Collaborative Prediction for Multi-entity Interaction With Hierarchical Representation

Qiang Liu, Shu Wu, Liang Wang
Center for Research on Intelligent Perception and Computing
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
{qiang.liu, shu.wu, wangliang}@nlpr.ia.ac.cn

ABSTRACT

With the rapid growth of Internet applications, there are more and more entities in interaction scenarios, and thus collaborative prediction for multi-entity interaction is becoming a significant problem. The state-of-the-art methods, e.g., tensor factorization and factorization machine, predict multi-entity interaction based on calculating the similarity among all entities. However, these methods are usually not able to reveal the joint characteristics of entities in the interaction. Besides, some methods may succeed in one specific application, but they can not be extended effectively for other applications or interaction scenarios with more entities. In this work, we propose a Hierarchical Interaction Representation (HIR) model, which models the mutual action among different entities as a joint representation. We generate the interaction representation of two entities via tensor multiplication, which is preformed iteratively to construct a hierarchical structure among all entities. Moreover, we employ several hidden layers to reveal the underlying properties of this interaction and enhance the model performance. After generating final representation, the prediction can be calculated using a variety of machine learning methods according to different tasks (i.e., linear regression for regression tasks, pair-wise ranking for ranking tasks and logistic regression for classification tasks). Experimental results show that our proposed HIR model yields significant improvements over the competitive compared methods in four different application scenarios (i.e., general recommendation, context-aware recommendation, latent collaborative retrieval and click-through rate prediction).

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Filtering

CIKM'15, October 19–23, 2015, Melbourne, Australia.

DOI: http://dx.doi.org/10.1145/2806416.2806530.

Keywords

Collaborative Prediction, Multi-Entity, Factorization Model, Hierarchical Representation

1. INTRODUCTION

Nowadays, collaborative prediction plays an important role in many real-world applications, e.g., recommender systems, information retrieval and social network analysis. With the rapid growth of web applications, there are more and more entities in one interaction scenario. For example, there are three entities in tag recommendation (i.e., user, item and tag) [25, 17, 20] and latent collaborative retrieval (i.e., user, query and document) [27, 2], and three or more entities in context-aware recommendation (i.e., user, query and several contexts) [4, 19, 21, 10]. Moreover, in some probability prediction applications, e.g., click-through rate prediction [12, 29, 14], there are even more entities in an interaction situation (e.g., device, website, application, advertisement and position). Accordingly, it is necessary to model joint characteristics of entities and collaboratively predict interaction among multiple entities.

Matrix factorization [13, 8] is widely used for interaction prediction between two entities, e.g., users and items in general recommendation. And matrix factorization based methods have been extensively studied, some of which, e.g., tensor factorization [28] and factorization machine [16], are implemented for predicting multi-entity interaction, e.g., user, query and document in latent collaborative retrieval. Both tensor factorization and factorization machine predict the interaction based on the similarity among all entities, which seems intuitive and reasonable. However, this does not conform to situations of various applications. Using latent collaborative retrieval as an example, both tensor factorization and factorization machine compute the similarity among user, query and document to predict their interaction. However, the similarity between the user and the query does not contribute to document selection of this user, and final selection is not merely based on the similarity between the user and the document or the similarity between the query and the document. We should measure the correlation between the document and the joint representation of user and query. In addition, there is another disadvantage of conventional methods. They only compute the similarity based on corresponding dimensional values in the latent vectors, which does not allow rich high-order calculation among values of different dimensions. Therefore, to be general, we provide a latent representation for a multi-entity interaction, which describes how these entities would act being together with

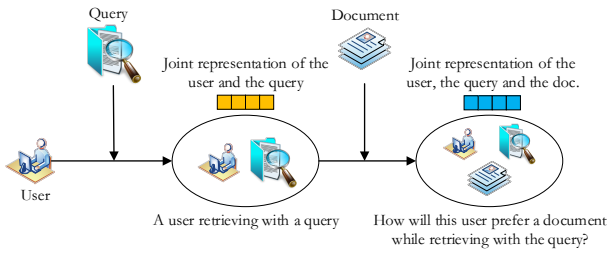


Figure 1: Illustration of the hierarchical structure of multi-entity interaction, using latent collaborative retrieval as an example.

each other and can reveal their high-order relation. As illustrated in Figure 1, when a user retrieves with a query, capturing the mutual action of the user and the query, a joint representation is generated. Similarly, when the user views a document, there will be a joint representation of user, query and document, which indicates how this user will prefer a document while retrieving with the query. Based on this joint representation, we could make prediction for the multi-entity interaction.

Recent studies on representation learning [1] in different areas give us great inspiration. Recently, joint representation of entities is employed for linked data [9] and structured data [24]. In recent works of natural language processing, using tensor multiplication, neural tensor networks are successfully used in learning the representation of two entities for knowledge base completion [22] and the representation of a sentence for semantic compositionality [23]. Tensor multiplication is also used in modeling contextual representation for context-aware recommendation [21, 10].

In this paper, we present a novel method to learn a Hierarchical Interaction Representation (HIR) model for predicting interaction among multiple entities. In our method, each entity is represented as a latent vector. And we use three-dimensional tensor multiplication to capture the joint characteristics of two entities. Tensor multiplication allows high-order calculation between these two entities. Moreover, rather than achieving a score of the interaction in conventional methods, HIR generates an interaction representation, which allows interacting with the next entity and can be widely implemented for various tasks. In a multi-entity interaction scenario with n entities, this tensor multiplying procedure can be performed iteratively, which forms a hierarchical architecture with n layers. The ith layer presents the interaction representation of the first i entities. The nth layer is the final representation of all the entities, which describes their joint characteristics. With such a hierarchical structure, we can also add new entities directly without re-training the former representations. Based on the final representation, we could select learning methods according to different application tasks for collaborative prediction (i.e., linear regression for regression tasks, pair-wise ranking method for ranking tasks and logistic regression for classification tasks).

The main contributions of this work are listed as follows:

- We model the joint representation of multiple types of entities in an interaction scenario, which presents a novel perspective on collaboratively modeling and predicting the entity interaction.
- Using the tensor multiplication, HIR allows high-order calculation among different types of entities, which reveals the underlying properties and relations among entities.
- Experiments conducted on four different tasks show that HIR is effective and clearly outperforms both general methods and state-of-the-art task-specific methods.

The rest of the paper is organized as follows. In section 2, we review some related work on both general methods and task-specific methods. Section 3 details our HIR model. In section 4, we introduce the learning methods of the HIR model. In section 5, we report experimental results in terms of four tasks and compare with several state-of-the-art methods. Section 6 concludes our work and discusses future research.

2. RELATED WORK

In this section, we briefly review related work on both general and task-specific methods, i.e., context-aware recommendation, latent collaborative retrieval, tag recommendation and click-through rate prediction.

2.1 General Methods

Matrix Factorization (MF) [13, 8] has become one of the state-of-the-art approaches to collaborative prediction. The basic objective of MF is to factorize a matrix into two low rank matrices, each of which represents the latent factors of entities, e.g., users and items in general recommendation. The original matrix can be approximated via the multiplying calculation. However, MF is not able to be implemented directly for collaborative prediction problem of multiple entities.

MF has been extended nowadays. Tensor Factorization (TF) [28] extends MF from two dimensions to three or even higher dimensions, which can be utilized for predicting multi-entity interaction. And Factorization Machine (FM) [16] gives a further extension of TF by modeling all interactions between pairs of entities. FM is a flexible model, and other state-of-the-art factorization models including TF, SVD++ [5] and timeSVD++ [6] can be implemented using FM by defining the input data or features [16]. Both TF and FM are successfully used in tag recommendation [25, 17, 20] and context-aware recommendation [4, 19]. FM also leads great improvement in click-through rate prediction [14]. However, predicting multi-entity interaction based on simply calculating the similarity among all entities, TF and FM are not able to better reveal the joint characteristics of interacting entities. Moreover, because FM uses the pair-wise interaction between entities to model the prediction of multi-entity interaction, this brings FM another disadvantage that the number of pairs grows exponentially with the number of entities.

2.2 Task-Specific Methods

In this subsection, we introduce some research works in different tasks that are related to the collaborative prediction for multi-entity interaction.

The context modeling approaches have made significant improvement for context-aware recommendation, which contains three or more entities (i.e., user, query and several contexts). Recent works on context modeling have focused on integrating contextual information with user-item rating matrix and building factorization models. Incorporating tensor factorization, multiverse recommendation [4] represents the user-item rating matrix with contextual information as a user-item-context rating tensor, which is factorized with Tucker decomposition [26]. And FM is applicable to a variety of contexts by specifying only the input data [19]. Random decision trees are also been applied here, in which contexts are split and general matrix factorization is performed on each leaf [11]. Furthermore, the work of [30] considers user attributes as priors for user latent vectors, and a transfer matrix is used to generate latent vectors from original ones. Similarly, Heterogeneous Matrix Factorization (HeteroMF) [3] generates context-specific latent vectors of entities using a context-dependent transfer matrix and the original latent vectors of entities. CARS2 [21] and Contextual Operating Tensor (COT) [10] model represents entities in a specific context using contextual representation and outperform previous methods in context-aware recommendation. Both of them model contexts as latent vectors, and generate contextual representations of users and items using tensor multiplication.

Recently, modeling the interaction among user, query and document, Latent Collaborative Retrieval (LCR) [27] has been proposed and extended to social network [2]. LCR represents a user with a matrix and generates the joint representation of user and query via multiplication. Then the prediction is given based on the inner product of latent representation of query and joint representation of user and query. Modeling the interaction among user, item and tag, tag recommendation can also be viewed as a LCR problem, in which a user retrieves tags with an item. Treating user, item and tag as three dimensions of a tensor, TF has been successfully used in this problem [25, 17]. As an extended version of TF, FM is also applied in tag recommendation and has become one of the state-of-the-art methods [20].

In the complex click-through rate (CTR) prediction problem, there may exist several entities such as user, device, website, application, advertisement, position and so on. Due to its ease of implementation and promising performance, Logistic regression (LR) has been widely used for CTR prediction, especially in industrial systems [12, 29]. LR can be used for the multi-entity interaction prediction with one-hot representation of each entity. However, LR or other classifiers have difficulties in discovering latent relation among entities, and the final prediction of LR could be viewed as the sum of biases of all the entities. Recently, as a representative method of factorization models, FM is applied for CTR prediction and brings a great improvement [14].

The methods mentioned above achieve delightful results in respective applications, which also greatly motivates our work. But they are unable to be extended effectively for other applications or interaction with more entities.

3. REPRESENTATION OF MULTI-ENTITY INTERACTION

In this section, we introduce our proposed hierarchical interaction representation. We introduce the definition and notations of our problem at first. Then we present the interaction representation of two entities, followed by the hierarchical representation of more entities. Finally, we show how to enhance the model via employing hidden layers.

3.1 Problem Definition

The problem we study in this paper can be described as follows. In an interaction scenario, suppose that we have an interaction prediction task with n types of entities denoted by $\{E^{(1)}, ..., E^{(n)}\}$. For each type of entity, we have $E^{(m)} = \{e_1^{(m)}, e_2^{(m)}, ...\}$, where $e_i^{(m)} \in \mathbb{R}^d$ and $m \in \{1, ..., n\}$. In the multi-entity interaction, the relation among $e_{k_1}^{(1)}, ..., e_{k_n}^{(n)}$ is denoted by $y_{k_1,...,k_n}$. The task of collaborative prediction for multi-entity interaction is to give a prediction $\hat{y}_{k_1,...,k_n}$ based on all the entities $e_{k_1}^{(1)}, ..., e_{k_n}^{(n)}$.

Here, using latent collaborative retrieval as an example, there will be three types of entities (i.e., user, query, document) denoted by $\{E^{(1)}, E^{(2)}, E^{(3)}\}$. Specifically, users, queries and documents are denoted by $E^{(1)} = \{e_1^{(1)}, e_2^{(1)}, ...\}$, $E^{(2)} = \{e_1^{(2)}, e_2^{(2)}, ...\}$ and $E^{(3)} = \{e_1^{(3)}, e_2^{(3)}, ...\}$ respectively. Each user, query and document are denoted as $e_i^{(1)} \in \mathbb{R}^d$, $e_i^{(2)} \in \mathbb{R}^d$ and $e_i^{(3)} \in \mathbb{R}^d$. Then the relation among user, query and document is denoted as y_{k_1,k_2,k_3}.

In this work, we would jointly model all the entities with an interaction representation $r_{k_1,...,k_n}^{(n)}$, and m-subset of the entities with an interaction representation $r_{k_1,...,k_m}^{(m)}$. The interaction representation describes the joint characteristics of entities. We could make prediction of their interaction based on this representation using various learning methods (i.e., linear regression, logistic regression and pair-wise ranking).

3.2 Interaction Representation

We first introduce the case with two entities. To get the joint representation of two entities, we need to employ an interaction function that satisfies:

$$r_{k_1,k_2}^{(2)} = f^{(2)}\left(e_{k_1}^{(1)}, e_{k_2}^{(2)}\right) , \tag{1}$$

where $f^{(2)}(\cdot)$ denotes an interaction function of two entities. Latent vector $r_{k_1,k_2}^{(2)}$ describes how the two entities would behave being together in an interaction scenario. For instance, if there are user and movie as two entities, their joint representation captures the characteristics of a user watching a movie, and models how much he or she likes the movie.

To capture the high-order relation between two entities, we use tensor multiplication, as shown in the bottom square of Figure 2. Here, we compute the interaction representation as:

$$r_{k_1,k_2}^{(2)} = \left[\left(e_{k_2}^{(2)}\right)^T T^{(1)}\right] e_{k_1}^{(1)} , \tag{2}$$

where $T^{(1)}$ is a $d \times d \times d$ tensor, modeling the interaction of these two entities. The tensor multiplication $\left(e_{k_2}^{(2)}\right)^T T^{(1)}$ can be defined as:

$$\left(e_{k_2}^{(2)}\right)^T T^{(1)} = \begin{bmatrix} \left(e_{k_2}^{(2)}\right)^T T_1^{(1)} \\ ... \\ \left(e_{k_2}^{(2)}\right)^T T_d^{(1)} \end{bmatrix} , \tag{3}$$

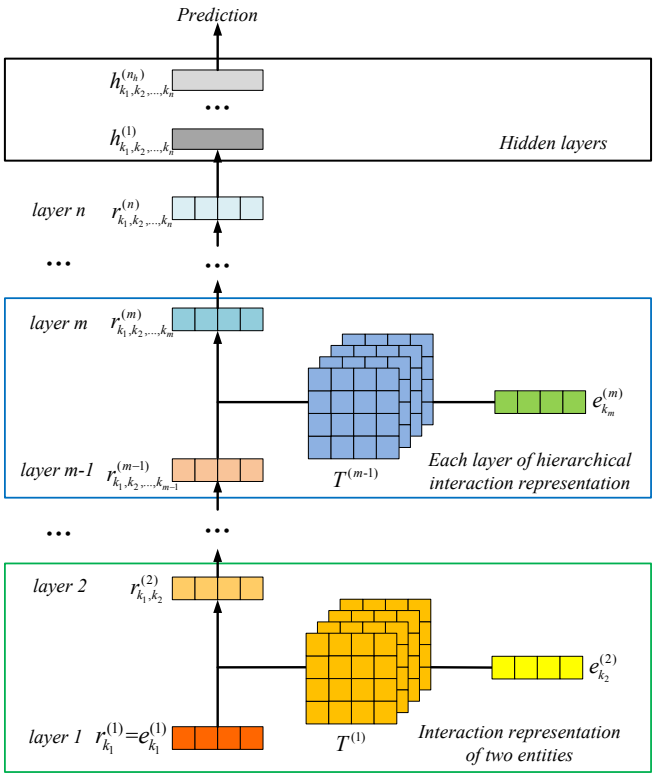


Figure 2: Overview of the learning procedure of HIR. Interaction representation of two entities is shown in the bottom square. Each layer of HIR is shown in the middle square. The top part illustrates hidden layers between interaction representation and prediction.

and Equation 2 can be calculated as:

$$r^{(2)}_{k_1,k_2} = \begin{bmatrix} \left(e^{(2)}_{k_2}\right)^T T^{(1)}_1 e^{(1)}_{k_1} \\ \cdots \\ \left(e^{(2)}_{k_2}\right)^T T^{(1)}_d e^{(1)}_{k_1} \end{bmatrix} , \quad (4)$$

where $T^{(1)}_m$ denotes the mth slice of the tensor $T^{(1)}$, and each slice is a $d \times d$ matrix.

With such computation, the tensor can model the rich high-order relation between entities. Each slice of the tensor can capture a specific type of relation between these entities.

Now, to predict the interaction of the two entities, we can use a simple linear regression, which can be concluded as:

$$\hat{y}_{k_1,k_2} = W^T \left[\left(e^{(2)}_{k_2}\right)^T T^{(1)} \right] e^{(1)}_{k_1} , \quad (5)$$

where W is a n-dimensional vector denoting the weights of regression.

3.3 Hierarchical Interaction Representation

After introducing the joint representation of two entities, we would model the joint representation of more entities in general cases. For more entities, we need another interaction function that satisfies:

$$r^{(n)}_{k_1,\ldots,k_n} = f^{(n)}\left(e^{(1)}_{k_1}, \ldots, e^{(n)}_{k_n}\right) , \quad (6)$$

where $f^{(n)}(\cdot)$ denotes an interaction function of n entities. $r^{(n)}_{k_1,\ldots,k_n}$ describes the joint characteristics of all the entities in an interaction scenario.

To generate the interaction representation of n entities, we perform the tensor multiplication iteratively, which forms a hierarchical structure with n layers. These procedures can be concluded as:

$$\begin{cases} r^{(1)}_{k_1} = e^{(1)}_{k_1} \\ r^{(m)}_{k_1,\ldots,k_m} = \left[\left(e^{(m)}_{k_m}\right)^T T^{(m-1)} \right] r^{(m-1)}_{k_1,\ldots,k_{m-1}} \end{cases} , \quad (7)$$

where $1 < m \leq n$, and $T^{(m-1)}$ is a $d \times d \times d$ tensor, generating the interaction representation in the mth layer. Each layer of HIR is illustrated in the middle square of Figure 2. Thus, the joint representation of all entities can be calculated as:

$$r^{(n)}_{k_1,\ldots,k_n} = \prod_{i=2}^{n} \left[\left(e^{(i)}_{k_i}\right)^T T^{(i-1)} \right] e^{(1)}_{k_1} . \quad (8)$$

Taking latent collaborative retrieval for an example, there are three entities, i.e., user, query and document. Their interaction representation captures how a user prefers a document while retrieving with a query, which can be calculated as:

$$r^{(3)}_{k_1,k_2,k_3} = \left[\left(e^{(3)}_{k_3}\right)^T T^{(2)} \right] [\left(e^{(2)}_{k_2}\right)^T T^{(1)}] e^{(1)}_{k_1} . \quad (9)$$

With the joint representation, we can compute the collaborative prediction. A variety of machine learning methods can be used according to the specific task. For example, with a linear regression method, the prediction of multi-entity interaction can be generated as:

$$\hat{y}_{k_1,\ldots,k_n} = W^T \prod_{i=2}^{n} \left[\left(e^{(i)}_{k_i}\right)^T T^{(i-1)} \right] e^{(1)}_{k_1} , \quad (10)$$

where W is a n-dimensional vector denoting the weights of regression. Moreover, in the task of latent collaborative retrieval, the prediction can be generated as:

$$\hat{y}_{k_1,k_2,k_3} = W^T \left[\left(e^{(3)}_{k_3}\right)^T T^{(2)} \right] \left[\left(e^{(2)}_{k_2}\right)^T T^{(1)} \right] e^{(1)}_{k_1} . \quad (11)$$

3.4 Multiple Hidden Layers

Deep models with multiple layers have shown delightful performance in different areas [1]. As shown in the top square of Figure 2, to enhance the ability of representation and to improve the performance of our model, we can add several hidden layers between the joint representation $r^{(n)}_{k_1,\ldots,k_n}$ and the prediction $\hat{y}_{k_1,\ldots,k_n}$.

Suppose we add n_h hidden layers between the final representation and the prediction. With each hidden layer modeled by a matrix, the joint representation after hidden layers can be calculated iteratively as:

$$\begin{cases} h^{(0)}_{k_1,k_2,\ldots,k_n} = r^{(n)}_{k_1,k_2,\ldots,k_n} \\ h^{(m_h)}_{k_1,k_2,\ldots,k_n} = H_{m_h} h^{(m_h-1)}_{k_1,k_2,\ldots,k_n} \end{cases} , \quad (12)$$

where $1 \leq m_h \leq n_h$, $h^{(m_h)}_{k_1,k_2,\ldots,k_n}$ denotes the joint representation after m_h hidden layers, and H_{m_h} is a $d \times d$ matrix of the m_hth hidden layer. Combining interaction layers and hidden layers, the overall formulation generating the joint

representation becomes:

$$h^{(n_h)}_{k_1,\dots,k_n} = \prod_{i=1}^{n_h} H_i \prod_{i=2}^{n} \left[\left(e^{(i)}_{k_i} \right)^T T^{(i-1)} \right] e^{(1)}_{k_1} \ . \quad (13)$$

Furthermore, with a linear regression method, the final prediction can be made via:

$$\hat{y}_{k_1,\dots,k_n} = W^T \prod_{i=1}^{n_h} H_i \prod_{i=2}^{n} \left[\left(e^{(i)}_{k_i} \right)^T T^{(i-1)} \right] e^{(1)}_{k_1} \ , \quad (14)$$

4. LEARNING FOR DIFFERENT TASKS

In this section, we introduce the learning process of the HIR model in three different tasks, i.e., personalised recommendation, personalised ranking and click-through rate prediction.

4.1 Collaborative Recommendation

In recommender systems with explicit rating values, the interaction prediction in HIR becomes rating prediction, which can be modeled as a regression problem. The model can be calculated via a linear regression:

$$\hat{y}_{k_1,\dots,k_n} = W^T h^{(n_h)}_{k_1,\dots,k_n} \ , \quad (15)$$

where W denotes the weights of linear regression. Therefore, HIR can be learned by minimizing the squared error [8], and the objective function can be written as:

$$\begin{aligned} J &= \sum (y_{k_1,\dots,k_n} - \hat{y}_{k_1,\dots,k_n})^2 + \frac{\lambda}{2} \|\Theta\|^2 \\ &= \sum (y_{k_1,\dots,k_n} - W^T h^{(n_h)}_{k_1,\dots,k_n})^2 + \frac{\lambda}{2} \|\Theta\|^2 \end{aligned} \ , \quad (16)$$

where $\Theta = \{E, T, H, W\}$ denotes all the parameters to be estimated, and λ is a parameter to control the power of regularization.

The derivations of J with respect to the parameters can be calculated as:

$$\frac{\partial J}{\partial W} = -2 \sum (y_{k_1,\dots,k_n} - W^T h^{(n_h)}_{k_1,\dots,k_n}) h^{(n_h)}_{k_1,\dots,k_n} + \lambda W \ ,$$

$$\frac{\partial J}{\partial h^{(n_h)}_{k_1,\dots,k_n}} = -2 \sum (y_{k_1,\dots,k_n} - W^T h^{(n_h)}_{k_1,\dots,k_n}) W + \lambda h^{(n_h)}_{k_1,\dots,k_n} \ .$$

4.2 Personalized Ranking

In recommender systems and information retrieval, different from scenarios with explicit feedbacks, scenarios with implicit feedbacks (i.e., a binary signal such as click, buy, view, etc) are more common in real-world applications. This can be treated as a ranking problem. Bayesian Personalized Ranking (BPR) [18] is a state-of-the-art pairwise ranking method for the implicit feedback data. The basic assumption of BPR is that a user prefers a selected item than a negative item. So, using HIR in the BPR framework, we need to maximize the following function:

$$p(k_n \succ_{k_1,k_2,\dots} k'_n) = g(\hat{y}_{k_1,k_2,\dots,k_n} - \hat{y}_{k_1,k_2,\dots,k'_n}) \ , \quad (17)$$

where k_n denotes a positive sample, k'_n denotes a negative sample, and $g(x)$ is a nonlinear function which we select as $g(x) = 1/(1 + exp(-x))$. Note that, the prediction here is based on linear regression as shown in Equation 15. Incorporating the negative log likelihood, we can solve the following objective function equivalently:

$$\begin{aligned} J &= -\sum \ln(p(k_n \succ_{k_1,k_2,\dots} k'_n)) + \frac{\lambda}{2} \|\Theta\|^2 \\ &= -\sum \ln \left(\frac{1}{1 + \exp(-y(k_n \succ_{k_1,k_2,\dots} k'_n))} \right) + \frac{\lambda}{2} \|\Theta\|^2 \\ &= \sum \ln(1 + \exp(-y(k_n \succ_{k_1,k_2,\dots} k'_n)) + \frac{\lambda}{2} \|\Theta\|^2 \\ &= \sum \ln(1 + \exp(-W^T h(k_n \succ_{k_1,k_2,\dots} k'_n))) + \frac{\lambda}{2} \|\Theta\|^2 \end{aligned} \quad (18)$$

where

$$y(k_n \succ_{k_1,k_2,\dots} k'_n) = \hat{y}_{k_1,k_2,\dots,k_n} - \hat{y}_{k_1,k_2,\dots,k'_n} \ ,$$

$$h(k_n \succ_{k_1,k_2,\dots} k'_n) = h^{(n_h)}_{k_1,k_2,\dots,k_n} - h^{(n_h)}_{k_1,k_2,\dots,k'_n} \ .$$

The derivations of J with respect to the parameters can be calculated as:

$$\frac{\partial J}{\partial W} = -\sum q(k_n \succ_{k_1,k_2,\dots} k'_n) h(k_n \succ_{k_1,k_2,\dots} k'_n) + \lambda W \ ,$$

$$\frac{\partial J}{\partial h^{(n_h)}_{k_1,k_2,\dots,k_n}} = -\sum q(k_n \succ_{k_1,k_2,\dots} k'_n) W + \lambda h^{(n_h)}_{k_1,k_2,\dots,k_n} \ ,$$

$$\frac{\partial J}{\partial h^{(n_h)}_{k_1,k_2,\dots,k'_n}} = \sum q(k_n \succ_{k_1,k_2,\dots} k'_n) W + \lambda h^{(n_h)}_{k_1,k_2,\dots,k'_n} \ ,$$

where

$$q(k_n \succ_{k_1,k_2,\dots} k'_n) = \frac{\exp(-W^T h(k_n \succ_{k_1,k_2,\dots} k'_n))}{1 + \exp(-W^T h(k_n \succ_{k_1,k_2,\dots} k'_n))} \ .$$

4.3 Click-Through Rate Prediction

Predicting the probability of clicking under a specific context, click-through rate prediction can be treated as a classification problem. As regular models for CTR prediction [12, 29], we incorporate logistic regression, and the prediction becomes:

$$\hat{y}_{k_1,\dots,k_n} = \frac{1}{1 + exp(-W^T h^{(n_h)}_{k_1,\dots,k_n})} \ , \quad (19)$$

where W denotes the weights of logistic regression. As in regular logistic regression, we use the negative log likelihood for model learning. The objective function can be written as:

$$\begin{aligned} J &= -\sum (1 - y_{k_1,k_2,\dots,k_n}) \ln (1 - \hat{y}_{k_1,k_2,\dots,k_n}) \\ &\quad - \sum y_{k_1,k_2,\dots,k_n} \ln (\hat{y}_{k_1,k_2,\dots,k_n}) + \frac{\lambda}{2} \|\Theta\|^2 \\ &= \sum \ln(1 + \exp(-W^T h^{(n_h)}_{k_1,\dots,k_n})) \\ &\quad + \sum (1 - y_{k_1,k_2,\dots,k_n}) W^T h^{(n_h)}_{k_1,\dots,k_n} + \frac{\lambda}{2} \|\Theta\|^2 \end{aligned} \quad (20)$$

The derivations of J with respect to the parameters can be calculated as:

$$\begin{aligned} \frac{\partial J}{\partial W} = -\sum \frac{\exp(-W^T h^{(n_h)}_{k_1,\dots,k_n})}{1 + \exp(-W^T h^{(n_h)}_{k_1,\dots,k_n})} h^{(n_h)}_{k_1,\dots,k_n} \\ + \sum (1 - \hat{y}_{k_1,k_2,\dots,k_n}) + \lambda W \end{aligned} \ ,$$

$$\frac{\partial J}{\partial h^{(n_h)}_{k_1,\ldots,k_n}} = -\sum \frac{\exp(-W^T h^{(n_h)}_{k_1,\ldots,k_n})}{1+\exp(-W^T h^{(n_h)}_{k_1,\ldots,k_n})} W + \sum (1-\hat{y}_{k_1,k_2,\ldots,k_n}) + \lambda h^{(n_h)}_{k_1,\ldots,k_n} .$$

4.4 Iterative Parameter Learning

In three tasks mentioned above, the derivation $\frac{\partial J}{\partial h^{(n_h)}_{k_1,\ldots,k_n}}$ has been calculated. Based on this derivation, we can calculate the derivations of hidden layers and joint representation iteratively. Suppose we have $\frac{\partial J}{\partial h^{(m_h)}_{k_1,\ldots,k_m}}$ of the m_hth $(0 < m_h \leq n_h)$ hidden layer, we can calculate the derivations of parameters on this layer as:

$$\frac{\partial J}{\partial H_{m_h}} = \frac{\partial J}{\partial h^{(m_h)}_{k_1,k_2,\ldots,k_n}} h^{(m_h-1)}_{k_1,k_2,\ldots,k_n} ,$$

$$\frac{\partial J}{\partial h^{(m_h-1)}_{k_1,k_2,\ldots,k_n}} = H^T_{m_h} \frac{\partial J}{\partial h^{(m_h)}_{k_1,k_2,\ldots,k_n}} .$$

On the first hidden layer, we can obtain the derivation of the joint representation:

$$\frac{\partial J}{\partial r^{(n)}_{k_1,\ldots,k_n}} = \frac{\partial J}{\partial h^{(0)}_{k_1,\ldots,k_n}} .$$

Now, with the derivation of the joint representation $\frac{\partial J}{\partial r^{(n)}_{k_1,\ldots,k_n}}$, we can calculate all the derivations iteratively. Suppose we have $\frac{\partial J}{\partial r^{(m)}_{k_1,\ldots,k_m}}$ of the mth $(1 < m \leq n)$ layer, we can calculate the derivations of parameters on this layer as:

$$\frac{\partial J}{\partial r^{(m-1)}_{k_1,\ldots,k_{m-1}}} = \left[\left(e^{(m)}_{k_m}\right)^T T^{(m-1)}\right]^T \frac{\partial J}{\partial r^{(m)}_{k_1,\ldots,k_n}} ,$$

$$\frac{\partial J}{\partial e^{(m)}_{k_m}} = \left[(r^{(m-1)}_{k_1,\ldots,k_{m-1}})^T (T^{(m-1)})^T\right]^T \frac{\partial J}{\partial r^{(m)}_{k_1,\ldots,k_n}} ,$$

$$\frac{\partial J}{\partial T^{(m-1)}_i} = e^{(m)}_{k_m} \left(r^{(m-1)}_{k_1,\ldots,k_{m-1}}\right)^T \left(\frac{\partial J}{\partial r^{(m)}_{k_1,\ldots,k_n}}\right)_i .$$

On the first layer, we can obtain the derivation of the first entity:

$$\frac{\partial J}{\partial e^{(1)}_{k_1}} = \frac{\partial J}{\partial r^{(1)}_{k_1}} .$$

After calculating all the derivations, a solution of learning HIR can be obtained by using stochastic gradient descent (SGD), which has been widely employed [8, 7]. Note that, the learning method can be changed according to different applications.

5. EXPERIMENTS

In this section, we empirically investigate the performance of HIR. As shown in Table 1, we conduct our experiments in four tasks: **general recommendation** with *user* and *movie*, **context-aware recommendation** with *user*, *item* and *context*, **latent collaborative retrieval** with *user*, *URL* and *tag*, and **click-through rate prediction** with *device*, *website*, *application*, *advertisement* and *position*. In all experiments, we implement HIR with 0, 1 and 2 hidden layers, denoted as HIR, HIR+ and HIR++ respectively. Then we analyze the impact of the dimensionality of latent representations and examine the interacting order of entities in context-aware recommendation and latent collaborative retrieval. Finally, we visualize representations in latent collaborative retrieval.

5.1 Experimental Settings

Our experiments are conducted on four real datasets for different applications:

- **Movielens-10M**[1] is a widely used dataset for rating prediction in recommender systems, which can be used for performance evaluation for general recommendation. There are two entities in the dataset: *user* and *movie*, with $10M$ observations.
- **Food** [15] is a dataset for rating prediction with contextual information. It is a suitable dataset for context-aware recommendation and has been used in previous works [4, 19, 21, 10]. This dataset contains three entities: *user*, *item* and *context*, with $6.3k$ observations. The contextual information captures the user's hunger degree and whether the user's feeling about hunger is real or virtual.
- **Delicious**[2] contains three entities: *user*, *URL* and *tag*, with about $0.4M$ observations. This dataset can be used for latent collaborative retrieval when considering *URL* as query and *tag* as document, which means a *user* retrieves *tags* for an untagged *URL*. Similar to [17], we use p-core[3] for filtering the dataset and p is chosen as 10.
- **Avazu**[4] is a dataset for click-through rate prediction, in which we extract five entities: *device*, *website*, *application*, *advertisement* and *position*, with $24M$ observations. We conduct two experiments on this dataset with the only difference of whether the entity *position* is included.

For all the datasets, we use 70% for training, 20% for testing, and the remaining 10% data as the validation set for tuning parameters, i.e., the dimensions of latent representations. And the regulation parameter is set as $\lambda = 0.01$. Moreover, we have different evaluation metrics for different tasks:

- **Root Mean Square Error (RMSE)** and **Mean Average Error (MAE)** are the most popular metrics for rating prediction. We use them for general recommendation and context-aware recommendation. The smaller the value, the better the performance.
- **Recall@k**, **Precision@k** and **F1-score@k** are three important metrics for ranking tasks. We use them for latent collaborative retrieval. The evaluation score for a given *user-URL-tag* triple is computed according to

[1]http://grouplens.org/datasets/movielens/
[2]http://grouplens.org/datasets/hetrec-2011/
[3]The p-core of the dataset is the largest subset of the dataset with the property that every user, every item and every tag has to occur in at least p posts.
[4]https://www.kaggle.com/c/avazu-ctr-prediction/data

Table 1: Experimental summarization

task	problem	dataset	#entities	entities	compared methods	evaluation metrics
recommendation	regression	Movielen-10M	2	*user*, *movie*	MF, FM	RMSE MAE
context-aware recommendation	regression	Food dataset	3	*user*, *item*, *context*	MF, LR, TF, FM, CARS2, COT	RMSE, MAE
lantent collaborative retreval	ranking	Delicious dataset	3	*user*, *url*, *tag*	MF, LR, TF, FM, LCR	recall@5, recall@10, recall@20, MAP
click-through rate prediction	classification	Avazu dataset	4/5	*device*, *application*, *website*, *advertisment*, (*position*)	LR, FM	LogLoss

where the tag appears in the ranked list. We report recall@k, precision@k and F1-score@k with $k = 5$, 10 and 20 in our experiments. The larger the value, the better the performance.

- **Mean Average Precision (MAP)** is another commonly used metric for evaluation in ranking tasks. MAP is a standard metric for evaluating the quality of ranked lists, and its top-bias property is particularly important for ranking tasks such as top-N recommendation and information retrieval. We use it as a global evaluation for latent collaborative retrieval. The larger the value, the better the performance.

- **LogLoss** is commonly used for evaluation in click-through rate prediction. The smaller the value, the better the performance.

5.2 Compared Methods

We compare HIR with both general and task-specific methods:

- **MF** [13] is used as a baseline for modeling interaction between two entities, i.e., general recommendation, ignoring *context* in context-aware recommendation and ignoring *user* in latent collaborative retrieval.

- **LR** is a widely used classifier and used as a method for click-through rate prediction [12, 29]. It can be implemented based on one-hot representation for context-aware recommendation and latent collaborative retrieval.

- **TF** [28] is a state-of-the-art method for context-aware recommendation [4] and tag recommendation [25, 17]. We use it as a baseline for both context-aware recommendation and latent collaborative retrieval.

- **FM** [16] is a compared and used as a baseline method in all the four experiments, as it is a state-of-the-art method used for different applications [20, 19, 14]. We use LibFM[5] to implement the method.

- **CARS2** [21] and **COT** [10] are two state-of-the-art methods for learning contextual representation and used as compared methods for the context-aware recommendation

- **LCR** [27, 2] is a state-of-the-art method for latent collaborative retrieval and implemented on the Delicious dataset in our experiments.

[5]http://www.libfm.org/

5.3 Performance Comparison

5.3.1 General Recommendation

The top part of Table 2 illustrates the results measured by RMSE and MAE on Movielens-10M, which show that HIR achieves the best results in general recommendation compared with MF and FM. HIR, HIR+ and HIR++ improve the performance of FM by 0.6%, 1.1% and 1.5% respectively. The results show that the high-order tensor multiplication in HIR can better represent the joint characteristics of *user-item* pair and thus improve the performance of collaborative prediction.

Table 2: Performance of general recommendation and context-aware recommendation evaluated by RMSE and MAE.

dataset	method	HIRSE	MAE
Movielens-10M	MF	0.8946	0.7159
	FM	0.8912	0.7128
	HIR	0.8863	0.7092
	HIR+	0.8814	0.7054
	HIR++	**0.8772**	**0.7025**
Food dataset	MF	1.1552	0.9484
	LR	1.1263	0.9136
	TF	1.0635	0.8415
	FM	1.0554	0.8453
	CARS2	1.0201	0.8162
	COT	1.0019	0.7921
	HIR	0.9757	0.7816
	HIR+	0.9643	0.7723
	HIR++	**0.9586**	**0.7684**

5.3.2 Context-aware Recommendation

The bottom part of Table 2 shows the results of context-aware recommendation measured by RMSE and MAE. HIR outperforms the state-of-the-art context-aware methods. And from the results, we can see that, comparing with COT, the performance improvements of HIR, HIR+ and HIR++ are 2.7%, 3.8% and 4.4% respectively. The improvement involved by the first and second hidden layer are 1.1% and 0.6% respectively, which proves that hidden layers can improve the performance but yield a smaller improvement than that of HIR itself. Moreover, we can observe that the improvement of hidden layers decreases gradually with its number, which means that the effort of hidden layers has its

Table 3: Performance of latent collaborative retrieval evaluated by recall, precision and F1-score.

dataset	method	recall			precision			F1-score			MAP
		@5	@10	@20	@5	@10	@20	@5	@10	@20	
Delicious dataset	MF	0.1116	0.1552	0.1952	0.0915	0.0636	0.0401	0.1006	0.0902	0.0665	0.0919
	LR	0.1397	0.1963	0.2576	0.1146	0.0804	0.0527	0.1259	0.1141	0.0875	0.1051
	TF	0.1613	0.2246	0.2931	0.1323	0.0918	0.0601	0.1454	0.1303	0.0997	0.1144
	FM	0.1638	0.2268	0.2965	0.1343	0.0927	0.0607	0.1476	0.1316	0.1008	0.1155
	LCR	0.1654	0.2297	0.3013	0.1356	0.0939	0.0617	0.1491	0.1333	0.1024	0.1164
	HIR	0.1792	0.2484	0.3274	0.1469	0.1017	0.0671	0.1615	0.1443	0.1114	0.1222
	HIR+	0.1856	0.2553	0.3282	0.1522	0.1046	0.0672	0.1672	0.1484	0.1116	0.1255
	HIR++	**0.1867**	**0.2565**	**0.3314**	**0.1531**	**0.1049**	**0.0679**	**0.1682**	**0.1489**	**0.1127**	**0.1266**

own limitation and we do not need to set too many hidden layers.

5.3.3 Latent Collaborative Retrieval

Table 3 illustrates the results of latent collaborative retrieval measured by recall, precision, F1-score and MAP on the Delicious dataset. Note that, all the methods are implemented under the BPR framework. The results obviously show that HIR greatly outperform the compared methods. On recall@5, HIR, HIR+ and HIR++ improve the performance of LCR by 8.5%, 12.1% and 12.7% respectively. And on recall@10 and recall@20, the corresponding improvements become 8.3%, 11.4%, 11.8% and 8.6%, 8.9%, 10.1%. Measured by precision and F1-score, the same improvements can be observed. Moreover, on global evaluation metric MAP, our proposed HIR, HIR+ and HIR++ improve the performance by 5.0%, 7.8% and 8.8% respectively. We can draw similar conclusion that the performance improvement of hidden layers decreases gradually with the number of hidden layers in most experiments. The table also shows that, compared with the results on @5 and @10, the second hidden layer of HIR++ on @20 brings a greater improvement, which means that when we generate more predicted results, more hidden layers can be used to further improve the performance.

5.3.4 Click-through Rate Prediction

The results of click-through rate prediction measured by LogLoss are illustrated in Table 4. Note that, all the methods are implemented with the same objective function under LogLoss. We can observe that, both latent factor based methods, i.e., FM and HIR, outperform LR, which shows that latent factor based methods can better discover underlying relation of different entities. Moreover, in both experiments, without or with the *position* entity in the advertisement impression, HIR achieves the best performance. Compared with FM, HIR, HIR+ and HIR++ improve the performance by 1.4%, 2.2%, 2.5% and 1.5%, 2.3%, 2.5% respectively in these two experiments, which shows the promising performance of HIR for multi-entity interaction prediction with a number of entities.

5.4 Impact of Dimension

Performance of HIR and compared methods on the Food dataset and the Delicious dataset with varying d is shown in Figure 3. For simplicity, we do not illustrate performance of MF and LR in the figure. Instead of that, we illustrate the performance of several state-of-the-art methods and HIR with 0, 1 and 2 hidden layers in our experiments.

Table 4: Performance of click-through rate prediction evaluated by LogLoss.

dataset	method	*position*	
		excluded	included
Avazu dataset	LR	0.4186	0.4145
	FM	0.4053	0.4025
	HIR	0.3994	0.3962
	HIR+	0.3962	0.3934
	HIR++	**0.3953**	**0.3921**

As shown in Figure 3(a), for all the methods, with increasing d, the value of RMSE decreases at first, then increases after $d = 5$, which means that 5 is the best dimension for all methods. We can also observe that HIR outperforms the compared methods consistently with all dimensions. Moreover, as shown in Figure 3(b), $d = 8$ is the best parameter for all ethods on the Delicious dataset. We can observe that the impact of dimensionality for latent collaborative retrieval is smaller than that for context-aware recommendation, and HIR performs the best compared to other methods.

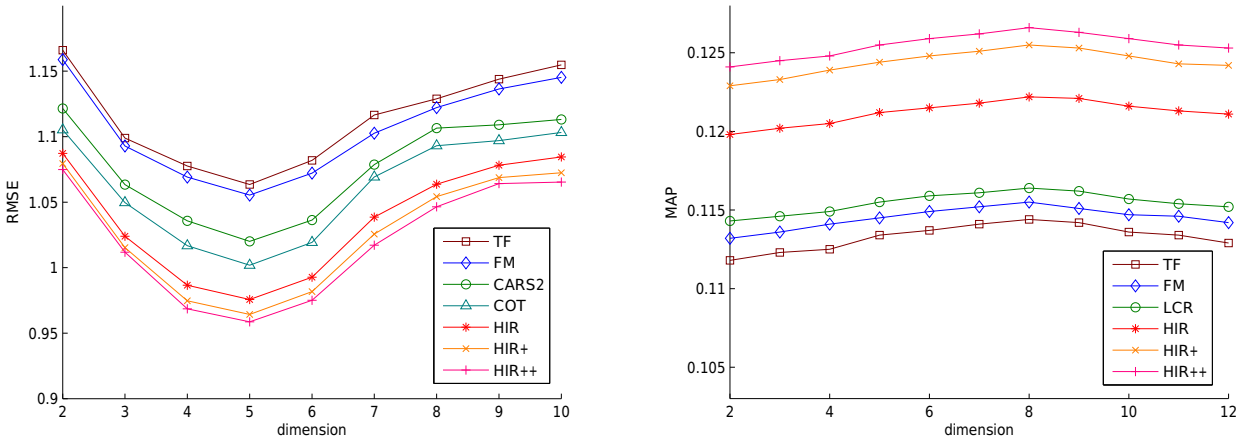
5.5 Impact of Interacting Order

In previous sections, we have illustrated the order *user*, *query*, *document* of information retrieval and the order *user*, *context*, *item* of context-aware recommendation. However, for some applications, the orders are sometimes difficult to be determined exactly. In such situations, it will cost lots of labor to determine the proper order. Thus, to examine the impact of interacting order of entities in HIR, we conduct experiments with all possible orders for two tasks: context-aware recommendation on the Food dataset and latent collaborative retrieval on the Delicious dataset. We implement HIR with 0, 1 and 2 hidden layers in this experiment.

Table 5: Performance under different interacting order on the Food dataset evaluated by RMSE of HIR.

order	HIR	HIR+	HIR++
(user,context,item)	0.9757	0.9643	0.9586
(item,context,user)	0.9763	0.9652	0.9583
(user,item,context)	0.9744	0.9633	0.9582
(context,user,item)	0.9786	0.9655	0.9596
(item,user,context)	0.9782	0.9663	0.9603
(context,item,user)	0.9798	0.9687	0.9594
variance	4.12E-06	3.45E-06	6.95E-07

The RMSE results of HIR under different interacting orders on the Food dataset is shown in Table 5. The vari-



(a) Food dataset

(b) Delicious dataset

Figure 3: Performance of HIR and compared methods on the Food dataset and the Delicious dataset with varying d.

Table 6: Performance under different interacting order on the Delicious dataset evaluated by MAP.

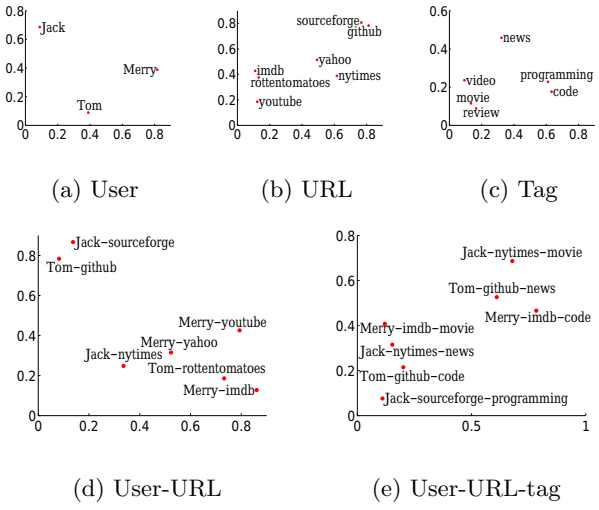
order	HIR	HIR+	HIR++
(user,URL,tag)	0.1222	0.1255	0.1266
(tag,URL,user)	0.1243	0.1263	0.1275
(user,tag,URL)	0.1196	0.1234	0.1243
(URL,user,tag)	0.1217	0.1242	0.1262
(tag,user,URL)	0.1232	0.1251	0.1267
(URL,tag,user)	0.1206	0.1228	0.1231
variance	2.91E-06	1.75E-06	2.80E-06

ances of RMSE results are also shown in the table. Note that, these results are all based on the same initialization. With the same number of hidden layers, performances of HIR are approximately the same with different interacting orders. The small variances also show that the impact of the interacting order for context-aware recommendation is not significant. Similar conclusion that the impact of the interacting order for latent collaborative retrieval is not significant can also be drawn from Table 6. In a word, in these applications, different interacting orders often achieve similar performance. The unnecessary of the settled structure can greatly reduce the labor for analyzing the raw data and selecting a suitable order when there is no obvious one.

5.6 Visualization of Representation

Here, using the Delicious dataset as an example, we plan to demonstrate the representations in HIR, and describe some interesting observations. In Figure 4, we use principal component analysis (PCA) to project the representations into a two-dimensional space. The distance in this figure reveals the relation of different representations.

We select three distinguishable users and name them as *Jack*, *Tom* and *Marry*, as shown in Figure 4(a). The representations of *URL* and *tag* are illustrated in Figure 4(b) and Figure 4(c) respectively, in which we can observe that the representations of similar *URLs* or *tags* are close. Figure



(a) User (b) URL (c) Tag

(d) User-URL (e) User-URL-tag

Figure 4: Visualization of interaction representation in the Delicious dataset.

4(d) shows the joint representations of *user* and *URL*. Although *Jack* and *Tom* have different representations, they have similar interaction representations when visiting programming websites. Similarly, although *Tom* and *Marry* have different representations, they have similar interaction representations when visiting movie websites. Figure 4(e) illustrates the interaction representations of *user*, *URL* and *tag*. We can observe that positive samples and negative samples are clearly divided into two groups in the two-dimensional space according to their interaction representations.

6. CONCLUSIONS AND FUTURE WORK

In this paper, for modeling collaborative multi-entity interaction, a novel method, i.e., HIR, has been proposed. In HIR, we generate the interaction representation of two en-

tities via tensor multiplication and this process is repeated iteratively for the interaction of several entities. This procedure forms a hierarchical structure and can generate the final joint representation. Then the prediction can be calculated based on this representation using various learning methods. The experimental results of four different applications show that HIR outperforms both general methods and state-of-the-art task-specific methods.

In the future, we can further investigate the following directions. First, in HIR, the dimensions of entities can be adjusted for different datasets, which leads lots of hyper-parameters to be determined. So, we need to find a method to determine the dimensions automatically. Second, since there are rich features in real-world applications, we plan to incorporate the entities' feature information in HIR.

7. ACKNOWLEDGMENTS

This work is jointly supported by National Basic Research Program of China(2012CB316300), and National Natural Science Foundation of China (61403390, U1435221, 61175003, 61420106015).

8. REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *TPAMI*, 35(8):1798–1828, 2013.

[2] K.-J. Hsiao, A. Kulesza, and A. Hero. Social collaborative retrieval. In *WSDM*, pages 293–302, 2014.

[3] M. Jamali and L. Lakshmanan. Heteromf: recommendation in heterogeneous information networks using context dependent factor models. In *WWW*, pages 643–654, 2013.

[4] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *RecSys*, pages 79–86, 2010.

[5] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD*, pages 426–434, 2008.

[6] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[7] Y. Koren and R. Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*, pages 145–186. 2011.

[8] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[9] K. Li, J. Gao, S. Guo, N. Du, X. Li, and A. Zhang. Lrbm: A restricted boltzmann machine based approach for representation learning on linked data. In *ICDM*, pages 300–309, 2014.

[10] Q. Liu, S. Wu, and L. Wang. Cot: Contextual operating tensor for context-aware recommender systems. In *AAAI*, pages 203–209, 2015.

[11] X. Liu and K. Aberer. Soco: a social network aided context-aware recommender system. In *WWW*, pages 781–802, 2013.

[12] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, et al. Ad click prediction: a view from the trenches. In *SIGKDD*, pages 1222–1230, 2013.

[13] A. Mnih and R. Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2007.

[14] R. J. Oentaryo, E.-P. Lim, J.-W. Low, D. Lo, and M. Finegold. Predicting response in mobile advertising with hierarchical importance-aware factorization machine. In *WSDM*, pages 123–132, 2014.

[15] C. Ono, Y. Takishima, Y. Motomura, and H. Asoh. Context-aware preference model based on a study of difference between real and supposed situation data. In *UMAP*, pages 102–113. 2009.

[16] S. Rendle. Factorization machines with libfm. *TIST*, 3(3):57–78, 2012.

[17] S. Rendle, L. Balby Marinho, A. Nanopoulos, and L. Schmidt-Thieme. Learning optimal ranking with tensor factorization for tag recommendation. In *SIGKDD*, pages 727–736, 2009.

[18] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[19] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *SIGIR*, pages 635–644, 2011.

[20] S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *WSDM*, pages 81–90, 2010.

[21] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, and A. Hanjalic. Cars2: Learning context-aware representations for context-aware recommendations. In *CIKM*, pages 291–300, 2014.

[22] R. Socher, D. Chen, C. D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, pages 926–934, 2013.

[23] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013.

[24] V. Srikumar and C. D. Manning. Learning distributed representations for structured output prediction. In *NIPS*, pages 3266–3274, 2014.

[25] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *RecSys*, pages 43–50, 2008.

[26] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[27] J. Weston, C. Wang, R. Weiss, and A. Berenzweig. Latent collaborative retrieval. In *ICML*, pages 9–16, 2012.

[28] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, pages 211–222, 2010.

[29] L. Yan, W.-J. Li, G.-R. Xue, and D. Han. Coupled group lasso for web-scale ctr prediction in display advertising. In *ICML*, pages 802–810, 2014.

[30] L. Zhang, D. Agarwal, and B.-C. Chen. Generalizing matrix factorization through flexible regression priors. In *RecSys*, pages 13–20, 2011.