are illustrated in TABLE I. As described in the Fig. 1, we randomly initialize the embedding of rating scores for the two datasets.

[1]https://pytorch.org/
[2]www.librec.net/datasets.html
[3]www.trustlet.org/downloaded_epinions.html

TABLE I
General statistics of the FilmTrust and Epinions

statistics	**FilmTrust**	**Epinions**
Users	1,508	40,163
Items	2,071	139,738
Ratings	35,497	664,824
Density(Ratings)	1.14%	0.051%
Social Relations	1,853	487,183
Density(Social Relations)	0.42%	0.029%

2) Evaluation Metrics: We used two classical metrics to evaluate accuracy of rating prediction: mean absolute error (MAE) and root mean square error (RMSE):

$$\begin{aligned} MAE &= \frac{\sum_{i,j} |r_{ij} - \hat{r}_{ij}|}{N} \\ RMSE &= \sqrt{\frac{\sum_{(i,j)} (r_{ij} - \hat{r}_{ij})^2}{N}} \end{aligned} \tag{19}$$

where N is the number of test ratings. Smaller values of MAE and RMSE indicate better predictive accuracy.

3) Baselines: We choose the following baselines compared with our model:

- **PMF** [1]. Probabilistic Matrix Factorization is a basic recommendation system model with only user-item rating matrix.
- **SoRec** [8]. SoRec combines the user-item interaction matrix with the social network information by matrix factorization.
- **SoReg** [9]. SoReg implements the matrix factorization with social regularization and solves the problem of trust-aware recommendation.
- **SocialMF** [10]. SocialMF applies the trust information and propagation of trust information into the matrix factorization.
- **TrustMF** [12]. TrustMF generates truster and trustee model to map users into the same latent feature spaces but with different implications.
- **TrustSVD** [11]. TrustSVD incorporates both the explicit and implicit influence of trusted users on the prediction of items for an active user.
- **NeuMF** [5]. NeuMF presents Neural Collaborative Filtering framework to explore the deep and internal interactions between users and items without social relation information.
- **GraphRec** [26]. GraphRec provides a state-of-the-art recommender system to jointly capture interactions with opinions in the user-item graph and introduces the attention mechanism into the model.

4) Parameter Settings: In view of the effectiveness and efficiency, we finally set the parameters of our model with the following values: For each dataset, we use 80% as training set to learn model parameters, 10% as validation set to verify the validity of the parameters and 10% as test set to generate the final experimental performance; we adopt the data preprocessing method as described in TrustSVD [11]; we set batch size to 128 and embedding size to 64; the learning rate of the whole model is 0.001 and the activation function is ReLU; we employe three hidden layers for the Multi-Layer Perceptron in Eq.1, Eq.10 and Eq.11; we adopt Gaussian distribution to randomly initialize model parameters, where the mean and standard deviation is 0 and 0.1; for the baseline algorithms, we follow the articles to get the best performance.

B. Performance

Tabel. II shows the perfomance comparison of our model and baseline methods. $*$ represents the best performace except for our method and the boldface represents the best result among all the algorithm. By careful comparison, we can find the following conclusions:

- PMF performs worst on both datasets because it's a traditional matrix factorization model without social relation information. SoRec, SoReg, SocialMF, TrustMF and TrustSVD all fuse the user-item matrix and social matrix so they perform better than PMF. NeuMF don't include social information but it uses advanced deep neural network structure to simulate the interaction of users and items, so it has a better experimental results than matrix factorization. The comparison results prove the significance of social information and deep neural networks for recommendation system
- TrustSVD has a very good performance among baseline algorithms. It extends the method SVD++ by further incorporating both the explicit and implicit influence of trusted users on the prediction of items for an active user.
- GraphRec is a state-of-the-art recommender system and shows best experimental performance than other baseline algorithms. Analysis from his model architecture, we can discover that graph neural networks plays an important role in the recommendation system beacuse it allows for a more accurate node embedding in the graph. At the same time, GraphRec uses attention mechanism for user-item graph and social graph.
- It is clear from the table II that our method performs best among all the algorithms. Compared to GraphRec, our approach also uses graph neural networks and attention mechanism with the social information. But more importantly, we have two important improvements: one is that we use multi-head attention for message passing to get more information from different perspectives, the other is that we use neural collaborative recommendation to capture the internal relationships of user–item interaction behavior. We will analyze the impact of these modules for our framework in detail in the next section.

C. Model Analysis

We now study the performance of our approach under different parameter settings. We mainly analyze the multi-head attention part and neural collaborative recommendation part.

1) The impact of multi-head attention: Transformer framework firstly proposed the multi-head attention mechanism, which is widely used in many research fields. Because traditional attention method learn concerns from only one perspective, the information learned from the model is not perfect. Graph attention networks(GAT) [28] applies multi-head attention to the graph neural networks and achieves excellent experimental results. GAT proves the effectiveness of multi-head attention and naturally we introduce it into the social recommendation.

In our model, as shown in Fig. 2, user aggregation, item aggregation in user-item interaction graph and social aggregation in social graph all use multi-head attention mechanism. Eq. 6, Eq. 9 and Eq. 14 introduced the message passing process in detail. In this method, the number of attention heads is an important parameter, which has a crucial impact on the performance of the model. And then we will compare the effects of different quantities of heads on the results. It's worth noting that different quantities of heads come with different dimensions of attention layer. As mentioned in the parameter settings, the embedding size=64 for a single-head attention but when the number of attention heads is k, the embedding size = 64/k for each head. Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality [27]. In our experiment, we set the number of heads k=1, 2, 4, 8, 16 and the experimental performance is shown in Tabel. III, **d** represents the dimension of attention layer.

From the table, we can clearly see that experimental performance of multi-head attetion is better than single-head attention. In our experimental environment, when the number of heads is 4, the performance is best for the two datasets. however, when k=16, the results are worse than the baseline GraphRec bacause the dimension of attention layer is only 4, which is difficult to learn all the useful information in the two graphs. So for the multi-head attetion mechanism, we need to find a balance between the number of attention heads and the dimension of attention layer to make the experiment performance best.

2) The impact of neural collaborative recommendation: The core technology of recommendation system is collaborative filtering, which purpose is to mine interactions behavior between users and items. The classical methods still resort to matrix factorization and apply an inner product on the latent features of users and items. NeuMF [5] points out the shortcomings of this approach and proposes a deep learning framework neural collaborative filtering to explore deep and internal interactions between users and items. As shown in Fig. 3, we adopt a collaboration Layer to integrate not only user latent vector $\mathbf{h}_i$ and item latent vector $\mathbf{h}_j$ but also the inner product of the two latent vectors. To verify the importance of the neural collaborative recommendation module, we design the following two modules for comparison:

– **Inner product**. We replace the neural collaborative recommendation module with inner product of user latent vector $\mathbf{h}_i$ and item latent vector $\mathbf{h}_j$. The inner product is

TABLE II
PERFOMANCE COMPARISON OF OUR MODEL AND BASELINE METHODS

Datasets	**Metrics**	Algorithms									
		PMF	**SoRec**	**SoReg**	**SocialMF**	**TrustMF**	**TrustSVD**	**NeuMF**	**GraphRec**	**GAT-NSR**	**Improve**
FilmTrust	MAE	0.714	0.628	0.668	0.638	0.631	0.607	0.656	0.599*	**0.594**	**0.8%**
	RMSE	0.949	0.810	0.875	0.837	0.810	0.787	0.845	0.783*	**0.756**	**3.4%**
Epinions	MAE	0.909	0.882	0.932	0.825	0.818	0.804	0.874	0.787*	**0.778**	**1.1%**
	RMSE	1.197	1.114	1.232	1.070	1.069	1.043	1.122	1.039*	**1.019**	**1.9%**

TABLE III
PERFOMANCE COMPARISON OF DIFFERENT QUANTITIES OF HEADS

k	**d**	**FilmTrust**		**Epinions**	
		MAE	RMSE	MAE	RMSE
1	64	0.602	0.779	0.791	1.045
2	32	0.598	0.762	**0.778**	1.026
4	16	**0.594**	**0.756**	**0.778**	**1.109**
8	8	0.606	0.784	0.805	1.055
16	4	0.629	0.811	0.817	1.069

a classical method for matrix factorization in recommendation system.

– **NCR–IP** [8]. NCR–IP stands for **n**eural **c**ollaborative **r**ecommendation removing **i**nner **p**roduct. In the collaboration Layer, we combine $\mathbf{h}_i$, $\mathbf{h}_j$ and $\mathbf{h}_i \odot \mathbf{h}_j$ to fuse the shallow information(inner product) into the model. So NCR–IP removes the inner product part and compares to the original model.

The comparison results are shown in the Table IV, and NCR representes our original model: neural collaborative recommendation. From the table, we can find that the classical inner

TABLE IV
PERFOMANCE COMPARISON OF DIFFERENT RECOMMENDATION MODULES

module	**FilmTrust**		**Epinions**	
	MAE	RMSE	MAE	RMSE
Inner product	0.612	0.791	0.799	1.051
NCR–IP	0.601	0.762	0.780	1.028
NCR	**0.594**	**0.756**	**0.778**	**1.109**

product module performs worst among the three methods, which demonstrates the importance of deep neural networks on the collaborative filtering technique. Deep neural networks can capture the interaction of users and items more accurately in the recommendation system. Comparing the results of NCR–IP and NCR methods, when we add the information of inner product $\mathbf{h}_i \odot \mathbf{h}_j$ to the Neural CF layers, the experimental results perform better. This illustrates the classical inner product is an effective supplement for the deep neural networks. So when we combine the deep model with the shallow model, the framework will perform better.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a graph attention networks for social neural recommendation. Our model integrates the information of user-item interaction graph and social graph to obtain the user latent vector accurately. At the same time, we use the multi-head attention mechanism for message passing on the two graphs, which learns more comprehensive information from different perspectives. And then, we use a neural collaborative recommendation module to mine interactions behavior between users and items deeply and inherently. The final experiments are conducted on two real-world datasets, which demonstrats the effectiveness of our proposed model. Through further analysis of model parameters, we verify the significance of multi-head attention mechanism and neural collaborative recommendation module.

In the future, to enhance the performance of the recommendation system, we will use the side information of users and items. In the real world, users contain rich properties as well as items, which are an important addition to characterizing the interaction for recommendation task. Moreover, in the recommendation problem, we need to consider the user history behavior record, and at this time, the order of behavior time will affect the result of our recommendation. So in many scenarios, user-item interaction graph and social graph are dynamic. How to make recommendations in a temporal dynamic process is an interesting topic.

ACKNOWLEDGMENT

This work is partially supported by National Natural Science Foundation of China grant (No. U163620068) and National Key Research and Development Program of China(No. 2016YFB0800504).

REFERENCES

[1] Mnih A, Salakhutdinov R R. Probabilistic matrix factorization[C]//Advances in neural information processing systems. 2008: 1257-1264.

[2] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems[J]. Computer, 2009 (8): 30-37.

[3] Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]//Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008: 426-434.

[4] Rendle S. Factorization machines[C]//2010 IEEE International Conference on Data Mining. IEEE, 2010: 995-1000.

[5] He X, Liao L, Zhang H, et al. Neural collaborative filtering[C]//Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2017: 173-182.

[6] Marsden P V, Friedkin N E. Network studies of social influence[J]. Sociological Methods & Research, 1993, 22(1): 127-151.
[7] McPherson M, Smith-Lovin L, Cook J M. Birds of a feather: Homophily in social networks[J]. Annual review of sociology, 2001, 27(1): 415-444.
[8] Ma H, Yang H, Lyu M R, et al. Sorec: social recommendation using probabilistic matrix factorization[C]//Proceedings of the 17th ACM conference on Information and knowledge management. ACM, 2008: 931-940.
[9] Ma H, Zhou D, Liu C, et al. Recommender systems with social regularization[C]//Proceedings of the fourth ACM international conference on Web search and data mining. ACM, 2011: 287-296.
[10] Jamali M, Ester M. A matrix factorization technique with trust propagation for recommendation in social networks[C]//Proceedings of the fourth ACM conference on Recommender systems. ACM, 2010: 135-142.
[11] Guo G, Zhang J, Yorke-Smith N. TrustSVD: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings[C]//Twenty-Ninth AAAI Conference on Artificial Intelligence. 2015.
[12] Yang B, Lei Y, Liu J, et al. Social collaborative filtering by trust[J]. IEEE transactions on pattern analysis and machine intelligence, 2016, 39(8): 1633-1647.
[13] Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations[C]//Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2014: 701-710.
[14] Grover A, Leskovec J. node2vec: Scalable feature learning for networks[C]//Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2016: 855-864.
[15] Tang J, Qu M, Wang M, et al. Line: Large-scale information network embedding[C]//Proceedings of the 24th international conference on world wide web. International World Wide Web Conferences Steering Committee, 2015: 1067-1077.
[16] Wang D, Cui P, Zhu W. Structural deep network embedding[C]//Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2016: 1225-1234.
[17] Zhang Z, Cui P, Zhu W. Deep learning on graphs: A survey[J]. arXiv preprint arXiv:1812.04202, 2018.
[18] Zhou J, Cui G, Zhang Z, et al. Graph neural networks: A review of methods and applications[J]. arXiv preprint arXiv:1812.08434, 2018.
[19] Wu Z, Pan S, Chen F, et al. A comprehensive survey on graph neural networks[J]. arXiv preprint arXiv:1901.00596, 2019.
[20] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.
[21] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs[C]//Advances in Neural Information Processing Systems. 2017: 1024-1034.
[22] Wu L, Sun P, Hong R, et al. SocialGCN: An Efficient Graph Convolutional Network based Model for Social Recommendation[J]. arXiv preprint arXiv:1811.02815, 2018.
[23] Bao H, Wu L, Sun P. Contextual Attention Model for Social Recommendation[C]//Pacific Rim Conference on Multimedia. Springer, Cham, 2018: 630-641.
[24] Wu L, Sun P, Hong R, et al. Collaborative Neural Social Recommendation[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018.
[25] Berg R, Kipf T N, Welling M. Graph convolutional matrix completion[J]. arXiv preprint arXiv:1706.02263, 2017.
[26] Fan W, Ma Y, Li Q, et al. Graph Neural Networks for Social Recommendation[C]//The World Wide Web Conference. ACM, 2019: 417-426.
[27] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in neural information processing systems. 2017: 5998-6008.
[28] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks[J]. arXiv preprint arXiv:1710.10903, 2017.
[29] Sedhain S, Menon A K, Sanner S, et al. Autorec: Autoencoders meet collaborative filtering[C]//Proceedings of the 24th International Conference on World Wide Web. ACM, 2015: 111-112.
[30] Cheng H T, Koc L, Harmsen J, et al. Wide & deep learning for recommender systems[C]//Proceedings of the 1st workshop on deep learning for recommender systems. ACM, 2016: 7-10.