






# Meta-path Enhanced Lightweight Graph Neural Network for Social Recommendation

Hang Miao<sup>1,2</sup>, Anchen Li<sup>1,2</sup>, and Bo Yang<sup>1,2</sup>

<sup>1</sup> Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun, China

<sup>2</sup> College of Computer Science and Technology, Jilin University, Changchun, China  
{miaohang19, liac20}@mails.jlu.edu.cn, ybo@jlu.edu.cn

**Abstract.** Social information is widely used in recommender systems to alleviate data sparsity. Since users play a central role in both user-user social graphs and user-item interaction graphs, many previous social recommender systems model the information diffusion process in both graphs to obtain high-order information. We argue that this approach does not explicitly encode high-order connectivity, resulting in potential collaborative signals between user and item not being captured. Moreover, direct modeling of explicit interactions may introduce noises into the model and we expect users to pay more attention to reliable links. In this work, we propose a new recommendation framework named Meta-path Enhanced Lightweight Graph Neural Network (ME-LGNN), which fuses social graphs and interaction graphs into a unified heterogeneous graph to encode high-order collaborative signals explicitly. We consider using a lightweight GCN to model collaborative signals. To enable users to capture reliable information more efficiently, we design meta-paths to further enhance the embedding learning by calculating meta-path dependency probabilities. Empirically, we conduct extensive experiments on three public datasets to demonstrate the effectiveness of our model.

**Keywords:** Social recommendation · Recommender systems · Graph convolutional network · Collaborative filtering

## 1 Introduction

The volume of data is growing with the rapid development of society, leading to information overload. In response, recommender systems have been proposed for personalized information filtering, which centers on predicting whether a

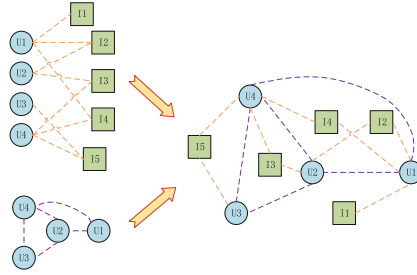
---

Supported by the National Key R&D Program of China under Grant Nos. 2021ZD0112501 and 2021ZD0112502; the National Natural Science Foundation of China under Grant Nos. 62172185 and 61876069; Jilin Province Key Scientific and Technological Research and Development Project under Grant Nos. 20180201067GX and 20180201044GX; and Jilin Province Natural Science Foundation under Grant No. 20200201036JC.

user will interact with a certain item. Collaborative filtering (CF) [4] has been extensively studied in recommender systems, which uses user-item interaction data to learn the user and item embeddings [9–13, 16, 17, 19, 23]. However, the data sparsity that accompanies CF hinders the development of recommender systems. With the rapid increase in social platforms, users are willing to make friends and express their preferences on social platforms. Inspired by this, the social recommendation was proposed to alleviate the data sparsity issue through abundant social relationships.

As users take on an important role in both user-user social networks and user-item interaction graphs, the key to social recommendation is that the final embedding contains information from two interaction graphs. Traditional social recommendation methods [5, 10, 20, 26] aim at adding the influence of social neighbors to the user embedding representation via a user-user interaction matrix with the matrix factorization. These improvements can be seen as utilizing first-order neighbors of the graph structure to make improvements. In the following research, Wu et al. proposed Diffnet [18] and Diffnet++ [21] to further enhance the embedding learning by incorporating high-order neighborhood information into the embedding learning process. Although the performance of the recommendation models of these social networks has improved, we believe that there is still room for improvement in the present social recommendation models. Specifically, these methods are not explicitly coded between the user and the long-distance item. For example, there is such an association  $\mathcal{U}_4 \rightarrow \mathcal{U}_1 \rightarrow \mathcal{I}_1$  in Fig. 1, then  $\mathcal{U}_4$  and  $\mathcal{I}_1$  are intrinsically related, but Diffnet does not explicitly model these latent collaborative signals, we hope to encode such collaboration signals in an explicit way. As well, during the training of the previous models, all interactions between users and users (items) are coded uniformly, regardless of the reliability of the connections between them. Some unreliable connections may distract the nodes and lead to diminishing embedding representation capabilities. Therefore, we expect users to focus on those users and items that are more relevant to them in the modeling.

In this work, we propose a new recommendation framework named Meta-path Enhanced Lightweight Graph Neural Network (ME-LGNN), which models high-order collaborative signals explicitly. Specifically, we merge the user-user social network and the user-item interaction graph into a unified social recommendation HIN, then we utilize a lightweight graph convolution operation to aggregate neighbors' information. By stacking multiple layers, users and items can obtain the characteristics of their high-order neighbors. In the aggregation process, we use the attention network to adaptively aggregate the embedding representation of different neighbors. In addition, to enable users to capture collaborative signals more efficiently, we devise a series of interpretable meta-paths in HIN. To make the model focus more on reliable connections during training, we reduce noises by constraining the dependency probability of meta-paths, making the embedding representation more capable. The two training processes are alternated to improve the overall recommendation performance.



**Fig. 1.** Heterogeneous information network (HIN) constructed by the user-item interaction graph and the social network. The blue node represents the user, the green node represents the item, the orange dashed line indicates the interaction between the user and the item, the purple dashed line indicates the trust relationship between the users. (Color figure online)

In summary, the contributions of this work are as follows:

- We emphasize the significance of explicitly incorporating high-order collaborative signals into embedding in social recommendation models.
- We fuse the user-user social network and the user-item interaction graph into a unified heterogeneous network. On the basis of the network, we propose a new social recommendation framework called ME-LGNN, which can model high-order collaborative signals explicitly.
- To enable users to focus more on reliable connections, we designed a series of meaningful meta-paths, which further improve embedding learning by constraining meta-path dependency probabilities.
- We conduct extensive experiments on three public datasets. Results demonstrate competitive performance of ME-LGNN and the effectiveness of explicit modelling of unified HINs.

## 2 Related Work

In this section, we briefly review the work related to social recommendation.

SoRec [15] is an early social recommendation model based on matrix factorization, which proposed a factor analysis method using the probability matrix. To incorporate the preferences of friends trusted by the user, Ma et al. proposed a recommendation model RSTE [14] with weighted social information on top of SoRec. However, this model does not propagate information about users' preferences in the network, so SocialMF [10] was proposed, which is based on the trust propagation mechanism of social networks to obtain the embedding representation of users. Observing that users tend to give higher scores to items that their friends like, Zhao et al. proposed the SBPR [26] model, which uses social information to select training examples and incorporates social relationships into the model. TBPR [20] was built on it by incorporating strong and weak

ties in social relationships into social recommendations. These models proposed different methods to address the sparse issue of collaborative filtering.

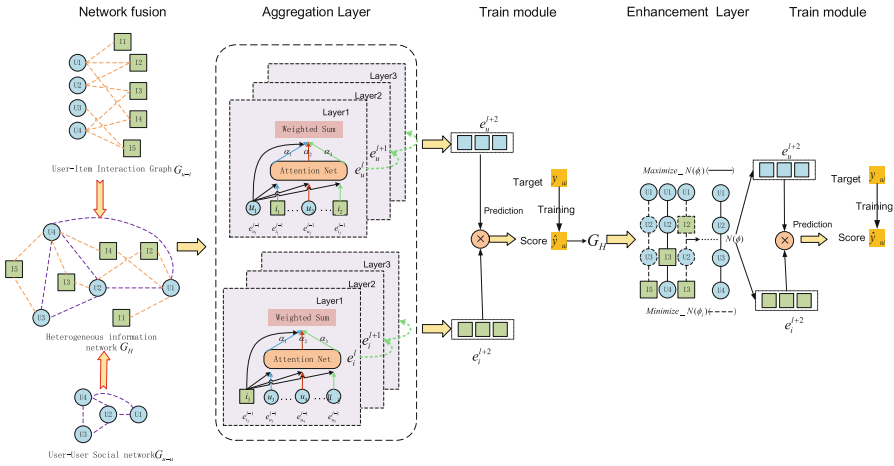
The recommender systems based deep learning aim to capture non-linear features from the interaction graph. NeuMF [8] combines traditional matrix factorization and Multilayer Perceptron (MLP) to extract both low and high dimensional features with promising recommendation performance. DeepSoR [2] proposed to learn embedding representations from social relationships and then integrated the user’s embedding representation into a probability matrix factorization for evaluating prediction. However, these methods do not encode the interaction of information in an explicit way.

In recent years, graph neural networks are becoming well known for their powerful performance in learning graph data. Recommendation tasks also can be well represented as graph structures, so GNNs provide great potential for the development of recommendation tasks. GC-MC [1] was proposed to construct embedding of users and items by passing messages on the user-item interaction graph, but this passing only operates on a layer of links and does not incorporate high order interaction information. The NGCF [19] proposed by Wang et al. designs a graph convolution operation on the user-item interaction graph to capture collaborative filtering signals in high-order connections. LightGCN [7] makes the model more suitable for collaborative filtering tasks by removing feature transformations and non-linear variations from the GCN of the NGCF. To incorporate social relationships, GraphRec [3] learns an adequate embedding representation from the rich social relationships by fusing first-order interactions on social networks and user-item interaction graphs with neural network processes. Diffnet [18] leverages convolutional operations to perform recursive diffusion in social networks to obtain high-order collaborative signals from users. With these existing models, our work performs differently in that we fuse the social network and user-item interaction graphs into a unified heterogeneous information graph that explicitly encodes potential collaborative signals by propagating information over the heterogeneous graph.

Considered from the perspective of heterogeneous graphs, a number of meta-path-related models have been proposed to solve the recommendation task. The recommendation model based on the meta-path can perform interpretable analysis of the recommendation results. Yu et al. proposed HteRec [25] which targets implicit feedback and obtains different user preference matrices based on different meta-paths, and then goes through a matrix factorization model to implement recommendation tasks. Han et al. proposed the NeuACF [6] model to extract different dimensions of information through multiple meta-paths in an attempt to fuse different aspects of information. The IF-BPR [24] model also learns the user’s embedding representation through a meta-path approach and then discovers the user’s potential friends through an adaptive approach. On the contrary, our work aims to enhance representation by calculating meta-path dependency probabilities in such a way that users can focus on connections that are more closely related to themselves and ignore connections that are more dissimilar to themselves in a heterogeneous graph.

### 3 Methodology

In general, ME-LGNN is composed of four parts: (1) Embedding Layer: providing the initial embedding of users and items. (2) Aggregation layer: learning embedding representations of users and items through a lightweight GCN with the attention mechanism; (3) Enhancement layer: designing some reasonable meta-paths, through meta-path constraints to further enhance the embedding representation capabilities. (4) Predicting Layer. We show the overall neural architecture of ME-LGNN in Fig. 2. We first describe the problem formulation, then introduce the four components of our model, and finally discuss the training optimization process.



**Fig. 2.** Illustration of the ME-LGNN model architecture (arrow lines indicate information flow). In the aggregation layer user (blue) and item (green) are aggregated through multiple propagation layers for neighborhood aggregation, the output is subjected to the first round of training, then further enhanced by a meta-path enhancement layer for embedding learning, and its output is subjected to the final prediction. (Color figure online)

#### 3.1 Notation and Problem Formulation

We introduce necessary notations and definitions by describing the graph-based collaborative filtering problem. In graph-based social recommendation, there are two types of entities, user set  $U$  ( $|U| = M$ ) and item set  $I$  ( $|I| = N$ ). Two interaction graphs are formed from  $U$  and  $I$ : (1) User-item interaction graph  $G_{u-i}$ : we can use the adjacency matrix to represent the interactions in the graph.  $R \in \mathbb{R}^{M \times N}$  is the interaction matrix between users and items. If there is an interaction between the user  $u$  and the item  $i$ ,  $y_{ui}$  is 1, otherwise it is 0. (2) User-user social network  $G_{u-u}$ :  $S \in \mathbb{R}^{M \times M}$  is the interaction matrix between

users. If the user  $u_1$  and the user  $u_2$  trust each other,  $y_{u_1 u_2}$  is 1, otherwise it is 0. The type of relationship between entities  $a$  and  $b$  is  $r_{ab}$ . Relationship types include social relationships between users and user ratings of items. We merge the two networks to obtain a unified heterogeneous information network  $\mathbf{G}_H$ . For social recommendation, given the evaluation matrix  $R$  of users and items and the social network interaction matrix  $S$ , our goal is to predict the user's preference for unknown items. Next, we define the problems investigated in this work as follows:

**Input:** user-user social network  $\mathbf{G}_{u-u}$ , user-item interaction graph  $\mathbf{G}_{u-i}$ , user set  $U$  and item set  $I$ .

**Output:** a personalized ranking function that maps an item to a real value for each user:  $f_u : I \rightarrow \mathbb{R}$ .

In this process, we first merge the two input interaction graphs to obtain a new heterogeneous information network  $\mathbf{G}_H$ . The rest of the training is carried out on  $\mathbf{G}_H$ .

### 3.2 Embedding Layer

For social recommendation, we consider it as a representation learning problem. Following some mainstream models [7, 18, 19, 21], we use the embedding vector  $\mathbf{e}_u \in \mathbb{R}^d$  ( $\mathbf{e}_i \in \mathbb{R}^d$ ) to describe the user (item) and use the embedding vector  $\mathbf{r} \in \mathbb{R}^d$  to describe the relationship type,  $d$  represents the dimension of the embedding. We construct two parameter matrices as embedding lookup tables:

$$\mathbf{E} = [ \overbrace{\mathbf{e}_{u_1}, \dots, \mathbf{e}_{u_N}}^{\text{user embeddings}}, \overbrace{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_M}}^{\text{item embeddings}} ], \mathbf{R} = [ \overbrace{\mathbf{r}_{uu}, \mathbf{r}_{ui}}^{\text{relationship type embeddings}} ]. \quad (1)$$

### 3.3 Aggregation Layer

At the aggregation layer, we employ the attention network structure to aggregate adaptively the learning embeddings of different neighbors in the heterogeneous graph  $\mathbf{G}_H$ . Through iterative aggregation, information can be diffused and propagated in the network. Some high-order cooperative information can also be captured, so the model can model high-order collaborative signals. Recent work [7] has found that the most common feature transformations and non-linear activations in GCN contribute little to collaborative filtering and instead make training more difficult and reduce recommendation performance. In our work, we remove the two steps of feature transformation and nonlinear activation to reduce model complexity. We aggregate all nodes in the heterogeneous information network, the graph convolution operation in the ME-LGNN is defined as:

$$\mathbf{e}_a^* = AGG(\mathbf{e}_a, \mathcal{N}(a)) = \mathbf{e}_a + \sum_{b \in \mathcal{N}(a)} \tilde{\pi}_{ab} \cdot \mathbf{e}_b, \quad (2)$$

where  $\mathbf{e}_a^*$  represents the embedding representation of node  $a$ ,  $\mathcal{N}(a)$  represents the set of neighbors of  $a$ , and  $\tilde{\pi}_{ab}$  is the attention value, which indicates that how strongly node  $a$  is influenced by node  $b$ . The attention value is specifically defined as follows:

$$\tilde{\pi}_{ab} = \frac{\exp(\pi_{ab})}{\sum_{b' \in \mathcal{N}(a)} \exp(\pi_{ab'})}, \quad (3)$$

$$\pi_{ab} = (\mathbf{e}_a \odot \mathbf{e}_b)^T \tanh(\mathbf{r}_{ab} \odot \mathbf{e}_b), \quad (4)$$

where  $r_{ab}$  is the interaction type between node  $a$  and node  $b$ . High-order correlation is crucial for encoding collaborative signals and estimating correlation scores between users and items. We iteratively execute the above aggregation process and encode high-order collaborative signals to explore high-order connectivity information.

### 3.4 Enhancement Layer

In the training process of the previous model, Fig. 1 shows that the interaction information between all users and users (items) is uniformly coded, regardless of the user’s high-order information relevance. Some unreliable links may distract nodes. To alleviate the above problems, we do the following work to enhance the representation learning ability.

**Design Meta-path.** Referring to [24], the interaction in this paper is undirected. Here we have designed some reasonable meta-paths as shown in Table 1. We generate these meta-paths by means of a random walk method on the heterogeneous graph  $\mathbf{G}_H$ . Then, a new path set  $\mathbf{P}$  is obtained.

**Table 1.** Meta-paths designed for social recommendation.

Path	Schema	Description
$P_1$	U-I-U	Users who have consumed the same item are similar
$P_2$	U-U-U	Users may trust their friends’ friends
$P_3$	U-U-I	Users may have similar preferences to their friends
$P_4$	U-U-U-U	Users who share the same friends are similar with each other
$P_5$	U-U-U-I	Users may have similar preferences as their friends’ friends
$P_6$	U-U-I-U	Users’ preferences may be similar to those of their friends who have similar preferences
$P_7$	U-I-U-U	Users who have consumed the same items may have similar preferences

**Calculate Dependent Path Probability.** We reconstruct a dependency path. For the path head and tail nodes, if there are interactions between the two nodes in the ground-truth, the path will have a higher score, otherwise, it will have a lower score. The goal is to enable users to pay more attention to the nodes that are more likely to interact with themselves when learning embedding, so as to further enhance the embedding learning.

Inspired by [22], we calculate the dependency probabilities of paths with the following approach. Unlike [22] which proposed to model reconstructed sequences as sequence generation, the sequence nature of paths in our study is not obvious. Specifically, since there is a relationship between the two nodes in the path, we use self-attention to calculate the hidden state  $\mathbf{p}_{b_l}$  of each node  $\mathbf{q}_{b_{l-1}}$  on the path  $\phi(n)$ , as follows:

$$\mathbf{p}_{b_l} = \text{self-attention}(\mathbf{p}_{b_{l-1}}, \mathbf{q}_{b_{l-1}}), \quad (5)$$

where  $\mathbf{p}_{b_l}$  is the embedding of the node, and the initialization of  $\mathbf{p}_{b_0}$  is the output of the aggregation layer. Then  $\mathbf{p}_{b_l}$  is given to the softmax layer to calculate the probability of node  $v_{b_l}$  in the path:

$$P(v_{b_l} | v_{<b_l}) = \frac{\exp(\mathbf{p}_{b_l} \mathbf{W}_r \mathbf{q}_{b_l})}{\sum_n \exp(\mathbf{p}_{b_l} \mathbf{W}_r \mathbf{q}_{b_n})}, \quad (6)$$

where  $\mathbf{W}_r \in \mathbb{R}^{d \times d}$  is the parameter matrix. Then we get the set of node probabilities for path  $\phi(n)$  as  $\{P(v_{b_1} | v_{<b_1}), P(v_{b_2} | v_{<b_2}), \dots, P(v_{b_L} | v_{<b_L})\}$ , finally, the probability of path  $\phi(n)$  is calculated as:

$$N(\phi(n)) = \prod_{l=1}^L P(v_{b_l} | v_{<b_l}). \quad (7)$$

### 3.5 Predicting Layer

After propagation through  $k$  layers, we obtain a representation of the user embedding  $\{\mathbf{e}_u^1, \mathbf{e}_u^2, \dots, \mathbf{e}_u^k\}$  and item embedding  $\{\mathbf{e}_i^1, \mathbf{e}_i^2, \dots, \mathbf{e}_i^k\}$  for each layer. Since each layer may reflect different dimensions of user preferences, we concatenate each layer of user representation as the final representation:  $\mathbf{e}_u^* = [\mathbf{e}_u^0 \parallel \mathbf{e}_u^1 \parallel \dots \parallel \mathbf{e}_u^k]$ . we use a similar approach to get the final representation of item:  $\mathbf{e}_i^* = [\mathbf{e}_i^0 \parallel \mathbf{e}_i^1 \parallel \dots \parallel \mathbf{e}_i^k]$ . Ultimately we calculate the user's preference  $\hat{y}_{ui}$  for a certain item via inner product:

$$\hat{y}_{ui} = \mathbf{e}_u^{*T} \mathbf{e}_i^*. \quad (8)$$

### 3.6 Model Training

To learn the model parameters, for the aggregation layer, we adopt cross-entropy loss as the loss function:

$$\mathcal{L}_A = \sum_{(u,i,j) \in \mathcal{O}} -\ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}), \quad (9)$$



where  $\mathcal{O} = \{(u, i, j) \mid (u, i) \in \mathcal{R}^+, (u, j) \in \mathcal{R}^-\}$  denotes the pairwise training data,  $\mathcal{R}^+$  denotes positive samples,  $\mathcal{R}^-$  denotes negative samples.

For the enhancement layer, the  $\hat{y}_{ui}$  was calculated using the same method as in Eq. (8). We calculate the additional reconstruction loss for the generated training set of reconstructed paths. We get fewer positive samples by random walk through the meta-path, so we give a larger weight to the positive samples during training. The difference with Eq. (9) is that here the training is performed through a cross-entropy loss function with weights:

$$\mathcal{L}_E = \sum_{(u,i,j) \in \mathcal{O}} -\ln \sigma(\mu \hat{y}_{ui} - \hat{y}_{uj}), \quad (10)$$

$$\mu = \sqrt{\frac{\|\mathcal{R}^+\| + \|\mathcal{R}^-\|}{\|\mathcal{R}^+\|}}, \quad (11)$$

where  $\mu$  is the weight value of the positive sample.

The complete loss function of ME-LGNN is as follows:

$$\mathcal{L} = \mathcal{L}_A + \mathcal{L}_E + \lambda \|\Theta\|^2, \quad (12)$$

where  $\Theta = \{\mathbf{E}, \mathbf{R}, \mathbf{W}_r\}$  represents all the trainable model parameters. It should be noted here that the only training for our aggregation layer is the embedding of the  $0^{th}$  layer. Compared with the traditional graph convolution operation, the amount of parameters is reduced. Our enhancement layer model training parameters and the conversion matrix in self-attention.  $\lambda$  is used to balance the power of  $L_2$  regularization and prevent overfitting.  $\mathcal{L}_A$  and  $\mathcal{L}_E$  are trained alternately to jointly improve the performance of the model. We adopt Adam optimizer to optimize the aggregation layer and enhancement layer to update the parameters. Adam is a commonly used optimizer which can adaptively adjust the learning rate.

The specific process of ME-LGNN is presented in Algorithm 1. A training epoch involves two stages: the aggregation layer (line 3–5) and the enhancement layer (line 6–8). In each iteration, we execute the aggregation layer and the enhancement layer in turn to enhance embedding learning.

### 3.7 Complexity Analysis

In this section, we analyse the complexity of our model.

**Model Size.** We adopt the alternative optimization strategy, the training matrix of our model consists of three parts: user and item embeddings, the parameter matrix, transformation matrixes. For the aggregation layer, we only need to learn the  $0^{th}$  layer user embeddings  $\mathbf{U}^{(0)} \in \mathbb{R}^{N \times d}$  and item embeddings  $\mathbf{I}^{(0)} \in \mathbb{R}^{M \times d}$ . For the enhancement layer, model training matrix  $\mathbf{W}^r$  and three conversion matrixes in self-attention need to be learned. The number of parameters in each matrix is  $d \times d$ . In total, the overall model size is approximately  $(N + M + 4d)d$ . It can be seen that our model is very lightweight.

**Algorithm 1.** ME-LGNN

---

**Input:** User-item interaction matrix  $R$ , user-user interaction matrix  $S$ , heterogeneous information network  $\mathbf{G}_H$ , user set  $U$  and item set  $I$

**Output:** Prediction function  $f_u : I \rightarrow \mathbb{R}$

- 1: Initialize model parameters
- 2: **for** number of training iteration **do**
- 3:   //the aggregation layer
- 4:   Sample minibatch of positive and negative interactions from  $R$ ;
- 5:   Update parameters by gradient descent on Equation (2)-(4), (8), (12);
- 6:   //the enhancement layer
- 7:   Obtain meta-paths by meta-path-based random walk method according to Table 1 from  $\mathbf{G}_H$ ;
- 8:   Update parameters by gradient descent on Equation (5)-(8), (12);
- 9: **end for**
- 10: Get the trained model parameters and calculate the prediction score

---

**Time Complexity.** Time consumption comes in five main parts: graph convolution, aggregation attention, self-attention, dependent path probability, and prediction. We first calculate the number of interactions  $t = |R^+| + |S^+|$  in the adjacency matrix, where  $|R^+|$  and  $|S^+|$  denote the number of nonzero elements in  $R$  and  $S$ . For the graph convolution and attention through  $k$  layers, time consumption both are  $O(tdk)$ ,  $d$  is the dimension of embedding. The self-attention and dependent path probability have computational complexity  $O(Nd)$ . For the prediction layer, only inner product calculations were carried out, taking time  $O(NMd^2)$ . Since our model removes the feature transformations and non-linear transformations from the GCN according to [7], our model is also more efficient in the training process than traditional GNN-based social recommendation models.

## 4 Experiments

In this section, we conduct experiments on three real datasets, Yelp, Douban and Lastfm-2k, to answer the following three questions:

**RQ1** Can ME-LGNN perform better than other competitive methods?

**RQ2** Does our proposed meta-path enhancement module improve recommendation performance?

**RQ3** Is ME-LGNN effective in mitigating data sparsity problems?

### 4.1 Experimental Settings

**Dataset.** To validate the performance of the model proposed in this paper, we conduct experiments on three public datasets, Yelp, Douban and Lastfm-2k. The three datasets are described in detail as follows.

**Table 2.** The statistics of the three datasets.

Dataset	Yelp	Douban	Lastfm-2k
Users	10580	12748	1892
Items	13870	22347	17632
Total links	79295	84575	84845
Ratings	102662	555739	55625
Link density	0.07%	0.05%	2.37%
Rating density	0.07%	0.19%	0.17%

- **Yelp:** The dataset is an online location-based social network where users rate and interact with each other on top of the site.
- **Douban:** The dataset is a crawl of Douban reading data, which describes the interaction behaviour of users on Douban.
- **Lastfm-2k:** The dataset contains the social networks of the 2K users set from the Last.fm online music system, as well as information about the users' listening.

With the dataset described above, we randomly selected 60% of the user interaction dataset as the training set, 20% as the test set and the remaining 20% as the validation set to tune the hyperparameters. Next, we perform a negative sampling strategy on the dataset, sampling the items that have not been consumed by the users as negative samples. The information of the dataset after pre-processing is shown in Table 2.

**Evaluation Indicators.** In this article, with an aim of evaluating the top-N, We use two benchmarks which are commonly used in recommender systems, Recall and Normalized Discounted Cumulative Gain (NDCG). Recall measures how many positive examples are judged to be positive, NDCG considers not only the ranking but also the relevance of the top positive examples.

**Baselines.** To verify the performance of our model, we compare ME-LGNN with the following methods.

- **TBPR [20]:** The approach incorporates the important concept of strong and weak ties into social recommendation, combining the BPR [16] model to discriminate between strong and weak ties. The authors propose an EM model-based algorithm for discriminating strong and weak ties in social networks, which learns the potential feature vectors of users and items based on the optimal recommendation accuracy.
- **SocialMF [10]:** The method constrains that a user's preferences should be as similar as possible to the average preferences of the social neighbors to which the user is connected, and introduces trust propagation in the matrix factorization so that users are represented as being close to the users they trust.

- **NeuMF** [8]: The method is a typical recommendation algorithm based on deep learning. It combines traditional matrix factorization and a multi-layer perceptron to extract both low and high dimensional features with impressive recommendation results.
- **Diffnet** [18]: This approach uses GCN to model the user’s social network to obtain the user’s embeddings then leverages the SVD++ [11] framework to implement the recommendation task.
- **LightGCN** [7]: This model is also a graph-based model, where it explores recommendation tasks on a user-item interaction graph. In this work, the authors’ goal is to simplify the design of the GCN to make it cleaner and more suitable for collaborative filtering tasks.

**Parameter Settings.** We use tensorflow to implement our model. For all models relying on gradient descent-based approaches in the model learning process, we utilise Adam as the optimization method. For all comparison models, we tune the learning rate between [0.001, 0.005, 0.01, 0.02, 0.05] to get the best results. The regularization factor was tuned between  $[10^{-6}, 10^{-5}, \dots, 10^1, 10^2]$  to obtain the best results. The training batch size is set to 1024. For NeuMF, we set the hidden layers as suggested in [8]. For Diffnet, LightGCN, we tune the GCN layers between [1, 2, 3]. For our model, the number of GCN layers is set to 2. Finally, we set the embedding dimension to 32 and 64 respectively to compare the recommended performance.

## 4.2 Overall Comparison (RQ1)

In Table 3 and Table 4, we show the overall performance of the top-10 recommendations for all models with different dimensional embeddings in the three datasets. It can be observed that almost all the performances are improved accordingly with increasing embedding dimension  $d$ . Both TBPR [20] and SocialMF [10] leverage the social connections of users to mitigate the problem of sparsity, and while TBPR [20] incorporates the strength of ties, SocialMF [10] introduces the propagation of trust. NeuMF [8] introduce high-dimensional features on the basis of traditional methods and achieve considerable results. Graph convolution models Diffnet [18] and LightGCN [7] show a significant improvement, as reflected in the great potential of graph-based recommendation models for recommendation tasks. Our ME-LGNN is superior on almost all data sets, which shows the effectiveness of explicit modeling for high-order collaborative signals. We carry out further experiments on this aspect, the results are depicted in Table 5 and Table 6. They present the experimental performance under different Top-N when  $d = 64$ , which are consistent with our previous analysis, further verifying the effectiveness of our model. Such results and explanations prove that our model is effective through specific experiments. As can be seen from the results of the experiment, the results perform best when the dimension is 64, so we will use  $d = 64$  for model analysis in the following experiments.

**Table 3.** Recall@10 comparisons for different dimension size  $d$ .

Models	Yelp		Douban		Lastfm-2k	
	d = 32	d = 64	d = 32	d = 64	d = 32	d = 64
TBPR	0.0247	0.0256	0.0246	0.0358	0.0621	0.0605
SocialMF	0.0182	0.0245	0.0410	0.0438	0.0766	0.0725
NeuMF	0.0258	0.0283	0.0473	0.0456	0.0934	0.0936
Diffnet	0.0223	0.0218	0.0309	0.0453	0.0834	0.0865
LightGCN	0.0344	0.0361	0.0422	0.0512	0.1118	0.1033
ME-LGNN	<b>0.0407</b>	<b>0.0412</b>	<b>0.0497</b>	<b>0.0514</b>	<b>0.1159</b>	<b>0.1200</b>

**Table 4.** NDCG@10 comparisons for different dimension size  $d$ .

Models	Yelp		Douban		Lastfm-2k	
	d = 32	d = 64	d = 32	d = 64	d = 32	d = 64
TBPR	0.0201	0.0198	0.0324	0.0631	0.0754	0.0734
SocialMF	0.0159	0.0218	0.0542	0.0605	0.0882	0.0873
NeuMF	0.0216	0.0226	0.0617	0.0636	0.1028	0.1171
Diffnet	0.0195	0.0207	0.0397	0.0585	0.0988	0.1068
LightGCN	0.0316	0.0334	0.0657	<b>0.0696</b>	0.1271	0.1282
ME-LGNN	<b>0.0346</b>	<b>0.0353</b>	<b>0.0670</b>	0.0691	<b>0.1396</b>	<b>0.1415</b>

**Table 5.** Recall@N comparisons for different top-N values.

Models	Yelp		Douban		Lastfm-2k	
	N = 10	N = 20	N = 10	N = 20	N = 10	N = 20
TBPR	0.0256	0.0353	0.0358	0.0546	0.0605	0.0898
SocialMF	0.0245	0.0462	0.0439	0.0727	0.0724	0.1263
NeuMF	0.0283	0.0499	0.0456	0.0720	0.0937	0.1461
Diffnet	0.0219	0.0338	0.0453	0.0648	0.0866	0.1354
LightGCN	0.0361	0.0582	0.0512	0.0735	0.1033	0.1596
ME-LGNN	<b>0.0412</b>	<b>0.0653</b>	<b>0.0514</b>	<b>0.0815</b>	<b>0.1200</b>	<b>0.1728</b>

**Table 6.** NDCG@N comparisons for different top-N values.

Models	Yelp		Douban		Lastfm-2k	
	N = 10	N = 20	N = 10	N = 20	N = 10	N = 20
TBPR	0.0198	0.0219	0.0631	0.0623	0.0733	0.0847
SocialMF	0.0218	0.0281	0.0605	0.0661	0.0874	0.1113
NeuMF	0.0226	0.0283	0.0636	0.0689	0.1171	0.1385
Diffnet	0.0207	0.0240	0.0585	0.0607	0.1060	0.1279
LightGCN	0.0334	0.0383	<b>0.0696</b>	0.0722	0.1282	0.1521
ME-LGNN	<b>0.0353</b>	<b>0.0421</b>	0.0691	<b>0.0740</b>	<b>0.1415</b>	<b>0.1609</b>

### 4.3 Ablation Experiments (RQ2)

In this subsection, we examine whether our proposed meta-path enhancement module is effective through ablation experiments. We remove the meta-path enhancement module to train the model and obtain experimental results for the model LGNN. The results of the experiment are shown in Table 7 and Table 8, it can be observed that the addition of the meta-path enhancement module to the LGNN yields the ME-LGNN, which further improves performance and demonstrates the effectiveness of valuing reliable connections for modeling. Moreover, after removing the meta-path enhancement module, our model still exhibits impressive performance, further demonstrating its effectiveness in explicit modeling of HINs.

**Table 7.** Recall@N comparisons for different top-N values.

Models	Yelp		Douban		Lastfm-2k	
	N = 10	N = 20	N = 10	N = 20	N = 10	N = 20
LGNN	0.0407	0.0646	0.0509	0.0821	0.1163	0.1686
ME-LGNN	<b>0.0412</b>	<b>0.0653</b>	<b>0.0514</b>	<b>0.0815</b>	<b>0.1200</b>	<b>0.1728</b>

**Table 8.** NDCG@N comparisons for different top-N values.

Models	Yelp		Douban		Lastfm-2k	
	N = 10	N = 20	N = 10	N = 20	N = 10	N = 20
LGNN	0.0345	0.0409	0.0685	0.0737	0.1390	0.1598
ME-LGNN	<b>0.0353</b>	<b>0.0421</b>	<b>0.0691</b>	<b>0.0740</b>	<b>0.1415</b>	<b>0.1609</b>

### 4.4 Performance Comparison Under Different Sparsity (RQ3)

To verify whether our model can mitigate the data sparsity problem, we conduct sparsity experiments on the Yelp and Douban datasets. For the users in the training data, we first group them according to the quantity of interactions. For example, [16, 32) means that the user interacts with the item at least 16 times and at most 32 times in the training set. The experimental results are presented in Fig. 3, from which we are able to observe a general trend of improved performance with increasing numbers of interactions. The result is cognitive, since the more interactions there are, the more information that can be captured. It can also be noted that our model exhibits respectable performance in most cases, with a significant improvement on the Yelp dataset in particular, indicating that our model can mitigate the data sparsity problem.

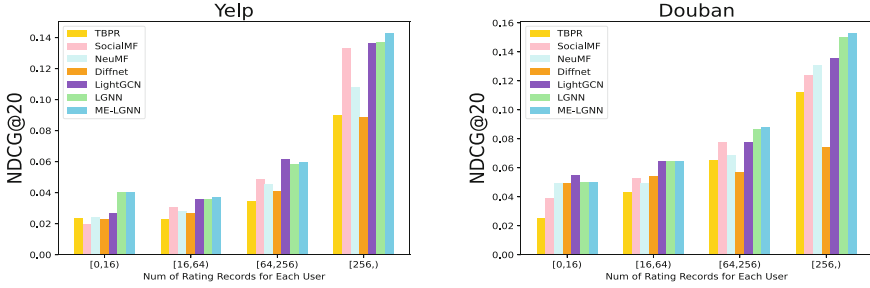


Fig. 3. Performance under different rating sparsity on two datasets.

## 5 Conclusion

In this work, we propose a new recommendation framework called Meta-path Enhanced Lightweight Graph Neural Network (ME-LGNN) to explicitly model high-order collaborative signals. In order to make users pay more attention to reliable links, we design a series of meaningful meta-paths, random walk based on the meta-paths, and constrain the links by calculating the dependency probabilities of the meta-paths. Extensive experimental analysis on three public datasets shows the effectiveness of our proposed model.

Currently, our approach shows promising performance in handling simple heterogeneous information networks. In future, we consider improving our approach to enable the handling of complex network structures with more attribute information. We will also consider how to make more rational use of meta-paths to improve the interpretability of recommendation results.

## References

1. van den Berg, R., Kipf, T.N., Welling, M.: Graph convolutional matrix completion. arXiv preprint [arXiv:1706.02263](https://arxiv.org/abs/1706.02263) (2017)
2. Fan, W., Li, Q., Cheng, M.: Deep modeling of social relations for recommendation. In: AAAI (2018)
3. Fan, W., et al.: Graph neural networks for social recommendation. In: WWW, pp. 417–426 (2019)
4. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. *Commun. ACM* **35**(12), 61–70 (1992)
5. Guo, G., Zhang, J., Yorke-Smith, N.: TrustSVD: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In: AAAI, vol. 29 (2015)
6. Han, X., Shi, C., Wang, S., Philip, S.Y., Song, L.: Aspect-level deep collaborative filtering via heterogeneous information networks. In: IJCAI, pp. 3393–3399 (2018)
7. He, X., et al.: LightGCN: simplifying and powering graph convolution network for recommendation. In: SIGIR, pp. 639–648 (2020)
8. He, X., et al.: Neural collaborative filtering. In: WWW, pp. 173–182 (2017)

9. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: 2008 Eighth IEEE International Conference on Data Mining, pp. 263–272. IEEE (2008)
10. Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: RecSys, pp. 135–142 (2010)
11. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: SIGKDD, pp. 426–434 (2008)
12. Koren, Y.: Collaborative filtering with temporal dynamics. In: SIGKDD, pp. 447–456 (2009)
13. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**(8), 30–37 (2009)
14. Ma, H., King, I., Lyu, M.R.: Learning to recommend with social trust ensemble. In: SIGIR, pp. 203–210 (2009)
15. Ma, H., Yang, H., Lyu, M.R., King, I.: SoRec: social recommendation using probabilistic matrix factorization. In: CIKM, pp. 931–940 (2008)
16. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. arXiv preprint [arXiv:1205.2618](https://arxiv.org/abs/1205.2618) (2012)
17. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) *The Adaptive Web*. LNCS, vol. 4321, pp. 291–324. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-72079-9\\_9](https://doi.org/10.1007/978-3-540-72079-9_9)
18. Seah, B.S., Bhowmick, S.S., Dewey Jr., C.F.: DiffNet: automatic differential functional summarization of de-map networks. *Methods* **69**(3), 247–256 (2014)
19. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: SIGIR, pp. 165–174 (2019)
20. Wang, X., Lu, W., Ester, M., Wang, C., Chen, C.: Social recommendation with strong and weak ties. In: CIKM, pp. 5–14 (2016)
21. Wu, L., et al.: DiffNet++: a neural influence and interest diffusion network for social recommendation. *TKDE* (2020)
22. Xu, W., Chen, K., Zhao, T.: Document-level relation extraction with reconstruction. In: AAI (2021)
23. Xue, H.J., Dai, X., Zhang, J., Huang, S., Chen, J.: Deep matrix factorization models for recommender systems. In: IJCAI, vol. 17, Melbourne, Australia, pp. 3203–3209 (2017)
24. Yu, J., Gao, M., Li, J., Yin, H., Liu, H.: Adaptive implicit friends identification over heterogeneous network for social recommendation. In: CIKM, pp. 357–366 (2018)
25. Yu, X., et al.: Personalized entity recommendation: a heterogeneous information network approach. In: WSDM, pp. 283–292 (2014)
26. Zhao, T., McAuley, J., King, I.: Leveraging social connections to improve personalized ranking for collaborative filtering. In: CIKM, pp. 261–270 (2014)