Collaborative Denoising Graph Attention Autoencoders for Social Recommendation

Nan Mu*†, Daren Zha*†, Lin Zhao*, Rui Gong*
*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,zhaolin1,gongrui}@iie.ac.cn

***Abstract*—In recent years, social recommendation has attracted more and more attention from researchers, because it can effectively solve the problems of data sparsity and cold start. But social recommendation faces two problems: The first is how to deeply integrate social information with user-item preference information to obtain accurate user and item latent vectors. The second problem is how to generate a robust top-N recommendation model from the implicit feedback information. To solve these problems, we propose a novel model: collaborative denoising graph attention autoencoders for social recommendation(CDGAAE). This model uses the currently popular graph neural networks to fuse the interaction preference information graph and social information graph through a multi-head attention message passing mechanism. At the same time, this model delicately merges graph neural networks with denoising autoencoders, which uses the corrupt versions of the original data to make the model more robust and generalized. Finally, we conduct comparative experiments of our model with other baseline algorithms on two real world datasets, and the experimental results prove the superiority of our model.**

***Index Terms*—Denoising Graph Attention Autoencoders , Social Recommendation , Implicit Feedback , Top-N Recommendation**

I. Introduction

Recently, with the boom of the Internet and social media, social recommendation is a hot topic in academia and industry. Previous works [1], [2] incorporate social information into the recommendation system from different perspectives and these works have achieved good performance. However, the above methods are shallow models and they cannot deeply merge social information with user-item interaction information. In addition, the robustness and generalization of the recommendation system is also crucial, so how to resist the interference of noise is also a problem we need to consider. In real world, the user's explicit rating information for items is difficult to obtain and what we can usually get is the implicit feedback data (click, browse, etc). At the same time, for ordinary users, they prefer to get a list of items that are most interesting to them. So for the social-based top-K recommendation system, we face the following two problems: The first is how to deeply integrate social information with user-item preference information to obtain accurate user and item latent vectors. The second problem is how to generate a robust top-k recommendation model from the implicit feedback information.

DOI reference number: 10.18293/SEKE2020-105

The rapid development of deep learning technology has brought us new ideas for solving the above problems. Graph neural networks [9] generate the accurate representation for nodes through deeply exploring the graph structure, which has achieved significant results. The denoising autoencoders [6] provide a robust feature representation algorithm by using the corrupted version of the original data. Collaborative neural filtering [14] deeply learns the interaction behaviors between users and items.

So in this paper, we propose a novel model: Collaborative Denoising Graph Attention Autoencoders for Social Recommendation (CDGAAE). We deeply integrate the graph neural networks with the denoising autoencoders. For the user-item implicit preference interaction matrix, we use its corrupted version as input to train our model. CDGAAE includes graph attention encoder and collaborative neural filtering decoder. The encoder performs the message passing process based multi-head attention mechanism on the user-item preference graph and the social network graph. Then we can obtain the user latent vector by deeply merging the results of these two graphs, and the item latent vector is generated from the user-item preference graph. The decoder designs a multi-layer neural collaborative recommendation module which takes the latent vectors of users and items as input, and then the decoder reconstructs the original user-item preferences, which can enhance the robustness of the model. For model learning process, since the original data is implicit feedback data, we use a binary cross-entropy loss function and a stochastic gradient descent optimization method. So the main contributions of this article are as follows:

1) We propose a novel model Collaborative Denoising Graph Attention Autoencoders for Social Recommendation (CDGAAE), which can deeply integrate social information and user-item preference information to accurately represent users and items.
2) The CDGAAE model delicately integrates graph neural networks and denoising autoencoders, which making the model more robust and generalized. At the same time, it can make top-K recommendations from implicit feedback data.
3) We compare CDGAAE and many baseline algorithms on two real world datasets, and the experimental results prove the superiority of our model.

II. Related Work

A. Social Recommendation

With the booming development of online social media, social recommendation technology has attracted more and more researchers' attention. SocialMF [1] incorporates the mechanism of trust propagation into the social recommendation model because the trust propagation has been shown to be a crucial phenomenon in the social sciences. In order to further improve the performance of recommendation, TrustSVD [2] integrates explicit and implicit influence into the model at the same time. And then the top-K recommendation is also an important research hotspot in social recommendation. SBPR [3] extends the classical BPR [4] method with observation that users tend to assign higher rankings to their friends' favorite items. SPF [5] develops a social poisson factorization method to closely combine ratings with social information.

B. Denoising Autoencoders

The Denoising Autoencoder (DAE) [6] is an extension of the classical autoencoder and the purpose of DAE is to reconstruct the raw input data χ from its (partially) corrupted version $\tilde{\chi}$. The common corrupted methods consist of the additive Gaussian noise and the mask-out/drop-out noise. With this setup, DAE can generate more robust features than the classical autoencoder. The DAE model is widely used in recommendation systems to improve the performance of the framework. Collaborative denoising auto-encoders(CDAE) [7] proposes a top-N recommendation algorithm which utilizes the idea of DAE and the CDAE model is a generalization of several well-known collaborative filtering models but with more flexible components. And then to tackle the data sparsity and cold start problems, the Trust-aware Collaborative Denoising AutoEncoder (TDAE) [8] learn compact and effective representations from both rating and trust data for top-N recommendation.

C. Graph Neural Networks

In recent years, graph neural networks [9] have developed rapidly in the field of representation learning and make remarkable achievements. The representation of each node on the graph structure data has always been a research hotspot and graph neural networks achieve better performance in terms of accuracy and speed than the classical network repersentation methods [10] in many application scenarios. Due to the advantages of graph neural networks more and more recommendation systems adopt these algorithms. GCMC [12] provides a framework which considers matrix completion as a link prediction task and leverage graph autoencoders combining interaction data with side information. GraphRec [13] 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 the classical recommendation system, we usually have a userset U with N users $\{u_1, u_2, \ldots, u_N\}$ and itemset V with M items $\{v_1, v_2, \ldots, v_M\}$. We also have a matrix $\mathbf{R} \in \mathbb{R}^{N \times M}$ that represents the users' preferences for the items. And in this paper we focus on the implicit feedback information because the explicit feedback is often hard to get. In the preference matrix $\mathbf{R}$, if there is an interaction between the user u_i and the item v_j, $r_{ij} = 1$, otherwise $r_{ij} = 0$. And then we also 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 u_k has a relation to u_i, $s_{ki} = 1$, otherwise $s_{ki} = 0$.

Now we have the implicit preference matrix $\mathbf{R}$ and the social matrix $\mathbf{S}$, so the goal of our social recommendation system is to pick a top-k list of the most interesting items from the unobserved item set for each user.

A. Overall Structure of the Proposed Framework

In this part, we will introduce the overall structure of our proposed model Collaborative Denoising Graph Attention Autoencoders(CDGAAE). The classical denoising autoencoder has been described in the related work II-B. The common corrupted methods consist of the additive Gaussian noise and the multiplicative mask-out/drop-out noise. And in this paper, we use the mask-out/drop-out corruption, which is widely used in the previous works [7], [8]. The drop-out corruption can be explained that the non-zero values ($r_{ij} = 1$) in the preference matrix $\mathbf{R}$ are randomly dropped out independently with probability q:

$$\begin{aligned} P(r_{ij} = 0) &= q \\ P(r_{ij} = r_{ij}) &= 1 - q \end{aligned} \tag{1}$$

And then the autoencoder framework of CDGAAE consists of graph attention encoder and neural collaborative filtering decoder. The encoder performs the message passing process based multi-head attention mechanism on the user-item implicit preference graph and the social network information graph. Then we can obtain the user latent vector by deeply merging the results of these two graphs, and the item latent vector is generated from the user-item preference graph. The decoder designs a multi-layer neural collaborative filtering module which takes the latent vectors of users and items as input, and then the decoder reconstructs the original preferences between users and items, which can enhance the robustness and generalization of the model. For model learning process, since the original data is implicit feedback data, we use a binary cross-entropy loss function and a stochastic gradient descent optimization method.

B. Graph Attention Encoder

The purpose of the encoder module is to learn user latent vector $\mathbf{h}_i$ and and item latent vector $\mathbf{h}_j$. The graph attention encoder part shown in Fig.1 contains three message passing processes. So $\mathbf{h}_i$ is from user-item implicit preference graph and social network graph, and $\mathbf{h}_j$ from user-item preference graph. Following the initialize method of NeuMF [14], for the one hot embedding user i and item j, we pass them through two multi-Layer perceptrons, then we can get the initial embedding $\mathbf{u}_i \in \mathbb{R}^d$ and $\mathbf{v}_j \in \mathbb{R}^d$. Next we will introduce the generation method of these two latent vectors $\mathbf{h}_i$ and $\mathbf{h}_j$.

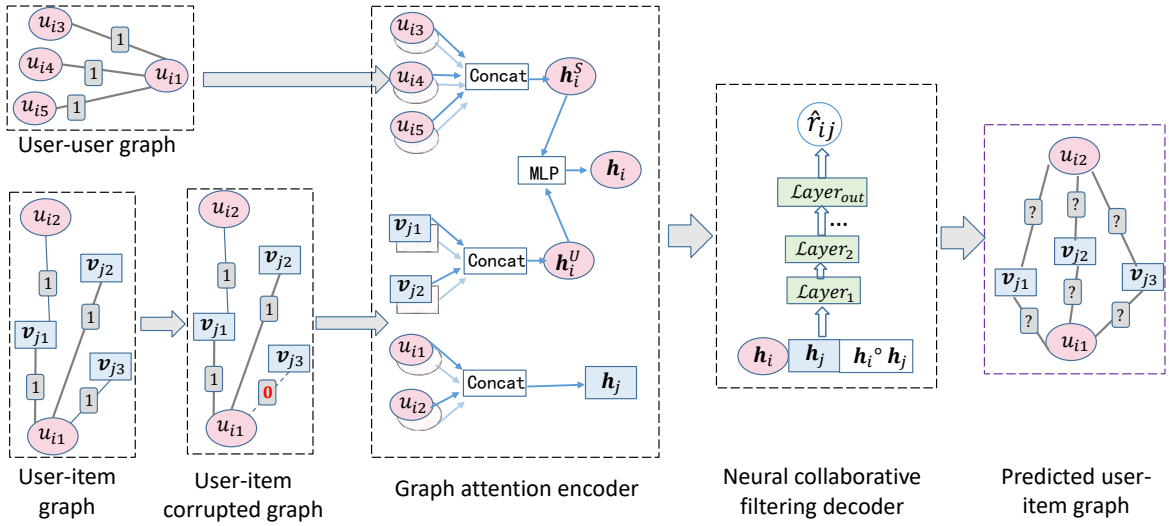

Fig. 1. Overall Structure of the Proposed Framework.

1) User Latent Vector: For each user, firstly, the features of all directly connected items in the user-item preference graph are collected to obtain 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, the deep fusion of these two vectors can form a new embedding vector, that is user latent vector $\mathbf{h}_i$.

In the user-item preference graph, for user i, $\mathcal{O}_i$ denotes the set of items which this user interacted with. Firstly we need to calculate how important the item $\mathbf{v}_j$ is to the user $\mathbf{u}_i$, which we call the attention coefficients e_{ij}:

$$e_{ij} = f(\mathbf{W}_u\mathbf{u}_i, \mathbf{W}_u\mathbf{v}_j) \tag{2}$$

In this equation, f is a function mapping and we can implement it with a variety of neural network structures. $\mathbf{W}_u$ denotes a weight matrix, of which the purpose is to make reasonable linear transformations for the $\mathbf{u}_i$ and $\mathbf{v}_j$. For each node in the graph, weight matrix $\mathbf{W}_u$ and function f are shared. This sharing strategy is inspired by the weight sharing of convolutional neural networks. During the message passing process, we consider all the items from the set $\mathcal{O}_i$, so 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 model, we use a standard multi-Layer perceptron g_u to implement the mapping function f, so the calculation of specific α_{ij} is as follows:

$$\alpha_{ij} = \frac{exp(g_u(\mathbf{W}_u\mathbf{u}_i \oplus \mathbf{W}_u\mathbf{v}_j))}{\sum_{k\in\mathcal{O}_i} exp(g_u(\mathbf{W}_u\mathbf{u}_i \oplus \mathbf{W}_u\mathbf{v}_k))} \tag{4}$$

$\oplus$ denotes the concatenation of two vectors, so Next we can get the single user embedding vector $\mathbf{h}_i^U$ from the aggregation of items' characteristics:

$$\mathbf{h}_i^U = \sigma(\sum_{j\in\mathcal{O}_i} \alpha_{ij}\mathbf{W}_u\mathbf{v}_j) \tag{5}$$

To make the model aggregate more information from different perspectives, we adopt a very popular multi-head attention mechanism [11], which can be described as:

$$\mathbf{h}_i^U = \Big\Vert_{k=1}^{K} \sigma(\sum_{j\in\mathcal{O}_i} \alpha_{ij}^k\mathbf{W}_u^k\mathbf{v}_j) \tag{6}$$

In this equation, where $\Vert$ 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.

Now we hava generated the vector $\mathbf{h}_i^U$ for user u_i from the user-item preference graph. Next, we will introduce the generation method of social embedding vector $\mathbf{h}_i^S$ from the social network graph. we also use the multi-head attention for message passing and the calculation of attention coefficients is similar to Eq.3:

$$\alpha_{ij} = \frac{exp(g_s(\mathbf{W}_s\mathbf{u}_i \oplus \mathbf{W}_s\mathbf{u}_j))}{\sum_{o\in\mathcal{S}_i} exp(g_s(\mathbf{W}_s\mathbf{u}_i \oplus \mathbf{W}_s\mathbf{u}_o))} \tag{7}$$

g_s is a multi-Layer perceptron and $\mathbf{W}_s$ denotes a weight matrix. $\mathcal{S}_i$ is the collection of users which have relationship with user $\mathbf{u}_i$, then we can generate the vector $\mathbf{h}_i^S$ through the multi-head attention mechanism:

$$\mathbf{h}_i^S = \Big\Vert_{k=1}^{K} \sigma(\sum_{o\in\mathcal{S}_i} \alpha_{io}^k\mathbf{W}_s^k\mathbf{u}_o) \tag{8}$$

Through the above illustration we've generated the two parts of the user latent vector: user embedding vector $\mathbf{h}_i^U$ from the user-item implicit preference graph and social embedding vector $\mathbf{h}_i^S$ from the social network graph. Both of these vectors are important components of the representation of user

characteristics,so in order to deeply merge the information of two vectors we also use a multi-layer perceptron:

$$\mathbf{h}_i = g_{us}(\mathbf{h}_i^U \oplus \mathbf{h}_i^S) \tag{9}$$

So $\mathbf{h}_i$ is the final user latent vector which deeply integrates the information from user-item implicit preference graph and social network graph.

2) Item Latent Vector: In this part, we introduce the generation of item latent vector $\mathbf{h}_j$ from the user-item interaction graph. For each item $\mathbf{v}_j$, we aggregate the characteristics of users who have interated with this item. So this process is very similar to the generation of user embedding vector $\mathbf{h}_i^U$ and due to the limit length of this article, we only list the important formulas below:

$$\alpha_{ij} = \frac{exp(g_v(\mathbf{W}_v\mathbf{v}_j \oplus \mathbf{W}_v\mathbf{u}_i))}{\sum_{t\in\mathcal{B}_j} exp(g_v(\mathbf{W}_v\mathbf{v}_j \oplus \mathbf{W}_v\mathbf{u}_t))} \tag{10}$$

$$\mathbf{h}_j = \Big\Vert_{k=1}^{K} \sigma(\sum_{i\in\mathcal{B}_j} \alpha_{ij}^k \mathbf{W}_v^k \mathbf{u}_i) \tag{11}$$

Through the above statement, we finally get the user latent vector $\mathbf{h}_i$ and item latent vector $\mathbf{h}_j$.

C. Neural Collaborative Filtering Decoder

For reconstructing the user-item relations in the preference graph, we proposes a neural collaborative filtering decoder inspired by NeuMF framework [14]. The neural collaborative filtering decoder part is shown in Fig.1, which consists of collaboration layer and neural collaborative filtering layers.

The collaboration layer combines user latent vector $\mathbf{h}_i$, item latent vector $\mathbf{h}_j$ and the element-wise product $\mathbf{h}_i \odot \mathbf{h}_j$:

$$\mathbf{P}_{ij} = [\mathbf{h}_i \oplus \mathbf{h}_j \oplus (\mathbf{h}_i \odot \mathbf{h}_j)] \tag{12}$$

$\mathbf{h}_i, \mathbf{h}_j$ is the results from the graph attention encoder part. But in our framework we also introduce the element-wise product of these two vectors $\mathbf{h}_i \odot \mathbf{h}_j$, which depicts the shallow linear user–item interaction. Next, we take $\mathbf{P}_{ij}$ as the input for the neural collaborative filtering layers to get the deep and intrinsic interaction of user-item pairs.

We now define the neural collaborative filtering layers as a multi-layer neural network formulated as:

$$\hat{r}_{ij} = \mathcal{L}_{out}(\mathcal{L}_X(\dots \mathcal{L}_2(\mathcal{L}_1(\mathbf{P}_{ij}))\dots)) \tag{13}$$

$\hat{r}_{ij}$ is the reconstructed value of user i with item j and the $\mathcal{L}_{out}$ is the Logistic function for the output predicted value. $\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_X$ are the mapping function and in our experiment, we all use the standard multi-layer perceptron (MLP) to implement the function.

D. Model Learning

When we talk about the model learning process, we first need to generate the negative instances. The set $\mathcal{O}$ represents the user-item pairs $(u_i, v_j, r_{ij} = 1)$ with observed interactions and the set $\tilde{\mathcal{O}}$ denotes the other user-item pairs $(u_i, v_j, r_{ij} = 0)$. In general, the size of $\tilde{\mathcal{O}}$ is much larger than the size of $\mathcal{O}$, so we can't just take the set $\tilde{\mathcal{O}}$ as negative instances. To balance the positive and negative instances, we randomly sample set $\tilde{\mathcal{O}}$ and get a new set $\mathcal{O}^-$, which size matches the size of set $\mathcal{O}$.

Then we will introduce the model learning process. For recommendation system based on explicit feedback, the loss function is mainly a regression with squared loss. But for the implicit feedback data used in our paper, the square loss function doesn't perform very well because the target value r_{ij} is a binarized 1 or 0, which refers to whether there is any interaction between user i and item j. So in order to learn the parameters of the model better, we constrain the output value $\hat{r}_{ij}$ in the range of [0, 1] with the Logistic function in the output layer. Just like the NCF model [14] described by He et al, we define the finall objective function as follows:

$$L = -\sum_{(i,j)\in(\mathcal{O}\cup\mathcal{O}^-)} r_{ij}\log\hat{r}_{ij} + (1 - r_{ij})\log(1 - \hat{r}_{ij}) \tag{14}$$

This function is the classical binary cross-entropy loss and we use the stochastic gradient descent algorithm to optimize the model.

IV. Experiment

A. Experimental Settings

1) Datasets: In our experiments, we choose two real-world public datasets: Ciao[1] and Epinions[2]. These datasets are crawled from two famous commerce website, Ciao.com and Epinions.com, which contain user-item interaction information and social relationships. The ratings in Ciao and Epinions are integers from 1 to 5: $\{1, 2, 3, 4, 5\}$ and the statistics of datasets are illustrated in TABLE I.

To generate the implicit feedback data for our model, we take records greater than or equal to 4 as observed preference interactions and other records as the unobserved preference. Then we iteratively drop users and items with less than 5 interactions. This data processing method is widely used in the previous works [7], [8]. For negative sampling, we randomly sample the unobserved set $\tilde{\mathcal{O}}$ to get the negative instances set $\mathcal{O}^-$. In our experiment, the sampling strategy is that for each user the number of negative instances is 5 times the number of observed instances of this user.

TABLE I
General statistics of the Ciao and Epinions

statistics	Ciao	Epinions
Users	7,375	40,163
Items	106,797	139,738
Ratings	284,086	664,824
Density(Ratings)	0.036%	0.051%
Social Relations	111,781	487,183
Density(Social Relations)	0.205%	0.029%

[1]https://www.cse.msu.edu/ tangjili/datasetcode/ciao.zip
[2]www.trustlet.org/downloaded_epinions.html

2) Evaluation Metrics: We use two classical metrics to evaluate the performance of our top-k recommendation system: NDCG@K and MAP@K.

DCG@K is computed by:

$$DCG@K = \sum_{i=1}^{K} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{15}$$

and then NDCG@K is the normalized DCG@K over the ideal iDCG@K.

AP@K is computed by:

$$AP@K = \frac{\sum_{m=1}^{K} P@m \times rel_m}{\min\{K, |\mathcal{Y}_i|\}} \tag{16}$$

$P@m$ represents the precision with m recommended items and by calculating the average of AP@K from all the users we can get the MAP@K.

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

- **BPR** [4]. BPR is a typical pairwise ranking method for item recommendation and it achieved competitive results in many datasets.
- **SBPR** [3]. This model extends the BPR algorithm by adding the social network information. SBPR is based on the same idea that a user's behavior can be influenced by the users associated with him on social networks.
- **NCF** [14]. NCF is a neural network based method which learn the internal interactions between users and items through the multi-layer perceptrons.
- **CDAE** [7]. CDAE formulates the top-N recommendation problem using the Denoising Auto-Encoder framework and learns distributed representations of the users and items from the implicit feedback data.
- **TDAE** [8]. TDAE extends the CDAE model and learns compact and effective representations from both rating and trust data for top-N recommendation.
- **GraphRec** [13]. GraphRec provides a state-of-the-art model for the social recommender system, which captures interactions in the user-item graph and social graph.
- **CDGAAE**$_{den}$. It is a variant of CDGAAE, which uses the uncorrupted input for the whole model.

4) Parameter Settings: We implement our model CDGAAE with the famous framework Pytorch and in view of the effectiveness and efficiency we set the final parameters of our model with the following values: For each user, we select 80% of the data as train set to learn the model parameters, 10% for validation and 10% for test. Moreover, we set the batch size and embedding size to 128 and 64, and also the learning rate is 0.001. Through multiple experiments, we set the number of attention heads to 4. Then we use the Gaussian distribution to randomly initialize the model parameters and the activation function is ReLU. For all the baseline algorithms, we read the articles and implement methods carefully to get the best performance.

B. Performance

Table. 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 algorithms. By careful comparison, we can find the following conclusions:

- Deep neural networks have better performance than the shallow models. In the absence of social network information, the results of NCF and CDAE are better than that of BPR, and if we add the social network data, the models TDAE and CDGAAE perform better than SBPR.
- The social network can improve the performace of the recommendation system. SBPR add social information to BPR and in the table we can find that SBPR performs better than BPR. The same conclusion can be drawn from the comparison of CDAE and TDAE.
- GraphRec shows best experimental results apart from CDGAAE$_{den}$ and CDGAAE. This model uses graph neural networks to generate more accurate embedding for users and items. At the same time, GraphRec uses attention mechanism for user-item graph and social graph.
- CDGAAE$_{den}$ uses the uncorrupted input for the model, so it's less robust than CDGAAE. From the results, we can clearly find that the performance of CDGAAE$_{den}$ is worse than that of CDGAAE, and even worse than that of GraphRec method under certain metrics.
- It is clear from the table II that our method CDGAAE performs best among all the algorithms. We deeply fuse the graph neural networks with the Denoising Auto-Encoder, and we also adopt multi-head attention mechanism for message passing.

C. Model Analysis

We now study the performance of our approach under different parameter settings. We mainly analyze the multi-head attention and the embedding size of the latent vector.

1) The impact of multi-head attention: In our graph attention encoder part, user aggregation, item aggregation in user-item interaction graph and social aggregation in social graph all use multi-head attention mechanism. In this method, the number of attention heads is an important parameter, which has a crucial impact on the performance of the model. So we will compare the effects of different quantities of heads on the results. In the parameter analysis setting, we set the number of heads k=1, 2, 4, 8, 16 and the experimental performance is shown in Fig. 2.

From the figure, 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 single attention layer is small, which is difficult to learn all the useful information in the two graphs. So for the multi-head attetion mechanism, an appropriate number of heads is the key to improve performance.

TABLE II
PERFOMANCE COMPARISON OF OUR MODEL AND BASELINE METHODS

Datasets	Metrics	Algorithms								
		BPR	SBPR	NCF	CDAE	TDAE	GraphRec	CDGAAE$_{den}$	*CDGAAE*	**Improve**
Ciao	NDCG@10	0.0369	0.0421	0.0437	0.0461	0.0496	0.0503	0.0506*	**0.0532**	**4.89%**
	MAP@10	0.0210	0.0237	0.0241	0.0261	0.0299	0.0307*	0.0305	**0.0315**	**2.61%**
Epinions	NDCG@10	0.0153	0.0188	0.0191	0.0218	0.0244	0.0240	0.0248*	**0.0258**	**4.03%**
	MAP@10	0.0080	0.0105	0.0107	0.0104	0.0127	0.0132*	0.0120	**0.0135**	**2.27%**

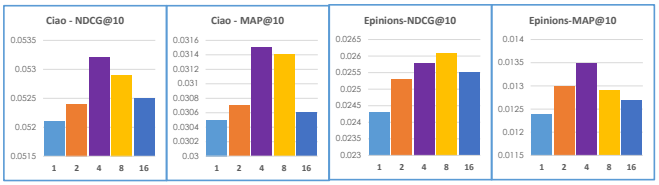

Fig. 2. Experimental results under different number of attention heads on two datasets.

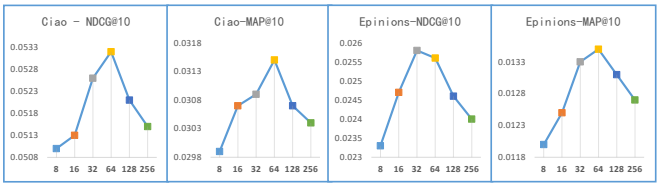

Fig. 3. Experimental results under different embedding size d on two datasets.

2) The impact of the embedding size: In this part, we will discuss the effect of embedding size on model performance. We adopt six different embedding sizes 8, 16, 32, 64, 126, 256 for parameter analysis on the two datasets and the experimental performance comparison is shown in Fig.3. In general, with the embedding size increases, the recommendation performance of our model first increases and then decreases. The embedding size of less than 8 and greater then 128 significantly degrades the model performance. This phenomenon demonstrates that if embedding size is small, the model can not fully and accurately represent user and item characteristics, but if the size is large, the complexity of the model is high, leading to performance degradation. So we need to find a suitable embedding size to balance the representation performance and complexity of the model.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a novel model: Collaborative Denoising Graph Attention Autoencoders for Social Recommendation (CDGAAE). CDGAAE uses a corrupted version of the original data as input, making the representation of the model more robust. At the same time, this autoencoder framework consists of graph attention encoder and collaborative neural decoder, which is used to deeply merge the user-item preference information and social network information. The final experiments are conducted on two real-world datasets, and the results show the superiority of our proposed model.

In the future, in order to improve the accuracy of the recommendation list, we can use the side information of users and items, which is a significant supplement to depict the rich characteristics of users and items. Moreover, in addition to denoising autoencoders, variational autoencoders are also widely used in recommendation systems. Therefore, we will integrate the idea of variational autoencoders into the social recommendation systems in the future, expecting to get better performance.

REFERENCES

[1] 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.

[2] 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.

[3] Zhao T, McAuley J, King I. Leveraging social connections to improve personalized ranking for collaborative filtering[C]//Proceedings of the 23rd ACM international conference on conference on information and knowledge management. ACM, 2014: 261-270.

[4] Rendle S, Freudenthaler C, Gantner Z, et al. BPR: Bayesian personalized ranking from implicit feedback[C]//Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence. AUAI Press, 2009: 452-461.

[5] Chaney A J B, Blei D M, Eliassi-Rad T. A probabilistic model for using social networks in personalized item recommendation[C]//Proceedings of the 9th ACM Conference on Recommender Systems. ACM, 2015: 43-50.

[6] Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising autoencoders[C]//Proceedings of the 25th international conference on Machine learning. ACM, 2008: 1096-1103.

[7] Wu Y, DuBois C, Zheng A X, et al. Collaborative denoising auto-encoders for top-n recommender systems[C]//Proceedings of the Ninth ACM International Conference on Web Search and Data Mining. ACM, 2016: 153-162.

[8] Pan Y, He F, Yu H. Trust-aware collaborative denoising auto-encoder for top-n recommendation[J]. arXiv preprint arXiv:1703.01760, 2017.

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

[10] 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.

[11] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks[J]. arXiv preprint arXiv:1710.10903, 2017.

[12] Berg R, Kipf T N, Welling M. Graph convolutional matrix completion[J]. arXiv preprint arXiv:1706.02263, 2017.

[13] Fan W, Ma Y, Li Q, et al. Graph Neural Networks for Social Recommendation[C]//The World Wide Web Conference. ACM, 2019: 417-426.

[14] 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.