Relational Collaborative Filtering:
Modeling Multiple Item Relations for Recommendation

Xin Xin
School of Computing Science
University of Glasgow
x.xin.1@research.gla.ac.uk

Xiangnan He
School of Information Science and
Technology, USTC
xiangnanhe@gmail.com

Yongfeng Zhang
Department of Computer Science
Rutgers University
yongfeng.zhang@rutgers.edu

Yongdong Zhang
School of Information Science and
Technology, USTC
zhyd73@ustc.edu.cn

Joemon M. Jose
School of Computing Science
University of Glasgow
Joemon.Jose@glasgow.ac.uk

ABSTRACT

Existing item-based collaborative filtering (ICF) methods leverage only the relation of *collaborative similarity* — i.e., the item similarity evidenced by user interactions like ratings and purchases. Nevertheless, there exist multiple relations between items in real-world scenarios, e.g., two movies share the same director, two products complement with each other, etc. Distinct from the collaborative similarity that implies co-interact patterns from the user's perspective, these relations reveal fine-grained knowledge on items from different perspectives of meta-data, functionality, etc. However, how to incorporate multiple item relations is less explored in recommendation research.

In this work, we propose *Relational Collaborative Filtering* (RCF) to exploit multiple item relations in recommender systems. We find that both the relation *type* (e.g., shared director) and the relation *value* (e.g., Steven Spielberg) are crucial in inferring user preference. To this end, we develop a two-level hierarchical attention mechanism to model user preference — the first-level attention discriminates which types of relations are more important, and the second-level attention considers the specific relation values to estimate the contribution of a historical item. To make the item embeddings be reflective of the relational structure between items, we further formulate a task to preserve the item relations, and jointly train it with user preference modeling. Empirical results on two real datasets demonstrate the strong performance of RCF[1]. Furthermore, we also conduct qualitative analyses to show the benefits of explanations brought by RCF's modeling of multiple item relations.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Retrieval models and ranking**; **Novelty in information retrieval**.

---

[1]Codes are available at https://github.com/XinGla/RCF.

KEYWORDS

Collaborative Filtering, Attention Mechanism, Relation Learning

1 INTRODUCTION

Recommender system has been widely deployed in Web applications to address the information overload issue, such as E-commerce platforms, news portals, lifestyle apps, etc. It not only can facilitate the information-seeking process of users, but also can increase the traffic and bring profits to the service provider [1]. Among the various recommendation methods, item-based collaborative filtering (ICF) stands out owing to its interpretability and effectiveness [16, 20], being highly preferred in industrial applications [8, 10, 32]. The key assumption of ICF is that a user shall prefer the items that are similar to her historically interacted items [31, 39, 45]. The similarity is typically judged from user interactions — how likely two items are co-interacted by users in the past.

Despite prevalence and effectiveness, we argue that existing ICF methods are insufficient, since they only consider the *collaborative similarity* relation, which is macro-level, coarse-grained and lacks of concrete semantics. In real-world applications, there typically exist multiple relations between items that have concrete semantics, and they are particularly helpful to understand user behaviors. For example, in the movie domain, some movies may share the same director, actors, or other attributes; in E-commerce, some products may have the same functionality, similar image, etc. These relations reflect the similarity of items from different perspectives, and more importantly, they could affect the decisions of different users differently. For example, after two users ($u1$ and $u2$) watch the same movie "E.T. the Extra-Terrestrial", $u1$ likes the director and chooses "Schindler's List" to watch next, while $u2$ likes the fiction theme and watches "The Avenger" in the next. Without explicitly modeling such micro-level and fine-grained relations between items, it is conceptually difficult to reveal the true reasons behind a user's decision, not to mention to recommend desired items with persuasive explanations like *"The Avenger" is recommended to*
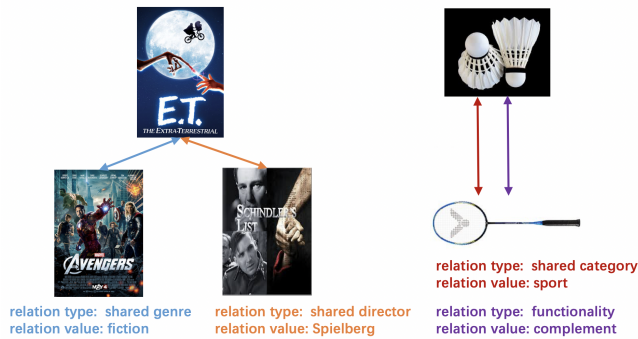
Figure 1: An example of multiple item relations. Each relation is described with a two-level hierarchy of type and value. Multiple relations may exist between two items and the same value may occur in relations of different types.
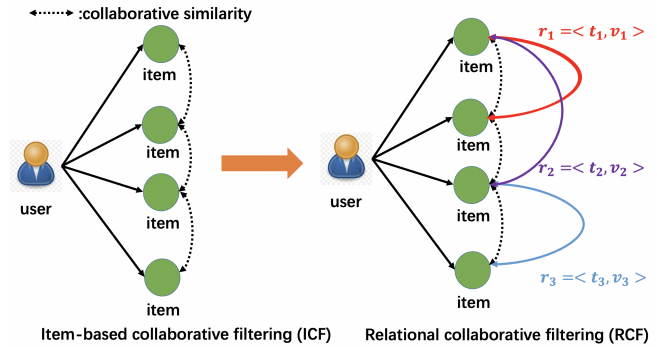

Figure 2: Comparison between ICF and RCF. The links between items of ICF are implicit and single, which denote the collaborative similarity. However, the links between items of RCF are explicit and multiple.

you because it is a fiction movie like "E.T. the Extra-Terrestrial" you watched before for the user $u2$.

In this paper, we propose a novel ICF framework *Relational Collaborative Filtering* (RCF), aiming to integrate multiple item relations for better recommendation. To retain the fine-grained semantics of a relation and facilitate the reasoning on user preference, we represent a relation as a concept with a two-level hierarchy:

(1) Relation *type*, which can be shared director and genre in the above movie example, or functionality and visually similar for E-commerce products. It describes how items are related with each other in an abstract way. The collaborative similarity is also a relation type from the macro view of user behaviors.

(2) Relation *value*, which gives details on the shared relation of two items. For example, the value of relation shared director for "E.T. the Extra-Terrestrial" and "Schindler's List" is Steven Spielberg, and the values for relation shared genre include fiction, action, romantic, etc. The relation values provide important clues for scrutinizing a user's preference, since a user could weigh different values of a relation type differently when making decisions.

Figure 1 gives an illustrative example on the item relations. Note that multiple relations may exist between two items; for example, badminton birdies balls and badminton rackets have two relations of complementary functionality and shared category. Moreover, a relation value may occur in multiple relations of different types; for example, a director can also be the leading actor of other movies, thus it is likely that two types of relations have the same value which refers to the same stuff. When designing a method to handle multiple item relations, these factors should be taken into account, making the problem more complicated than the standard ICF.

To integrate such relational data into ICF, we devise a two-level neural attention mechanism [3] to model the historically interacted items. Specifically, to predict a user's preference on a target item, the first-level attention examines the types of the relations that connect the interacted items with the target item, and discriminates which types affect more on the user. The second-level attention is operated on the interacted items under each relation type, so as to estimate the contribution of an interacted item in recommending the target item. The two-level attention outputs a weight for each interacted item, which is used to aggregate the embeddings of all

interacted items to obtain the user's representation. Furthermore, to enhance the item embeddings with the multi-relational data, we formulate another learning task that preserves the item relations with embedding operations. Finally, we jointly optimize the two tasks to make maximum usage of multiple relations between items.

To summarize, this work makes the key contributions as follows: rimental

2 METHODOLOGY

We first introduce the problem of using multiple item relations for CF, and then elaborate our proposed RCF method.

2.1 Problem Formulation

Given a user and his interaction history, conventional ICF methods aim at generating recommendations based on the collaborative similarity which encode the co-interact patterns of items. Its interaction graph can be shown as the left part of Figure 2, where the links between items are just the implicit collaborative similarity. However, there are multiple item relations in the real world which have meaningful semantics. In this work, we define the item relations as:

Definition 2.1. Given an item pair (i, j), the relations between them are defined as a set of $r = < t, v >$ where t denotes the relation type and v is the relation value.

The target of RCF is to generate recommendations based on both the user-item interaction history and item relational data. Generally speaking, the links between items in the interaction graph of RCF contain not only the implicit collaborative similarity, but also the explicit multiple item relations, which are represented by the heterogeneous edges in the right part of Figure 2. Our notations are summarized in Table 1.

In the remainder of this section, we first present the attention-based model to infer user-item preference. We then illustrate how to model the item relational data to introduce the relational structure between item embeddings. Based on that, we propose to integrate the two parts in an end-to-end fashion through a multi-task learning framework. Finally, we provide a discussion on the relationship between RCF and some other models.

Table 1: Notations

| Notation	Description
$\mathcal{U}, \mathcal{I}$	the set of users and items
$\mathcal{T}$	the set of relation types
$\mathcal{V}$	the set of relation values
$\mathcal{I}_u^+$	the item set which user u has interacted with
$\mathcal{I}_{u,i}^t$	the items in $\mathcal{I}_u^+$ that have the relation of type t with the target item i
$I_r(i,j)$	an indicator function where $I_r(i,j) = 1$ if relation r holds for item i and j, otherwise 0
$\mathbf{p}_u \in \mathbb{R}^d$	the ID embedding for user $u \in \mathcal{U}$, which represents the user's inherent interests
$\mathbf{q}_i \in \mathbb{R}^d$	the embedding for item $i \in \mathcal{I}$
$\mathbf{x}_t \in \mathbb{R}^d$	the embedding for relation type $t \in \mathcal{T}$
$\mathbf{z}_v \in \mathbb{R}^d$	the embedding for relation value $v \in \mathcal{V}$
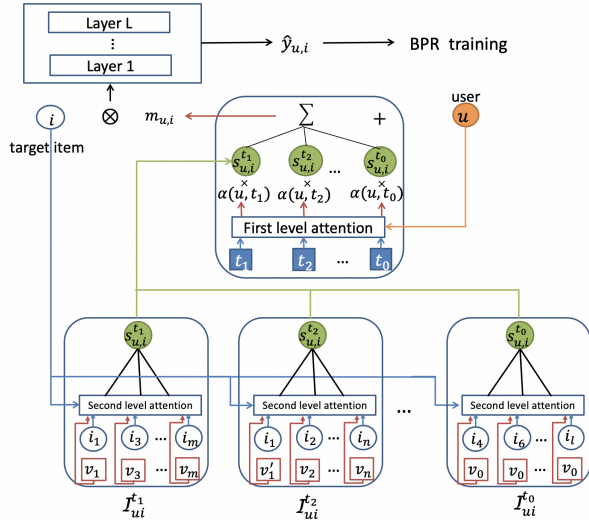
Figure 3: Illustration of the proposed recommendation model. The target-aware user embedding ($m_{u,i}$) is modeled with a two-level hierarchy attention mechanism. The input of the first level attention contains the user ID embedding and relation types. The second level attention is used to calculate the weights of specific historical items. There are three inputs during this state, including the target item, the historical item and the relation value. Note that one historical item (e.g., i_1) can occur in different $\mathcal{I}_{u,i}^t$ when there are multiple relations between it and the target item.

2.2 User-Item Preference Modeling

An intuitive motivation when modeling user preference is that users tend to pay different weights to relations of different types (e.g., some users may prefer movies which share same actors, some users may prefer movies fall into same genres). Given multiple item relations which consist of relation types and relation values, we propose to use a hierarchy attention mechanism to model the user preference. Figure 3 demonstrates the overall structure of our model.

Given the item relational data, we first divide the interacted items of user u (i.e., $\mathcal{I}_u^+$) into different sets (i.e., $\mathcal{I}_{u,i}^t$) according to the relation types between these items and the target item. Note that a single item may occur in different $\mathcal{I}_{u,i}^t$ when there are multiple relations between this item and i. Besides, there may be some items which have no explicit relation with the target item. To tackle with these items, we introduce a latent relation $r_0 = < t_0, v_0 >$ and put these items into $\mathcal{I}_{u,i}^{t_0}$, as shown in Figure 3. Here r_0 can be regarded as the collaborative similarity which just indicates the item co-interact patterns. Then the target-aware user embedding can be formulated as

$$\mathbf{m}_{u,i} = \mathbf{p}_u + \sum_{t \in \mathcal{T}} \alpha(u,t) \cdot \mathbf{s}_{u,i}^t , \qquad (1)$$

where $\alpha(u,t)$ is the first-level attention which aims to calculate the importance of different relation types for this user and $\mathbf{s}_{u,i}^t$ describes the user's profile based on the items in $\mathcal{I}_{u,i}^t$. More precisely, we define $\alpha(u,t)$ with the standard $softmax$ function:

$$\alpha(u,t) = \frac{exp(a(\mathbf{p}_u, \mathbf{x}_t))}{\sum_{t' \in \mathcal{T}} exp(a(\mathbf{p}_u, \mathbf{x}_{t'}))} , \qquad (2)$$

where $a(\mathbf{p}_u, \mathbf{x}_t)$ is the attention score between user u and relation type t. We define it with a feedforward neural network, as shown in Eq.(3)

$$a(\mathbf{p}_u, \mathbf{x}_t) = \mathbf{h}_1^T (ReLU(\mathbf{W}_1(\mathbf{p}_u \otimes \mathbf{x}_t) + \mathbf{b}_1)). \qquad (3)$$

$\mathbf{W}_1$ and $\mathbf{b}_1$ are corresponding weight matrix and bias vector that project the input into a hidden state, and $\mathbf{h}_1^T$ is the vector which projects the hidden state into the attention score. We term the size of hidden state as "attention factor", for which a larger value brings a stronger representation power for the attention network. $\otimes$ denotes the element-wise product.

The next step is to model $\mathbf{s}_{u,i}^t$. It's obvious that the relation value accounts for an important part during this process. For example, a user may pay attention to genres when watching a movie. However, among all the genres, he is most interested in fiction other than romantic. As a result, we should consider both the items and the corresponding relation values when modeling the second-level attentions. From that view, we define $\mathbf{s}_{u,i}^t$ as

$$\mathbf{s}_{u,i}^t = \sum_{j \in \mathcal{I}_{u,i}^t} \beta_t(i,j,v) \cdot \mathbf{q}_j , \qquad (4)$$

where $\beta_t(i,j,v)$ represents the specific weight of item j.

Similar to Eq.(2), a straight-forward solution to calculate $\beta_t(i,j,v)$ is to use the $softmax$ function. However we found that such a simple solution would lead to bad performance. Same observations can also be found in [16] under similar circumstances. The reason is that the number of items between different $\mathcal{I}_{u,i}^t$ vary greatly. For those items in large $\mathcal{I}_{u,i}^t$, the standard $softmax$ function will have very big denominator, causing the gradient vanishing problem of corresponding $\mathbf{q}_j$.

To tackle with this problem, we utilize a *smoothed softmax* function to replace the standard solution. As a result, the weight $\beta_t(i,j,v)$ is formulated as

$$\beta_t(i,j,v) = \frac{exp(b_t(\mathbf{q}_i, \mathbf{q}_j, \mathbf{z}_v))}{[\sum_{j' \in \mathcal{I}_{u,i}^t} exp(b_t(\mathbf{q}_i, \mathbf{q}_{j'}, \mathbf{z}_{v'})]^\rho} , \qquad (5)$$

where ρ is a smoothing factor between $(0,1]$ and is commonly set as 0.5 [16]. $b_t(\mathbf{q}_i, \mathbf{q}_j, \mathbf{z}_v)$ is the second-level attention score which is defined as

$$b_t(\mathbf{q}_i, \mathbf{q}_j, \mathbf{z}_v) = \mathbf{h}_{2,t}^T (ReLU(\mathbf{W}_{2,t} \begin{bmatrix} \mathbf{q}_i \\ \mathbf{q}_j \\ \mathbf{z}_v \end{bmatrix} + \mathbf{b}_{2,t})), \qquad (6)$$

where $[\cdot]$ denotes the vector concatenation. $\mathbf{W}_{2,t}$, $\mathbf{b}_{2,t}$ and $\mathbf{h}_{2,t}$ are corresponding attention parameters. Different from Eq.(3) which utilizes element-wise product to learn signals from inputs, here we concatenate the input embeddings and send it to a feedforward neural netwrok. The reason is that there are three inputs when modeling the second-level attentions. Utilizing element-wise product under such situation would have a high risk of suffering from vanishing or exploding gradients.

Now we have completed the modeling of the target-aware user embedding $\mathbf{m}_{u,i}$. Based on that, we utilize a multilayer perceptron (MLP) to calculate the final predicted score of user u on item i, which is shown as:[2]

$$\hat{y}_{ui} = MLP(\mathbf{m}_{u,i} \otimes \mathbf{q}_i), \qquad (7)$$

Given the final predicted score $\hat{y}_{ui}$, we want the positive items to have a higher rank than negative ones. We utilize the BPR pairwise learning framework [30] to define the objective function, which is shown as

$$L_{rec} = - \sum_{(u,i,k) \in \mathcal{D}_I} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uk}), \qquad (8)$$

where σ denotes the sigmoid function and $\mathcal{D}_I$ is the set of training triplets:

$$\mathcal{D}_I = \left\{ (u,i,k) | u \in \mathcal{U} \wedge i \in \mathcal{I}_u^+ \wedge k \in \mathcal{I} \backslash \mathcal{I}_u^+ \right\}. \qquad (9)$$

2.3 Item-Item Relational Data Modeling

The second task of RCF is to model the item relational data. Typically, the relational data is organized as knowledge graphs (KG). A knowledge graph is a directed heterogeneous graph in which nodes correspond to entities and edges correspond to relations. It can be represented by a set of triplets (e_1, r, e_2) where e_1 denotes the head entity, r is the relation and e_2 represents the tail entity. Knowledge graph embedding (KGE) is a popular approach to learn signals from relational data which aims at embedding a knowledge graph into a continuous vector space.

However, directly using techniques from KGE [4, 25, 46] to model the item relations of RCF is infeasible due to the following challenges in our specific domain:

(1) The item relation is defined with a two-level hierarchy: relation type and relation value. As shown in Figure 1, the relation between "E.T. the Extra-Terrestrial" and "The Avenger" is described as <shared genre,fiction>. To represent this relation properly, we must consider both the first-level (i.e., shared genre) for type constrains and the second-level (i.e., fiction) for model fidelity. As a result, we can not assign a single embedding for an item relation $r = <t, v>$, which is a common case in the field of KGE [4, 25, 46].

(2) Different from the conventional KG which is represented as a directed graph, the item relations are reversible (i.e., the relation r holds for both (e_1, e_2) and (e_2, e_1)), resulting in an undirected graph structure. Traditional KGE methods [4, 25] may encounter difficulties under such situations. For example, the most popular TransE [4] models the relation between two entities as a translation operation between their embeddings, that is, $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2$ when (e_1, r, e_2) holds, where $\mathbf{e}_1, \mathbf{r}, \mathbf{e}_2$ are corresponding embeddings for head entity, relation and tail entity. Based on that, TransE defines the scoring function for this triplet as $f(e_1, r, e_2) = \|\mathbf{e}_1 + \mathbf{r} - \mathbf{e}_2\|_2$ where $\| \cdot \|_2$ denotes the L_2 norm of a vector. However, because of the undirected structure, we will get both $\mathbf{e}_1 + \mathbf{r} \approx \mathbf{e}_2$ and $\mathbf{e}_2 + \mathbf{r} \approx \mathbf{e}_1$ on our item relational data. Optimizing objective functions based on such equations may lead to a trivial solution that $\mathbf{r} \approx \mathbf{0}$ and $\mathbf{e}_1 \approx \mathbf{e}_2$.

To tackle with the first challenge, we use the summation of the two-level hierarchy components as relation embeddings. More precisely, the representation of a specific relation $r = <t, v>$ is formulated as the following equation:

$$\mathbf{r} = \mathbf{x}_t + \mathbf{z}_v. \qquad (10)$$

By doing so, we can make sure that relations with the same type keep similar with each other in some degree. Meanwhile, the model fidelity is also guaranteed because of the value embedding. It also empowers the model with the ability to tackle the situation that same values occur in relations of different types.

To address the second challenge, we find that the source of the trivial solution is the minus operation in TransE, which only suits for directed structures. To model undirected graphs, we need the model which satisfies the commutative law (i.e., $f(e_1, r, e_2) = f(e_2, r, e_1)$). Another state-of-the-art methods of KGE is DistMult [46]. It defines the scoring function as $f(e_1, r, e_2) = \mathbf{e}_1^T \mathbf{M}_r \mathbf{e}_2$, where $\mathbf{M}_r$ is a matrix representation of r. It's obvious that DistMult is based on the multiply operation and satisfies the desired commutative property. Based on that, given a triplet (i, r, j) which means item i and j has relation r, we define the scoring function for this triplet as

$$f(i, r, j) = \mathbf{q}_i^T \cdot diag(\mathbf{r}) \cdot \mathbf{q}_j. \qquad (11)$$

Here $diag(\mathbf{r})$ denotes a diagonal matrix whose diagonal elements equal to $\mathbf{r}$ correspondingly.

Similar to the BPR loss used in the recommendation part, we want to maximize $f(i, r, j)$ for positive examples and minimize it for negative ones. Based on that, the objective function is defined by contrasting the scores of observed triplets (i, r, j) versus unobserved ones (i, r, j^-):

$$L_{rel} = - \sum_{(i,r,j,j^-) \in \mathcal{D}_R} \ln \sigma(f(i, r, j) - f(i, r, j^-)), \qquad (12)$$

where $\mathcal{D}_R$ is defined as

$$\mathcal{D}_R = \{(i, r, j, j^-) | i, j, j^- \in \mathcal{I} \wedge I_r(i,j) = 1 \wedge I_r(i, j^-) = 0 \wedge r \neq r_0 \}. \qquad (13)$$

The above objective function encourages the positive item j to be ranked higher than negative items j^- given the context of the head item i and relation r. Because r_0 is defined as a latent relation so we don't include it during this process.

2.4 Multi-Task Learning

To effectively learn parameters for recommendation, as well as preserve the relational structure between item embeddings, we integrate the recommendation part (i.e., L_{rec}) and the relation modeling part (i.e., L_{rel}) in an end-to-end fashion through a multi-task learning framework. The total objective function of RCF is defined as

$$\min_{\Theta} \quad L = L_{rec} + \gamma L_{rel} \;,$$
$$s.t. \quad \|\mathbf{p}_u\|_2 \leq 1, \|\mathbf{q}_i\|_2 \leq 1, \|\mathbf{x}_t\|_2 \leq 1, \|\mathbf{z}_v\|_2 \leq 1 \qquad (14)$$
$$\forall u \in \mathcal{U}, i \in \mathcal{I}, t \in \mathcal{T}, v \in \mathcal{V}$$

where Θ is the total parameter space, including all embeddings and variables of attention networks. It's obvious that both L_{rec} and L_{rel} can be decreased by simply scaling up the norm of corresponding embeddings. To avoid this problem during the training process, we explicitly constrain the embeddings to fall into a unit vector space. This constraint differs from traditional L_2 regularization which pushes parameters to the origin. It has been shown to be effective in both fields of KGE [4, 25] and recommendation [13, 22, 35]. The training procedure of RCF is illustrated in Algorithm 1.

Algorithm 1 Learning algorithm for RCF

Input: user-item interaction data $\mathcal{D}_I$, item relationl data $\mathcal{D}_R$, learning rate η, smoothing factor ρ, γ
Output: all parameters in the learning space Θ
1: Initialize all parameters in Θ
2: **repeat**
3:     Draw a mini-batch of (u, i, k) from $\mathcal{D}_I$
4:     Draw a mini-batch of (i, r, j, j^-) from $\mathcal{D}_R$
5:     Compute L_{rec} according to Eq.(1)-(9)
6:     Compute L_{rel} according to Eq.(10)-(13)
7:     $L \leftarrow L_{rec} + \gamma L_{rel}$
8:     **for** each parameter $\vartheta \in \Theta$ **do**
9:         Compute $\partial L/\partial \vartheta$ on the mini-batch by back-propagation
10:         Update $\vartheta \leftarrow \vartheta - \eta \cdot \partial L/\partial \vartheta$
11:     **end for**
12:     **for** $\theta \in \{\mathbf{p}_u, \mathbf{q}_i, \mathbf{x}_t, \mathbf{z}_v\}$ **do**
13:         $\theta \leftarrow \theta/\max(1, \|\theta\|_2)$
14:     **end for**
15: **until** converge
16: **return** all parameters in Θ

2.5 Discussion

Here we examine three types of related recommendation models and discuss the relationship between RCF and them.

2.5.1 Conventional collaborative filtering. RCF extends the item relations from the collaborative similarity to multiple and semantically meaningful relations. It can easily generalize the conventional CF methods. If we downgrade the MLP in Eq.(7) to inner product

and only consider one item relation (i.e., the collaborative similarity), we can get the following predicted score:

$$\hat{y}_{ui} = \underbrace{\mathbf{p}_u^T \mathbf{q}_i}_{\text{MF}} + \underbrace{\mathbf{q}_i^T \left(\sum_{j \in \mathcal{I}_u^+ \setminus \{i\}} \beta(i, j) \cdot \mathbf{q}_j \right)}_{\text{NAIS}}, \qquad (15)$$

which can be regarded as en ensemble of matrix factorization (MF) [23] and the item-based NAIS model [20]. In fact, compared with conventional CF methods, RCF captures item relations in an explicit and fine-grained level, and thus enjoys much more expressiveness to model user preference.

2.5.2 Knowledge graph enhanced recommendation. Recently, incorporating KG as an additional data source to enhance recommendation has become a hot research topic. These works can be categorized into embedding-based methods and path-based methods. Embedding-based methods [5, 18, 38, 47] utilize KG to guide the representation learning. However, the central part of ICF is the item similarity and none of these methods is designed to explicitly model it. On the contrary, RCF aims at directly modeling the item similarity from both the collaborative perspective and the multiple concrete relations. Path-based methods [2, 17, 34, 37, 40] first construct paths to connect users and items, then the recommendation is generated by reasoning over these paths. However, constructing paths between users and items isn't a scalable approach when the number of users and items are very large. Under such situation, sampling [2, 37] and pruning [34, 40] must be involved. However, RCF is free from this problem. Besides, the recommendation model of RCF is totally different from the path-based methods.

2.5.3 Relation-aware recommendation. MCF [28] proposed to utilize the "also-viewed" relation to enhance rating prediction. However, the "also-viewed" relation is just a special case of the item co-interact patterns and thus still belongs to the collaborative similarity. Another work which considers heterogeneous item relations is MoHR [22]. But it only suits for the sequential recommendation. The idea of MoHR is to predict both the next item and the next relation. The major drawback of MoHR is that it can only consider the relation between the last item of $\mathcal{I}_u^+$ and the target item. As a result, it fails to capture the long-term dependencies. On the contrary, RCF models the user preference based on all items in $\mathcal{I}_u^+$. The attention mechanism empowers RCF to be effective when capturing both long-term and short-term dependencies.

3 EXPERIMENTS

In this section, we conduct experiments on two real-world datasets to evaluate the proposed RCF model. We aim to answer the following research questions:

RQ1: Compared with state-of-the-art recommendation models, how does RCF perform?

RQ2: How do the multiple item relations affect the model performance?

RQ3: How does RCF help to comprehend the user behaviour? Can it generate more convincing recommendation?

In the following parts, we will first present the experimental settings and then answer the above research questions one by one.

Table 2: Dataset statistics.

	Dataset	MovieLens	KKBox
User-Item Interactions	#users	943	24,613
	#items	1,682	61,877
	#interactions	100,000	2,170,690
Item-Item Relations	#types	4	5
	#values	5,126	42,532
	#triplets	924,759	70,237,773

3.1 Experimental Settings

3.1.1 Datasets. We perform experiments with two publicly accessible datasets: MovieLens[3] and KKBox[4], corresponding to movie and music recommendation, respectively. Table 2 summarizes the statistics of the two datasets.

1. MovieLens. This is the stable benchmark published by GroupLens [12], which contains 943 users and 1,682 movies. We binarize the original user ratings to convert the dataset into implicit feedback. To introduce item relations, we combine it with the IMBD dataset[5]. The two datasets are linked by the titles and release dates of movies. The relation types of this data contains genres[6], directors, actors, and t_0, which is the relation type of the latent relation.

2. KKBox. This dataset is adopted from the WSDM Cup 2018 Challenge[7] and is provided by the music streaming service KKBox. Besides the user-item interaction data, this dataset also contains description of music, which can help us to introduce the item relations. We process this dataset by removing the songs that have missing description. The final version contains 24,613 users, 61,877 items and 2,170,690 interactions. The relation types of this dataset contain genre, artist, composer, lyricist, and t_0.

3.1.2 Evaluation protocols. To evaluate the performance of item recommendation, we adopt the leave-one-out evaluation, which has been widely used in literature [6, 16, 20]. More precisely, for each user in MovieLens, we leave his latest two interactions for validation and test and utilize the remaining data for training. For the KKBox dataset, because of the lack of timestamps, we randomly hold out two interactions for each user as the test example and the validation example and keep the remaining for training. Because the number of items is large in this dataset, it's too time consuming to rank all items for every user. To evaluate the results more efficiently, we randomly sample 999 items which have no interaction with the target user and rank the validation and test items with respect to these 999 items. This has been widely used in many other works [6, 16, 35, 40].

The recommendation quality is measured by three metrics: hit ratio (HR), mean reciprocal rank (MRR) and normalized discounted cumulative gain (NDCG). HR@k is a recall-based metric, measuring whether the test item is in the top-k positions of the recommendation list (1 for yes and 0 otherwise). MRR@k and NDCG@k are

---

[3] https://grouplens.org/datasets/movielens/
[4] https://www.kaggle.com/c/kkbox-music-recommendation-challenge/data
[5] https://www.imdb.com/interfaces/
[6] Here, genres means that two movies share at least one same genre, as shown in Figure 1. Same definition also suits for the following relation types.
[7] https://wsdm-cup-2018.kkbox.events/

weighted versions which assign higher scores to the top-ranked items in the recommendation list [19].

3.1.3 Compared methods. We compare the performance of the proposed RCF with the following baselines:

- MF [30]: This is the standard matrix factorization which models the user preference with inner product between user and item embeddings.
- FISM [20]: This is a state-of-the-art ICF model which characterizes the user with the mean aggregation of the embeddings of his interacted items.
- NAIS [16]: This method enhances FISM through a neural attention network. It replaces the mean aggregation of FISM with an attention-based summation.
- FM [29]: Factorization machine is a feature-based baseline which models the user preference with feature interactions. Here we treat the auxiliary information of both datasets as additional input features.
- NFM [14]: Neural factorization machine improves FM by utilizing a MLP to model the high-order feature interactions.
- CKE [47]: This is an embedding-based KG-enhanced recommendation method, which integrates the item embeddings from MF and TransR [25].
- MoHR [22]: This method is a state-of-the-art relation-aware CF method. We only report its results on the MovieLens dataset because it's designed for sequential recommendation and the KKBox dataset contains no timestamp information.

3.1.4 Parameter settings. To fairly compare the performance of models, we train all of them by optimizing the BPR loss (i.e.,Eq(8)) with mini-batch Ada-grad [9]. The learning rate is set as 0.05 and the batch size is set as 512. The embedding size is set as 64 for all models. For all the baselines, the L_2 regularization coefficients are tuned between $[1e^{-5}, 1e^{-4}, 1e^{-3}, 0]$. For FISM, NAIS and RCF, the smoothing factor ρ is set as 0.5. We pre-train NAIS with 100 iterations of FISM. For the attention-based RCF and NAIS, the attention factor is set as 32. Regarding NFM, we use FM embeddings with 100 iterations as pre-training vectors. The number of MLP layers is set as 1 with 64 neurons, which is the recommended setting of their original paper [14]. The dropout ratio is tuned between $[0, 0.1, \cdots, 0.9]$. For the MLP of RCF, we adopt the same settings with NFM to guarantee a fair comparison. For MoHR, we set the multi-task learning weights as 1 and 0.1 according to their original paper [22]. For RCF, we find that it achieves satisfactory performance when $\gamma = 0.01$. We report the results under this setting if there is no special mention.

3.2 Model Comparison (RQ1)

Table 3 demonstrates the comparison between all methods when generating top-k recommendation. It's obvious that the proposed RCF achieves the best performance among all methods on both datasets regarding to all different top-k values.

Compared with the conventional item-based FISM and NAIS which only consider the collaborative similarity, our RCF is based on the multiple and concrete item relations. We argue that this is the major source of the the improvement. From this perspective,

Table 3: Top-k recommendation performance comparison of different models ($k = 5, 10, 20$). The last column RI denotes the relative improvement on average of RCF over the baseline. $*$ denotes the significance p-value < 0.05 compared with the best baseline on the corresponding metric (indicated by boldface).

Models	MovieLens									RI
	HR@5	MRR@5	NDCG@5	HR@10	MRR@10	NDCG@10	HR@20	MRR@20	NDCG@20	
MF	0.0774	0.0356	0.0458	0.1273	0.0430	0.0642	0.2110	0.0482	0.0833	+25.2%
FISM	0.0795	0.0404	0.0500	0.1325	0.0474	0.0671	0.2099	0.0526	0.0865	+20.3%
NAIS	0.0827	0.0405	0.0508	0.1367	0.0477	0.0683	0.2142	0.0528	0.0876	+17.9%
FM	0.0827	0.0421	0.0521	0.1410	**0.0496**	0.0707	0.1994	0.0535	0.0852	+18.6%
NFM	**0.0880**	0.0427	**0.0529**	**0.1495**	0.0495	0.0725	0.2153	0.0540	**0.0889**	+13.4%
CKE	0.0827	0.0414	0.0515	0.1404	0.0476	0.0688	0.2089	0.0528	0.0884	+15.2%
MoHR	0.0832	**0.0490**	0.0499	0.1463	0.0485	**0.0733**	0.2249	0.0554	0.0882	+11.2%
RCF	**0.1039***	**0.0517***	**0.0646***	**0.1591***	**0.0598***	**0.0821***	**0.2354***	**0.0642***	**0.1015***	

Models	KKBox									RI
	HR@5	MRR@5	NDCG@5	HR@10	MRR@10	NDCG@10	HR@20	MRR@20	NDCG@20	
MF	0.5575	0.3916	0.4329	0.6691	0.4065	0.4690	0.7686	0.4135	0.4942	+29.1%
FISM	0.5676	0.4084	0.4356	0.6866	0.4103	0.4844	0.7654	0.4258	0.5244	+26.2%
NAIS	0.5862	0.4156	0.4409	0.6932	0.4153	0.4966	0.7810	0.4333	0.5315	+24.0%
FM	0.5793	0.4064	0.4495	0.6949	0.4219	0.4869	**0.7941**	0.4288	0.5121	+24.4%
NFM	**0.5973**	0.4183	**0.4630**	**0.7178**	**0.4432**	**0.5088**	0.7768	**0.4476**	0.5244	+19.9%
CKE	0.5883	**0.4191**	0.4613	0.6930	0.4332	0.4952	0.7865	0.4397	**0.5389**	+21.3%
RCF	**0.7158***	**0.5612***	**0.5999***	**0.7940***	**0.5718***	**0.6253***	**0.8563***	**0.5762***	**0.6412***	

the results demonstrate the importance of multiple item relations when modeling the user preference.

Compared with the feature-based FM and NFM, RCF still achieves significant improvement. The reason is that although FM and NFM also incorporate the auxiliary information, they fail to explicitly model the item relations based on that data. Besides, we can also see that NFM achieves better overall performance than FM because it introduces a MLP to learn high-order interaction signals. However, RCF achieves higher performance under the same MLP settings, which confirms the effectiveness of modeling item relations.

Compared with CKE, we can see that although CKE utilizes KG to guide the learning of item embeddings, it fails to directly model user preference based on multiple item relations, resulting in lower performance than RCF. Besides, we can see that although MoHR is also relation-aware, RCF still achieves better results than it. The reason is that MoHR only considers the relation between the last historical item and the target item, and thus fails to capture the long-term dependencies among the user interaction history.

3.3 Studies of Item Relations (RQ2)

3.3.1 Effect of the hierarchy attention. RCF utilizes a hierarchy attention mechanism to model user preference. In this part, we conduct experiments to demonstrate the effect of the two-level attentions. Table 4 shows the results of top-10 recommendation when replacing the corresponding attention with average summation. It's obvious that both the first-level and the second-level attentions are necessary to capture user preference, especially the second-level attention, which aims at calculating a specific weight for every historical item and thus largely improves the model expressiveness.

Table 4: Performance when replacing the attention with average summation. Avg-1 denotes the first-level attention (i.e.,$a(u, t)$) is replaced. Avg-2 means the second-level attention (i.e., $\beta_t(i, j, v)$) is replaced. Avg-both denotes replacing both attentions. Dec is the average decrease of performance. $*$ denotes the statistical significance for $p < 0.05$.

Models	MovieLens			Dec
	HR@10	MRR@10	NDCG@10	
Avg-1	0.1478	0.0556	0.0746	-7.6%
Avg-2	0.1346	0.0501	0.0694	-15.6%
Avg-both	0.1294	0.0495	0.0684	-17.8%
RCF	**0.1591***	**0.0598***	**0.0821***	

Models	KKBox			Dec
	HR@10	MRR@10	NDCG@10	
Avg-1	0.7657	0.5484	0.5773	-5.0%
Avg-2	0.6983	0.4331	0.5249	-16.8%
Avg-both	0.6792	0.4103	0.4946	-20.4%
RCF	**0.7940***	**0.5718***	**0.6253***	

3.3.2 Ablation studies on relation modeling. The proposed RCF defines the item relations with relation types and relation values. To demonstrate the effectiveness of these two components, we modify the proposed RCF by masking the corresponding parts. Table 5 shows the detail of the masked models. Table 6 reports the performance when masking different relation components. We can draw the following conclusions from this table.

Table 5: Modification of RCF. Single denotes only considering one relation (i.e., collaborative similarity). RCF-type only considers relation types for the attention. RCF-value only considers relation values. g denotes the attention function.

	Modification
Single	Eq.(1)$\Rightarrow \mathbf{m}_{u,i} = \mathbf{p}_u + \sum_{j \in \mathcal{I}_u^+} g(i,j) \cdot \mathbf{q}_j$
RCF-type	Eq.(4)$\Rightarrow \mathbf{s}_{u,i}^t = \sum_{j \in \mathcal{I}_{u,i}^t} g(i,j) \cdot \mathbf{q}_j$
RCF-value	Eq.(1)$\Rightarrow \mathbf{m}_{u,i} = \mathbf{p}_u + \sum_{j \in \mathcal{I}_u^+} g(i,j,v) \cdot \mathbf{q}_j$

Table 6: Performance of different relation ablations when generating top-10 recommendation. Dec is the average decrease of performance. ∗ denotes the statistical significance for $p < 0.05$.

Ablations	MovieLens			
	HR@10	MRR@10	NDCG@10	Dec
Single	0.1399	0.0481	0.0691	-14.6%
RCF-type	0.1484	0.0587	0.0804	-4.5%
RCF-value	0.1548	0.0558	0.0801	-3.4%
RCF	**0.1591***	**0.0598***	**0.0821***	

Ablations	KKBox			
	HR@10	MRR@10	NDCG@10	Dec
Single	0.6923	0.4666	0.5207	-15.7%
RCF-type	0.7523	0.5431	0.5723	-6.2%
RCF-value	0.7708	0.5579	0.5867	-3.8%
RCF	**0.7940***	**0.5718***	**0.6253***	

(1) RCF-type achieves better performance than the single model, demonstrating the importance of relation types. Generally speaking, the type component describes item relations in an abstract level. It helps to model the users' preference on a class of items which share particular similarity in some macro perspectives.

(2) The performance of RCF-value is also better than the single model. This finding verifies the effectiveness of relation values, which describe the relation between two specific items in a much fine-grained level. The relation value increases the model fidelity and expressiveness largely through capturing the user preference from micro perspective.

(3) RCF achieves the best performance. It demonstrates that both relation types and relation values are necessary to model the user preference. Moreover, it also confirms the effectiveness of the proposed two-level attention mechanism to tackle with the hierarchical item relations.

3.3.3 Effect of multi-task learning. RCF utilizes the item relational data in two ways: constructing the target-aware user embeddings and introducing the relational structure between item embeddings through the multi-task learning framework. In this part, we conduct experiments to show the effect of the later.
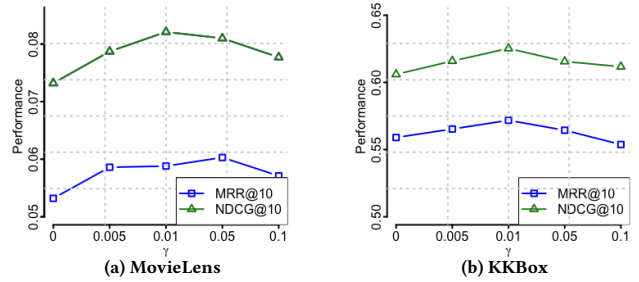

Figure 4: Effect of γ on the two datasets

Figure 4 reports the results of MRR@10 and NDCG@10 when changing the multi-task learning weight γ[8]. It's obvious the performance of RCF boosts when γ increases from 0 to positive values on both two datasets. Because $\gamma = 0$ means only the recommendation task (i.e., L_{rec}) is considered, we can draw a conclusion that jointly training L_{rec} and L_{rel} can definitely improve the model performance. In fact, the function of L_{rel} is to introduce a constraint that if there is relation between two items, there must be an inherent structure among their embeddings. This constraint explicitly guides the learning process of both item and relation embeddings and thus helps to improve the model performance. We can also see that with the increase of γ, the performance improves first and then starts to decrease. Because the primary target of RCF is recommendation other than predicting item relations, we must make sure that L_{rec} accounts the crucial part in the total loss. Actually, we can see from Table 2 that the number of item-item relational triplets is much larger than the number of user-item interactions, leading to situation that γ is commonly set as a small value.

3.4 Qualitative Analyses (RQ3)

In this part, we conduct qualitative analyses to show how RCF helps us to comprehend user behaviors and generate more convincing recommendation.

3.4.1 Users as a whole. Figure 5 illustrates the average $a(u,t)$ for all $u \in \mathcal{U}$ on the two datasets. We can see that on the MovieLens dataset, the largest $a(u,t)$ falls into genre, which means that users tend to watch movies that share same genres. The second position is actor. This finding is in concert with the common sense that genres and actors are the most two important elements that affect the users' choices on movies. Director is in the third position. Moreover, we can see that all these three relation types are more important than t_0, which denotes the collaborative similarity. It further confirms that only considering collaborative similarity is not enough to model user preference. Multiple and fine-grained item relations should be involved to generate better recommendation.

For the music domain, we can see that the most important relation type falls into artist. Following that are comp. (short for composer) and lyri. (short for lyricist). They are the most three important factors that affect users when listening to music. Besides, compared with the movie domain, the attention $a(u,t_0)$ in the music domain is much smaller. It indicates that user behaviour patterns when listening to music are more explicit than the ones

---

[8]Results of HR@10 show similar trends and are omitted due to the reason of space.

when watching movies. As a result, our proposed RCF achieves bigger improvement on the KKBox dataset, as shown in Table 3.

3.4.2 Invididual case studies. We randomly select a user $u54$ in the MovieLens dataset to see how RCF helps us to comprehend the individual user behavior. Figure 6 shows the attention visualization of this user. We can see that this user pays the most attention (0.4003) on the relation type "shared genres" when watching movies. Among the second-level relation values, he is most interested in "crime" (0.4477) and "sci-fic" (0.3928). Based on his historical interacted movies "Breakdown" and "The Fifth Element", the recommended movie is "Face/Off". From this perspective, we can also generate the explanation as *"Face/Off" is recommended to you because it is a crime movie like "Breakdown" you have watched before.* It's obvious that a side benefit of RCF is that it can generate reasonable explanations for recommendation results.

4 RELATED WORK

4.1 Item-based Collaborative Filtering

The idea of ICF is that the user preference on a target item i can be inferred from the similarity of i to all items the user has interacted in the past [16, 20, 26, 31]. Under this case, the relation between items is referred as the collaborative similarity, which measures the co-occurrence in the user interaction history. A popular approach of ICF is FISM [20], which characterizes the user representation as the mean aggregation of item embeddings which occur in his interaction history. Plenty of work has been done following this research line, such as incorporating user information [11, 43], neural network-enhanced approaches [15, 16, 41] and involving local latent space [7, 24].

Although these methods has improved the performance of ICF, all of them are based solely on the collaborative similarity between items. This item relation is coarse-grained and lacks of semantic meaning, introducing the bottleneck of the model and the difficulty of generating convincing results.

4.2 Attention Mechanism

The attention mechanism has become very popular in fields of computer vision [27, 44] and natural language processing [3, 36] because of its good performance and interpretability for deep learning models. The key insight of attention is that human tends to pay different weights to different parts of the whole perception space. Based on this motivation, [16] improved FISM by replacing the mean aggregation with attention-based summation. [6] proposed to utilize the attention mechanism to generate multimedia recommendation. [21] exploited self-attention for sequential recommendation. There are many other works focusing on involving attention mechanism for better recommendation [35, 42]. However, all of them fail to model the multiple item relations. In fact, users tend to pay different weights on different item relations and it's a promising direction to utilize attention mechanism under such circumstance.

5 CONCLUSION

In this work, we proposed a novel ICF framework namely RCF to model the multiple item relations for better recommendation.
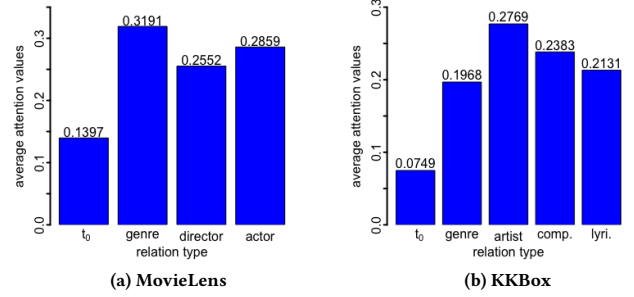

(a) MovieLens   (b) KKBox

Figure 5: Average $a(u, t)$ on two datasets. $a(u, t)$ denotes the user u's attention on the relation type t.
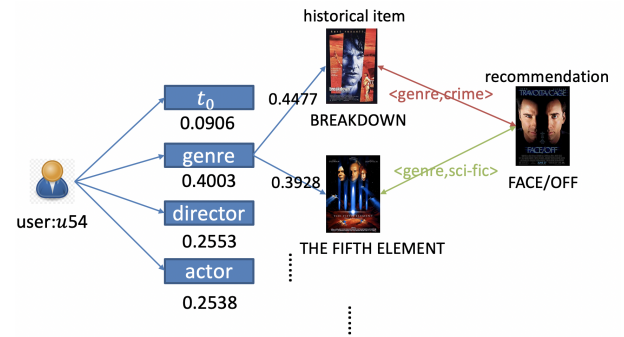

Figure 6: Attention visualization of user $u54$ in MovieLens.

RCF extends the item relations of ICF from collaborative similarity to fine-grained and concrete relations. We found that both the relation type and the relation value are crucial for capturing user preference. Based on that, we proposed to utilize a hierarchy attention mechanism to construct user representations. Besides, to maximize the usage of relational data, we further defined another task which aims to preserve the relational structure between item embeddings. We jointly optimize it with the recommendation task in an end-to-end fashion through a multi-task learning framework. Extensive experiments on two real-world datasets show that RCF achieves significantly improvement over state-of-the-art baselines. Moreover, RCF also provides us an approach to better comprehend user behaviors and generate more convincing recommendation.

Future work includes deploying RCF on datsts with more complex item relations. Besides, we also want to extend RCF to empower it with the ability to tackle with not only the static item relations but also the dynamic user relations. Another promising direction is how to utilize item relations to develop adaptive samplers for the pair-wise ranking framework.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible

extensions. *IEEE Transactions on Knowledge & Data Engineering* 6 (2005), 734–749.

[2] Qingyao Ai, Vahid Azizi, Xu Chen, and Yongfeng Zhang. 2018. Learning Heterogeneous Knowledge Base Embeddings for Explainable Recommendation. *Algorithms* 11, 137 (2018).

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*. 2787–2795.

[5] Yixin Cao, Xiang Wang, Xiangnan He, Tat-Seng Chua, et al. 2019. Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences. In *Proceedings of the 30th international conference on World Wide Web*. ACM.

[6] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 335–344.

[7] Evangelia Christakopoulou and George Karypis. 2018. Local latent space models for top-n recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1235–1243.

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[9] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[10] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A System for Recommending 3+ Billion Items to 200+ Million Users in Real-Time. In *Proceedings of the 27th International Conference on World Wide Web*. ACM, 1775–1784.

[11] Asmaa Elbadrawy and George Karypis. 2015. User-specific feature-based similarity models for top-n recommendation of new items. *ACM Transactions on Intelligent Systems and Technology (TIST)* 6, 3 (2015), 33.

[12] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (TIIS)* 5, 4 (2016), 19.

[13] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 161–169.

[14] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 355–364.

[15] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial personalized ranking for recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 355–364.

[16] Xiangnan He, Zhenkui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2018), 2354–2366.

[17] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1531–1540.

[18] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving sequential recommendation with knowledge-enhanced memory networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 505–514.

[19] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[20] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 659–667.

[21] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[22] Wang-Cheng Kang, Mengting Wan, and Julian McAuley. 2018. Recommendation Through Mixtures of Heterogeneous Item Relationships. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1143–1152.

[23] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[24] Joonseok Lee, Seungyeon Kim, Guy Lebanon, and Yoram Singer. 2013. Local low-rank matrix approximation. In *International Conference on Machine Learning*. 82–90.

[25] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, Vol. 15. AAAI Press, 2181–2187.

[26] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 1 (2003), 76–80.

[27] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. 2014. Recurrent models of visual attention. In *Advances in neural information processing systems*. 2204–2212.

[28] Chanyoung Park, Donghyun Kim, Jinoh Oh, and Hwanjo Yu. 2017. Do Also-Viewed Products Help User Rating Prediction?. In *Proceedings of the 26th International Conference on World Wide Web*. ACM, 1113–1122.

[29] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 995–1000.

[30] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[31] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[32] Brent Smith and Greg Linden. 2017. Two Decades of Recommender Systems at Amazon.Com. *IEEE Internet Computing* 21, 3 (May 2017), 12–18.

[33] 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.

[34] Zhu Sun, Jie Yang, Jie Zhang, Alessandro Bozzon, Long-Kai Huang, and Chi Xu. 2018. Recurrent Knowledge Graph Embedding for Effective Recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*. ACM, 297–305.

[35] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent Relational Metric Learning via Memory-based A ention for Collaborative Ranking. In *Proceedings of the 27th International Conference on World Wide Web*. ACM.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[37] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 417–426.

[38] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. *Proceedings of the 27th International Conference on World Wide Web* (2018), 1835–1844.

[39] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*.

[40] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable Reasoning over Knowledge Graphs for Recommendation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*. AAAI Press.

[41] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 153–162.

[42] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 3119–3125.

[43] Xin Xin, Dong Wang, Yue Ding, and Chen Lini. 2016. FHSM: factored hybrid similarity methods for top-n recommender systems. In *Asia-Pacific Web Conference*. Springer, 98–110.

[44] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.

[45] Feng Xue, Xiangnan He, Xiang Wang, Jiandong Xu, Kai Liu, and Richang Hong. 2018. Deep Item-based Collaborative Filtering for Top-N Recommendation. *ACM Transactions on Information Systems (TOIS)*.

[46] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).

[47] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 353–362.