Graph Neural Networks for Social Recommendation

Wenqi Fan
Department of Computer Science
City University of Hong Kong
wenqifan03@gmail.com

Yao Ma
Data Science and Engineering Lab
Michigan State University
mayao4@msu.edu

Qing Li
Department of Computing
The Hong Kong Polytechnic University
csqli@comp.polyu.edu.hk

Yuan He
JD.com
heyuan6@jd.com

Eric Zhao
JD.com
ericzhao@jd.com

Jiliang Tang
Data Science and Engineering Lab
Michigan State University
tangjili@msu.edu

Dawei Yin
JD.com
yindawei@acm.org

ABSTRACT

In recent years, Graph Neural Networks (GNNs), which can naturally integrate node information and topological structure, have been demonstrated to be powerful in learning on graph data. These advantages of GNNs provide great potential to advance social recommendation since data in social recommender systems can be represented as user-user social graph and user-item graph; and learning latent factors of users and items is the key. However, building social recommender systems based on GNNs faces challenges. For example, the user-item graph encodes both interactions and their associated opinions; social relations have heterogeneous strengths; users involve in two graphs (e.g., the user-user social graph and the user-item graph). To address the three aforementioned challenges simultaneously, in this paper, we present a novel graph neural network framework **(GraphRec)** for social recommendations. In particular, we provide a principled approach to jointly capture interactions and opinions in the user-item graph and propose the framework GraphRec, which coherently models two graphs and heterogeneous strengths. Extensive experiments on two real-world datasets demonstrate the effectiveness of the proposed framework GraphRec. Our code is available at https://github.com/wenqifan03/GraphRec-WWW19

CCS CONCEPTS

• **Information systems** → **Social recommendation**; • **Computing methodologies** → **Neural networks**; **Artificial intelligence**.

KEYWORDS

Social Recommendation; Graph Neural Networks; Recommender Systems; Social Network; Neural Networks

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313488

ACM Reference Format:
Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313488

1 INTRODUCTION

The exploitation of social relations for recommender systems has attracted increasing attention in recent years [18, 28, 30]. These social recommender systems have been developed based on the phenomenon that users usually acquire and disseminate information through those around them, such as classmates, friends, or colleagues, implying that the underlying social relations of users can play a significant role in helping them filter information [23]. Hence, social relations have been proven to be helpful in boosting the recommendation performance [8, 29].

Recent years have witnessed great developments in deep neural network techniques for graph data [15]. These deep neural network architectures are known as Graph Neural Networks (GNNs) [5, 10, 19], which have been proposed to learn meaningful representations for graph data. Their main idea is how to iteratively aggregate feature information from local graph neighborhoods using neural networks. Meanwhile, node information can be propagated through a graph after transformation and aggregation. Hence, GNNs naturally integrate the node information as well as the topological structure and have been demonstrated to be powerful in representation learning [5, 7, 15]. On the other hand, data in social recommendation can be represented as graph data with two graphs. As demonstrated in Figure 1, these two graphs include a social graph denoting the relationships between users, and a user-item graph denoting interactions between users and items. Users are simultaneously involved in both graphs, who can bridge them. Moreover, the natural way of social recommendation is to incorporate the social network information into user and item latent factors learning [37]. Learning representations of items and users is the key to build social recommender systems. Thus, given

their advantages, GNNs provide unprecedented opportunities to advance social recommendation.

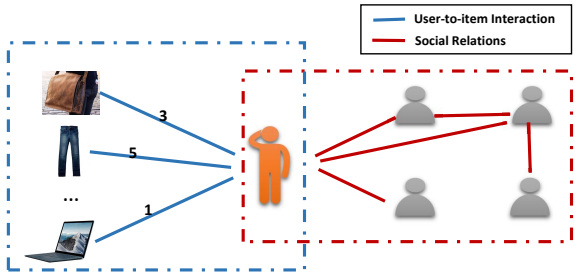


Figure 1: Graph Data in Social Recommendation. It contains two graphs including the user-item graph (left part) and the user-user social graph (right part). Note that the number on the edges of the user-item graph denotes the opinions (or rating score) of users on the items via the interactions.

Meanwhile, building social recommender systems based on GNNs faces challenges. The social graph and the user-item graph in a social recommender system provide information about users from different perspectives. It is important to aggregate information from both graphs to learn better user representations. Thus, the first challenge is how to inherently combine these two graphs. Moreover, the user-item graph not only contains interactions between users and items but also includes users' opinions on items. For example, as shown in Figure 1, the user interacts with the items of "trousers" and "laptop"; and the user likes "trousers" while disliking "laptop". Therefore, the second challenge is how to capture interactions and opinions between users and items jointly. In addition, the low cost of link formation in online worlds can result in networks with varied tie strengths (e.g., strong and weak ties are mixed together) [36]. Users are likely to share more similar tastes with strong ties than weak ties. Considering social relations equally could lead to degradation in recommendation performance. Hence, the third challenge is how to distinguish social relations with heterogeneous strengths.

In this paper, we aim to build social recommender systems based on graph neural networks. Specially, we propose a novel graph neural network **GraphRec** for social recommendations, which can address three aforementioned challenges simultaneously. Our major contributions are summarized as follows:

- We propose a novel graph neural network GraphRec, which can model graph data in social recommendations coherently;
- We provide a principled approach to jointly capture interactions and opinions in the user-item graph;
- We introduce a method to consider heterogeneous strengths of social relations mathematically; and
- We demonstrate the effectiveness of the proposed framework on various real-world datasets.

The remainder of this paper is organized as follows. We introduce the proposed framework in Section 2. In Section 3, we conduct experiments on two real-world datasets to illustrate the effectiveness of the proposed method. In Section 4, we review work related to our framework. Finally, we conclude our work with future directions in Section 5.

2 THE PROPOSED FRAMEWORK

In this section, we will first introduce the definitions and notations used in this paper, next give an overview about the proposed framework, then detail each model component and finally discuss how to learn the model parameters.

Table 1: Notation

Symbols	Definitions and Descriptions
r_{ij}	The rating value of item v_j by user u_i
$\mathbf{q}_j$	The embedding of item v_j
$\mathbf{p}_i$	The embedding of user u_i
$\mathbf{e}_r$	The opinion embedding for the rating level r, such as 5-star rating, $r \in \{1, 2, 3, 4, 5\}$
d	The length of embedding vector
$C(i)$	The set of items which user u_i interacted with
$N(i)$	The set of social friends who user u_i directly connected with
$B(j)$	The set of users who have interacted the item v_j
$\mathbf{h}_i^I$	The item-space user latent factor from item set $C(i)$ of user u_i
$\mathbf{h}_i^S$	The social-space user latent factor from the social friends $N(i)$ of user u_i
$\mathbf{h}_i$	The user latent factor of user u_i, combining from item space $\mathbf{h}_i^I$ and social space $\mathbf{h}_i^S$
$\mathbf{x}_{ia}$	The opinion-aware interaction representation of item v_a for user u_i
$\mathbf{f}_{jt}$	The opinion-aware interaction representation of user u_t for item v_j
$\mathbf{z}_j$	The item latent factor of item v_j
α_{ia}	The item attention of item v_a in contributing to $\mathbf{h}_i^I$
β_{io}	The social attention of neighboring user u_o in contributing to $\mathbf{h}_i^S$
μ_{jt}	The user attention of user u_t in contributing to $\mathbf{z}_j$
r'_{ij}	The predicted rating value of item v_j by user u_i
$\oplus$	The concatenation operator of two vectors
T	The user-user social graph
R	The user-item rating matrix (user-item graph)
W, **b**	The weight and bias in neural network

2.1 Definitions and Notations

Let $U = \{u_1, u_2, ..., u_n\}$ and $V = \{v_1, v_2, ..., v_m\}$ be the sets of users and items respectively, where n is the number of users, and m is the number of items. We assume that $\mathbf{R} \in \mathbb{R}^{n \times m}$ is the user-item rating matrix, which is also called the user-item graph. If u_i gives a rating to v_j, r_{ij} is the rating score, otherwise we employ 0 to represent the unknown rating from u_i to v_j, i.e., $r_{ij} = 0$. The observed rating score r_{ij} can be seen as user u_i's opinion on the item v_j. Let $O = \{\langle u_i, v_j \rangle | r_{ij} \neq 0\}$ be the set of known ratings

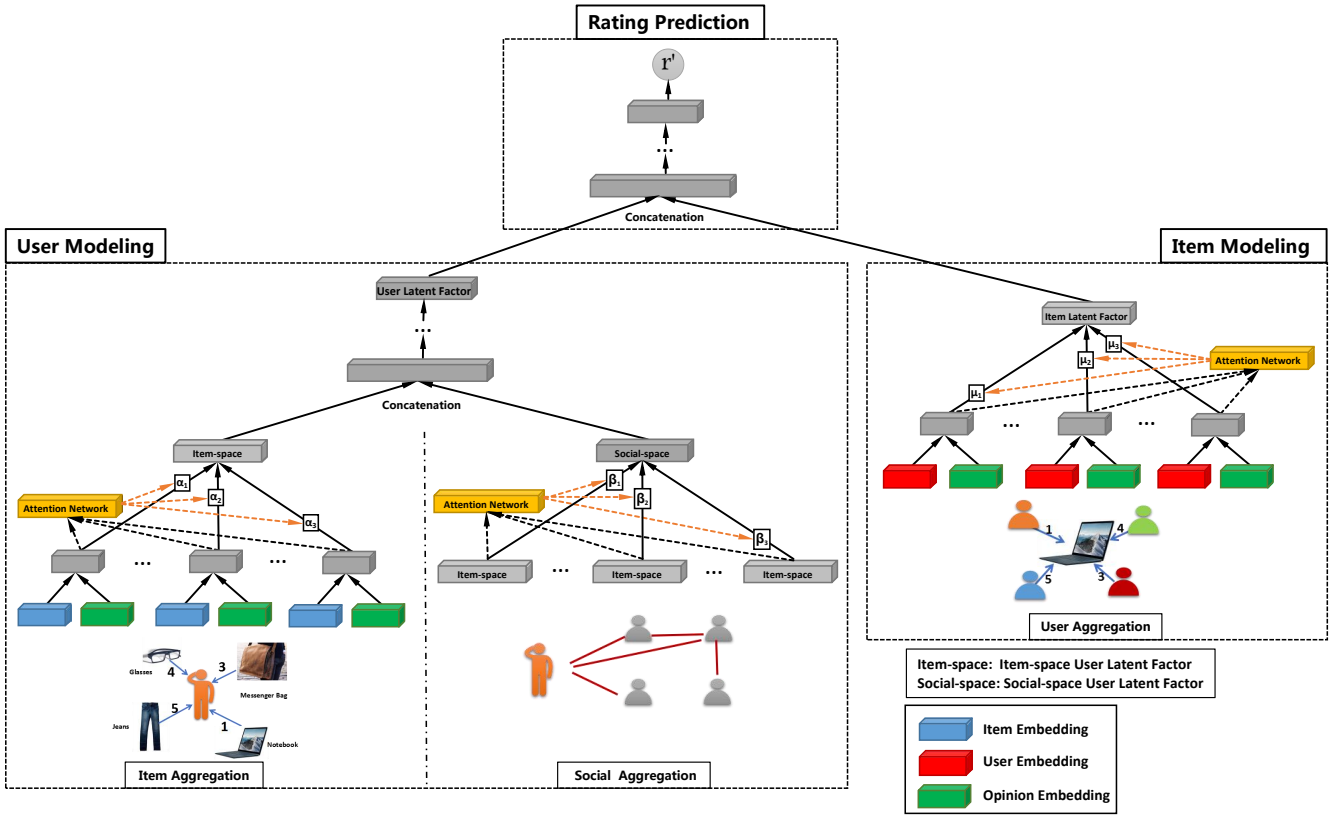


Figure 2: The overall architecture of the proposed model. It contains three major components: user modeling, item modeling, and rating prediction.

and $\mathcal{T} = \{\langle u_i, v_j \rangle | r_{ij} = 0\}$ be the set of unknown ratings. Let $N(i)$ be the set of users whom u_i directly connected with, $C(i)$ be the set of items which u_i have interacted with, and $B(j)$ be the set of users who have interacted with v_j. In addition, users can establish social relations to each other. We use $\mathrm{T} \in \mathbb{R}^{n \times n}$ to denote the user-user social graph, where $\mathrm{T}_{ij} = 1$ if u_j has a relation to u_i and zero otherwise. Given the user-item graph $\mathbf{R}$ and social graph T, we aim to predict the missing rating value in $\mathbf{R}$. Following [11], we use an embedding vector $\mathbf{p}_i \in \mathbb{R}^d$ to denote a user u_i and an embedding vector $\mathbf{q}_j \in \mathbb{R}^d$ to represent an item v_j, where d is the length of embedding vector. More details will be provided about these embedding vectors in the following subsections. The mathematical notations used in this paper are summarized in Table 1.

2.2 An Overview of the Proposed Framework

The architecture of the proposed model is shown in Figure 2. The model consists of three components: user modeling, item modeling, and rating prediction. The first component is user modeling, which is to learn latent factors of users. As data in social recommender systems includes two different graphs, i.e., a social graph and a user-item graph, we are provided with a great opportunity to learn user representations from different perspectives. Therefore, two aggregations are introduced to respectively process these two different graphs. One is item aggregation, which can be utilized to understand users via interactions between users and items in the user-item graph (or item-space). The other is social aggregation, the relationship between users in the social graph, which can help model users from the social perspective (or social-space). Then, it is intuitive to obtain user latent factors by combining information from both item space and social space. The second component is item modeling, which is to learn latent factors of items. In order to consider both interactions and opinions in the user-item graph, we introduce user aggregation, which is to aggregate users' opinions in item modeling. The third component is to learn model parameters via prediction by integrating user and item modeling components. Next, we will detail each model component.

2.3 User Modeling

User modeling aims to learn user latent factors, denoted as $\mathbf{h}_i \in \mathbb{R}^d$ for user u_i. The challenge is how to inherently combine the user-item graph and social graph. To address this challenge, we first use two types of aggregation to learn factors from two graphs, as shown in the left part in Figure 2. The first aggregation, denoted as item aggregation, is utilized to learn item-space user latent factor $\mathbf{h}_i^I \in \mathbb{R}^d$ from the user-item graph. The second aggregation is social aggregation where social-space user latent factor $\mathbf{h}_i^S \in \mathbb{R}^d$ is learned from the social graph. Then, these two factors are combined together to form the final user latent factors $\mathbf{h}_i$. Next, we will

introduce item aggregation, social aggregation and how to combine user latent factors from both item-space and social-space.

Item Aggregation. As user-item graph contains not only interactions between users and items but also users' opinions (or rating scores) on items, we provide a principled approach to jointly capture interactions and opinions in the user-item graph for learning item-space user latent factors $\mathbf{h}_i^I$, which is used to model user latent factor via interactions in the user-item graph.

The purpose of item aggregation is to learn item-space user latent factor $\mathbf{h}_i^I$ by considering items a user u_i has interacted with and users' opinions on these items. To mathematically represent this aggregation, we use the following function as:

$$\mathbf{h}_i^I = \sigma(\mathbf{W} \cdot Aggre_{items}(\{\mathbf{x}_{ia}, \forall a \in C(i)\}) + \mathbf{b}) \tag{1}$$

where $C(i)$ is the set of items user u_i has interacted with (or u_i's neighbors in the user-item graph), $\mathbf{x}_{ia}$ is a representation vector to denote opinion-aware interaction between u_i and an item v_a, and $Aggre_{items}$ is the items aggregation function. In addition, σ denotes non-linear activation function (i.e., a rectified linear unit), and $\mathbf{W}$ and $\mathbf{b}$ are the weight and bias of a neural network. Next we will discuss how to define opinion-aware interaction representation $\mathbf{x}_{ia}$ and the aggregation function $Aggre_{items}$.

A user can express his/her opinions (or rating scores), denoted as r, to items during user-item interactions. These opinions on items can capture users' preferences on items, which can help model item-space user latent factors. To model opinions, for each type of opinions r, we introduce an opinion embedding vector $\mathbf{e}_r \in \mathbb{R}^d$ that denotes each opinion r as a dense vector representation. For example, in a 5-star rating system, for each $r \in \{1, 2, 3, 4, 5\}$, we introduce an embedding vector $\mathbf{e}_r$. For an interaction between user u_i and item v_a with opinion r, we model opinion-aware interaction representation $\mathbf{x}_{ia}$ as a combination of item embedding $\mathbf{q}_a$ and opinion embedding $\mathbf{e}_r$ via a Multi-Layer Perceptron (MLP). It can be denoted as g_v to fuse the interaction information with the opinion information as shown in Figure 2. The MLP takes the concatenation of item embedding $\mathbf{q}_a$ and its opinion embedding $\mathbf{e}_r$ as input. The output of MLP is the opinion-aware representation of the interaction between u_i and v_a, $\mathbf{x}_{ia}$, as follows:

$$\mathbf{x}_{ia} = g_v([\mathbf{q}_a \oplus \mathbf{e}_r]) \tag{2}$$

where $\oplus$ denotes the concatenation operation between two vectors.

One popular aggregation function for $Aggre_{items}$ is the mean operator where we take the element-wise mean of the vectors in $\{\mathbf{x}_{ia}, \forall a \in C(i)\}$. This mean-based aggregator is a linear approximation of a localized spectral convolution [15], as the following function:

$$\mathbf{h}_i^I = \sigma(\mathbf{W} \cdot \left\{ \sum_{a \in C(i)} \alpha_i \mathbf{x}_{ia} \right\} + \mathbf{b}) \tag{3}$$

where α_i is fixed to $\frac{1}{|C(i)|}$ for all items in the mean-based aggregator. It assumes that all interactions contribute equally to understand the user u_i. However, this may not be optimal, due to the fact that the influence of interactions on users may vary dramatically. Hence, we should allow interactions to contribute differently to a user's latent factor by assigning each interaction a weight.

To alleviate the limitation of mean-based aggregator, inspired by attention mechanisms [3, 38], an intuitive solution is to tweak α_i to be aware of the target user u_i, i.e., assigning an individualized weight for each (v_a, u_i) pair,

$$\mathbf{h}_i^I = \sigma(\mathbf{W} \cdot \left\{ \sum_{a \in C(i)} \alpha_{ia} \mathbf{x}_{ia} \right\} + \mathbf{b}) \tag{4}$$

where α_{ia} denotes the attention weight of the interaction with v_a in contributing to user u_i's item-space latent factor when characterizing user u_i's preference from the interaction history $C(i)$. Specially, we parameterize the *item attention* α_{ia} with a two-layer neural network, which we call as the *attention network*. The input to the attention network is the opinion-aware representation $\mathbf{x}_{ia}$ of the interaction and the target user u_i's embedding $\mathbf{p}_i$. Formally, the attention network is defined as,

$$\alpha_{ia}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{x}_{ia} \oplus \mathbf{p}_i] + \mathbf{b}_1) + b_2 \tag{5}$$

The final attention weights are obtained by normalizing the above attentive scores using Softmax function, which can be interpreted as the contribution of the interaction to the item-space user latent factor of user u_i as:

$$\alpha_{ia} = \frac{exp(\alpha_{ia}^*)}{\sum_{a \in C(i)} exp(\alpha_{ia}^*)} \tag{6}$$

Social Aggregation. Due to the social correlation theories [20, 21], a user's preference is similar to or influenced by his/her directly connected social friends. We should incorporate social information to further model user latent factors. Meanwhile, tie strengths between users can further influence users' behaviors from the social graph. In other words, the learning of social-space user latent factors should consider heterogeneous strengths of social relations. Therefore, we introduce an attention mechanism to select social friends that are representative to characterize users social information and then aggregate their information.

In order to represent user latent factors from this social perspective, we propose social-space user latent factors, which is to aggregate the item-space user latent factors of neighboring users from the social graph. Specially, the social-space user latent factor of u_i, $\mathbf{h}_i^S$, is to aggregate the item-space user latent factors of users in u_i's neighbors $N(i)$, as the follows:

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot Aggre_{neigbhors}(\left\{\mathbf{h}_o^I, \forall o \in N(i)\right\}) + \mathbf{b}) \tag{7}$$

where $Aggre_{neigbhors}$ denotes the aggregation function on user's neighbors.

One natural aggregation function for $Aggre_{neigbhors}$ is also the mean operator which take the element-wise mean of the vectors in $\left\{\mathbf{h}_o^I, \forall o \in N(i)\right\}$, as the following function:

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot \left\{ \sum_{o \in N(i)} \beta_i \mathbf{h}_o^I \right\} + \mathbf{b}) \tag{8}$$

where β_i is fixed to $\frac{1}{|N(i)|}$ for all neighbors for the mean-based aggregator. It assumes that all neighbors contribute equally to the representation of user u_i. However, as mentioned before, strong and weak ties are mixed together in a social network, and users are likely to share more similar tastes with strong ties than weak ties.

Thus, we perform an attention mechanism with a two-layer neural network to extract these users that are important to influence u_i, and model their tie strengths, by relating *social attention* β_{io} with $\mathbf{h}_o^I$ and the target user embedding $\mathbf{p}_i$, as below,

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot \left\{ \sum_{o \in N(i)} \beta_{io} \mathbf{h}_o^I \right\} + \mathbf{b}) \tag{9}$$

$$\beta_{io}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{h}_o^I \oplus \mathbf{p}_i] + \mathbf{b}_1) + b_2 \tag{10}$$

$$\beta_{io} = \frac{exp(\beta_{io}^*)}{\sum_{o \in N(i)} exp(\beta_{io}^*)} \tag{11}$$

where the β_{io} can be seen as the strengths between users.

Learning User Latent Factor. In order to learn better user latent factors, item-space user latent factors and social-space user latent factors are needed to be considered together, since the social graph and the user-item graph provide information about users from different perspectives. We propose to combine these two latent factors to the final user latent factor via a standard MLP where the item-space user latent factor $\mathbf{h}_i^I$ and the social-space user latent factor $\mathbf{h}_i^S$ are concatenated before feeding into MLP. Formally, the user latent factor $\mathbf{h}_i$ is defined as,

$$\mathbf{c}_1 = \left[\mathbf{h}_i^I \oplus \mathbf{h}_i^S \right] \tag{12}$$

$$\mathbf{c}_2 = \sigma(\mathbf{W}_2 \cdot \mathbf{c}_1 + \mathbf{b}_2) \tag{13}$$

$$\dots$$

$$\mathbf{h}_i = \sigma(\mathbf{W}_l \cdot \mathbf{c}_{l-1} + \mathbf{b}_l) \tag{14}$$

where l is the index of a hidden layer.

2.4 Item Modeling

As shown in the right part of Figure 2, item modeling is used to learn item latent factor, denoted as $\mathbf{z}_j$, for the item v_j by user aggregation. Items are associated with the user-item graph, which contains interactions as well as user's opinions. Therefore, interactions and opinions in the user-item graph should be jointly captured to further learn item latent factors.

User Aggregation. Likewise, we use a similar method as learning item-space user latent factors via item aggregation. For each item v_j, we need to aggregate information from the set of users who have interacted with v_j, denoted as $B(j)$.

Even for the same item, users might express different opinions during user-item interactions. These opinions from different users can capture the characteristics of the same item in different ways provided by users, which can help model item latent factors. For an interaction from u_t to v_j with opinion r, we introduce an opinion-aware interaction user representation $\mathbf{f}_{jt}$, which is obtained from the basic user embedding $\mathbf{p}_t$ and opinion embedding $\mathbf{e}_r$ via a MLP, denoted as g_u. g_u is to fuse the interaction information with the opinion information, as shown in Figure 2:

$$\mathbf{f}_{jt} = g_u([\mathbf{p}_t \oplus \mathbf{e}_r]) \tag{15}$$

Then, to learn item latent factor $\mathbf{z}_j$, we also propose to aggregate opinion-aware interaction representation of users in $B(j)$ for item v_j. The users aggregation function is denoted as $Aggre_{users}$, which is to aggregate opinion-aware interaction representation of users in $\{\mathbf{f}_{jt}, \forall t \in B(j)\}$ as:

$$\mathbf{z}_j = \sigma(\mathbf{W} \cdot Aggre_{users}(\{\mathbf{f}_{jt}, \forall t \in B(j)\}) + \mathbf{b}) \tag{16}$$

In addition, we introduce an attention mechanism to differentiate the importance weight μ_{jt} of users with a two-layer neural attention network, taking $\mathbf{f}_{jt}$ and $\mathbf{q}_j$ as the input,

$$\mathbf{z}_j = \sigma(\mathbf{W} \cdot \left\{ \sum_{t \in B(j)} \mu_{jt} \mathbf{f}_{jt} \right\} + \mathbf{b}) \tag{17}$$

$$\mu_{jt}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{f}_{jt} \oplus \mathbf{q}_j] + \mathbf{b}_1) + b_2 \tag{18}$$

$$\mu_{jt} = \frac{exp(\mu_{jt}^*)}{\sum_{t \in B_{(j)}} exp(\mu_{jt}^*)} \tag{19}$$

This *user attention* μ_{jt} is to capture heterogeneous influence from user-item interactions on learning item latent factor.

2.5 Rating Prediction

In this subsection, we will design recommendation tasks to learn model parameters. There are various recommendation tasks such as item ranking and rating prediction. In this work, we apply the proposed GraphRec model for the recommendation task of rating prediction. With the latent factors of users and items (i.e., $\mathbf{h}_i$ and $\mathbf{z}_j$), we can first concatenate them $\left[\mathbf{h}_i \oplus \mathbf{z}_j\right]$ and then feed it into MLP for rating prediction as:

$$\mathbf{g}_1 = \left[\mathbf{h}_i \oplus \mathbf{z}_j\right] \tag{20}$$

$$\mathbf{g}_2 = \sigma(\mathbf{W}_2 \cdot \mathbf{g}_1 + \mathbf{b}_2) \tag{21}$$

$$\dots$$

$$\mathbf{g}_{l-1} = \sigma(\mathbf{W}_l \cdot \mathbf{g}_{l-1} + \mathbf{b}_l) \tag{22}$$

$$r'_{ij} = \mathbf{w}^T \cdot \mathbf{g}_{l-1} \tag{23}$$

where l is the index of a hidden layer, and r'_{ij} is the predicted rating from u_i to v_j.

2.6 Model Training

To estimate model parameters of GraphRec, we need to specify an objective function to optimize. Since the task we focus on in this work is rating prediction, a commonly used objective function is formulated as,

$$Loss = \frac{1}{2\,|O|} \sum_{i,j \in O} (r'_{ij} - r_{ij})^2 \tag{24}$$

where $|O|$ is the number of observed ratings , and r_{ij} is the ground truth rating assigned by the user i on the item j.

To optimize the objective function, we adopt the RMSprop [31] as the optimizer in our implementation, rather than the vanilla SGD. At each time, it randomly selects a training instance and updates each model parameter towards the direction of its negative gradient. There are three embedding in our model, including item embedding $\mathbf{q}_j$, user embedding $\mathbf{p}_i$, and opinion embedding $\mathbf{e}_r$. They are randomly initialized and jointly learned during the training stage. We do not use one-hot vectors to represent each user and item, since the raw features are very large and highly sparse. By embedding high-dimensional sparse features into a low-dimensional latent space, the model can be easy to train [11].

Opinion embedding matrix **e** depends on the rating scale of the system. For example, for a 5-star rating system, opinion embedding matrix **e** contains 5 different embedding vectors to denote scores in $\{1, 2, 3, 4, 5\}$. Overfitting is a perpetual problem in optimizing deep neural network models. To alleviate this issue, the dropout strategy [26] has been applied to our model. The idea of dropout is to randomly drop some neurons during the training process. When updating parameters, only part of them will be updated. Moreover, as dropout is disabled during testing, the whole network is used for prediction.

3 EXPERIMENT

3.1 Experimental Settings

3.1.1 Datasets. In our experiments, we choose two representative datasets Ciao and Epinions[1], which are taken from popular social networking websites Ciao (http://www.ciao.co.uk) and Epinions (www.epinions.com). Each social networking service allows users to rate items, browse/write reviews, and add friends to their 'Circle of Trust'. Hence, they provide a large amount of rating information and social information. The ratings scale is from 1 to 5. We randomly initialize opinion embedding with 5 different embedding vectors based on 5 scores in $\{1, 2, 3, 4, 5\}$. The statistics of these two datasets are presented in Table 2.

Table 2: Statistics of the datasets

Dataset	Ciao	Epinions
# of Users	7,317	18,088
# of Items	10,4975	261,649
# of Ratings	283,319	764,352
# of Density (Ratings)	0.0368%	0.0161%
# of Social Connections	111,781	355,813
# of Density (Social Relations)	0.2087%	0.1087%

3.1.2 Evaluation Metrics. In order to evaluate the quality of the recommendation algorithms, two popular metrics are adopted to evaluate the predictive accuracy, namely Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) [34]. Smaller values of MAE and RMSE indicate better predictive accuracy. Note that small improvement in RMSE or MAE terms can have a significant impact on the quality of the top-few recommendations [16].

3.1.3 Baselines. To evaluate the performance, we compared our GraphRec with three groups of methods including traditional recommender systems, traditional social recommender systems, and deep neural network based recommender systems. For each group, we select representative baselines and below we will detail them.

- **PMF** [24]: **P**robabilistic **M**atrix **F**actorization utilizes user-item rating matrix only and models latent factors of users and items by Gaussian distributions.
- **SoRec** [17]: **So**cial **Rec**ommendation performs co-factorization on the user-item rating matrix and user-user social relations matrix.
- **SoReg** [18]: **So**cial **Reg**ularization models social network information as regularization terms to constrain the matrix factorization framework.
- **SocialMF** [13]: It considers the trust information and propagation of trust information into the matrix factorization model for recommender systems.
- **TrustMF** [37]: This method adopts matrix factorization technique that maps users into two low-dimensional spaces: truster space and trustee space, by factorizing trust networks according to the directional property of trust.
- **NeuMF** [11]: This method is a state-of-the-art matrix factorization model with neural network architecture. The original implementation is for recommendation ranking task and we adjust its loss to the squared loss for rating prediction.
- **DeepSoR** [8]: This model employs a deep neural network to learn representations of each user from social relations, and to integrate into probabilistic matrix factorization for rating prediction.
- **GCMC+SN** [1]: This model is a state-of-the-art recommender system with graph neural network architecture. In order to incorporate social network information into GCMC, we utilize the *node2vec* [9] to generate user embedding as user side information, instead of using the raw feature social connections ($\mathrm{T} \in \mathbb{R}^{n \times n}$) directly. The reason is that the raw feature input vectors is highly sparse and high-dimensional. Using the network embedding techniques can help compress the raw input feature vector to a low-dimensional and dense vector, then the model can be easy to train.

PMF and NeuMF are pure collaborative filtering model without social network information for rating prediction, while the others are social recommendation. Besides, we compared GraphRec with two state-of-the-art neural network based social recommender systems, i.e., DeepSoR and GCMC+SN.

3.1.4 Parameter Settings. We implemented our proposed method on the basis of Pytorch[2], a well-known Python library for neural networks. For each dataset, we used $x\%$ as a training set to learning parameters, $(1-x\%)/2$ as a validation set to tune hyper-parameters, and $(1-x\%)/2$ as a testing set for the final performance comparison, where x was varied as $\{80\%, 60\%\}$. For the embedding size d, we tested the value of [8, 16, 32, 64, 128, 256]. The batch size and learning rate was searched in [32, 64, 128, 512] and [0.0005, 0.001, 0.005, 0.01, 0.05, 0.1], respectively. Moreover, we empirically set the size of the hidden layer the same as the embedding size and the activation function as ReLU. Without special mention, we employed three hidden layers for all the neural components. The early stopping strategy was performed, where we stopped training if the RMSE on validation set increased for 5 successive epochs. For all neural network methods, we randomly initialized model parameters with a Gaussian distribution, where the mean and standard deviation is 0 and 0.1, respectively. The parameters for the baseline algorithms were initialized as in the corresponding papers and were then carefully tuned to achieve optimal performance.

[1] http://www.cse.msu.edu/~tangjili/index.html

[2] https://pytorch.org/

Table 3: Performance comparison of different recommender systems

Training	Metrics	Algorithms								
		PMF	SoRec	SoReg	SocialMF	TrustMF	NeuMF	DeepSoR	GCMC+SN	**GraphRec**
Ciao (60%)	MAE	0.952	0.8489	0.8987	0.8353	0.7681	0.8251	0.7813	0.7697	0.7540
	RMSE	1.1967	1.0738	1.0947	1.0592	1.0543	1.0824	1.0437	1.0221	1.0093
Ciao (80%)	MAE	0.9021	0.8410	0.8611	0.8270	0.7690	0.8062	0.7739	0.7526	0.7387
	RMSE	1.1238	1.0652	1.0848	1.0501	1.0479	1.0617	1.0316	0.9931	0.9794
Epinions (60%)	MAE	1.0211	0.9086	0.9412	0.8965	0.8550	0.9097	0.8520	0.8602	0.8441
	RMSE	1.2739	1.1563	1.1936	1.1410	1.1505	1.1645	1.1135	1.1004	1.0878
Epinions (80%)	MAE	0.9952	0.8961	0.9119	0.8837	0.8410	0.9072	0.8383	0.8590	0.8168
	RMSE	1.2128	1.1437	1.1703	1.1328	1.1395	1.1476	1.0972	1.0711	1.0631

3.2 Performance Comparison of Recommender Systems

We first compare the recommendation performance of all methods. Table 3 shows the overall rating prediction error $w.r.t.$ RMSE and MAE among the recommendation methods on Ciao and Epinions datasets. We have the following main findings:

- SoRec, SoReg, SocialMF, and TrustMF always outperform PMF. All of these methods are based on matrix factorization. SoRec, SoReg, SocialMF, and TrustMF leverage both the rating and social network information; while PMF only uses the rating information. These results support that social network information is complementary to rating information for recommendations.
- NeuMF obtains much better performance than PMF. Both methods only utilize the rating information. However, NeuMF is based on neural network architecture, which suggests the power of neural network models in recommender systems.
- DeepSoR and GCMC+SN perform better than SoRec, SoReg, SocialMF, and TrustMF. All of them take advantage of both rating and social network information. However, DeepSoR and GCMC+SN are based on neural network architectures, which further indicate the power of neural network models in recommendations.
- Among baselines, GCMC+SN shows quite strong performance. It implies that the GNNs are powerful in representation learning for graph data, since it naturally integrates the node information as well as topological structure.
- Our method GraphRec consistently outperforms all the baseline methods. Compared to DeepSoR and GCMC+SN, our model provides advanced model components to integrate rating and social network information. In addition, our model provides a way to consider both interactions and opinions in the user-item graph. We will provide further investigations to better understand the contributions of model components to the proposed framework in the following subsection.

To sum up, the comparison results suggest (1) social network information is helpful for recommendations; (2) neural network models can boost recommendation performance and (3) the proposed framework outperforms representative baselines.

3.3 Model Analysis

In this subsection, we study the impact of model components and model hyper-parameters.

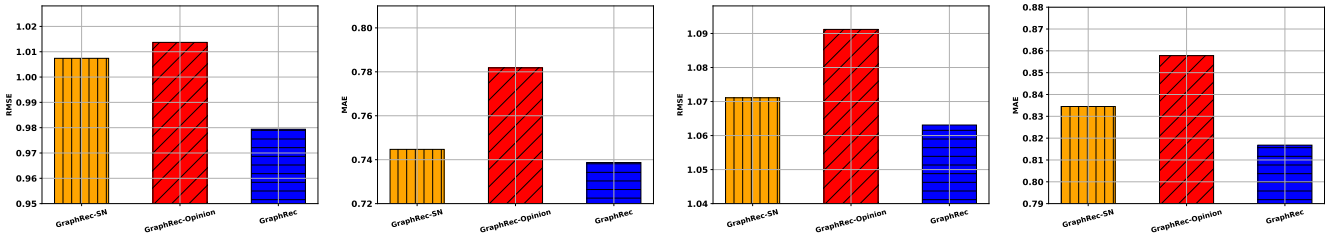
3.3.1 Effect of Social Network and User Opinions. In the last subsection, we have demonstrated the effectiveness of the proposed framework. The proposed framework provides model components to (1) integrate social network information and (2) incorporate users' opinions about the interactions with items. To understand the working of GraphRec, we compare GraphRec with its two variants: GraphRec-SN, and GraphRec-Opinion. These two variants are defined in the following:

- GraphRec-SN: The social network information of GraphRec is removed . This variant only uses the item-space user latent factor $\mathbf{h}_i^I$ to represent user latent factors $\mathbf{h}_i$; while ignoring the social-space user latent factors $\mathbf{h}_i^S$.
- GraphRec-Opinion: For learning item-space user latent factor and item latent factor, the opinion embedding is removed during learning $\mathbf{x}_{ia}$ and $\mathbf{f}_{jt}$. This variant ignores the users' opinion on the user-item interactions.

The performance of GraphRec and its variants on Ciao and Epinions are given in Figure 3. From the results, we have the following findings:

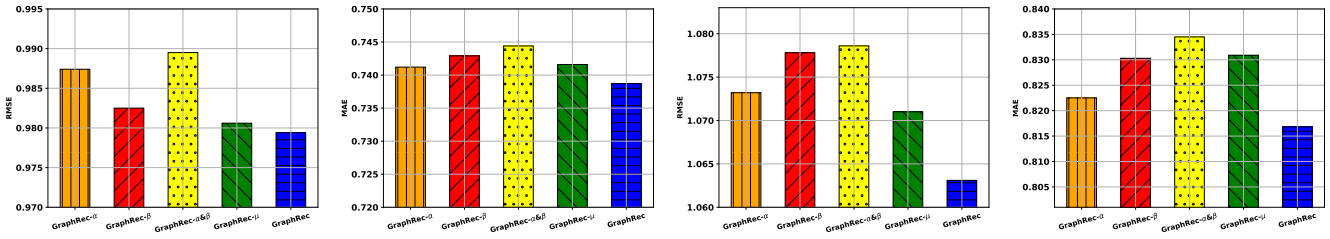
- **Social Network Information.** We now focus on analyzing the effectiveness of social network information. GraphRec-SN performs worse than GraphRec. It verifies that social network information is important to learn user latent factors and boost the recommendation performance.
- **Opinions in Interaction.** We can see that without opinion information, the performance of rating prediction is deteriorated significantly. For example, on average, the relative reduction on Ciao and Epinions is 3.50% and 2.64% on RMSE metric, and 5.84% and 5.02% on MAE metric, respectively. It justifies our assumption that opinions on user-item interactions have informative information that can help to learn user or item latent factors and improve the performance of recommendation.

3.3.2 Effect of Attention Mechanisms. To get a better understanding of the proposed GraphRec model, we further evaluate the key components of GraphRec - *Attention mechanisms*. There are three different attention mechanisms during aggregation, including item attention α, social attention β, and user attention μ. We



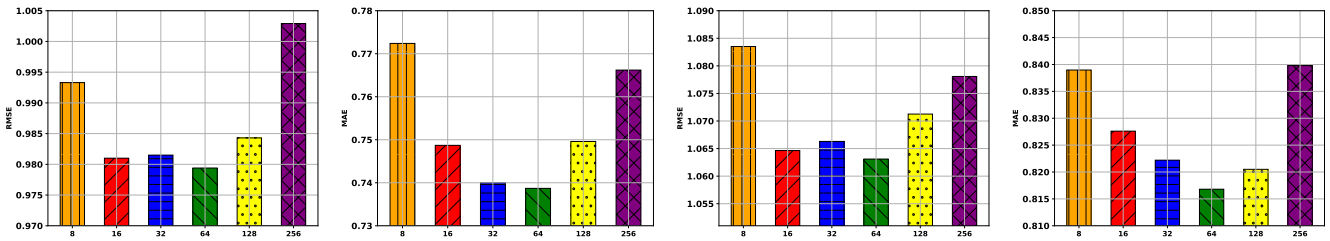
(a) Ciao-RMSE (b) Ciao-MAE (c) Epinions-RMSE (d) Epinions-MAE

Figure 3: Effect of social network and user opinions on Ciao and Epinions datasets.



(a) Ciao-RMSE (b) Ciao-MAE (c) Epinions-RMSE (d) Epinions-MAE

Figure 4: Effect of attention mechanisms on Ciao and Epinions datasets.



(a) Ciao-RMSE (b) Ciao-MAE (c) Epinions-RMSE (d) Epinions-MAE

Figure 5: Effect of embedding size on Ciao and Epinions datasets.

compare GraphRec with its four variants: GraphRec-α, GraphRec-β, GraphRec-$\alpha\&\beta$, and GraphRec-μ. These four variants are defined in the following:

- GraphRec-α: The item attention α of GraphRec is eliminated during aggregating the opinion-aware interaction representation of items. This variant employs the mean-based aggregation function on item aggregation for modeling item-space user latent factors.
- GraphRec-β: The social attention α is to model users' tie strengths. The social attention α of GraphRec in this variant is eliminated during aggregating user's neighbors. This variant employs the mean-based aggregation function on social aggregation for modeling social-space user latent factors.
- GraphRec-$\alpha\&\beta$: This variant eliminates two attention mechanisms (item attention α and social attention β) on item aggregation and social aggregation for modeling user latent factors.
- GraphRec-μ: The user attention μ of GraphRec is eliminated during aggregating opinion-aware interaction user representation. This variant employs the mean-based aggregation function on user aggregation for modeling item latent factors.

The results of different attention mechanisms on GraphRec are shown in Figure 4. From the results, we have the following findings,

- Not all interacted items (purchased history) of one user contribute equally to the item-space user latent factor, and not all interacted users (buyers) have the same importance to learning item latent factor. Based on these assumptions,

our model considers these difference among users and items by using two different attention mechanisms (α and μ). From the results, we can observe that GraphRec-α and GraphRec-μ obtain worse performance than GraphRec. These results demonstrate the benefits of the attention mechanisms on item aggregation and user aggregation.

- As mentioned before, users are likely to share more similar tastes with strong ties than weak ties. The attention mechanism β at social aggregation considers heterogeneous strengths of social relations. When the attention mechanism β is removed, the performance of GraphRec-β is dropped significantly. It justifies our assumption that during social aggregation, different social friends should have different influence for learning social-space user latent factor. It's important to distinguish social relations with heterogeneous strengths.

To sum up, GraphRec can capture the heterogeneity in aggregation operations of the proposed framework via attention mechanisms, which can boost the recommendation performance.

3.3.3 Effect of Embedding Size. In this subsection, to analyze the effect of embedding size of user embedding $\mathbf{p}$, item embedding $\mathbf{q}$, and opinion embedding $\mathbf{e}$, on the performance of our model.

Figure 5 presents the performance comparison $w.r.t.$ the length of embedding of our proposed model on Ciao and Epinions datasets. In general, with the increase of the embedding size, the performance first increases and then decreases. When increasing the embedding size from 8 to 64 can improve the performance significantly. However, with the embedding size of 256, GraphRec degrades the performance. It demonstrates that using a large number of the embedding size has powerful representation. Nevertheless, if the length of embedding is too large, the complexity of our model will significantly increase. Therefore, we need to find a proper length of embedding in order to balance the trade-off between the performance and the complexity.

4 RELATED WORK

In this section, we briefly review some related work about social recommendation, deep neural network techniques employed for recommendation, and the advanced graph neural networks.

Exploiting social relations for recommendations has attracted significant attention in recent years [27, 28, 37]. One common assumption about these models is that a user's preference is similar to or influenced by the people around him/her (nearest neighbours), which can be proven by social correlation theories [20, 21]. Along with this line, SoRec [17] proposed a co-factorization method, which shares a common latent user-feature matrix factorized by ratings and by social relations. TrustMF [37] modeled mutual influence between users, and mapped users into two low-dimensional spaces: truster space and trustee space, by factorizing social trust networks. SoDimRec [30] first adopted a community detection algorithm to partition users into several clusters, and then exploited the heterogeneity of social relations and weak dependency connections for recommendation. Comprehensive overviews on social recommender systems can be found in surveys [29].

In recent years, deep neural network models had a great impact on learning effective feature representations in various fields, such as speech recognition [12], Computer Vision (CV) [14] and Natural Language Processing (NLP) [4]. Some recent efforts have applied deep neural networks to recommendation tasks and shown promising results [41], but most of them used deep neural networks to model audio features of music [32], textual description of items [3, 33], and visual content of images [40]. Besides, NeuMF [11] presented a Neural Collaborative Filtering framework to learn the non-linear interactions between users and items.

However, the application of deep neural network in social recommender systems is rare until very recently. In particular, NSCR [35] extended the NeuMF [11] model to cross-domain social recommendations, i.e., recommending items of information domains to potential users of social networks, and presented a neural social collaborative ranking recommender system. However, the limitation is NSCR requires users with one or more social networks accounts (e.g., Facebook, Twitter, Instagram), which limits the data collections and its applications in practice. SMR-MNRL [42] developed social-aware movie recommendation in social media from the viewpoint of learning a multimodal heterogeneous network representation for ranking. They exploited the recurrent neural network and convolutional neural network to learn the representation of movies' textual description and poster image, and adopted a random-walk based learning method into multimodal neural networks. In all these works [35] [42], they addressed the task of cross-domain social recommendations for ranking metric, which is different from traditional social recommender systems.

Most related to our task with neural networks includes DLMF [6] and DeepSoR [8]. DLMF [6] used auto-encoder on ratings to learn representation for initializing an existing matrix factorization. A two-phase trust-aware recommendation process is proposed to utilize deep neural networks in matrix factorization's initialization and to synthesize the user's interests and their trust friends' interests together with the impact of community effect based on matrix factorization for recommendations. DeepSoR [8] integrated neural networks for user's social relations into probabilistic matrix factorization. They first represented users using pre-trained node embedding technique, and further exploited k-nearest neighbors to bridge user embedding features and neural network.

More recently, Graph Neural Networks (GNNs) have been proven to be capable of learning on graph structure data [2, 5, 7, 15, 25]. In the task of recommender systems, the user-item interaction contains the ratings on items by users, which is a typical graph data. Therefore, GNNs have been proposed to solve the recommendation problem [1, 22, 39]. sRMGCNN [22] adopted GNNs to extract graph embeddings for users and items, and then combined with recurrent neural network to perform a diffusion process. GCMC [1] proposed a graph auto-encoder framework, which produced latent features of users and items through a form of differentiable message passing on the user-item graph. PinSage [39] proposed a random-walk graph neural network to learn embedding for nodes in web-scale graphs. Despite the compelling success achieved by previous work, little attention has been paid to social recommendation with GNNs. In this paper, we propose a graph neural network for social recommendation to fill this gap.

5 CONCLUSION AND FUTURE WORK

We have presented a Graph Network model (GraphRec) to model social recommendation for rating prediction. Particularly, we provide a principled approach to jointly capture interactions and opinions in the user-item graph. Our experiments reveal that the opinion information plays a crucial role in the improvement of our model performance. In addition, our GraphRec can differentiate the ties strengths by considering heterogeneous strengths of social relations. Experimental results on two real-world datasets show that GraphRec can outperform state-of-the-art baselines.

Currently we only incorporate the social graph into recommendation, while many real-world industries are associated rich other side information on users as well as items. For example, users and items are associated with rich attributes. Therefore, exploring graph neural networks for recommendation with attributes would be an interesting future direction. Beyond that, now we consider both rating and social information static. However, rating and social information are naturally dynamic. Hence, we will consider building dynamic graph neural networks for social recommendations with dynamic.

ACKNOWLEDGMENTS

The work described in this paper has been supported, in part, by a general research fund from the Hong Kong Research Grants Council (project PolyU 1121417/17E), and an internal research grant from the Hong Kong Polytechnic University (project 1.9B0V). Yao Ma and Jiliang Tang are supported by the National Science Foundation (NSF) under grant numbers IIS-1714741, IIS-1715940 and CNS-1815636, and a grant from Criteo Faculty Research Award.

REFERENCES

[1] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).

[2] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42.

[3] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *Proceedings of the 27th International Conference on World Wide Web*. 1583–1592.

[4] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter* 19, 2 (2017), 25–35.

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*. 3844–3852.

[6] Shuiguang Deng, Longtao Huang, Guandong Xu, Xindong Wu, and Zhaohui Wu. 2017. On deep learning for trust-aware recommendations in social networks. *IEEE transactions on neural networks and learning systems* 28, 5 (2017), 1164–1177.

[7] Tyler Derr, Yao Ma, and Jiliang Tang. 2018. Signed Graph Convolutional Networks. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 929–934.

[8] Wenqi Fan, Qing Li, and Min Cheng. 2018. Deep Modeling of Social Relations for Recommendation. In *AAAI*.

[9] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 855–864.

[10] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW*. 173–182.

[12] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29, 6 (2012), 82–97.

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

[14] Hamid Karimi, Jiliang Tang, and Yanen Li. 2018. Toward End-to-End Deception Detection in Videos. In *2018 IEEE Big Data*. IEEE, 1278–1283.

[15] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

[16] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[17] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. 2008. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and Knowledge Management*. ACM, 931–940.

[18] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web Search and Data Mining*. ACM, 287–296.

[19] Yao Ma, Suhang Wang, Charu C. Aggarwal, Dawei Yin, and Jiliang Tang. 2019. Multi-dimensional Graph Convolutional Networks. In *Proceedings of the 2019 SIAM International Conference on Data Mining(SDM)*.

[20] Peter V Marsden and Noah E Friedkin. 1993. Network studies of social influence. *Sociological Methods & Research* 22, 1 (1993), 127–151.

[21] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 27, 1 (2001), 415–444.

[22] Federico Monti, Michael Bronstein, and Xavier Bresson. 2017. Geometric matrix completion with recurrent multi-graph neural networks. In *Advances in Neural Information Processing Systems*. 3700–3710.

[23] Paul Resnick and Hal R Varian. 1997. Recommender systems. *Commun. ACM* 40, 3 (1997), 56–58.

[24] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *21th Conference on Neural Information Processing Systems*, Vol. 1. 2–1.

[25] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine* 30, 3 (2013), 83–98.

[26] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[27] Jiliang Tang, Charu Aggarwal, and Huan Liu. 2016. Recommendations in signed social networks. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 31–40.

[28] Jiliang Tang, Xia Hu, Huiji Gao, and Huan Liu. 2013. Exploiting local and global social context for recommendation.. In *IJCAI*, Vol. 13. 2712–2718.

[29] Jiliang Tang, Xia Hu, and Huan Liu. 2013. Social recommendation: a review. *Social Network Analysis and Mining* 3, 4 (2013), 1113–1133.

[30] Jiliang Tang, Suhang Wang, Xia Hu, Dawei Yin, Yingzhou Bi, Yi Chang, and Huan Liu. 2016. Recommendation with Social Dimensions. In *AAAI*. 251–257.

[31] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-RMSProp, COURSERA: Neural networks for machine learning. *University of Toronto, Technical Report* (2012).

[32] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in neural Information Processing Systems*. 2643–2651.

[33] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[34] Suhang Wang, Jiliang Tang, Yilin Wang, and Huan Liu. 2018. Exploring Hierarchical Structures for Recommender Systems. *IEEE Transactions on Knowledge and Data Engineering* 30, 6 (2018), 1022–1035.

[35] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 185–194.

[36] Rongjing Xiang, Jennifer Neville, and Monica Rogati. 2010. Modeling relationship strength in online social networks. In *Proceedings of the 19th international conference on World wide web*. ACM, 981–990.

[37] Bo Yang, Yu Lei, Jiming Liu, and Wenjie Li. 2017. Social collaborative filtering by trust. *IEEE transactions on pattern analysis and machine intelligence* 39, 8 (2017), 1633–1647.

[38] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[39] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD '18*. ACM, 974–983.

[40] Lili Zhao, Zhongqi Lu, Sinno Jialin Pan, and Qiang Yang. 2016. Matrix Factorization+ for Movie Recommendation. In *IJCAI*. 3945–3951.

[41] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning. In *KDD'18*. ACM, 1040–1048.

[42] Zhou Zhao, Qifan Yang, Hanqing Lu, Tim Weninger, Deng Cai, Xiaofei He, and Yueting Zhuang. 2018. Social-aware movie recommendation via multimodal network learning. *IEEE Transactions on Multimedia* 20, 2 (2018), 430–440.